Online resource allocation in Markov Chains

Jianhao Jia
ITCS, Shanghai University of Finance and Economics
Shanghai, China
jianhao.jia@outlook.com

Hao Li
ITCS, Shanghai University of Finance and Economics
Shanghai, China
sufelihao@outlook.com

Kai Liu
Ant Group
Hangzhou, China
lk325626@antgroup.com

Ziqi Liu
Ant Group
Hangzhou, China
ziqiliu@antfin.com

Jun Zhou
Ant Group
Hangzhou, China
jun.zhoujun@antfin.com

Nick Gravin
ITCS, Shanghai University of Finance and Economics
Shanghai, China
nikolai@mail.shufe.edu.cn

Zhihao Gavin Tang
ITCS, Shanghai University of Finance and Economics
Shanghai, China
tang.zhihao@mail.shufe.edu.cn

ABSTRACT

A large body of work in Computer Science and Operations Research study online algorithms for stochastic resource allocation problems. The most common assumption is that the online requests have randomly generated i.i.d. types. This assumption is well justified for static markets and/or relatively short time periods. We consider dynamic markets, whose states evolve as a random walk in a market-specific Markov Chain. This is a new model that generalizes previous i.i.d. settings. We identify important parameters of the Markov chain that is crucial for obtaining good approximation guarantees to the expected value of the optimal offline algorithm which knows realizations of all requests in advance. We focus on a stylized single-resource setting and: (i) generalize the well-known Prophet Inequality from the optimal stopping theory (single-unit setting) to Markov Chain setting; (ii) in multi-unit setting, design a simple algorithm that is asymptotically optimal under mild assumptions on the underlying Markov chain.

ACM Reference Format:
Jianhao Jia, Hao Li, Kai Liu, Ziqi Liu, Jun Zhou, Nick Gravin, and Zhihao Gavin Tang. 2023. Online resource allocation in Markov Chains. In *Proceedings of the ACM Web Conference 2023 (WWW '23), May 1–5, 2023, Austin, TX, USA.* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/3543507.3583428

1 INTRODUCTION

There has been a lot of interest in online resource allocation problems from computer science and operation research communities largely motivated by their applications in domains such as the multi-billion dollar industry of Internet advertising, flight and hotel bookings, and network revenue management. The earlier work in computer science such as well-known Adwords problem introduced by Mehta et al. [Mehta et al. 2007] was focused on the traditional approach to describe the online algorithm's uncertainty about the future via worst-case competitive analysis. However, most of the recent papers (see, e.g., [Devanur et al. 2019]) study *stochastic* settings, where it is assumed that the input stream is drawn i.i.d. from a prior distribution and the objective is taken in expectation over this distribution.

The resource allocation framework can be informally described as follows. A stream of requests arrive online; upon arrival, the type of the request is realized and the algorithm needs to decide whether to serve the request or not. The request type specifies the amounts of each resource it would consume and the request's value (e.g., how much revenue it generates), if the request is served. Each resource has a certain allowed budget/capacity that the algorithm may use. It is commonly assumed that the request types are drawn *i.i.d.* from a known prior distribution F. The goal is to maximize the total value collected by the online algorithm while satisfying the budget constraints. The expected value of the online algorithm is usually compared against the expected value of the offline optimum benchmark. This problem is often treated with the help of linear program (LP) relaxations, primal-dual LPs, and the online linear programming framework (OLP) (see, e.g., [Li and Ye 2022]). The typical results are quite optimistic: they give near optimal performance guarantees of $1 - o(1)$, under the practically motivated *large budgets* assumption, i.e., the assumption that each request may only consume negligibly small amount of a resource relative to the corresponding budget.

While the i.i.d. assumption on the request types greatly simplifies the problem's theoretical analysis and allows to use powerful LP machinery, it is a rather strong assumption. Indeed, by making it we assume that the environment does not change over time, which

WWW '23, May 1–5, 2023, Austin, TX, USA

ACM ISBN 978-1-4503-9416-1/23/04…$15.00
https://doi.org/10.1145/3543507.3583428

in practice is often violated: e.g., the types of search requests directed to advertising platforms might exhibit significant statistical difference based on external factors such as the weather, apartment prices, recent political events, and the general state of the market and economy. There are more realistic models describing the evolution of a market state via transitions in a *Markov chain*. A famous example is the Markov chain describing evolution of a stock market by three states: "bear market", "bull market", and "stagnant market" states.

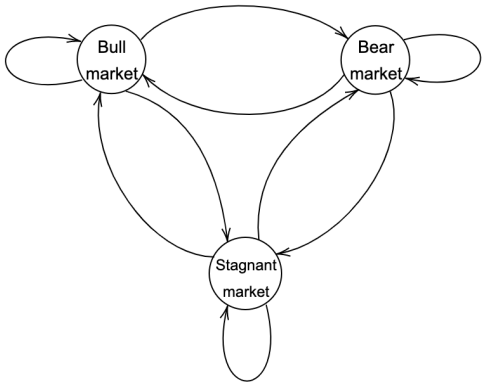

Figure 1: market example

This kind of Markov models is precisely what we propose to use in the online resource allocation framework. Specifically, we propose a new model where the online requests are sampled from a distribution F_s which *depends on the market state* s, that changes over time according to a random walk in a market specific Markov chain. Our main focus is on a stylized model with only one resource. Of course it is a simplification of the actual problem, but all theoretical works in this area consider somewhat simplified abstractions of the problem. Fortunately, this model already provides interesting insights about the power and limitations of the new framework and lets us identify right assumptions on the Markov chains that allow interesting positive results. Moreover, even this stylized model allows to capture other important stochastic online problems such as the well-known prophet inequality from the optimal stopping theory literature and its generalization to k-unit prophet inequality.

1.1 Our Results

We start by exploring the optimal online algorithm in Section 3 and derive a few interesting structural results for the multi-resource and single-resource cases of the Markov chain model. Note that our model can be interpreted as a special case of Markov Decision Processes (MDP) and one can find the optimal policy in any MDP by dynamic programming. However, even for a small number of resources the respective MDP has a large number of states and the optimal online policy described by the dynamic program would be too complex to run in practice. Note that simpler and more robust algorithms (like Samuel-Cahn's single-threshold algorithm [Samuel-Cahn 1984]) in a more basic setting of prophet inequality (PI) are often preferred over the optimal online algorithm described by a simple dynamic program. Thus, in this section, we are interested in structural properties of the optimal solution with the hope that these properties might indicate the existence of simpler and more robust online algorithms with good performance guarantees. We prove that the expected reward function at each state of the Markov chain is an increasing concave function of the remaining resources (in multi-resource case we allow the algorithm to serve requests fractionally, in the single-resource case the result holds for $\{0, 1\}$ decisions). As a consequence, in the single-resource setting, the optimal policy can be interpreted as a decreasing-threshold-based algorithm analogous to the optimal online policy for the prophet inequality.

In the following Sections 4 and 5, we focus on competitive analysis and try to find online algorithms with good guarantees on competitive ratios against the optimal offline solution. We focus on the stylized model with a single resource and identify the right assumptions on the Markov chain model that would allow interesting theoretical results.

In Section 4, we study the single-unit setting when the algorithm may only serve one request. We generalize the classic prophet inequality from optimal stopping theory to the Markov chain model by allowing the algorithm to select the starting state and achieve tight competitive ratio of $\frac{1}{2}$. Without the extra assumption, no constant approximation result is attainable. This assumption is justified when the Markov chain has a small hitting time compared to the total number of time steps. Indeed, if the expected number of steps needed to travel from any given state to another is relatively small compared to the time frame, our algorithm is able to wait until the random walk visits the desired starting state.

We extend our results to the multi-unit setting when the algorithm can serve k requests in Section 5. We design simple online algorithms with $1-o(1)$ competitive ratio when the resource budget and the total number of steps are sufficiently large (the algorithm and its analysis can be extended to multi-resource case). Our result generalizes the multi-unit prophet inequality and the previous results on online resource allocation with i.i.d. requests.

1.2 Related Work

Online resource allocation is closely related to the online matching literature. [Karp et al. 1990] proposed the online bipartite matching problem and designed an optimal $1 - 1/e$ competitive algorithm. Later, their result was generalized to vertex-weighted setting [Aggarwal et al. 2011]. These settings can be viewed as online resource allocation problems for which we only have 1 copy of each resource and each online request can be served by a subset of the resources. The AdWords problem [Mehta et al. 2007] and online b-matching [Kalyanasundaram and Pruhs 2000] are further generalizations of the model in which each offline vertex can be matched multiple times. These settings are typically studied under the large budgets assumption which we mentioned in the introduction. All the above settings are originally studied in the adversarial framework in which no stochastic information about the online requests is known. In contrast to the $1 - o(1)$ competitive ratios achieved in the stochastic settings (e.g., [Devanur and Hayes 2009; Devanur et al. 2019; Li and Ye 2022]), a constant fraction of loss is unavoidable in the adversarial setting.

The AdWords problem is studied in the known i.i.d. model, e.g., in [Alaei et al. 2012; Devanur and Hayes 2009; Devanur et al. 2012], and in the unknown i.i.d. model, e.g., in [Agrawal and Devanur 2015; Devanur et al. 2019; Gupta and Molinaro 2016; Kesselheim et al. 2018; Li and Ye 2022]. The later model only requires that the online requests are drawn from an i.i.d. distribution which is not known in advance to the online algorithm. In other words, the online algorithm needs to learn the distribution and to make online decisions simultaneously. We leave it as an interesting future direction to incorporate the learning part into our Markov chain model. For instance, how should we design algorithms if the Markov chain is unknown to the algorithm beforehand? Going beyond the i.i.d. model, [Devanur et al. 2019] proposed the adversarial stochastic input (ASI) model to allow non-i.i.d. online requests. They provide results for both the known and unknown distribution models. Our model is more general than the ASI model in the known distribution setting. In the unknown distribution setting, they studied a relaxed offline optimum in order to achieve non-trivial theoretical results and hence are not comparable to ours.

