REAL TIME OPTIMIZATION IN NETWORKS: PRACTICAL ALGORITHMS WITH PROVABLE GUARANTEES

A Dissertation

Presented to the Faculty of the Graduate School

of Cornell University

in Partial Fulfillment of the Requirements for the Degree of

Doctor of Philosophy

by

Alberto Abel Vera Azócar

May 2020

REAL TIME OPTIMIZATION IN NETWORKS: PRACTICAL ALGORITHMS WITH PROVABLE GUARANTEES

Alberto Abel Vera Azócar, Ph.D.

Cornell University 2020

We consider several optimization problems where the main challenge is to answer to a stream of requests. We provide algorithms that are efficient, practical, enjoy performance guarantees, and work well in practice.

The first problem we study is computing *Constrained Shortest Paths* at scale. Even though the underlying optimization problem is NP-Hard, we can give parametric guarantees for the space requirement and running time of our algorithm. We use precomputation techniques, generalize the notion of Highway Dimension, and prove that, under mild conditions, we can scale Constrained Shortest Path whenever we can scale Shortest Path.

We study also a large family of *Resource Allocation and Pricing* problems. We give a unified treatment and provide an algorithm that achieves *constant regret, i.e., constant additive loss,* in several problems. As special cases we can cover, we mention Online Packing (Network Revenue Management), Online Matching, Online Probing, and Pricing of Multiple Items. Our results hold even under several generalizations, such as restocking of resources and queues of customers with abandonment.

Additionally, we present several robustness results in cases where the parameters of the problem are misspecified, similar to Bandits problems, or when there are two conflicting objectives, namely reward maximization and cost minimization. Furthermore, we also study the case when the arrival distribution can be accessed only via historical data and show that a single trace is enough to maintain constant regret guarantees.

BIOGRAPHICAL SKETCH

Alberto was born on 1989 in Santiago, Chile. He attended Universidad de Chile and completed a B.S. in Industrial Engineering in 2012, a M.S. in Industrial Engineering in 2015, and a M.S. in Operations Management in 2015 under the supervision of José Correa. Two months after graduation he began his Ph.D. at Cornell.

This document is dedicated to whomever may profit from its content.

ACKNOWLEDGEMENTS

This would not have been possible without Itai Gurvich and Samitha Samaranayake, who not only instructed me, but showed me friendship and understanding.

Thanks to my parents and my sister who accompanied and supported me through this tortuous road. Thanks to my eternal friend Alan, whose love I feel from 8000km away.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

We want answers and we want them now. New applications require to solve problems of ever increasing size and complexity, but users are forever impatient and, when they make a request, it has to be answered promptly. Consider, for example, the classical task of computing the shortest path in a graph. This is the fundamental problem that applications like Google Maps, Bing Maps, and Apple Maps have to solve, million of times every day, on graphs of massive size. From a classical perspective, this problem was solved optimally by Dijkstra in the last century; there is an optimal algorithm that runs in $O(n \log n)$ time, where n is the number of nodes in the graph. The hurdle is, $O(n \log n)$ takes too long for modern applications.

Let us consider another example. When booking flight tickets in an airline's website, the user wants to see the price instantaneously. On the other hand, from the airline's perspective, the problem of pricing tickets with thousand of flights, each with a large number of seats, over a long horizon, is a high dimensional a stochastic optimization problem. Again, from a classical point of view, there is an optimal solution, namely using Dynamic Programming. The conundrum is, the optimal solution is completely impractical and cannot be implemented even for small size instances.

In this thesis, we deal exactly with the problem of *real time optimization* in various settings. Our goal is to provide algorithms that are optimal or near optimal, but that also run in real time. In particular, computational efficiency is not enough. Consider again the case of Dijkstra's algorithm, that runs in polynomial time, but it is still too slow for modern applications. Using pre-processing and offline relaxations, we can solve and give parameterized performance guarantees for a variety of problems described next.

In Chapter 2, we study the computation of Constrained Shortest Path at scale. In contrast to Shortest Path, this problem is NP-hard, hence even more challenging to solve in real time. We leverage recent advancements in the computation of Shortest Paths. In particular, the notion of Highway Dimension, which is a property of the graph, has been used to parameterize the complexity of Shortest Path computation. We show that, under mild conditions on the cost structure, Constrained Shortest Path computation is scalable whenever Shortest Path is. Our work generalizes the notion of Highway Dimension to arbitrary path systems and average-case metrics. We test our algorithm using real world data and show that it is indeed practical and its computation time outperforms the traditional approach by four orders of magnitude.

We then turn a large family of problems called Dynamic Resource Allocation. In these problems, a controller allocates a finite set of resources to a stream of requests that arrive online. A request generates certain reward depending on the allocation made by the controller. The controller must decide on the allocation as soon as the request is presented, i.e., without knowledge of future requests, in an online fashion. The goal is to maximize the reward collected. This family of problems has an additional challenge; not only the decisions must be made in real time, but they must consider the uncertainty of future requests.

We focus on the stochastic setting, wherein requests are drawn from an underlying stochastic process. In all of our results, we present a real time algorithm and evaluate it in terms of its *regret*, which is the additive loss with respect to the optimal algorithm. In other words, the regret quantifies the amount of reward left on the table, i.e., reward that could have been collected if we used the optimal algorithm. Note that, despite the fact that the optimal algorithm is intractable, the regret is an informative measure in that, if it is small, it means that our algorithm is near optimal.

In Chapter 3, we present a novel technique for the analysis of online decision problems.

Our technique, called Compensated Coupling, facilitates the study of online problems and gives a regret bound in a completely generic setting. When we apply it to the special cases of Online Packing and Online Matching (both belong to the family of Dynamic Resource Allocation), we obtain the first constant regret bound for the multi dimensional setting. In other words, we provide an algorithm with a regret guarantee that is independent of the length of the horizon and the initial capacity of resources. We test our algorithm in an extensive numerical study and show that it outperforms other known algorithms.

In Chapter 4, we study these resource allocation problems from a different and novel angle. We use a geometric proof method and focus on the evolution of the remaining inventory, and hence of the LP that drives the algorithm, as a stochastic process. In particular, we can generalize our results to settings with resource restock and where requests have patience levels depending on their types, hence we may maintain a queue for certain requests. Our method has explanatory power and uncovers the robustness limits of our algorithm in regards to: (i) misspecification of the model parameters (rewards and probability distribution), which readily give bounds for the bandits version, where parameters must be learned; (ii) generalizations to restock of inventory and queues of requests; and (iii) the tuning of the algorithm's aggressiveness as captured by its "scoring" of requests. The latter flexibility allows us to achieve, simultaneously, near optimal reward maximization and near optimal holding cost minimization.

In Chapter 5, we develop a framework with a general purpose algorithm, called RABBI, to evaluate and design heuristics to include cases where the offline benchmark is not achievable. Our framework is based on the notion of Bellman Inequalities, which decompose the loss of an algorithm in two distinct sources of error: (1) our limited computational power and (2) estimation/prediction of the random trajectories The balancing of these sources of error guides the choice of benchmarks and the design of policies that are both tractable and have

strong performance guarantees. Using this framework we can obtain constant regret for budget-constrained probing, dynamic pricing, and online contextual bandits with knapsacks. We also show that the benchmark derived from our framework is necessary, in that there is no constant regret algorithm for these problems against traditional benchmarks.

In Chapter 6, we consider the case where the arrival distribution of requests is unknown, but we have historical data available. We extend RABBI to use only traces (sample paths) of the stochastic process. We prove that, for resource allocation problems, RABBI has near-optimal performance with the minimum possible sample-complexity; in particular, it obtains *constant regret with as few as one trace*. The algorithm is agnostic of the generative model of arrivals; however, the results hold even under time-varying and correlated arrival processes. Finally, even in settings beyond our theoretical guarantees, our framework generates data-friendly algorithms that match and beat the performance of specialized state-of-the-art algorithms in simulations.

CHAPTER 2

CONSTRAINED SHORTEST PATH

2.1 Introduction

Motivated by the requirements of modern transportation systems, we consider the fast computation of constrained shortest paths (CSP) in large-scale graphs. Though the unconstrained shortest-path (SP) problem has a long history, it has been revolutionized by recent algorithmic advancements that help enable large-scale mapping applications (cf. [20, 49] for surveys). In particular, the use of preprocessing techniques and network augmentation has led to dramatic improvements in the scalability of SP computation for road networks. These techniques, however, do not extend to the CSP, and hence can not fully leverage the rich travel-time distribution data available today.

The SP and CSP problems can be summarized as follows: We are given a graph G, where each edge has an associated *length* and *cost*. The SP problem requires finding an (s, t)-path of minimum length for any given nodes s and t. The CSP problem inputs an additional budget b, and requires finding a minimum length (s, t)-path *with total cost at most* b. The two problems, though similar, have very different runtime complexity: SP admits polynomial-time algorithms (in particular, the famous Dijkstra's algorithm), while CSP is known to be NP-Hard [63]. That said, a standard dynamic program computes CSPs in pseudo-polynomial time for discrete costs [44], and gives a natural scaling-based FPTAS for continuous costs (as in the knapsack problem).