Another closely related line of work concerns the prophet inequality. It was first proposed by [Krengel and Sucheston 1977] in the context of the optimal stopping theory. In our context, the optimal stopping problem can be modeled as a Markov chain given by a path of length T with a single-resource and unit capacity constraint. The random walk starts at the head of the path and makes exactly T transitions along the path. They [Krengel and Sucheston 1977] showed that the optimal online algorithm is 1/2 competitive with respect to the expected value of the optimal offline algorithm. Later [Samuel-Cahn 1984] showed that a much simpler single-threshold algorithm can achieve the same tight competitive ratio. More recently, the topic of prophet inequality has received much attention in the algorithmic mechanism design literature due to its close connection to the posted-price mechanisms [Hajiaghayi et al. 2007]. They extended the prophet inequality to the k-unit setting and proved a $1 - O\left(\sqrt{\frac{\log k}{k}}\right)$ competitive ratio. Prior to that, [Kennedy 1985, 1987; Kertz 1986] have also studied multi-choice optimal stopping problems but used different benchmarks. Later, Alaei [Alaei 2014] studied the problem in a more general setting that applies to the multi-unit prophet inequality, and achieved an improved tight ratio of $1 - \frac{1}{\sqrt{k+3}}$. Alaei et al. [Alaei et al. 2012] generalize k-unit prophet inequality to bipartite graphs that model ad allocation scenarios, which is also related to the online resource allocation literature. More recent work on prophet inequalities focuses on complex combinatorial settings, e.g., [Kleinberg and Weinberg 2019] consider a feasibility constraint on the feasible set of online requests given by the base of a matroid, or intersection of k matroids.

2 PRELIMINARIES

2.1 Markov Chains

Consider a discrete Markov chain with a set $N = \{s_1, s_2, \ldots, s_n\}$ of n states. We use M to denote the transition matrix of the Markov chain, where $M_{ij} = \mathbf{Pr}[s_i \to s_j]$ (i-th column and j-th row) contains the transition probability from state s_i to s_j. We use $W(s) = (s(t))_{t=1}^{T}$ to denote a random walk of length T starting from the state $s(1) = s$.

Ergodic Markov Chains. In Section 5, we focus on ergodic Markov chains, i.e., those Markov chains in which it is possible to go from every state to every state. We use $\mathbf{w} = (w_i)_{i=1}^{n}$ to denote its stationary distribution, i.e., $M \cdot \mathbf{w} = \mathbf{w}$. We denote by $\mathtt{Rt}(s_i) = \frac{1}{w_i}$ the expected first return time to each state s_i. The hitting time $\mathtt{Ht} = \max_{s_i, s_j \in N} \mathbf{E}_{W(s_i)}[\min_t \{s(t) = s_j\}]$ is the maximum expected time for the Markov chain to travel between any two states.

2.2 Online Resource Allocation

Next, we present our general model for online resource allocation. The market state $s(t)$ evolves according to a given Markov chain M. Each state s is associated with an a priori known type distribution F_s. At time t, a type $\theta(t)$ is drawn from $F_{s(t)}$. Every type θ is given by its value $v(\theta)$ and a cost vector of m resources which consumes $\mathbf{c}(\theta) = (c_i(\theta))_{i=1}^{m} \in \mathbb{R}^m$. The algorithm $\mathcal{A}$ observes the state $s(t)$ and realized type $\theta(t)$ and decides whether to serve a given type or not indicated by the binary decision variable $x(t) \in \{0, 1\}$; it gets the value $v(t) = v(\theta(t)) \cdot x(t)$ and consumes resources $x(t) \cdot \mathbf{c}(\theta(t))$. The algorithm $\mathcal{A}$ must satisfy capacity constraints: $\sum_{t=1}^{T} x(t) \cdot c_i(\theta(t)) \le C_i$ for each $i \in [m]$ given by the vector $\mathbf{C} = (C_i)_{i=1}^{m}$. That is, there are m types of resources and we have capacity C_i for each of the resource i. The algorithm aims to maximize the total value $\sum_{t=1}^{T} v(t) = \sum_{t=1}^{T} v(\theta(t)) \cdot x(t)$.

Remark 1. *Our process, given by the Markov chain M with a set of finitely supported distributions $(F_s)_{s \in N}$, can be equivalently simulated by another Markov chain $\widehat{M}$ with* deterministic types *in each state. Indeed, one can split each state $s \in N$ of M into m new states with respective types $\{\theta_s^1, \ldots, \theta_s^m\}$ and let transition probabilities from state θ_a^i to θ_b^j be $\widehat{M}_{\theta_a^i, \theta_b^j} \stackrel{\text{def}}{=} \mathbf{Pr}_M[a \to b] \cdot \mathbf{Pr}_{F_b}[\theta_b^j]$. While such a transformation helps to simplify the analysis of our Markov chain model, it also blows up the state space, turning a typically small market model into a huge Markov chain.*

2.3 Stylized Model and Examples

Most of our results focus on a stylized single-resource setting ($m = 1$) with uniform costs, i.e., $c(\theta) = 1$ for any request types θ. In other words, an algorithm can serve at most C requests and the only randomness is from the value of each online request. Our model generalizes the classical prophet inequality setting (PI) from optimal stopping theory and also the most commonly made assumption of i.i.d. request types.

- **Prophet inequality.** Consider a Markov chain M^{PI} representing a path, i.e., the transition probabilities $\mathbf{Pr}[s_i \to s_{i+1}] = 1$ for all $i \in [n-1]$, $\mathbf{Pr}[s_n \to s_n] = 1$. This is equivalent to the PI with a known arrival order. The capacity constraint can be $C = 1$ for the classic PI, or it can be $C = k$ for the multi-unit PI (generalization of PI).
- **I.i.d. requests.** Consider a Markov chain with a single state s_0 and the transition probability $\mathbf{Pr}[s_0 \to s_0] = 1$. This is equivalent to the setting when all online requests are drawn i.i.d..

3 OPTIMAL ONLINE ALGORITHM

In this section we describe the optimal online algorithm and discuss its structural properties. To simplify exposition and notations, we assume that all states in the original Markov chain M have deterministic types and will omit θ in the notations.

Our online problem is a special case of Markov Decision Process (MDP) which in general can be solved by dynamic programming. Specifically for our MDP, the following dynamic programming equation (DPE) on the optimal expected reward $R_s(\mathbf{r}, t)$ for the given time step $t \le T$, vector of remaining resources $\mathbf{r}$, and a starting state $s \in N$ is as follows.

$$\forall s \in N \qquad R_s(\mathbf{r}, t) = \max\big(v_s + \mathbf{R}(\mathbf{r} - \mathbf{c}(s), t+1)^\top \cdot M \cdot \mathbf{e}_s, \\ \mathbf{R}(\mathbf{r}, t+1)^\top \cdot M \cdot \mathbf{e}_s\big), \quad \text{where } \mathbf{R}(\mathbf{r}, t) = \Big(R_s(\mathbf{r}, t)\Big)_{s \in N} \tag{1}$$

The vector $\mathbf{e}_s \in \mathbb{R}^n$ in the above program is an elementary basis vector corresponding to the state s; also $R_s(\mathbf{r}, t) \stackrel{\text{def}}{=} -\infty$ if any coordinate of $\mathbf{r}$ is negative. This MDP is hard to solve even in a single-resource case. Indeed, solving the MDP for a simple path Markov chain with deterministic values and costs is equivalent to the knapsack problem. For multiple resources, our setting generalizes the online bipartite matching studied in [Papadimitriou et al. 2021], which was shown to be APX hard. We present the following nice properties of the MDP which suggest the plausibility of obtaining a PTAS in the single-resource setting. We leave the question of finding such a PTAS for future research.

3.1 Single-resource case

Here, we consider the case of the single-resource, i.e., when the consumption vectors $\mathbf{c}(s) \in \mathbb{R}^1$ for all states $s \in N$. We also assume that all consumption values are normalized[1] as $c(s) = 1$. Then equation (1) can be rewritten for the remaining supply $k \in \mathbb{N}_0$ as

$$\forall s \in N \qquad R_s(k, t) = \max\big(v_s + \mathbf{R}(k-1, t+1)^\top \cdot M \cdot \mathbf{e}_s, \\ \mathbf{R}(k, t+1)^\top \cdot M \cdot \mathbf{e}_s\big), \quad \text{where } \mathbf{R}(k, t) = \Big(R_s(k, t)\Big)_{s \in N}. \tag{2}$$

In the settings where the values $v_s \sim F_s$ are random, it is easy to see from the DPE (2) that the optimal online policy is a threshold policy given by $\boldsymbol{\tau}(k, t) = (\tau_s(k, t))_{s \in N}$, i.e., the online algorithm serves a request in the state s at time t and when the remaining amount of resources is k if and only if the realized value $v_s \ge \tau_s(k, t)$.

We observe nice structural properties on the values of $R_s(k, t)$.