Though there is a rich literature on CSP [63], existing approaches do not scale to support modern applications. To this end, we study *preprocessing and network augmentation for speeding up CSP computation*. Our work contributes to a growing field of algorithms for

large-scale problems (non-convex methods, sketching techniques, etc.), with relatively poor worst-case performance, but which are provably efficient for practically relevant settings.

Applications of large-scale CSP computation: Our primary motivation for scaling CSP comes from the requirements of modern transportation platforms (Lyft, Uber, etc.) for accurate routing and travel-time estimates. Modern SP engines like Google Maps do not make full use of available traffic information. In particular, they do not incorporate uncertainties in travel times, leading to inaccurate estimates in settings with high traffic variability. This can be addressed by computing shortest paths based on *probabilistic* travel-time estimates: if ℓ is the (possibly random) edge length function, given vertices s, t, tolerance δ and failure probability p, we want to find an (s,t)-path P minimizing $\mathbb{E}[\ell(P)]$, subject to $\mathbb{P}(\ell(P) > \mathbb{E}[\ell(P)] + \delta) \leq p$. Computing this exactly for general distributions is expensive due to the need for computing convolutions of distributions – indeed, there is no known polynomial time algorithm for doing this even with black-box access to convolution solvers [90]. However, conditioning on public state variables (e.g. weather and traffic in key neighborhoods), we can approximate the distributions with uncorrelated travel-times across different road segments [107]. Thus, Chebyshev's inequality gives us that $\mathbb{P}(\sum_e \ell_e > \mathbb{E}[\sum_e \ell_e] + \delta) \leq \sum_e \mathrm{Var}(\ell_e)/\delta^2$. Using this, we can reformulate the stochastic path optimization problem as $\min_{P \in \mathcal{P}_{s,t}} \sum_{i \in P} \mu_i$ s.t. $\sum_{i \in P} \sigma_i^2 \leq \delta^2 p$, which is now a CSP problem (see also [90] for a similar approach for Gaussian travel times). Note that, though we relax the condition $\mathbb{P}(\ell(P) > \mathbb{E}[\ell(P)] + \delta) \leq p$, our solution always respects this constraint – this is often critical for practical applications, e.g., for Lyft/Uber, accuracy of ETA estimates is more important than choosing the optimal route.

Another problem that can be modelled as a CSP is that of finding *reliable shortest paths*. Consider the case where each edge has a probability q_e of triggering a bad event, with resulting penalty p (for example, slowdowns due to accidents). In this case, we want

to minimize the travel time as well as the expected penalty. Assuming independence, we have the following natural problem: $\min_{P \in \mathcal{P}_{s,t}} \ell(P) + p\big(1 - \prod_{e \in P}(1 - q_e)\big)$. This model is considered in [48] for routing with fare evasion, where q_e is the probability of encountering an inspector, and p the penalty; the authors suggest using a CSP formulation, wherein the non-linear objective is replaced by a linear constraint by taking logarithms.

2.1.1 Our Contributions

We consider the problem of developing oracles that support fast CSP queries in large networks. Specifically, given integer edge-costs, edge-lengths, and a budget upper bound, we use preprocessing to create a data-structure supporting arbitrary source-destination-budget queries; moreover, we obtain formal guarantees on the performance: preprocessing, storage, and query time.

In the case of SP algorithms, the seminal work of Abraham et al. [2, 4] demonstrated that the preprocessing, storage, and query times of several widely-used heuristics could be parametrized using a graph structural metric called the *Highway Dimension* (HD). Our work extends these notions for the CSP; our contributions are summarized as follows:

Theoretical contributions: We define the *constrained highway dimension* (CHD) for the set of *efficient paths* (i.e., minimal solutions to the CSP; cf. Definition 2.2.1). We show how the CHD can be used to parametrize the performance of CSP algorithms. One hurdle, however, is that the CHD can be much bigger than the HD in general. Our main theoretical contribution is in showing that the *HD and CHD can be related under an additional partial witness condition* (Definition 2.3.12); therefore, in settings where SP computation is scalable, we can also solve the harder problem of CSP. Our next contribution is to show that, under *average performance metrics*, we can obtain even weaker conditions to relate the *average*

CHD and HD, thus certifying the performance of our algorithms. The conditions can be interpreted in terms of having few physical overpasses in road networks.

Practical contributions: We use our theoretical results to develop new practical data-structures for CSP queries, based on *hub labels* [45]. We evaluate our algorithm on datasets with detailed travel-time information for San Francisco and Luxembourg. In experiments, our algorithms exhibit query-times 10^4 faster than existing (non-preprocessing) techniques, have small storage requirements, and good preprocessing times even on a single machine.

Paper outline: In Section 2.2, we introduce the SP and CSP problems, and extend the notion of the HD (as defined in [2]) to directed graphs and general path systems; this allows us to define an analogous notion of a *constrained highway dimension* (CHD) for constrained shortest paths in Section 2.3. We then show that the two can be related under an additional *partial witness condition* (Section 2.3.3). In Section 2.3.4, we study average-case performance, and show how a small average CHD can be related to physical overpasses in road networks. Finally, in Section 2.4, we present our practical hub-label construction and our experiments on SF and Luxembourg data.

2.2 Setting and Overview of Results

We consider a directed graph $G = (V, E)$, where each edge $e \in E$ has an associated *length* $\ell(e) \in \mathbb{N}_+$, and *cost* $c(e) \in \mathbb{N}_+ \cup \{0\}$. The triplet (G, ℓ, c) is called a *network*. For any path P, we define its length $\ell(P)$ and cost $c(P)$ as the sum of edge lengths and edge costs in P. Our goal is to develop a data structure to answer *Constrained Shortest-Path* (CSP) queries efficiently as described next.

Constrained Shortest-Path Queries

Input:	Graph $G = (V, E)$, costs c, lengths ℓ, and maximum budget B.
Preprocessing Task:	Create a data structure $\mathcal{S}$.
Real Time Task:	For any source-terminal pair $s, t \in V$ and budget $b \leq B$, use $\mathcal{S}$ to return a path solving $\min\{\ell(P) : c(P) \leq b, P \text{ is an } (s,t)\text{-path}\}$. The triplet (s, t, b) defines a *query* and it is arbitrarily specified by the user.
Performance Metrics:	Size of $\mathcal{S}$, preprocessing time (to compute $\mathcal{S}$), and query time (to return a path given s, t, b).

Note that the two computation times are in different scales. Indeed, it is acceptable to have a preprocessing time of a few minutes, but the query time should be in milliseconds. Observe also that the performance metrics are in conflict. To illustrate this, define $n = |V|$ as the number of nodes and consider two extreme cases. (i) for each (s, t, b) we store in $\mathcal{S}$ the solution to the associated query, which has $O(1)$ query time, but space $|\mathcal{S}| = \Omega(Bn^2)$. (ii) use no data structure, which consumes no space, but requires $\Omega(bn \log n)$ query time using Dijkstra (see Proposition 2.3.3).

We stress that either storage $\Omega(n^2)$ or query time $\Omega(n)$ is unacceptable for modern applications. The goal is to find $\mathcal{S}$ in polynomial time with small storage and fast query time.

Our Results. We identify a parameter h_c, called the Constrained Highway Dimension (CHD), which generalizes the concept introduced in [2], and obtain the results below. In each case, the data structure can be computed in polynomial time. D represents the diameter

of the graph.

1. We give a data structure of size $\tilde{O}(nB \cdot Bh_c \log D)$ and query time $\tilde{O}(bh_c \log D)$ for any (s, t, b), see Theorems 2.3.6 and 2.3.9. We relate h_c to the HD in Theorem 2.3.14.
2. We introduce the notion of *Average CHD*, which is strictly weaker than CHD, and show that we can obtain a data structure with the same guarantees, but the query time is in average over (s, t), see Theorem 2.3.18.
3. Building on the average CHD, we give a condition, interpreted as having few overpasses, such that the query time is $\tilde{O}(bh\alpha \log D)$, average over (s, t), with space requirement $\tilde{O}(nB \cdot Bh\alpha \log D)$, where h is the HD and α the doubling constant, see Theorem 2.3.22. We observe that, for road networks, it is conjectured that $h = \text{polylog}(n)$ and $\alpha = O(1)$, hence our result explains the good empirical performance of our algorithm, see Section 2.4.

Final Setup Details. For any source-terminal pair $s, t \in V$, we denote by $\mathcal{P}_{s,t}$ the set of all simple (s, t)-paths (without loops or cycles). Throughout this work, we only consider simple paths, which we refer to as paths for brevity. We denote the shortest (s, t)-path (if it exists) as $P(s, t)$, and denote the set of all shortest paths in G as $\mathcal{P}^*$.

For each node v, we denote its degree $\Delta(v)$ as the sum of the in-degree and out-degree, and define the *maximum degree* $\Delta := \max_v \Delta(v)$. For $s, t \in V$, the distance from s to t, denoted $\text{dist}(s, t)$, is the smallest length among all paths $P \in \mathcal{P}_{s,t}$. We define $\text{dist}(s, t|b)$ to be the length of the shortest path with cost at most b, i.e., the length of the path returned by the query (s, t, b). If there is no feasible solution, we define $\text{dist}(s, t|b) = \infty$.

For a node v and a path P, we abuse notation to denote $\text{dist}(v, P)$ as the minimum distance from v to any node $w \in P$; the distance $\text{dist}(P, v)$ from P to v is defined analogously.

Note that $\text{dist}(P, v)$ and $\text{dist}(v, P)$ need not be the same as the graph is directed. We define $D \coloneqq \max_{P \in \mathcal{P}^*} \ell(P)$ to be the diameter of G.

For $r > 0$ and $v \in V$, we define the *forward and reverse balls of radius* r by $B_r^+(v) \coloneqq \{u \in V : \text{dist}(v, u) \leq r\}$ and $B_r^-(v) \coloneqq \{u \in V : \text{dist}(u, v) \leq r\}$, and also define $B_r(v) \coloneqq B_r^+(v) \cup B_r^-(v)$. Finally, a graph G is said to have a *doubling constant* α if, for any node v and any $r > 0$, the ball $B_{2r}(v)$ can be covered by at most α balls of radius r.

2.2.1 Related work

CSP problems have an extensive literature, surveyed in [63]. More recently, there has been significant interest in stochastic SP problems, including the related stochastic on-time arrival (SOTA) problem [59]; recent works have proposed both optimal and approximate policies [98, 74]. Existing approaches for these problems, however, are limited in their use of preprocessing and augmentation techniques, and consequently do not support the latencies required for mapping applications.

As we mentioned before, our work is inspired by the recent developments in shortest path algorithms [2, 3, 4, 49, 67, 81]; refer [20] for an excellent survey of these developments. The pre-processing technique we use for speeding up CSP computations is hub labels (HL), first introduced for SP computations in [45]. More recently, HL was proved to have the best query-time bounds for SP computation in low HD graphs [2, 4] (this was experimentally confirmed in [3], [20, Figure 7]). The notion of HD has been widely applied and therefore there are several variants (definitions) of this concept, we refer to Appendix A.1 and also [61, Section 9] for a more detailed discussion. Finally, the HD-based bound for hub labels was shown to be tight in [14, 106], and it was also shown that finding optimal hub labels is NP hard.

A related class of problems to CSP is that of SP under label constraints [19], where the aim is to find shortest paths that avoid certain labels (e.g. toll roads, ferries, etc.). In this setting, there is work on using preprocessing to improve query-times [95]. These problems are essentially concatenations of parallel SP problems, involving only local constraints. In contrast, the CSP involves global constraints on paths. Our results do in fact shed light on why preprocessing works well for label-constrained SP queries.

Finally, we note that the notion of Highway Dimension has been successfully applied to parameterize the complexity of other NP hard problems. Graphs with bounded HD admit a probabilistic embedding into graphs with bounded treewidth [61], this implies that several NP hard problems have quasi-polynomial approximation schemes in the context of transportation networks. Subsequently, [21] introduced new ideas to get an embedding that allows for polynomial time approximation schemes (as opposed to quasi-polynomial) for several problems when the HD is bounded. Let us consider an important problem, the k-center problem, arising in logistics and other domains. The k-center problem requires to find k center vertices such that the distance of every node to the closest center is minimized. This problem is known to be hard, but it surprisingly remains hard even if we parameterize the complexity jointly by k, the HD, and the treewitdh [62]. On the other hand, despite its parameterized hardness, there are approximation algorithms when k and the HD are combined [60]. Two fundamental problems are TSP (travelling salesman) and STP (Steiner tree). [53] gives improved results for TSP and STP when the HD is 1, obtaining a FPTAS and, on the negative side, they also show a hardness result when the HD is 6 or larger.

2.2.2 Preliminaries: Hitting Sets and the Highway Dimension

We give a generalization of the notion of *highway dimension*, introduced by [2, 4] to parametrize shortest-path computations in undirected graphs. The technical challenges of

these extensions may not be clear to a non-expert reader, thus we differ all discussions on the matter to Appendix A.1. Very broadly speaking, [2] deals only with undirected graphs and shortest paths, whereas our approach covers directed graphs and general sets of paths. In particular, if we restrict the set of paths accordingly, we recover the original notion.

To motivate general sets of paths, note that the CSP problem may have multiple solutions as there could be several paths with the same length and cost lower than b. To limit these solutions to those with minimal cost, we require that the path also be *efficient*.

Definition 2.2.1 (Efficient Path). A path $P \in \mathcal{P}_{s,t}$ is called *efficient* if there is no other path $P' \in \mathcal{P}_{s,t}$ such that $\ell(P') \leq \ell(P)$ and $c(P') \leq c(P)$ with at least one inequality strict. We denote the set of all efficient paths as $\mathcal{P}^E$.

Definition 2.2.2 (Path System). Recall that $G = (V, E)$ is fixed throughout. We define a *path system* $\mathcal{Q}$ as any collection of paths in G.

Our goal is to define the HD of any path system, then in Section 2.2.3 show how to answer queries parametrized it. We will use mainly two path systems, $\mathcal{P}^*$ (all shortest paths) and $\mathcal{P}^E$ (efficient paths), but note that our notion is general and encompasses other restrictions such as label constraints.

We say that a set $C \subseteq V$ *hits* any given path Q if some node in Q belongs to C. Moreover, we say that C is a *hitting set for a path system* $\mathcal{Q}$ if it hits every $Q \in \mathcal{Q}$. *Hitting sets are fundamental and they drive all the analysis and design of our algorithms.* A crucial intuitive concept we use is that of *compression*. As we discussed at the beginning of Section 2.2, if we want query time better than Dijkstra, i.e., $O(n \log n)$, a priori we need space $\Omega(n^2)$ to store all the shortest paths. We say that a data structure $\mathcal{S}$ compresses CSP if it has size $o(n^2)$ and query time $o(n \log n)$. The holy grail is a compression of space $O(n\text{polylog}(n))$ with query time $O(\text{polylog}(n))$. Hitting sets are fundamental because, as we will prove in

Proposition 2.2.6, they allow to compress path systems.

For any $r > 0$, we say a path Q is r-significant if $\ell(Q) > r$. For a given path system $\mathcal{Q}$, we denote $\mathcal{Q}_r$ as the set of all r-significant paths in $\mathcal{Q}$. In particular, even if the hitting set is large, the extent to which a path system can be compressed depends on the *local sparsity* of hitting sets with respect to *significant paths* of $\mathcal{Q}$.

Definition 2.2.3 (Locally-Sparse Hitting Sets). Given a path system $\mathcal{Q}$ and $r > 0$, an (h, r) locally-sparse hitting set (or (h, r)-LSHS) is a set $C \subseteq V$ with two properties:

1. Hitting: C is a hitting set for $\mathcal{Q}_r$.
2. Local sparsity: for every $v \in V$, $|B_{2r}(v) \cap C| \leq h$.

As we discuss in Section 2.2.3, the existence of (h, r)-LSHS immediately enables the compression of path system $\mathcal{Q}$ via the construction of *hub labels*. However, the existence of LSHS does not guarantee the ability to efficiently compute these objects. To address this, we need a stronger notion; the *highway dimension* is a property that ensures both existence and efficient computation of LSHS. To define the highway dimension (HD), we first need two additional definitions: for $v \in V, r > 0$, the *forward path-neighbourhood* with respect to a path system $\mathcal{Q}$ is $S_r^+(v, \mathcal{Q}) \coloneqq \{Q \in \mathcal{Q}_r : \text{dist}(v, Q) \leq 2r\}$ and similarly $S_r^-(v, \mathcal{Q}) \coloneqq \{Q \in \mathcal{Q}_r : \text{dist}(Q, v) \leq 2r\}$ is the reverse neighbourhood. As before, $S_r(v, \mathcal{Q}) \coloneqq S_r^+(v, \mathcal{Q}) \cup S_r^-(v, \mathcal{Q})$. Now we can define the HD of a path system $\mathcal{Q}$. Essentially, the HD re-orders the sequence of qualifiers in the definition of (h, r)-LSHS: it requires the existence of a small hitting set for each individual neighborhood, rather than a single hitting set which is locally sparse.

Definition 2.2.4 (Highway Dimension). A path system $\mathcal{Q}$ has HD h if, $\forall r > 0, v \in V$, there exists a set $H_{v,r} \subseteq V$ such that $|H_{v,r}| \leq h$ and $H_{v,r}$ is a hitting set for $S_r(v, \mathcal{Q})$.

As shorthand, we refer to the HD of (G, ℓ) as that of $\mathcal{P}^*$. Note that $HD \leq h$ is a more

stringent requirement than the existence of an (h, r)-LSHS C, since $C \cap B_{2r}(v)$ need not hit all the paths in $S_r(v, \mathcal{Q})$. However, if G has $HD \leq h$, then this guarantees the existence of a (h, r)-LSHS according to the following result, which can be proven by adapting the proof from [2, Theorem 4.2] to our general case.

Proposition 2.2.5. *If the path system $\mathcal{Q}$ has HD h, then, $\forall\, r > 0$, there exists an (h, r)-LSHS.*

More importantly, note that the result is about existence and does not touch on computability. As we discuss in Section 2.3.2, if G has $HD \leq h$, then this permits efficient computation of LSHS.