Claim 1. *For each state $s \in N$ and time $t \le T$, the expected reward $R_s(k, t)$ is increasing concave function in the amount of resource k.*

Proof. The proof proceeds by backward induction on t. Specifically, we show that $R_s(k+1, t) - R_s(k, t)$ is non negative decreasing function in $k \in \mathbb{N}_0$ for each $t \le T$ and $s \in N$. In the base case $t = T$, we have

$$\mathbf{R}(k, T) = \mathbf{0}, \ \text{ for } k = 0; \qquad \mathbf{R}(k, T) = \mathbf{v}, \ \text{ for } k \ge 1,$$

where $\mathbf{v} = (v_s)_{s \in N}$. Hence, $R_s(k+1, t) - R_s(k, t) \ge 0$ is decreasing in k for each $s \in N$ and $t = T$. Next, we assume that our claim holds for $t = t_0 + 1$ and we will show that it holds for $t = t_0$. To simplify notations we let $(Q_s(k))_{s \in N} = \mathbf{Q}(k) \stackrel{\text{def}}{=} M^\top \cdot \mathbf{R}(k, t_0 + 1)$, for each $k \in \mathbb{N}_0$. That allows us to rewrite DPE (2) as follows.

$$R_s(k, t_0) = \begin{cases} v_s + Q_s(k-1), & Q_s(k) - Q_s(k-1) < v_s \\ Q_s(k), & Q_s(k) - Q_s(k-1) \ge v_s \end{cases} \tag{3}$$

We note that $Q_s(k)$ is a linear non negative combination of $R_s(k, t_0 + 1)$ for $s \in N$ and all $k \in \mathbb{N}_0$. Thus, by the induction hypothesis $Q_s(\cdot)$ is an increasing concave function and $g(k) \stackrel{\text{def}}{=} Q_s(k) - Q_s(k-1) \ge 0$ is decreasing in k for every state $s \in N$. Now we can plot the discrete derivative $h(k) \stackrel{\text{def}}{=} R_s(k+1, t_0) - R_s(k, t_0)$ of $R_s(k, t_0)$, where $R_s(k, t_0)$ is given by equation (3), as a function of k. The function $h(k)$ is similar to $g(k)$. Indeed, as $g(k)$ is monotonically decreasing in k there is a threshold value $\tau \in \mathbb{N}_0$ such that $R_s(k, t_0) = Q_s(k)$ for all $k < \tau$ and $R_s(k, t_0) = Q_s(k-1) + v_s$ for all $k \ge \tau$. It is easy to verify that $h(k)$ is non negative and decreasing in k, which concludes the proof. □

This allows us to get structural property on the threshold vector $\boldsymbol{\tau}(k, t)$, which has the following explicit form that follows from equation (2).

$$\boldsymbol{\tau}(k, t) = M^\top \cdot \Big(\mathbf{R}(k, t+1) - \mathbf{R}(k-1, t+1)\Big). \tag{4}$$

Claim 2. *For each state $s \in N$ and time $t \le T$, the threshold $\tau_s(k, t)$ is decreasing in k.*

Proof. By claim 1 the discrete derivative $R_s(k, t+1) - R_s(k-1, t+1)$ is a decreasing function in k. I.e., $\mathbf{R}(k, t+1) - \mathbf{R}(k-1, t+1)$ in (4) is coordinate-wise smaller than $\mathbf{R}(k', t+1) - \mathbf{R}(k'-1, t+1)$ for any $k' > k$. Now, since all entries in matrix $M^\top$ are non negative, the resulting vector $\boldsymbol{\tau}(k, t)$ must be coordinate-wise larger than the vector $\boldsymbol{\tau}(k', t)$. □

We note that understanding the dependency of $\boldsymbol{\tau}(k, t)$ on the number of time steps t is less important than dependency on the amount of the resource. Indeed, one may consider a model with geometric stopping time (i.e., when stopping time T is a random variable with geometric distribution), or more generally consider Markov chains with an absorbing state and infinite time horizon $T = \infty$. In the latter case the optimal policy does not depend on the time, i.e., threshold function $\tau_s(k)$ is only a function of the amount of remaining resources and not the time t when each state s is visited.

We note that when vector of available resources $\mathbf{r}$ (or capacities $\mathbf{C}$) is multi-dimensional, then even the task of finding the optimal offline solution is equivalent to solving general integer linear program. On the other hand, the common assumption for the online resource allocation problems is large budgets, i.e., settings where fractional online algorithm is a close approximation to the online algorithm that has to make integral (accept/reject) decisions. Thus, in the multi-resource setting we study *fractional online algorithms* that can serve any fraction $x_s \in [0, 1]$ of a request at state s. The optimal fractional online algorithm is described by the following DPE: for all states $s \in N$ and time steps t (the vector $\mathbf{R}(\mathbf{r}, t) = (R_s(\mathbf{r}, t))_{s \in N}$)

$$R_s(\mathbf{r}, t) \ = \max_{x \in [0,1]} \big(v_s \cdot x + \mathbf{R}(\mathbf{r} - x \cdot \mathbf{c}(s), t+1)^\top \cdot M \cdot \mathbf{e}_s\big), \tag{5}$$

[1] This is not restrictive assumption in the setting with a single resource when all consumption values $c(s)$ are small relative to the capacity constraint C.

where $\mathbf{c}(s)$ is the consumption vector for state s. It turns out that, our claim 1 extends to multi-resource case (see full proof in Appendix A).

Claim 3. *For each time $t \le T$, and each starting state $s \in N$ the expected reward $R_s(\mathbf{r}, t)$ is concave function in $\mathbf{r}$.*

Proof. The proof proceeds by backward induction on t. In the base case for $t = T$,

$$R_s(\mathbf{r}, T) = \max_{x \in [0,1]} \left(v_s \cdot x \;\middle|\; \mathbf{r} \succeq x \cdot \mathbf{c}(s) \right), \tag{6}$$

where $\mathbf{r} \succeq x \cdot \mathbf{c}(s)$ means that vector $\mathbf{r}$ is coordinate-wise larger than vector $x \cdot \mathbf{c}(s)$. We need to check that for any $\alpha \in [0,1]$ and $\mathbf{r}_1, \mathbf{r}_2 \succeq \mathbf{0}$

$$\alpha \cdot R_s(\mathbf{r}_1, T) + (1-\alpha) \cdot R_s(\mathbf{r}_2, T) \le R_s(\alpha \cdot \mathbf{r}_1 + (1-\alpha) \cdot \mathbf{r}_2, T). \tag{7}$$

Let $x_1, x_2 \in [0,1]$ be respective values of x at which $R_s(\mathbf{r}_1, T)$ and $R_s(\mathbf{r}_2, T)$ attain their maximum value in (6). That means that $\mathbf{r}_1 \succeq x_1 \cdot \mathbf{c}(s)$ and $\mathbf{r}_2 \succeq x_2 \cdot \mathbf{c}(s)$. Then $x \stackrel{\text{def}}{=} \alpha \cdot x_1 + (1-\alpha) \cdot x_2 \in [0,1]$ must be feasible for $\alpha \cdot \mathbf{r}_1 + (1-\alpha)\mathbf{r}_2$, i.e., $\alpha \cdot \mathbf{r}_1 + (1-\alpha)\mathbf{r}_2 \succeq x \cdot \mathbf{c}(s)$. Therefore, $R_s(\alpha \cdot \mathbf{r}_1 + (1-\alpha) \cdot \mathbf{r}_2, T) \ge x \cdot v_s = v_s \cdot (\alpha \cdot x_1 + (1-\alpha) \cdot x_2) = \alpha \cdot R_s(\mathbf{r}_1, T) + (1-\alpha) \cdot R_s(\mathbf{r}_2, T)$.

In the induction step, we assume that $R_s(\mathbf{r}, t_0 + 1)$ is concave and we need to show that $R_s(\mathbf{r}, t_0)$ is also a concave function in $\mathbf{r}$. We first observe that the function $Q(\mathbf{r}) \stackrel{\text{def}}{=} \mathbf{R}(\mathbf{r}, t_0 + 1)^\top \cdot M \cdot \mathbf{e}_s$ is concave in $\mathbf{r}$, since it is a non negative linear combination of concave functions. We again need to check (7) for any $\alpha \in [0,1]$ and $\mathbf{r}_1, \mathbf{r}_2 \succeq \mathbf{0}$, where $T \leftarrow t_0$. Let $x_1, x_2 \in [0,1]$ be respective values of x at which $R_s(\mathbf{r}_1, t_0)$ and $R_s(\mathbf{r}_2, t_0)$ attain their maximum value. We set $x \stackrel{\text{def}}{=} \alpha \cdot x_1 + (1-\alpha) \cdot x_2 \in [0,1]$ in the equation (6) for $R_s(\alpha \cdot \mathbf{r}_1 + (1-\alpha) \cdot \mathbf{r}_2, t_0)$ and get that its value is at least

$$v_s \cdot \big(\alpha x_1 + (1-\alpha) x_2\big) + Q\Big(\alpha \mathbf{r}_1 + (1-\alpha)\mathbf{r}_2 - x \cdot \mathbf{c}(s)\Big) \ge$$
$$\alpha \cdot \Big(x_1 \cdot v_s + Q\big(\mathbf{r}_1 - x_1 \cdot \mathbf{c}(s)\big)\Big) + (1-\alpha) \cdot \Big(x_2 \cdot v_s + Q\big(\mathbf{r}_2 - x_2 \cdot \mathbf{c}(s)\big)\Big)$$
$$= \alpha \cdot R_s(\mathbf{r}_1, t_0) + (1-\alpha) \cdot R_s(\mathbf{r}_2, t_0),$$

where we used concavity of $Q(\mathbf{r})$ in the inequality. This concludes the proof of induction step. □

4 PROPHET INEQUALITY

In this section we consider a generalization of the well known Prophet Inequality (PI) from the optimal stopping theory to our Markov Chain setting. In the classic PI there are n boxes with given reward distributions $(F_i)_{i=1}^n$; an online algorithm opens these boxes one by one: at the i-th step, the algorithm opens the i-th box and observes the reward $v_i \sim F_i$; it can either take v_i, in which case the game terminates, or it can irrevocably discard the reward and continue to the next box. The goal is to maximize the online algorithm's expected reward. The *prophet inequality* refers to the competitive ratio of 2 between the expected reward of the best online algorithm[2] and the expected reward of the offline algorithm (the "prophet"), who can see all realized values in advance and would simply stop at the box with the maximum value. The PI can be viewed as a special case of online resource allocation: there is only a single resource with capacity $C = 1$, taking the reward from any box i would consume one unit of this resource. This is also the case in our Markov Chain setting: only a single resource $m = 1$ of capacity $C = 1$, and by serving any state s_i we consume $r = 1$ of the resource. The classic PI setting corresponds to the Markov Chain M^{PI} introduced in the previous section; the algorithm makes $T = n$ steps starting from the initial state of $s(0) = s_1$. We would like to have a constant approximation guarantee analogous to PI, for Markov Chains M beyond the path example. However, as the next simple example below illustrates, this is not possible in general.