2.2.3 Preliminaries: Shortest-Paths via Hub Labels

Two of the most successful data-structures enabling fast shortest path queries at scale are *contraction hierarchies* (CH) [67] and *hub labels* (HL) [45]. These are general techniques which always guarantee correct SP computation, but have no uniform storage/query-time bounds for all graphs. We now explain the construction for HL; for the construction and results of CH refer to Appendix A.2.

The basic HL technique for SP computations is as follows: Every node v is associated with a hub label $L(v) = \{L^+(v), L^-(v)\}$, comprising of a set of forward hubs $L^+(v) \subseteq V$ and reverse hubs $L^-(v) \subseteq V$. We also store $\text{dist}(v, w)\, \forall\, w \in L^+(v)$ and $\text{dist}(u, v)\, \forall\, u \in L^-(v)$. Hub Labels must satisfy the *cover property* defined as follows: for any $s \neq t \in V$, $L^+(s) \cap L^-(t)$ contains at least one node in $P(s, t)$. In the case that t is not reachable from s, it must be that $L^+(s) \cap L^-(t) = \varnothing$.

With the aid of the cover property, we can obtain $\text{dist}(s, t)$ by searching for the minimum

value of $\text{dist}(s,w) + \text{dist}(w,t)$ over all nodes $w \in L^+(s) \cap L^-(t)$. If the hubs are sorted by ID, this can be done in time $O(|L^+(s)| + |L^-(t)|)$ via a single sweep. Moreover, by storing the second node in $P(s,w)$ for each $w \in L^+(s)$, and the penultimate node in $P(w,t)$ for each $w \in L^-(t)$, we can also recover the shortest path recursively, as each HL query returns at least one new node $w \in P(s,t)$. Note that we need to store this extra information, otherwise we could have $L^+(s) \cap L^-(t) = \{s\}$. Let $L_{\max} \coloneqq \max_v |L^+(v)| + \max_v |L^-(v)|$ be the size of the maximum HL. The per-node storage requirement is $O(L_{\max})$, while the query time is $O(L_{\max}\ell(P(s,t)))$.

Although hub labels always exist (in particular, we can always choose $L^+(s)$ to be the set of nodes reachable from v, and $L^-(s)$ the set of nodes that can reach v), finding *optimal* hub-labels (in terms of storage/query-time bounds) is known to be NP-hard [14]. To construct hub labels with guarantees on preprocessing time and $L_{\max}$, we need the additional notion of a *multi-scale LSHS*. We assume that (G,ℓ) admits a collection of sets $\{C_i : i = 1,\dots,\log D\}$, such that each C_i is an $(h, 2^{i-1})$-LSHS. Given such a collection, we can now obtain small HL. We outline this construction for directed graphs, closely following the construction in [2, Theorem 5.1] for the undirected case.

Proposition 2.2.6. *For (G,ℓ), given a multi-scale LSHS collection $\{C_i : i = 0,\dots,\log D\}$, where each C_i is an $(h, 2^{i-1})$-LSHS, we can construct hub labels of size at most $h(1+\log D)$.*

Proof. For each node v, we define the hub label $L(v)$ as

$$L^+(v) \coloneqq \bigcup_{i=0}^{\log D} C_i \cap B^+_{2^i}(v) \quad \text{and} \quad L^-(v) \coloneqq \bigcup_{i=0}^{\log D} C_i \cap B^-_{2^i}(v).$$

Since each C_i is an $(h, 2^{i-1})$-LSHS which we intersect with balls of radius $2 \cdot 2^{i-1}$, every set in the union contributes at most h elements and the maximum size is as claimed.

To prove the cover property, we note that, if t is not reachable from s, by definition $L^+(s) \cap L^-(t) = \varnothing$. This is because the elements in $L^+(s)$ are reachable from s and the elements in $L^-(t)$ reach t. On the other hand, when $P(s,t)$ exists, we do a case analysis on $\ell(P)$ to prove the cover property. Let i be such that $2^{i-1} < \ell(P(s,t)) \leq 2^i$. Finally, any point in the path belongs to both $B^+_{2^i}(s)$ and $B^-_{2^i}(t)$, and hence $C_i \cap P(s,t)$ is in both hubs (which is not empty since C_i hits all SPs of length $\geq 2^{i-1}$). □

Finally, we need to compute the desired multi-scale LSHS in polynomial time. In Section 2.3.2 we show that, if the HD is h, in polynomial time we can obtain sparsity $h' = O(h\Delta \log(h\Delta))$. In other words, the HL have size $h'(1 + \log D)$ instead of $h(1 + \log D)$ if we are not given the multi-scale LSHS and have to compute them in polynomial time. A more subtle point is that the resulting algorithm, even though polynomial, is impractical for large networks. In Section 2.4, we discuss heuristics that work better in practice.

2.3 Scalable CSP Algorithms: Theoretical Guarantees

We develop a data-structure that supports fast queries for *efficient paths*. Specifically, given a graph $G = (V, E)$ and a maximum budget B, we construct a data-structure such that, for any $s, t \in V$ and $b \leq B$, we return the length of the shortest (s,t)-path with cost at most b, denoted $\text{dist}(s, t|b)$. The actual path can easily be recovered as discussed in Section 2.2.3, hence we focus on querying $\text{dist}(s, t|b)$ only.

In Section 2.2.3 we discussed that, if a graph G has HD h, we can simultaneously bound, as functions of h, the preprocessing time, storage requirements, and query time for hub labels. This suggests that, for the construction of provably efficient hub labels for the CSP problem, we need an analogous property for the set of *efficient paths*.

Definition 2.3.1 (Constrained Highway Dimension). The constrained highway dimension (CHD) of (G, ℓ, c), denoted h_c, is the HD of the efficient-path system $\mathcal{P}^E$.

Note that, since every shortest path is efficient, $h_c \geq h$.

We now have two main issues with this definition: first, it is unclear how this can be used to get hub labels, and second, it is unclear how the corresponding hub labels compare with those for shortest-path computations. To address this, we first convert efficient paths in G to shortest paths in a larger *augmented graph*. In Section 2.3.2, we use this to construct hub labels for CSP queries whose storage and query complexity can be bounded as Bh_c (which can be strengthened further to $g(b)h_c$, where $g(b)$ measures the size of the Pareto frontier, cf. Section 2.3.2.). Finally, in Section 2.3.3, we show that the hub labels for CSP queries can in fact be related to the hub labels for SP queries under an additional natural condition on the efficient paths.

2.3.1 Augmented Graph

In order to link the constrained highway dimension to hub labels, we first convert the original graph G (with length and cost functions) into an *augmented graph* G^B with only edge lengths, such that the *efficient paths of G are in bijection with the shortest paths of G^B*. We achieve this as follows: Each node in G^B is of the form $\langle v, b \rangle$, which encodes the information of the remaining budget $b \geq 0$ and location $v \in V$. A node is connected to neighbors (according to E) as long as the remaining budget of that transition is non-negative. Finally, we create n sink nodes, denoted v^-, and connect node $\langle v, b \rangle$ to v^- with length $1/(b+1)$. An illustration of the construction is presented in Figure 2.1. The following definition formalizes this.

Definition 2.3.2 (Augmented Graph). Given (G, ℓ, c) and $B \in \mathbb{N}$, the augmented version G^B has vertex set $V^B := \{\langle v, b \rangle : v \in V, b = 0, 1, \ldots, B\} \cup \{v^- : v \in V\}$, the edge set E^B

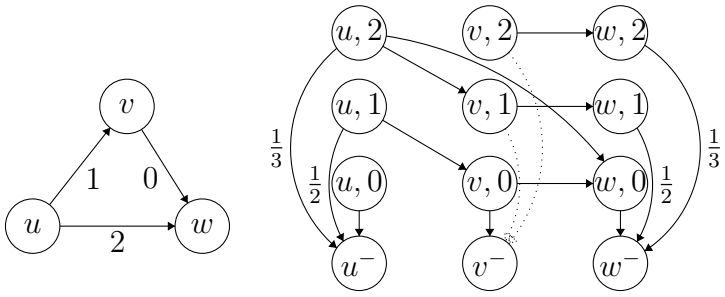


Figure 2.1: Example of a graph augmentation: The original graph G has all paths of unit length, and costs as indicated on the edges. In the augmented graph G^B, the labels represent the edge lengths (unlabeled edges have length 1). Note the additional edges from (w, b) to the sink node w^- (and similarly for u and v).

is

$\{\langle v, b\rangle\langle w, x\rangle : vw \in E, x = b - c_{vw}, x \geq 0\} \cup \{\langle v, b\rangle v^- : v \in V, 0 \leq b \leq B\}$. The length function in G^B is $\ell(\langle v, b\rangle, v^-) \coloneqq \frac{1}{b+1}$ and $\ell(\langle v, b\rangle, \langle w, x\rangle) \coloneqq \ell(vw)$.

Paths in G^B are mapped to paths in G in the intuitive way, by removing the budget labels and sink nodes. We call this mapping the *projection* of a path.