Counter-Example. The Markov chain M^c has $n+1$ states $\{s_i\}_{i=0}^n$: state s_0 is absorbing ($M^c_{0,0} = 1$); from each other state s_i, $i \in [n-1]$ we can either go to the next state s_{i+1} with a small probability $M^c_{i,i+1} = \varepsilon > 0$, or go to s_0 with probability $M^c_{i,0} = 1 - \varepsilon$, also $M^c_{n,0} = 1$ for the last state s_n. All states have *deterministic values*: $v_i = \frac{1}{\varepsilon^{i-1}}$ for $i \in [n]$; $v_0 = 0$ for the state s_0. The random walk starts at s_1 and proceeds for $T = n$ steps.

The offline optimum would take the reward from the last state $s_t, t \in [n]$ before the random walk jumps to the absorbing state. The expected value of the offline optimum is $\frac{1}{\varepsilon^{n-1}} \cdot \varepsilon^{n-1} + \sum_{t=0}^{n-2} \frac{1}{\varepsilon^t} \cdot \varepsilon^t(1-\varepsilon) = 1 + (1-\varepsilon)(n-1)$. The online algorithm, on the other hand, must choose upfront the state s_i, $i \in [n]$, until which it will wait and take the reward. For each $i \in [n]$, this strategy has expected reward of $\frac{1}{\varepsilon^{i-1}} \cdot \varepsilon^{i-1} = 1$. I.e., the competitive ratio of any online algorithm is not better than n.

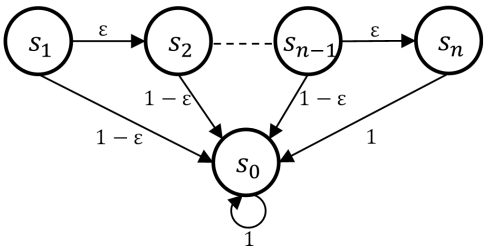

Figure 2: counter-example

The competitive ratio of n is discouraging. Let us take a closer look at the aforementioned example to find what makes it so difficult for the online algorithm. Notice that we do not use the randomness of the distributions F_i, but instead rely on the randomness of the transitions in M^c. The same holds true more generally for any Markov Chain model M with a set of finitely supported distributions $\{F_i\}_{i \in [n]}$: any such process can be equivalently modeled as a random walk in a larger Markov chain with deterministic values at every state. Next, notice that each interaction between a pair of states (s_i, s_{i+1}) in M^c is similar to the standard 2-approximation lower bound example in PI[3]. However, by getting to the state s_{i+1} in M^c, we increase our chances to get to the future high value states $s_j, j \ge i+2$. In other words, the significance of reaching a certain state s_i may be much higher than obtaining a given value from a distribution F_i in PI.

[2] In fact, a much simpler online algorithm that stops at the first value v_i exceeding a uniform threshold τ achieves the same competitive ratio of 2.

[3] The tight PI example has two boxes with distributions: F_1 – $\mathbf{Pr}[v = 1] = 1$; F_2 – $\mathbf{Pr}[v = \frac{1}{\varepsilon}] = \varepsilon, \mathbf{Pr}[v = 0] = 1 - \varepsilon$. Transition from s_i to s_{i+1} state captures the uncertainty of the distribution F_2 in the PI setting.

The latter behavior may only occur in random walks that have only small probability of reaching certain states in the Markov Chain. Such types of Markov Chains are perhaps not the best choice for modeling long term market evolution. It is also not surprising that an online algorithm may not perform well in comparison to the offline algorithm in markets with high unpredictability for a short time decision making processes.

Hence, to avoid the negative results it is reasonable to add an extra assumption that our random walk has a good chance to visit each state s_i of the Markov Chain. I.e., that the number of steps T in our random walk is at least $\Omega(\mathtt{Ht})$, where $\mathtt{Ht}(M)$ is the the *hitting time* of M which is formally defined as $\mathtt{Ht}(M) \stackrel{\text{def}}{=} \max_{s_i,s_j\in[n]} \mathbf{E}_{W(s_i)}[\min_t\{s(t)=s_j\}]$ (the maximal expected number of steps needed to travel from any given state s_i to any other given state s_j in M). This assumption allows the online algorithm to sacrifice a relatively small number of steps and get to any desired state s in M. I.e., by making this assumption, we get the power to *choose the starting point* of our random walk. In the following we obtain the analogue of the classic PI for Markov Chains with the free choice for the starting point. Specifically, we let the online algorithm to pick a starting position $s(1)=s$ and compare its expected performance to the offline algorithm that can also pick the starting state $s(1)$. Hence, our benchmark is

$$\mathtt{prophet} = \max_{s\in N} \mathop{\mathbf{E}}_{W\sim M(s)}\left[\mathop{\mathbf{E}}_{\mathbf{v}\sim\mathbf{F}}\left[\max_{t\in[T]}\{v_{s(t)}\}\right]\right], \tag{8}$$

where a random walk $W=(s(t))_{t=1}^T$ is generated from Markov chain M, starting from $s(1)=s$. Similar to the classic PI we shall use a uniform threshold algorithm with the threshold $\tau=\frac{1}{2}\cdot\mathtt{prophet}$, i.e., the online algorithm that stops at the any state s with the realized value $v_s\ge\tau$.

Theorem 4. *There is a starting state and a uniform threshold online algorithm with expected reward of at least* $0.5\cdot\mathtt{prophet}$.

Proof. First, we would like to specify the starting state s^* of our online algorithm $\mathcal{A}$. The most obvious choice of s^* would be the same state s as in (8) that maximizes prophet. Interestingly, while it is possible to show a constant approximation to prophet for such s^*, there are Markov Chains where the approximation ratio to prophet will be strictly worse than 0.5. Instead we use the following starting state s^*:

$$s^* = \operatorname*{argmax}_s \mathop{\mathbf{E}}_{W\sim M(s)}\left[\mathop{\mathbf{E}}_{\mathbf{v}\sim\mathbf{F}}\left[\max_{t\in[T]}\left\{\left(v_{s(t)}-\tau\right)^+\right\}\right]\right], \tag{9}$$

where $W=(s(t))_{t=1}^T$, $s(1)=s$ and $(x)^+\stackrel{\text{def}}{=}\max\{0,x\}$ for any $x\in\mathbb{R}$. Let us fix the distribution of random walks $W^*\sim M(s^*)$, $W^*=(s(t))_{t=1}^T$, $s(1)=s^*$. Let Q be the probability that our algorithm takes a value $v_{s(t)}\ge\tau$ on the random walk W^*, and let $\{q_{s,t}\}_{s\in N,t\in[T]}$ be the probabilities that our algorithm reaches state s at time t.

$$\begin{cases} Q = & \mathbf{Pr}_{W^*,\mathbf{v}}[\exists t\in[T]\ v_{s(t)}\ge\tau] \\ q_{s,t} = & \mathbf{Pr}_{W^*,\mathbf{v}}[s(t)=s\ \wedge\ \forall\ell<t\ v_{s(\ell)}<\tau], \quad \forall s\in N, t\in[T] \end{cases}$$

We can express the expected reward of online algorithm $\mathcal{A}(s^*)$ as a combination of two parts: revenue (a guaranteed reward of τ, whenever algorithm stops at $v_{s(t)}\ge\tau$), and surplus $((v_{s(t)}-\tau)^+)$.

$$\mathcal{A}(s^*) = Q\cdot\tau + \sum_{\substack{t\in[T],\\ s\in N}} q_{s,t}\cdot\mathbf{Pr}\left[v_s\ge\tau\right]\cdot\mathop{\mathbf{E}}_{v_s}\left[(v_s-\tau)^+\,\middle|\, v_s\ge\tau\right], \tag{10}$$

where the term $Q\cdot\tau$ corresponds to revenue part and in the surplus term the probabilities $\{q_{s,t}\cdot\mathbf{Pr}[v_s\ge\tau]\}_{s,t}$ represent disjoint events that $\mathcal{A}(s^*)$ has stopped at a specific step t in a given state s. The latter also means that

$$Q = \sum_{t\in[T],s\in N} q_{s,t}\cdot\mathbf{Pr}\left[v_s\ge\tau\right]. \tag{11}$$

Now, consider the expectation in (9) for the starting state s^*. Let s^o be the starting state of the prophet and $W^o\sim M(s^o)$ be the corresponding random walk. First, we have

$$\begin{aligned}\tau = \mathop{\mathbf{E}}_{W^o,\mathbf{v}}\left[\max_{t\in[T]} v_{s(t)}\right] - \tau &\le \mathop{\mathbf{E}}_{W^o,\mathbf{v}}\left[\max_{t\in[T]}\left\{\left(v_{s(t)}-\tau\right)^+\right\}\right] \\ &\le \mathop{\mathbf{E}}_{W^*,\mathbf{v}}\left[\max_{t\in[T]}\left\{\left(v_{s(t)}-\tau\right)^+\right\}\right] \stackrel{\text{def}}{=} \mathtt{Sur}^*, \end{aligned} \tag{12}$$

where the first inequality holds by linearity of expectation and since $\max v_{s(t)}-\tau\le\max(v_{s(t)}-\tau)^+$; the second inequality holds as s^* maximizes expression in (9). Second, we have

$$\begin{aligned}\mathtt{Sur}^* &= \sum_{t,s} q_{s,t}\,\mathbf{Pr}\left[v_s\ge\tau\right]\mathop{\mathbf{E}}_{W_s^{T-t},\mathbf{v}}\left[\max_{\ell\in[T-t]}(v_{s(\ell)}-\tau)^+\,\middle|\, v_s\ge\tau\right] \\ &\le \sum_{t,s} q_{s,t}\cdot\mathbf{Pr}\left[v_s\ge\tau\right]\left(\mathop{\mathbf{E}}_{v_s}\left[(v_s-\tau)^+\,\middle|\, v_s\ge\tau\right]+\mathtt{Sur}^*\right), \end{aligned} \tag{13}$$

where in the first equality W_s^{T-t} denotes the $T-t$ random walk from s; the inequality holds, since $\max(a,b)\le a+b$ for non negative a,b, the value $\max(v_{s(\ell)}-\tau)^+$ of the $T-t$ walk from s is not more than $\max(v_{s(\ell)}-\tau)^+$ of the T step walk from s, and $\mathtt{Sur}^*$ is the maximum expectation of $\max(v_{s(\ell)}-\tau)^+$ achieved at the starting state s^*. Now, we combine all bounds together to get the desired result. First, we get the following from inequality (13) using equation (11)

$$\mathtt{Sur}^*\cdot(1-Q)\le\sum_{t,s} q_{s,t}\cdot\mathbf{Pr}\left[v_s\ge\tau\right]\mathop{\mathbf{E}}_{v_s}\left[(v_s-\tau)^+\,\middle|\, v_s\ge\tau\right].$$

Next, by inequality (12) we have $\mathtt{Sur}^*\cdot(1-Q)\ge\tau\cdot(1-Q)$. Finally, we apply the last two bounds to (10) and get

$$\mathcal{A}(s^*)\ge Q\cdot\tau+\mathtt{Sur}^*\cdot(1-Q)\ge Q\cdot\tau+\tau\cdot(1-Q)=\tau.$$

This concludes the proof, as $\tau=0.5\cdot\mathtt{prophet}$. □

Remark 2. *Note that for the Markov Chain* $M^{\mathtt{PI}}$ *and* $T=n$*, the starting position at* $s(1)=s_1$ *dominates any other choice of the starting position. Thus our Markov Chain with free starting point generalizes the classic PI.*

5 MULTI-UNIT SETTING

In this section, we study the generalization of multi-unit PI from the optimal stopping theory to our Markov chain setting. In the multi-unit PI, one can use up to $k\ge1$ copies of the resource instead of 1 unit in the classical PI. The multi-unit prophet *inequality* refers to the competitive ratio of $1-o(1)$ between the expected reward of the best online algorithm and the offline optimum (the prophet). Multi-unit PI can be viewed as a special case of online resource

allocation problem with a single resource of capacity $C = k$. The corresponding Markov Chain that describes standard multi-unit PI is exactly the same Markov chain as in the previous section. We would like to have an $1-o(1)$ approximation analogue to multi-unit PI for more general Markov Chains. Note that we cannot achieve an $1 - o(1)$ approximation (even $\Omega(1)$ approximation) without the power to choose the starting state of our random walk, as was discussed in the previous section. It turns out that this power is still not sufficient to achieve an $1 - o(1)$ approximation for large k (see the counter-example on the figure 3 below).

Counter-Example. The Markov chain M^c first goes through a path with k states of identical *deterministic value* $v = 1$. Then the random walk would either go to a path with $k/2$ states with identical *deterministic value* $v = 2$, or go to a path with $k/3$ states with identical *deterministic value* $v = 3$, each path is chosen with half probability.

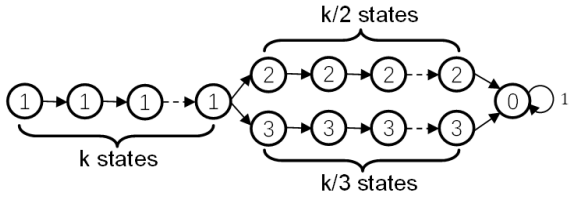

Figure 3: counter-example

The offline optimum sees which path the random walk followed after the first stage and, thus, can decide how many resources in the first stage it should consume. Hence, the expected value of offline optimum is $\frac{1}{2} \cdot \frac{3k}{2} + \frac{1}{2} \cdot \frac{5k}{3} = \frac{19k}{12}$. The best starting state for the online algorithm is the leftmost state (for example, if it picks a starting state on one of the paths in the second stage its reward would not be larger than $2 \cdot k/2 = k = 3 \cdot k/3$). Now, when the online algorithm starts from the leftmost state, it must decide how many resources to take in the first stage without knowing which path the random walk will follow in the second stage. The best expected value the online algorithm can achieve is $3k/2$ (Denote m as the number of resources an online algorithm reserves for the second stage. It is clear that m should be between $k/3$ and $k/2$ for the optimal algorithm. Thus the optimal online reward should be no more than $k - m + 0.5 \cdot m \cdot 2 + 0.5 \cdot k/3 \cdot 3 = 3k/2$). I.e., the competitive ratio of any online algorithm is no better than $\frac{18}{19}$, a constant smaller than 1 for any k.

When compared to the classic multi-unit PI where we have full information of all distributions beforehand, a random walk, like in the example above, may visit quite different sets of states and also quite a different number of times. On the other hand, if we want to reach an $1 - o(1)$ approximation, we could afford to make only a few different choices compared to the offline optimum. This is clearly impossible as the previous example has demonstrated.

Therefore, we need a stronger assumption on the Markov chain than the choice of the starting state. Specifically, we assume that the Markov chain is *recurrent* and our random walk is significantly *longer than the hitting time* $\mathtt{Ht}$. Recurrence of M ensures that every state will be visited eventually, and, as we show below, the assumption on the length of the random walk enables us to obtain good concentration bounds on the number of visits to every state. This gives us a power to make a good prediction about the set of visited distributions (note that this set is fixed in the classic multi-unit PI). This assumption often holds in practice, as the Markov Chain that could be reasonably inferred from data is rather small (most likely a constant number of states). For example, the Markov Chain may capture a daily cycle of the bidders' behavior, while the time horizon for the optimization could be a month.

The following theorem shows how we can get an $1 - o(1)$ approximation guarantee with such power. The main idea is that with an accurate enough prediction on the (multi) set of visited states, if we use the same online algorithm, as in the classic setting for the prediction, we get the desired approximation result.

The benchmark $\mathtt{kprophet}$ for the k-unit PI is as follows

$$\mathtt{kprophet} = \max_{s \in N} \mathop{\mathbf{E}}_{W \sim M(s)} \left[\mathop{\mathbf{E}}_{\mathbf{v} \sim \mathbf{F}} \left[\max_{|K|=k} \left\{ \sum_{i \in K} v_{s(i)} \right\} \right] \right], \tag{14}$$

Let ℓ_s be the expected number of visits to each state s. The following lemma establishes a concentration bound on the number of visits to each state (its full proof is deferred to Appendix).

Lemma 5. *For any $0 < \varepsilon < 0.5$, and any state $s \in N$, if $T = 16e \cdot \frac{\mathtt{Ht}}{\varepsilon^2} \cdot \left(\ln \left(\frac{\mathtt{Ht}}{\mathtt{Rt}(s)} \right) + 2 \right) \cdot \frac{1.1\Delta}{1-\varepsilon}$, for any constant $\Delta \geq 1$*

$$\mathbf{Pr} \left[\textit{number of visits to } s \geq (1-\varepsilon) \cdot \ell_s \right] \geq 1 - e^{-\Delta}$$

Proof Idea. The number of visits to any fixed state s is tightly connected with the sum of i.i.d. random variables X_i which are distributed as the random variable X describing return time to s. Its expectation is $\mathbf{E}[X] = \mathtt{Rt}(s) = \frac{1}{w_s}$, and it has exponentially decreasing tail probability related to the hitting time $\mathtt{Ht}$. Our main technical challenge is to extend Chernoff bound from bounded r.v. (like Bernoulli) to r.v. with an exponential tail bound. □

Theorem 6. *If $k \geq \frac{2\log(1/\varepsilon)}{\varepsilon^2}$ and T is at least the number stated in Lemma 5, there exists an online algorithm with expected reward of*

$$\left(1 - e^{-\Delta}\right) \cdot (1 - 2\varepsilon) \cdot \mathtt{kprophet}$$

Proof. We use the following convex program, known as the ex-ante relaxation in the mechanism design literature, as an upper bound on $\mathtt{kprophet}$. Let x_s be the expected number of requests served by any given algorithm in a state $s \in N$. Then, the algorithm's total expected reward from state s is upper bounded[4] by the following quantity $r_s(x_s)$:

$$r_s(x_s) \stackrel{\text{def}}{=} \ell_s \cdot \mathbf{E} \left[v_s \mid v_s \geq \theta_s(x_s) \right],$$

where the threshold $\theta_s(x_s)$ satisfies $\mathbf{Pr}[v_s \geq \theta_s(x_s)] = \frac{x_s}{\ell_s}$. In the ex-ante relaxation the function r_s is concave in x_s. The *ex-ante*

[4]The idea behind this relaxation is to let the offline algorithm satisfy the capacity constraint in expectation over the instance randomness, thus making its decisions independent across different states and/or distributions.

relaxation of the benchmark kprophet is as follows,

$$\max_{\mathbf{x}}: \sum_{s\in N} r_s(x_s) \qquad \geq \texttt{kprophet}$$
$$\text{subject to}: \sum_{s\in N} x_s \leq k; \qquad x_s \leq \ell_s, \quad \forall s \in N.$$

This program can be solved efficiently and we use $(x_s^*)_{s\in N}$ to denote its optimal solution. We have $\texttt{kprophet} \leq \sum_s r_s(x_s^*)$.