Proposition 2.3.3. *A shortest path from source $\langle s, b\rangle$ to sink node t^- projects to an efficient path in G solving dist$(s, t|b)$.*

Proof. Let P be the shortest path from $\langle s, b\rangle$ to t^-, and $\bar{P}$ its projection. To reach t^-, P must pass through some $\langle t, b'\rangle$, $b' \geq 0$. By construction, P consumes $b - b'$ units of resource, hence it is feasible; moreover, $\bar{P}$ is the shortest among (s, t)-paths with cost $b - b'$. Now assume, by way of contradiction, that $\bar{P}$ is not efficient. As $\bar{P}$ is the shortest using $b - b'$ units of resource, there exists P' such that $\ell(\bar{P}') \leq \ell(\bar{P})$ and $c(\bar{P}') < c(\bar{P})$. It must be that P' passes through $\langle t, b''\rangle$, with $b'' > b'$. We argue that, in this case, P would not be a shortest path to t^-. Indeed,

$$\ell(P') = \ell(\bar{P}') + \frac{1}{1 + b''} \leq \ell(\bar{P}) + \frac{1}{1 + b''} < \ell(\bar{P}) + \frac{1}{1 + b'},$$

where the last expression is exactly $\ell(P)$. □

Later we give a construction that depends on LSHS for the system $\mathcal{P}^E$, we call this object an efficient path hitting set (EPHS). The next result allows us to relate EPHS of the original graph to LSHS in the augmented graph. Note that, in G^B, we are interested only in shortest paths ending in sink nodes (since these map to efficient paths). Let $\mathcal{P}^B$ be the path system comprising all shortest paths in G^B ending in a sink node. A hitting set for $\mathcal{P}^E$ (in G) can be used to obtain a hitting set for $\mathcal{P}^B$ (in G^B), but, since the augmented graph has more information, the sparsity increases.

Proposition 2.3.4. *Given a (h_c, r)-EPHS for the path system $\mathcal{P}^E$, in polynomial time we can construct a $(h_c B, r)$-LSHS for $\mathcal{P}^B$.*

Proof. Given C, an (h_c, r)-EPHS for $\mathcal{P}^E$, define

$$C^B \coloneqq \{\langle v, b\rangle : v \in C, v \text{ hits } \bar{P} \in \mathcal{P}^B_r, c(\bar{P}) = b \leq B\}. \tag{2.1}$$

We prove that C^B hits $\mathcal{P}^B_r$ and is locally sparse. By Proposition 2.3.3, we know that shortest paths are efficient, hence C^B hits all the desired paths. Finally, we prove local sparsity. Take any node $\langle s, b\rangle$ and observe that

$$B^+_{2r}(\langle s, b\rangle) = \{\langle t, x\rangle : \exists P \in \mathcal{P}_{s,t}, \ell(P) \leq 2r, c(P) = b - x\} \subseteq \{\langle t, x\rangle : t \in B^+_{2r}(s), x \leq b\}. \tag{2.2}$$

We know that $|B^+_{2r}(s) \cap C| \leq h_c$, therefore $|B^+_{2r}(\langle s, b\rangle) \cap C^B| \leq h_c b \leq h_c B$. A similar argument shows the sparsity for the reverse ball. □

The proof above shows a stronger result: In Eq. (2.2) we see that the sparsity around the node $\langle u, b\rangle$ is $h_c b$. This is key for our subsequent query time guarantees.

Surprisingly, in this case we can also relate the HDs of the path systems $\mathcal{P}^E$ and $\mathcal{P}^B$. Note that this does not follow from Proposition 2.3.4, since the HD is a stronger notion than existence of locally-sparse hitting sets.

Proposition 2.3.5. *If the HD of the system $\mathcal{P}^E$ is h_c, then the HD of the system $\mathcal{P}^B$ is Bh_c.*

Proof. Fix $r > 0$ and $\langle v, b\rangle \in V^B$. Let $H_{v,r} \subseteq V$ be the set hitting $S_r(v, \mathcal{P}^E)$ and define $H \coloneqq H_{v,r} \times \{0, 1, \ldots, B\}$. We show that H hits $S_r^+(\langle v, b\rangle, \mathcal{P}^B)$.

Take $P \in S_r^+(\langle v, b\rangle, \mathcal{P}^B)$. Since $\mathrm{dist}(\langle v, b\rangle, P) \leq 2r$, $\mathrm{dist}(v, \bar{P}) \leq 2r$, therefore $\bar{P} \in S_r^+(v, \mathcal{P}^E)$. Finally, $H_{v,r}$ hits $\bar{P}$, thus H hits P. A similar argument shows that H hits $S_r^-(\langle v, b\rangle, \mathcal{P}^B)$. □

2.3.2 Solving CSP via Hub Labels

We present a construction similar to HL for shortest-paths (cf. Section 2.2.3). A subtle difference is that we are only interested in paths ending in a sink node. Each node $\langle v, b\rangle$ has a forward hub label $L^+(\langle v, b\rangle) \subseteq V^B$, and *only sink nodes* u^- have a reverse hub $L^-(u^-) \subseteq V^B$. The cover property must be satisfied for every $\langle s, b\rangle$ and t^-. Finally, if we want to reconstruct the path, we can proceed similarly as in Section 2.2.3; we can augment the hub labels with the next-hop node, and compute the entire path recursively. Putting things together, we can construct hub labels for answering CSP queries, with preprocessing time and storage parameterized by the CHD h_c.

Query Time and Data Requirements

Theorem 2.3.6. *For a network (G, ℓ, c), given a multi-scale EPHS $\{C_i : i = 0, 1, \ldots, \log D\}$, where C_i is an $(h_c, 2^{i-1})$-EPHS, we can construct hub labels to answer queries for s, t, b in time $O((B + 1)h_c \log D)$. The total space requirement is $O(nB \cdot Bh_c \log D)$.*

PROOF. Create C_i^B as in Eq. (2.1). Define $L(\langle v, b\rangle)^+ \coloneqq \bigcup_{i=1}^{\log D} C_i^B \cap B_{2^i}^+(\langle v, b\rangle)$ and $L(u^-)^- \coloneqq \bigcup_{i=1}^{\log D} C_i^B \cap B_{2^i}^-(u^-)$. The cover property is proved similarly as in Proposition 2.2.6; we are left to bound the hub size. For a reverse hub we use that $B_{2^i}^-(t^-) = \{\langle s, x\rangle : \exists P \in \mathcal{P}_{s,t}, c(P) = x, \ell(P) \leq 2^i\} \subseteq B_{2^i}^-(t) \times \{0, 1, \ldots, B\}$. Thus, $B_{2^i}^-(t^-) \cap C_i^B \leq (B+1)h_c$. For forward hubs, the size follows from observing that $|C_i^B \cap B_{2^i}^+(\langle v, b\rangle)| \leq (b+1)h_c$. □

Preprocessing

Computing hitting sets is difficult in general, but it becomes tractable when the underlying set has small VC-dimension [57]. The critical observation in [2] is that the set system of *unique* shortest paths has a VC-dimension of 2. Directed or non-shortest paths break the arguments in [2], so we need a different formulation of the set system. We recall the concept of VC-dimension. A set system $(X, \mathcal{X})$ is a ground set X together with a family $\mathcal{X} \subseteq 2^X$. A set $Y \in \mathcal{X}$ is shattered if $\{Z \cap Y : Z \in \mathcal{X}\} = 2^Y$. If d is the smallest integer such that no $Y \in \mathcal{X}$ with $|Y| = d + 1$ can be shattered, then this number d is the VC-dimension of $(X, \mathcal{X})$. The critical observation in [2] is that the set system of *unique* shortest paths has a VC-dimension of 2, but their argument does not hold in directed graphs or general path systems.

Let $\mathcal{Q}$ be any path system. We can obtain a set system with small VC-dimension by considering the ground set as E (instead of the usual choice of V), and mapping a path $Q = e_1 e_2 \ldots e_k$ to $\pi(Q) = \{e_1, e_2, \ldots, e_k\}$. Note that, since path systems contain no cycles, each set $\{e_1, e_2, \ldots, e_k\}$ corresponds uniquely to one path.

Proposition 2.3.7. *Given a path system $\mathcal{Q}$, the corresponding set system $(E, \{\pi(Q) : Q \in \mathcal{Q}\})$ has VC-dimension 2.*

Note that this argument also can be used for shortest paths in undirected graphs to

remove the uniqueness requirement. Finally, polynomial-time preprocessing now follows from combining Proposition 2.3.7 and the main result in [57]. The desired result is stated in Proposition 2.3.8, we defer the proof to Appendix A.4.

Proposition 2.3.8. *If a path system $\mathcal{Q}$ has HD h, then, for any $r > 0$, we can obtain in polynomial time a (h', r)-LSHS, where $h' = O(h\Delta \log(h\Delta))$ and Δ is the maximum degree.*

Using the size of the Pareto Frontier

The linear dependence on B in the bound on HL sizes (cf. Theorem 2.3.6) is somewhat weak. Essentially, this corresponds to a worst-case setting where the efficient paths between any pair of nodes is different for each budget level. In most practical settings, changing the budget does not change the paths too much, and ideally the hub label sizes should reflect this fact. This is achieved via a more careful construction of hub labels, resulting in the following bound.