Next, we present our online algorithm:

- For each state s, upon the first $(1-\varepsilon)\ell_s$ visits to s, we serve the request if and only if $v_s \geq \theta_s(x_s^*)$ and there is still some resource left;
- We completely ignore s after the first $(1-\varepsilon)\ell_s$ visits.

We study the probability that the budget k is exhausted by our algorithm. At each visit to state s, we consume a unit of the resource with probability $\frac{x_{s^*}}{\ell_s}$. Let y_s be the summation of $(1-\varepsilon)\cdot\ell_s$ i.i.d. random $\{0,1\}$ variables, each is realized with probability $\frac{x_{s^*}}{\ell_s}$. Then, the random variable y_s stochastically dominates the random number of requests served by our algorithm. Consequently, the probability that our budget is exhausted is upper bounded by the probability that $\sum_s y_s > k$. Observe that $\mathbf{E}[\sum_s y_s] = (1-\varepsilon)\sum_s x_s^* = (1-\varepsilon)\cdot k$. By a standard Chernoff bound[5], we have

$$\mathbf{Pr}\left[\text{budget exhausted}\right] \leq \mathbf{Pr}\left[\sum_s y_s > k\right] \leq \exp(-k\varepsilon^2/2) \leq \varepsilon,$$

where the last inequality holds as $k \geq \frac{2\ln(1/\varepsilon)}{\varepsilon^2}$ by the assumption.

Finally, we calculate the expected gain of our algorithm from each state s. We visit each state s with probability at least $(1-e^{-\Delta})$ more than $(1-\varepsilon)\ell_s$ times by Lemma 5. Per each of the first $(1-\varepsilon)\ell_s$ visits, our expected gain is $\mathbf{E}[v_s \mid v_s \geq \theta_s(x_s^*)]$ as long as we still have the budget. That is, our total expected gain from s is at least

$$\begin{aligned}
&\left(1-e^{-\Delta}\right)\cdot \sum_{t\leq(1-\varepsilon)\ell_s} \Big(\mathbf{E}\left[v_s \mid v_s \geq \theta_s(x_s^*)\right] \\
&\qquad \cdot \mathbf{Pr}\left[\text{budget not exhausted at } t\text{-th visit to } s\right]\Big) \\
&\geq \left(1-e^{-\Delta}\right)\cdot(1-\varepsilon)\cdot(1-\varepsilon)\cdot\ell_s\cdot\mathbf{E}\left[v_s \mid v_s \geq \theta_s(x_s^*)\right] \\
&\geq \left(1-e^{-\Delta}\right)\cdot(1-2\varepsilon)\cdot r_s(x_s^*)
\end{aligned}$$

Summing over all states s of the Markov chain, we conclude that

$$\begin{aligned}
\mathbf{E}\left[\mathcal{A}\right] &\geq \left(1-e^{-\Delta}\right)\cdot(1-2\varepsilon)\cdot\sum_s r_s(x_s^*) \\
&\geq \left(1-e^{-\Delta}\right)\cdot(1-2\varepsilon)\cdot\texttt{kprophet}
\end{aligned}$$

□

The above theorem guarantees a $1-\varepsilon$ approximation to the offline optimum for $k = \widetilde{\Omega}\left(\frac{1}{\varepsilon^2}\right)$ and $T = \widetilde{\Omega}\left(\frac{\texttt{Ht}}{\varepsilon^2}\right)$ (as usual $\widetilde{\Omega}$ notation hides logarithmic factors). The quadratic dependency of k on $1/\varepsilon$ is unavoidable even in the classic multi-unit PI. The dependency of T on hitting time Ht is unavoidable even if we want to get a constant approximation to kprophet; the quadratic dependency on $1/\varepsilon$ is also unavoidable for the ex-ante relaxation approach, as we need to predict the number of visits to each state s with accuracy $1\pm\varepsilon$.

[5]I.e., that a sum X of i.i.d. random variables with expectation $\mu = (1-\varepsilon)k$ satisfies $\mathbf{Pr}[X \geq (1+\delta)\mu] \leq e^{-\delta^2\mu/(2+\delta)} = e^{-k\varepsilon^2/(2-\varepsilon)} < e^{-k\varepsilon^2/2}$ for $\delta = \frac{1}{1-\varepsilon} - 1$.

Remark 3. *Our results in this section extend to multi-resource setting and non-uniform costs in different states. Indeed, Lemma 5 still holds and the ex-ante relaxation can be generalized if we replace budget constraint by $\sum_s x_s\cdot c_i(s) \leq C_i$ for each resource type $i \in [m]$. As long as the small-bid assumption holds, i.e., $\max_{i,s} c_i(s)/C_i(s) \to 0$ (which corresponds to $k \to \infty$ in the above stylized model), our algorithm achieves an $1-o(1)$ approximation. The analysis is essentially the same as above. We focus on the single-resource and multi-unit setting to avoid cumbersome notations.*

ACKNOWLEDGMENTS

This work is supported by Ant Group through CCF-AFSG Research Fund, Science and Technology Innovation 2030 – "New Generation of Artificial Intelligence" Major Project No.(2018AAA0100903), Innovation Program of Shanghai Municipal Education Commission, Program for Innovative Research Team of Shanghai University of Finance and Economics (IRTSHUFE), and the Fundamental Research Funds for the Central Universities. Nick Gravin is supported by NSFC grant 62150610500.

REFERENCES

Gagan Aggarwal, Gagan Goel, Chinmay Karande, and Aranyak Mehta. 2011. Online Vertex-Weighted Bipartite Matching and Single-Bid Budgeted Allocations. In *Proceedings of the Twenty-Second Annual ACM-SIAM Symposium on Discrete Algorithms* (San Francisco, California) *(SODA '11)*. Society for Industrial and Applied Mathematics, USA, 1253–1264.

Shipra Agrawal and Nikhil R. Devanur. 2015. *Fast Algorithms for Online Stochastic Convex Programming*. SIAM, USA, 1405–1424. https://doi.org/10.1137/1.9781611973730.93 arXiv:https://epubs.siam.org/doi/pdf/10.1137/1.9781611973730.93

Shipra Agrawal, Zizhuo Wang, and Yinyu Ye. 2014. A dynamic near-optimal algorithm for online linear programming. *Operations Research* 62, 4 (2014), 876–890.

Saeed Alaei. 2014. Bayesian Combinatorial Auctions: Expanding Single Buyer Mechanisms to Many Buyers. *SIAM J. Comput.* 43, 2 (2014), 930–972.

Saeed Alaei, MohammadTaghi Hajiaghayi, and Vahid Liaghat. 2012. Online Prophet-Inequality Matching with Applications to Ad Allocation. In *Proceedings of the 13th ACM Conference on Electronic Commerce* (Valencia, Spain) *(EC '12)*. Association for Computing Machinery, New York, NY, USA, 18–35. https://doi.org/10.1145/2229012.2229018

Nikhil R. Devanur and Thomas P. Hayes. 2009. The Adwords Problem: Online Keyword Matching with Budgeted Bidders under Random Permutations. In *Proceedings of the 10th ACM Conference on Electronic Commerce* (Stanford, California, USA) *(EC '09)*. Association for Computing Machinery, New York, NY, USA, 71–78. https://doi.org/10.1145/1566374.1566384

Nikhil R. Devanur, Kamal Jain, Balasubramanian Sivan, and Christopher A. Wilkens. 2019. Near Optimal Online Algorithms and Fast Approximation Algorithms for Resource Allocation Problems. *J. ACM* 66, 1 (2019), 7:1–7:41.

Nikhil R. Devanur, Balasubramanian Sivan, and Yossi Azar. 2012. Asymptotically Optimal Algorithm for Stochastic Adwords. In *Proceedings of the 13th ACM Conference on Electronic Commerce* (Valencia, Spain) *(EC '12)*. Association for Computing Machinery, New York, NY, USA, 388–404. https://doi.org/10.1145/2229012.2229043

Anupam Gupta and Marco Molinaro. 2016. How the Experts Algorithm Can Help Solve LPs Online. *Math. Oper. Res.* 41, 4 (2016), 1404–1431.