Theorem 2.3.9. *Let (G, ℓ, c) as in Theorem 2.3.6 and $g : \mathbb{N} \to \mathbb{N}$ be such that, for every $s, t \in V$, $b \in \mathbb{N}$, $|\{P \in \mathcal{P}^E_{s,t} : c(P) \leq b\}| \leq g(b)$. Then, we can construct hub labels of size $O(g(B)h_c \log D)$, and answer queries with budget b in time $O(g(b)h_c \log D)$.*

Note that there always exists such a function g and the worst case is $g(b) = b$. The proof depends on a different technique for constructing HL (refer to Algorithms 12 and 13 in Appendix A.4). The main idea is to sort the efficient paths for each source node s by cost, and then carefully mark nodes when they are added to the forward HL; these marked nodes are then used to construct the reverse HL. For brevity, the complete algorithmic details and proof are deferred to Appendix A.4.

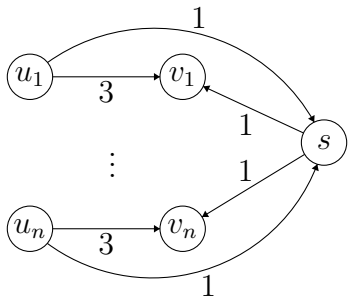


Figure 2.2: Graph with small HD but large CHD: The graph comprises $2n + 1$ nodes, with the edge labels representing the lengths. Note that the shortest paths in the graph are of the form sv_i, u_is and u_isv_j (for all combinations i, j). Thus, the HD is 1 as $H_{v,r} = \{s\}$ is a hitting set for all these paths. On the other hand, if we have costs such that $c(u_iv_i) = 0 \forall\, i$, while all other edges have cost 1, then we have n parallel efficient paths u_iv_i, which must all be hit by any EPHS.

2.3.3 Comparing HD and CHD

The previous sections show that we can construct hub labels for solving CSP whose preprocessing time, storage and query time can all be parameterized in terms of the constrained highway dimension h_c. This, however, still does not give a sense of how much worse the hub labels for the CSP problem can be in comparison with those for finding shortest paths. We now try to understand this question. Comparing the size of the *optimal* hub labels for SP and CSP is infeasible as even finding the optimal hub labels for SP is NP-hard [14]. However, since we can parametrize the complexity of HL construction for SP in terms of the HD, a natural question is whether graphs with small HD also have a small CHD. Note that the answer to this depends on both the graph and the costs. We now show that the CHD and, moreover, the sparsity of any EPHS, can be *arbitrarily worse* than the HD.

Proposition 2.3.10. *There are networks with HD* 1 *where the CHD is* n*. Furthermore, the sparsity of an EPHS is also* n*.*

Proof. Consider the directed graph G defined in Figure 2.2. It is easy to see that $H_{v,r} = \{s\}$ is a shortest-path hitting set for every $r > 0$ and $v \in V(G)$; hence the HD is 1. On the other

hand, suppose the costs are such that $c(u_iv_i) = 0$ for every i, while all other costs are set to 1. Note that the 1-significant efficient paths intersecting the ball $B_s(2)$ are u_iv_i, which are all disjoint. Therefore, the hitting set $H_{s,1}$ must contain at least n elements. This concludes the separation between HD and CHD. Finally, the same argument shows that there is LSHS for $\mathcal{P}^*$ with sparsity 1, whereas the sparsity of *any* EPHS is also lower bounded by n. Thus the sparsities of LSHS and EPHS are also separated. □

Remark 2.3.11. One criticism of the graph in Fig. 2.2 is that it has a maximum degree of n. However, the result holds even for bounded degree graphs. In Appendix A.1, where we discuss alternative notions of HD, we give a more involved example with bounded degrees that exhibits the same separation between LSHS and EPHS. Another criticism is that the edge u_iv_i is not the shortest (u_i, v_i)-path, but this is for exposition only and the same holds with the following modification: add a node w_i between u_i and v_i, then set $\ell(u_iw_i) = 2$, $\ell(w_iv_i) = 1$, and $c(u_iw_i) = c(w_iv_i) = 0$.

Intuitively, the separation between HD and CHD occurs due to the fact that, for arbitrary graphs and cost functions, the shortest and efficient paths may be completely unrelated. For real-world networks, however, this appears unlikely. In particular, intuition suggests that efficient paths largely comprise of segments which are in fact shortest-paths in their own right. This notion can be formalized via the following definition of a *partial witness*

Definition 2.3.12 (Partial Witness). Let $\beta \geq 0$. We say that a path system $\mathcal{Q}$ is β-witnessed by the path system $\mathcal{Q}'$ if, for every $Q \in \mathcal{Q}$, $\exists Q' \in \mathcal{Q}'$ such that $Q' \subseteq Q$ and $\ell(Q') \geq 2^{-\beta}\ell(Q)$.

We use the partial-witness property to provide a link between the CHD and HD, essentially showing that scaling CSP queries is of similar complexity to scaling SP queries as long as any efficient path contains large shortest-path segments, this is formally stated in The-

orem 2.3.14 below. We illustrate the definition and the intuition behind it in the following example.

Example 2.3.13 (Multimodal Network). Consider a *multimodal mapping service* which gives transit routes combining different modes of transportation, e.g., walking, bus, subway, trolley, etc. As a desirable feature, we ensure that routes have at most k transfers. We claim that, in this network, the partial witness property is satisfied with $\beta = \log_2(k+1)$. Indeed, with at most k transfers there are at most $k+1$ segments, hence one of them must be larger than a fraction $\frac{1}{k+1}$. Finally, we remark that, if each individual network has small HD, then the CHD is also small (cf. Proposition A.3.1).

We can now ask if the hub labels for computing SPs and CSPs can be related in settings where the shortest-path system $\mathcal{P}^*$ is a partial witness for the efficient path system $\mathcal{P}^E$. At an intuitive level, the partial witness property says that efficient and shortest paths are not completely different, i.e., if Q is efficient, a fraction $2^{-\beta}$ of Q is a shortest path. As a consequence, a node hitting numerous paths in $\mathcal{P}^*$, should also hit many paths in $\mathcal{P}^E$. Note that asking for the witness property to hold for all lengths is too extreme, as this essentially requires that all single-hop paths with 0 costs are shortest paths. Thus, we want this property only for 'long-enough' paths.

We now show that if, for some β, the network indeed has the partial witness property for paths longer than some r_β, then we can relate the HL sizes for the two problems in terms of β and the doubling constant α. Note that the doubling constant depends on G and ℓ; the partial witness property depends on the interplay between G, c and ℓ. Observe also that, if α is a constant, then the requirement in Theorem 2.3.14 is for paths longer than $r_\beta \sim h\alpha^{\beta-2}$.

Theorem 2.3.14. *Assume G is α-doubling and $\mathcal{P}_r^E$ is β-witnessed by $\mathcal{P}^*$ for every $r \geq r_\beta$, where $r_\beta = 2^{\log_\alpha(h\alpha^{\beta-2})}$. Then, for any $r > 0$, given an (h, r)-LSHS, we can construct, in polynomial time, an $(h\alpha^\beta, r)$-EPHS for (G, ℓ, c).*

PROOF. For any r, we need to construct a hitting set C^E for $\mathcal{P}_r^E$. Assume first $r \geq r_\beta$. Let C be the hitting set for $\mathcal{P}^*_{2^{-\beta}r}$ which is guaranteed to be sparse with respect to balls of radius $2^{-\beta+1}r$. Define the desired set by $C^E \coloneqq \{v \in C : v \text{ is in some } r\text{-efficient path}\}$.

Since $\mathcal{P}^*$ is a $2^{-\beta}$-witness for $\mathcal{P}_r^E$, C^E is indeed a hitting set for $\mathcal{P}_r^E$. We are only left to prove the sparsity. Take some $u \in V$, by doubling constant we can cover $B^+_{2r}(u)$ by at most α^β balls of radius $2^{-\beta+1}r$. Each of these balls contains at most h elements of C, therefore the sparsity is as claimed. The argument for reverse balls is identical.

Now we analyse the case $r < r_\beta$. It is no longer true that efficient paths are witnessed, but now the neigborhoods are small. We first claim that, for any $v \in V$ and $r > 0$, $|B_r(v)| \leq \alpha^{\log_2 r+1}$. Indeed, using the doubling property $\log_2 r + 1$ times, we can cover $B_r(v)$ with balls of radii $1/2$. Since the minimum edge length is 1, all of these balls must be singletons and the claim follows. Now we can take $C = V$ as the EPHS. Clearly C hits all the paths and the local sparsity is at most the size of the ball. Using our assumption on r_β, we verify that $|B_{2r}(v)| \leq \alpha^{\log_2 r_\beta+2} \leq h\alpha^\beta$. □

Remark 2.3.15. The existence of a β-witness is not enough to bound the CHD. Nevertheless, as we discussed in Section 2.2.3, the existence of (h, r)-LSHS already allows the construction of HL. Moreover, the above argument does indeed give a bound for a weaker definition of the highway dimension [4].

2.3.4 Average-Case Performance Guarantees

Converting the partial-witness condition to a more interpretable condition is difficult in general, as the structure of $\mathcal{P}^*$ and $\mathcal{P}^E$ may be complex. One way to get such a condition, however, is by considering *average-case* performance metrics. For this, we relax the definition of HD in two ways: (i) we require LSHS to be locally sparse "on average" over all nodes,

and (ii) we only require the existence of LSHS (as opposed to a hitting set for $S_r(v, \mathcal{Q})$).

Definition 2.3.16 (Average LSHS). Given $r > 0$ and a system $\mathcal{Q}$, a set $C \subseteq V$ is an average (h, r)-LSHS if it hits $\mathcal{Q}_r$ and is locally sparse in average, i.e., $\frac{1}{n}\sum_{v\in V}|B_{2r}(v)\cap C| \leq h$.

Definition 2.3.17 (Average HD). The system $\mathcal{Q}$ has average HD h if, for every $r > 0$, there exists an average (h, r)-LSHS.

Recall that all we need to construct HL is the ability to find LSHS, hence our definition is transparent in the sense that we ask only for the existence of LSHS. We note that [1] also introduces a notion of average HD, but they use a more restrictive concept. Indeed, their stronger condition implies both (i) existence and (ii) ability to find LSHS. Therefore, we need to prove that existence is enough to find these sets (approximately) in polynomial time, we do this in Theorem 2.3.18. Finally, we remark that our weaker notion of average HD allows us to get stronger results (because it is easier to satisfy and verify), hence we believe it can be useful in other problems too.

Theorem 2.3.18. *If $\mathcal{P}^*$ has average HD h, then we can obtain, in polynomial time, HL with average size $\frac{1}{n}\sum_{v\in V}|L^+(v)| \leq h' \log D$ and $\frac{1}{n}\sum_{v\in V}|L^-(v)| \leq h' \log D$, where $h' = O(\Delta h \log(hn\Delta))$.*

Note that since query time depends linearly on the hub size, the above result implies both storage and performance bounds.

Proof. We only show how to compute average LSHS, since the construction of HL is the same as in Proposition 2.2.6 and the bound for the size easily follows. The objective is to obtain a set C_i that is an average $(h', 2^i)$-LSHS. This turns out to be a minimum-cost hitting

set problem. Indeed, we want to solve

$$C_i = \text{argmin}\left\{\sum_{v\in V}|B_{2^{i+1}}(v)\cap C| : C\subseteq V, C \text{ hits } \mathcal{P}^*_{2^i}\right\}.$$

This follows from a symmetry argument, assigning to each node u the cost $c(u) = |\{v \in V : u \in B_{2^{i+1}}(v)\}|$. On the other hand, given a minimum cost hitting set problem with optimum value τ, if the set system has VC-dimension d, the algorithm in [57] finds a solution, in polynomial time, with cost at most $O(d\tau \log(d\tau))$.

By assumption, the minimum of the problem is at most hn. Now we perform a mapping from paths to sets, where the ground set is E and paths are sequences of edges. This system has VC-dimension 2, and now the minimum is at most $h\Delta n$. We apply the algorithm in [57] and obtain a solution C_i with cost at most $O(h\Delta n \log(h\Delta n))$; this gives the promised average $(h', 2^i)$-LSHS. □

Relaxed Witness

We describe a realistic setting where we can obtain small, in average, HL for the CSP.

First, we assume that individual edges are shortest paths. This assumption is mainly for exposition purposes and can be further relaxed using the same analysis. Second, we add an additional constraint wherein we insist our efficient paths have bounded *stretch* compared to the shortest path; note that this is natural in applications as users do not want to be presented solutions which are far away from the optimum, even if it saves them budget. Formally, we define:

Definition 2.3.19 (Stretch). An algorithm for CSP has stretch $S^t \geq 1$ if, $\forall s, t \in V$ and $b \leq B$, it outputs $\text{dist}(s, t|b)$ whenever $\text{dist}(s, t|b) \leq S^t \cdot \text{dist}(s, t)$ and outputs "infeasible" when $\text{dist}(s, t|b) > S^t \cdot \text{dist}(s, t)$.

We henceforth treat S^t as an extra constraint given by the application. Next, let $E_c \coloneqq \{e \in E : c_e > 0\}$ denote the set of 'costly' edges. We define the following notion of an *overpass*:

Definition 2.3.20 (Overpass). For $r > 0$, the edge $e = (u, v)$ is an r-overpass if:

(1) e belongs to a path $Q \in \mathcal{P}^E_{2r} \setminus \mathcal{P}^*$

(2) both u and v are endpoints of paths in $\mathcal{P}^*_{r(2/S^t-1)}$ and

(3) $\min(\text{dist}(e, E_c), \text{dist}(E_c, e)) \leq 3r/2$

Essentially, overpasses are edges connecting long shortest-paths in a costly zone; Fig. 2.3 shows an example. In case costs are contiguous (for example, tolls on highways or traffic jams), then the definition corresponds to the intuitive notion of an overpass. Our main requirement is the following *bounded growth* condition, controlling the number of r-overpasses for every scale $r > 0$.

Definition 2.3.21 (Bounded Growth). (G, c, ℓ) satisfies the bounded growth condition if, $\forall r > 0$, $|\{u \in V : \exists v, uv \text{ is an } r\text{-overpass}\}| \leq \phi(2r)$, where $\phi(r) \coloneqq nh\alpha^{\beta-2-\log_2 r}$

Observe that ϕ is a slowly decreasing function of r and, even when $r = D$, we allow for overpasses. Now, with these conditions, we get our main result of this section:

Theorem 2.3.22. *Let (G, ℓ, c) be a network with HD h and doubling constant α. If the bounded growth is satisfied, we can obtain, in polynomial time, hub labels for CSP queries*

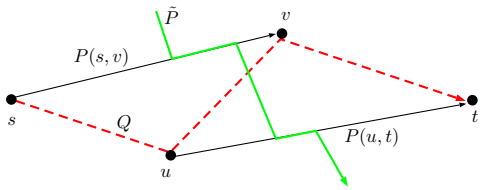


Figure 2.3: Edge uv here is an overpass, lying on efficient path Q (dashed red), connecting a pair of long shortest paths $P(s, v)$ and $P(u, t)$, and close to costly edges $\tilde{P}$ (solid green).

that guarantee average query-time $O((b+1)h'\alpha \log D)$ *and total storage* $O(nB \cdot Bh'\alpha \log D)$*, where* $h' = O(\Delta h \log(hn\Delta))$*.*

Roadmap of the proof. The remainder of this section is devoted to the proof of Theorem 2.3.22. We henceforth refer to the average HD of $\mathcal{P}^E$ as average CHD. We prove that, under the bounded growth condition, the average CHD is $h_c \leq 2h\alpha$, which intuitively allows for the construction of HL based on our previous result (Theorem 2.3.18). To bound the average CHD, we need to operate under even less restrictive assumptions than the partial-witness. We first weaken the partial-witness concept, as defined below, by allowing for an additional *supplementary witness set* D such that *every efficient path is either witnessed by a shortest-path or hit by* D. This corresponds to the intuition that a few bad efficient paths should not completely ruin the algorithm. This weakened concept drives our analysis and allows us to bound the average CHD.

Definition 2.3.23 (Weak Partial Witness). Given $\beta \geq 0$, we say a path system $\mathcal{Q}$ is weakly β-witnessed by the path system $\mathcal{Q}'$ if, for every $r > 0$, $\exists D_r \subseteq V$ such that, $\forall Q \in \mathcal{Q}_r$ either (1) Q is β-witnessed by $\mathcal{Q}'$ or (2) Q is hit by D_r. Additionally we require $|D_r| \leq \phi(r)$.

Intuitively, the supplementary witness set D_r takes care of all the corner cases where $\mathcal{Q}$ and $\mathcal{Q}'$ differ too much. We show that our requirement on $|D_r|$ guarantees a bound on the average HD.