Mohammad Taghi Hajiaghayi, Robert Kleinberg, and Tuomas Sandholm. 2007. Automated Online Mechanism Design and Prophet Inequalities. In *Proceedings of the 22nd National Conference on Artificial Intelligence - Volume 1* (Vancouver, British Columbia, Canada) *(AAAI'07)*. AAAI Press, USA, 58–65.

Bala Kalyanasundaram and Kirk Pruhs. 2000. An optimal deterministic algorithm for online b-matching. *Theor. Comput. Sci.* 233, 1-2 (2000), 319–325.

R. M. Karp, U. V. Vazirani, and V. V. Vazirani. 1990. An Optimal Algorithm for On-Line Bipartite Matching. In *Proceedings of the Twenty-Second Annual ACM Symposium on Theory of Computing* (Baltimore, Maryland, USA) *(STOC '90)*. Association for Computing Machinery, New York, NY, USA, 352–358. https://doi.org/10.1145/100216.100262

D. P. Kennedy. 1985. Optimal Stopping of Independent Random Variables and Maximizing Prophets. *The Annals of Probability* 13, 2 (1985), 566 – 571. https://doi.org/10.1214/aop/1176993009

Douglas P Kennedy. 1987. Prophet-type inequalities for multi-choice optimal stopping. *Stochastic Processes and their applications* 24, 1 (1987), 77–88.
Robert P Kertz. 1986. Comparison of optimal value and constrained maxima expectations for independent random variables. *Advances in applied probability* 18, 2 (1986), 311–340.
Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. 2018. Primal Beats Dual on Online Packing LPs in the Random-Order Model. *SIAM J. Comput.* 47, 5 (2018), 1939–1964.
Robert Kleinberg and S. Matthew Weinberg. 2019. Matroid prophet inequalities and applications to multi-dimensional mechanism design. *Games Econ. Behav.* 113 (2019), 97–115.
Ulrich Krengel and Louis Sucheston. 1977. Semiamarts and finite values. *Bull. Amer. Math. Soc.* 83, 4 (1977), 745–747.
Zachary J. Lee, Tongxin Li, and Steven H. Low. 2019. ACN-Data: Analysis and Applications of an Open EV Charging Dataset.
Xiaocheng Li and Yinyu Ye. 2022. Online Linear Programming: Dual Convergence, New Algorithms, and Regret Bounds. *Oper. Res.* 70, 5 (2022), 2948–2966. https://doi.org/10.1287/opre.2021.2164
Aranyak Mehta, Amin Saberi, Umesh V. Vazirani, and Vijay V. Vazirani. 2007. AdWords and generalized online matching. *J. ACM* 54, 5 (2007), 22.
Christos Papadimitriou, Tristan Pollner, Amin Saberi, and David Wajc. 2021. Online Stochastic Max-Weight Bipartite Matching: Beyond Prophet Inequalities. In *Proceedings of the 22nd ACM Conference on Economics and Computation* (Budapest, Hungary) *(EC '21)*. Association for Computing Machinery, New York, NY, USA, 763–764. https://doi.org/10.1145/3465456.3467613
Ester Samuel-Cahn. 1984. Comparison of Threshold Stop Rules and Maximum for Independent Nonnegative Random Variables. *The Annals of Probability* 12, 4 (1984), 1213 – 1216. https://doi.org/10.1214/aop/1176993150

A PROOF OF LEMMA 5

Let us first fix a state $s \in N$. Let ξ be the random variable equal to the number of visits of the random walk to state s. In the Lemma we need to show that ξ is well concentrated around its expected value. The behavior of ξ is tightly connected to another random variable X – the number of steps it takes random walk to return to state s (starting from s). Specifically, consider i.i.d. random variables $(X_i)_{i=1}^{\infty}$ with the same distribution as X. A random variable X_i represents the number of steps between i-th visit and $(i+1)$-th visit to s. We also define a random variable X_0 – the number of steps it takes to reach state s for the first time. Then

$$\xi = \max_i \left\{ i \;\middle|\; X_0 + X_1 + \ldots + X_i \le T \right\} + 1,$$

if $X_0 > T$, we let $i = -1$. In general, we would like to apply a Chernoff type bound to the sum $X_1 + X_2 + \ldots + X_j$ of i.i.d random variables, but, unfortunately, X_i is unbounded as the random walk may keep indefinitely avoiding state s. Luckily, we have the following exponential tail probability bound on each random variable X_i.

Claim 7. $\forall i \ge 0, j \in \mathbb{N} \qquad \mathbf{Pr}[X_i \ge j \cdot e \cdot \mathtt{Ht}] \le e^{-j}.$

Proof. Observe that for $j = 1$, we can apply Markov inequality to the random variable $\eta(s')$ – the number of steps to reach s from another state $s' \in N$. Indeed, $\mathbf{E}[\eta] \le \mathtt{Ht}$, by definition of the hitting time for any $s' \in N$.

For $j > 1$, the random walk does not reach s within the first $e \cdot \mathtt{Ht}$ steps with probability $\le 1/e$. If it fails, the walk gets to a state s'' and then it has another chance to reach s in the next $e \cdot \mathtt{Ht}$ steps; again it may only fail with probability $\le 1/e$. We repeat the argument j times, which gives the desired upper bound on the probability of not reaching s in $j \cdot e \cdot \mathtt{Ht}$ steps. □

For convenience of notations, we define $\Gamma \stackrel{\text{def}}{=} 2e \cdot \mathtt{Ht}$ and extend the tail bound from Claim 7 for $j \in \mathbb{N}$ to real values $\alpha \in \mathbb{R}_{\ge 1}$.

$$\mathbf{Pr}\left[X_i \ge \Gamma \cdot \alpha\right] \le e^{-\lfloor 2\alpha \rfloor} \le e^{-\alpha}, \qquad \forall i \in \mathbb{N}_0,\ \alpha \in \mathbb{R}_{\ge 1}. \tag{15}$$

Now, by applying Chernoff bound method for $t \in \mathbb{R}_{>0}$ we get

$$\mathbf{Pr}\left[\sum_{i=1}^{j} X_i \ge a\right] = \mathbf{Pr}\left[e^{\sum_{i\le j} t\cdot X_i} \ge e^{t\cdot a}\right] \le e^{-t\cdot a} \cdot \left(\mathbf{E}\left[e^{t\cdot X}\right]\right)^j,$$

where to get the inequality, we use Markov inequality and the fact that $\mathbf{E}[e^{\sum_{i\le j} t\cdot X_i}] = \mathbf{E}[e^{t\cdot X}]^j$. Next, we need to get an upper bound on $\mathbf{E}[e^{t\cdot X}]$. To do this we use the constraint (15) and the fact[6] that $\mathbf{E}[X] = \mathtt{Rt}(s)$.

Claim 8. *The following random variable X^* maximizes $\mathbf{E}[e^{t\cdot X}]$ under the constraint* (15) *and $\mathbf{E}[X] = \mathtt{Rt}(s) = \frac{1}{w_s}$.*

$$X^* = \begin{cases} \text{PDF} \quad f(\Gamma\cdot x) = \dfrac{e^{-x}}{\Gamma}, & \text{for } x \in [c_o, +\infty) \\ \mathbf{Pr}\left[X^* = 0\right] = 1 - e^{-c_o}, & \text{point mass at } 0. \end{cases} \tag{16}$$

Where c_o satisfies $\Gamma\cdot(c_o+1)e^{-c_o} = \frac{1}{w_s}$.

Proof. First, note that the constraint (15) is tight for random variable X^* for all $x \ge c_o$ and that $\mathbf{E}[X^*] = \mathtt{Rt}(s)$. We will show that X^* maximizes $\mathbf{E}[e^{t\cdot X}]$ for the relaxed optimization where constraint (15) holds only for $x \ge c_o$.

Observe that t must be strictly smaller than $\frac{1}{\Gamma}$ for $\mathbf{E}[e^{X\cdot t}]$ to converge. For each $t < \frac{1}{\Gamma}$, we can discretize support of X into integer multiples of a small $\varepsilon > 0$ up to a large constant B (such that $\mathbf{E}[e^{T\cdot X}\cdot \mathbb{I}[X^* > B]]$ is negligibly small for the given tail bound (15)), so that the support is a finite set. Then we optimize $\mathbf{E}[e^{t\cdot X}]$ for X supported on this finite set with the constraints (15) and $\mathbf{E}[X] = \mathtt{Rt}(s)$. This is optimization of a continuous function on a compact set, thus it achieves maximum at a finitely supported random variable X^o. Note that the expectation $\mathbf{E}[e^{t\cdot X^o}]$ is close to the expectation $\mathbf{E}[e^{t\cdot X}]$ of the continuous random variable X.

Now, we observe that if PDF of X^0 at $z > 0$ is strictly smaller than the respective PDF (its discretized version) of X^*, $\mathbf{Pr}[v = z] < e^{-z/\Gamma} - e^{-(z+\varepsilon)/\Gamma}$, and X^o has a point $0 < y < z$ with positive PDF, then we can feasibly modify X^o and strictly increase $\mathbf{E}[e^{t\cdot X^o}]$. Indeed, without loss of generality let us consider such y that is the closest to z. We can move a small $\delta > 0$ mass from y ($\mathbf{Pr}[X^o = y] \leftarrow \mathbf{Pr}[X^o = y] - \delta$) to z ($\mathbf{Pr}[X^o = z] \leftarrow \mathbf{Pr}[X^o = y] + \frac{y}{z}\cdot\delta$) and 0 ($\mathbf{Pr}[X^o = 0] \leftarrow \mathbf{Pr}[X^o = 0] + \frac{z-y}{z}\cdot\delta$) so that $\mathbf{E}[X^o] = \mathtt{Rt}(s)$ and (15) is still satisfied ((15) does not change for $\alpha > z/\Gamma$, (15) was not tight for $\Gamma\cdot\alpha \in [y, z]$, and (15) just gets extra slack for $\Gamma\cdot\alpha < y$). As $e^{t\cdot x}$ is a strictly convex function, the latter modification strictly increases the objective $\mathbf{E}[e^{t\cdot X^o}]$ – a contradiction to maximality of X^o. Hence, the PDF of X^o is equal to the PDF of X^* (its discretized version) on the interval $(a_0, B]$, it is smaller than PDF of X^* at a_0, and it has PDF 0 on $(0, a_0)$. Given the constraint $\mathbf{E}[X^o] = \mathbf{E}[X^*] = \mathtt{Rt}(s)$ it means that $X^o = X^*$. □

$$\text{Therefore,} \quad \mathbf{E}\left[e^{tX}\right] \le \mathbf{E}\left[e^{tX^*}\right] = 1 - e^{-c_o} + \frac{1}{1-\Gamma t} e^{(\Gamma t - 1)c_o}.$$

To simplify notation let $g(t) \stackrel{\text{def}}{=} \ln(1 - e^{-c_o} + \frac{1}{1-\Gamma t} e^{(\Gamma t-1)c_o})$. We first set $a = j \cdot \mathtt{Rt}(s) \cdot (1+\varepsilon)$ in the Chernoff bound. Then,

$$\mathbf{Pr}\left[\sum_{i=1}^{j} X_i \ge a\right] \le e^{-ta}\cdot e^{j\cdot g(t)} = exp\left[j\cdot\left(g(t) - t\cdot\mathtt{Rt}(s)\cdot(1+\varepsilon)\right)\right].$$

[6] A well known fact about return time and stationary distribution of a Markov chain.

We choose $t \stackrel{\text{def}}{=} \frac{\varepsilon}{2c_o\Gamma}$ and estimate $g(t) - t \cdot \mathtt{Rt}(s) \cdot (1+\varepsilon)$ as follows.

$$g(t) - t \cdot \mathtt{Rt}(s) \cdot (1+\varepsilon) = \ln\left(1 - e^{-c_o} + \frac{e^{(\Gamma t-1)c_o}}{1-\Gamma t}\right) - \frac{t(1+\varepsilon)}{w_s} \leq$$

$$e^{-c_o} \cdot \left(\frac{e^{t\cdot\Gamma\cdot c_o}}{1-\Gamma t} - 1\right) - \frac{t(1+\varepsilon)}{w_s} = \frac{\mathtt{Rt}(s)}{(1+c_o)\Gamma}\left(\frac{e^{\varepsilon/2}}{1-\frac{\varepsilon}{2c_o}} - 1\right) - \frac{\varepsilon(1+\varepsilon)}{2\Gamma c_o w_s}$$

$$\leq \frac{1}{(1+c_o)\Gamma w_s}\left(\frac{1+\frac{\varepsilon}{2}+\frac{\varepsilon^2}{4}}{1-\frac{\varepsilon}{2c_o}} - 1\right) - \frac{\varepsilon(1+\varepsilon)}{2\Gamma c_o w_s} =$$

$$\frac{\varepsilon^2 \cdot \mathtt{Rt}(s)}{(2c_o-\varepsilon)\Gamma}\left(-\frac{1}{2} + \frac{1}{2c_o(1+c_o)} + \frac{\varepsilon}{2c_o}\right) \leq \frac{-\varepsilon^2 \cdot \mathtt{Rt}(s)}{8c_o\Gamma},$$

where the first inequality holds as $\ln(1+x) \leq x$ for any $x \geq -1$; the second equality holds as $t = \frac{\varepsilon}{2c_o\Gamma}$ and $e^{-c_o} = \frac{\mathtt{Rt}(s)}{\Gamma(c_o+1)}$; the second inequality holds as $e^x \leq 1+x+x^2$ for any $x < 1.79$; the last inequality holds as $c_o \geq 2$ from equation $2e(c_o+1)e^{-c_o} = \frac{\mathtt{Rt}(s)}{\mathtt{Ht}} \leq 1$ and $\varepsilon < 0.5$. When we plug this estimate in the Chernoff bound we get

$$\mathbf{Pr}\left[\sum_{i=1}^{j} X_i \geq j \cdot \mathtt{Rt}(s)(1+\varepsilon)\right] \leq \exp\left[-j \cdot \frac{\varepsilon^2 \cdot \mathtt{Rt}(s)}{16e \cdot c_o \cdot \mathtt{Ht}}\right].$$

From $2e \cdot \mathtt{Ht} \cdot (c_o+1)e^{-c_o} = \mathtt{Rt}(s)$, we have $c_o \ln(c_o+1) = \ln\left(\frac{\mathtt{Ht}}{\mathtt{Rt}(s)}\right) + 1 + \ln(2)$. Thus, $c_o \leq \ln\left(\frac{\mathtt{Ht}}{\mathtt{Rt}(s)}\right) + 2$ as $c_o > 2$ and $\ln(c_o+1) > 1$.

Finally, we are ready to get the estimate on ξ – the total number of visits to state s. Recall that $T = 16e \cdot \frac{\mathtt{Ht}}{\varepsilon^2} \cdot \left(\ln\left(\frac{\mathtt{Ht}}{\mathtt{Rt}(s)}\right) + 2\right) \cdot \frac{1.1\cdot\Delta}{1-\varepsilon}$. We set $j = \frac{T(1-\varepsilon)}{\mathtt{Rt}(s)}$, then the Chernoff bound gives us

$$\mathbf{Pr}\left[\sum_{i=1}^{j} X_i \geq T(1-\varepsilon^2)\right] \leq \exp\left[\frac{-\varepsilon^2 \cdot T(1-\varepsilon)}{16e \cdot \left(\ln\left(\frac{\mathtt{Ht}}{\mathtt{Rt}(s)}\right) + 2\right) \cdot \mathtt{Ht}}\right] \leq e^{-1.1\Delta}$$

On the other hand, $\mathbf{Pr}[X_0 \geq \varepsilon^2 T] \leq e^{-32e\cdot\Delta}$ by Claim 7. Thus the probability that either of $X_0 \leq \varepsilon^2 T$ and $\sum_{i=1}^{j} X_i \geq T(1-\varepsilon^2)$ is not more than $1 - (1-e^{-32e\cdot\Delta})(1-e^{-1.1\Delta}) < e^{-\Delta}$. I.e., $\xi \geq j$ with probability at least $1 - e^{-\Delta}$.

B PRELIMINARY EXPERIMENTS

We conducted preliminary experiments on real data to test our Markovian modeling for the online resource allocation problem. Specifically, we used ACN-Data [Lee et al. 2019] of 50000 EV(electric vehicle) charging sessions and modeled it as a resource allocation problem as follows.

The charging sessions are sorted in time and thus can be naturally treated as online requests. For each session we look at the following features: connection time, required electricity, and the user ID. The value of serving a request is defined as the number of charging sessions for the respective user ID in the previous 3 months prior to the request. Every request consumes a certain amount of electric power based on the connection time and required electricity of the respective charging session. We artificially introduce a limited capacity on the electric power of the station, so that it could not serve all requests. This leads to the following resource allocation problem: each time an EV comes to the station, we need to decide whether to serve its charging request based on its value with the goal of maximizing the total value of the served requests.

We choose the data of each month from 4/2019 to 12/2019 be the online arrival request stream. Every time we use the data from the previous three months as our training data to learn the Markov transition matrix, then we solve online resource allocation problem on the next month data. We predict the number of requests in the next month with an average number of requests in the previous three months. We conduct experiments with different power capacity limitations of $1000, 2000, \ldots, 7000$ kwh. We use dynamic programming to solve the MDP as described in section 3, and use the corresponding threshold vectors to decide whether to serve the online requests.

To construct a Markov chain, we apply optimal k-means clustering algorithm in one dimension based on the values and divide the requests into $5, 10, 20$ states. Then we count the number of transitions from one state to another for each ordered 2-state pair and derive the Markov Chain transition matrix by the frequency of transitions.

We compare our algorithm to the OLA algorithm [Agrawal et al. 2014], a primal-dual based algorithm for the unknown i.i.d setting. The algorithm computes optimal dual price on training data and then makes online decisions according to the dual price.

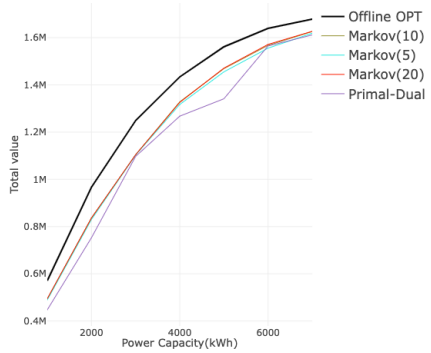

Figure 4: total value of the served requests

Figure 4 summarizes our experiments, where Markov(5), Markov(10), Markov(20) denote the results by our Markov chain algorithm with $5, 10, 20$ states respectively, Offline OPT denotes the offline optimum, Primal-Dual denotes the result by the OLA algorithm [Agrawal et al. 2014]. On each capacity level, Markov chain algorithms performs consistently better over the OLA algorithm and is not particularly sensitive on the number of states on this data set. The above observation together with the fact that the OLA algorithm is near optimal for the unknown i.i.d. setting, our preliminary experiment suggests that the i.i.d. assumption might be too optimistic for real world data and our Markov Chain model might be a good alternative modeling choice.