Proposition 2.3.24. *Assume that* G *is* α*-doubling and let* $\mathcal{Q}'$ *be weakly* β*-witnessed by* $\mathcal{Q}$*. If the HD of* $\mathcal{Q}$ *is* h*, then the average HD of* $\mathcal{Q}'$ *is* $h' \leq 2h\alpha^{\beta}$

Proof. Let $r > 0$, and C be an $(h, 2^{-\beta}r)$-LSHS for $\mathcal{Q}$. We show that $C \cup D_r$ is an average

(h, r)-LSHS for $\mathcal{Q}'$. Clearly it is a hitting set for $\mathcal{Q}'_r$ and we can compute

$$\begin{aligned} h' &\leq \frac{1}{n}\sum_{v\in V}|B_{2r}(v)\cap(C\cup D_r)| \\ &\leq \frac{1}{n}\left(\sum_{v\in V}|B_{2r}(v)\cap C| + \sum_{v\in V}|B_{2r}(v)\cap D_r|\right) \\ &\leq \frac{1}{n}\left(\sum_{v\in V} h\alpha^{\beta} + \sum_{v\in D_r}|B_{2r}(v)|\right) \leq h\alpha^{\beta} + \frac{|D_r|\alpha^{\log_2 r+2}}{n}. \end{aligned}$$

In the third inequality we used that C is sparse with respect to balls of radius $2^{-\beta+1}r$ and that $\sum_{v\in V}|B_{2r}(v)\cap D_r| = \sum_{v\in D_r}|B_{2r}(v)|$ by symmetry of the bi-directional balls. In the last inequality we used that, by doubling constant, balls of radius r have at most $\alpha^{\log_2 r+1}$ elements. Since $|D_r| \leq \phi(r)$, the result follows. □

This now allows us to link $\mathcal{P}^E$ and $\mathcal{P}^*$ as follows:

Proposition 2.3.25. *Under the bounded growth condition, $\mathcal{P}^*$ is a weak 1-witness for $\mathcal{P}^E$.*

Proof. We first need some additional notation: For a path Q and two vertices $u, v \in Q$ we denote $Q[u, v] \subseteq Q$ as the sub (u, v)-path; for two paths P, Q with a common endpoint, we denote $P|Q$ as their concatenation.

Consider $Q \in \mathcal{P}^E$ with endpoints s, t and set $\ell(Q) = 2r$. We will show how to obtain a vertex for the supplementary witness set in case Q is not witnessed and then we bound the size of the set. Assume that $Q \neq P(s, t)$, otherwise the path is trivially witnessed. Let $(u, v) \in Q$ be such that $\ell(Q[s, v]), \ell(Q[u, t]) \geq r$ (see Fig. 2.3). If either $Q[s, v]$ or $Q[u, t]$ is a shortest path or $\ell_{uv} \geq r$, then Q is witnessed. Thus, we henceforth assume that all of the above conditions fail.

We claim that uv is a r-overpass (in fact, this is the exact scenario depicted in Fig. 2.3). Condition 1 is clearly satisfied. Condition 2 also holds because both $Q[s, v]$ or $Q[u, t]$ have

stretch at most $\frac{2-S^t}{S^t}$. To see this, note that both $P(s,v)|Q[v,t]$ and $Q[s,u]|P(u,t)$ are no shorter than $P(s,t)$ and $S^t\ell(P(s,t)) \geq \ell(Q)$, hence $\frac{2r}{S^t} \leq \ell(P(s,t)) \leq \ell(P(s,v)) + \ell(Q[v,t])$ and $\frac{2r}{S^t} \leq \ell(P(s,t)) \leq \ell(Q[s,u]) + \ell(P(u,t))$. Since each path $Q[s,u], Q[v,t]$ has length less than r, it follows that both $P(s,v), P(u,t)$ have length strictly greater than $\frac{2-S^t}{S^t}r$. Finally, to show condition 3, we have $\ell(Q[s,v]) + \ell(Q[u,t]) = 2r + \ell_{uv}$ and since $\ell_{uv} < r$, one of $Q[s,v]$ or $Q[u,t]$ has length at most $3r/2$. Since neither of these paths is shortest, it must be that both $P(s,v), P(u,t)$ have costly edges and thus one of u, v is closer than $3r/2$ to E_1 and the condition is satisfied.

For every path of length $2r$, we can thus either exhibit a witness or show that it contains an overpass and add use the tail of the edge as a supplementary witness. For a fixed $r > 0$, we need to add at most $\phi(r)$ nodes to D_r to cover all the efficient paths. The result follows. □

We are almost ready to prove that bounded growth allows to solve the CSP. The last piece is Proposition 2.3.26, the proof of which follows form similar arguments as those in Theorems 2.3.6 and 2.3.18.

Proposition 2.3.26. *If the average HD of $\mathcal{P}^E$ is h_c, then we can construct, in polynomial time, hub labels for CSP, which guarantee average query-time $O((b+1)h'_c \log D)$ for queries with budget b, and total storage requirements $O(nB \cdot Bh'_c \log D)$.*

Proof of Theorem 2.3.22. We argue that the average CHD is $h_c \leq 2h\alpha$. By Proposition 2.3.25, $\mathcal{P}^E$ is weakly 1-witnessed by $\mathcal{P}^*$. It follows by Proposition 2.3.24 that the average CHD is at most $2h\alpha$ as needed. Applying Proposition 2.3.26 yields the result. □

2.4 Scalable CSP Algorithms: Implementations and Experiments

Our theoretical results in the preceding sections suggest that using hub labels for CSP queries should perform well in road networks, as these are known to have low highway dimension, and potentially also satisfy the (average) partial witness property. We use our theoretical findings to guide the construction of practical algorithms; the modifications of the theoretical algorithms are simply to replace some of the black box routines. We now describe how our techniques can be adapted to give practical hub label constructions, and discuss experimental results for two real world networks using these methods.

2.4.1 Practical CSP Algorithms

We start by defining a more scalable construction of G^B. The augmented graph G^B defined in Section 2.3.1 is not a minimal representation as it may contain a lot of redundant information. For example, the same efficient path uv can be repeated many times in the form $\langle u, 1\rangle\langle v, 0\rangle$, $\langle u, 2\rangle\langle v, 1\rangle$ and so on. By encoding this information more efficiently, we get considerable improvements both in query time and in data storage.

We construct our *pruned* augmented graph $\tilde{G}^B$ as follows: As before, nodes are pairs $\langle v, b\rangle$, but now we add an edge $\langle v, b\rangle\langle v', b'\rangle$ only if it is essential for some efficient path, i.e., removing said edge impacts correctness. If we let $\mathcal{P}^E_{s,t}$ be the set of all efficient paths from s to t, we take every $P \in \mathcal{P}^E_{s,t}$ with cost $b \leq B$ and trace it in the augmented graph such that it terminates at $\langle t, 0\rangle$.

Definition 2.4.1. The pruned augmented graph is defined by $\tilde{G}^B = (\tilde{V}^B, \tilde{E}^B)$, where

$$\tilde{V}^B := \{\langle v, b\rangle : v \in V, b = 0, 1, \ldots, B\},$$

$$\tilde{E}^B := \{\langle v, b\rangle\langle u, x\rangle : \exists s, t \in V, P \in \mathcal{P}^E_{s,t}, c(P) \leq B, vu \in P, b = c(P[v, t]), x = c(P[u, t])\}.$$

In $\tilde{G}^B$ all the lengths are preserved.

Note that in $\tilde{G}^B$ there are no sink nodes, hence it has at least n nodes and nB arcs fewer compared to G^B. In the worst case, those nB arcs are the only gain by doing this process. Nevertheless, in our experiments $\tilde{G}^B$ is up to 60% smaller than G^B. Observe that, by running Dijkstra in G^B, $\tilde{G}^B$ can be computed in time $O(n^2 B \log(nB))$.

HD of the pruned augmented graph

A shortest path in $\tilde{G}$ does not necessarily project to an efficient path, even if the path ends in a node of the form $\langle t, 0\rangle$. In contrast, if P projects to an efficient path, then necessarily P is shortest. To bound the HD, the correct system to study is

$$\tilde{\mathcal{P}^B} := \{P : P \text{ ends in a node } \langle t, 0\rangle, \bar{P} \in \mathcal{P}^E, c(\bar{P}) \leq B\}.$$

The following result shows how the HD of this system relates to that of $\mathcal{P}^E$. We omit the proof since it is identical as the one in Proposition 2.3.5.

Proposition 2.4.2. *Given CHD h_c, the HD of $\tilde{\mathcal{P}^B}$ is Bh_c.*

Types of queries

We test our algorithms with two different tasks. Recall that our preprocessing is done for some fixed maximum budget B. The first task we consider is a *frontier query*, wherein given s and t, we return the lengths of all efficient paths with costs $b = 0, 1, \ldots, B$. The second we call a *specific query*, we return $\text{dist}(s, t|b)$ for given s, t, b (i.e., a single efficient path).

Note that the pruned augmented graph $\tilde{G}^B$ is designed for frontier queries. To see this, fix the terminal $\langle t, 0\rangle$. As we ask for the shortest path from $\langle s, B\rangle, \langle s, B-1\rangle, \ldots, \langle s, 0\rangle$ we

are guaranteed to recover the entire frontier. On the other hand, it may be that the shortest path between $\langle s, b\rangle$ and $\langle t, 0\rangle$ does not correspond to $\text{dist}(s, t|b)$. This occurs when b is not a tight budget and the efficient path requires less.

To answer specific queries, we modify $\tilde{G}^B$ by adding extra edges. For every $v \in V(G)$ and $b = 1, 2, \ldots, B$, we include the edge $\langle v, b\rangle\langle v, b-1\rangle$ with length 0. A simple argument shows that with the added edges, the shortest path between $\langle s, b\rangle$ and $\langle t, 0\rangle$ has length $\text{dist}(s, t|b)$.

HL construction via Contraction Hierarchies

We use some techniques described in [3] combined with an approach tailored for augmented graphs. The CH algorithm takes as input any ranking (i.e., permutation) of the nodes, and proceeds by removing nodes from the lowest rank first. Whenever a node is removed, we add new edges, called shortcuts, if needed to preserve the shortest paths. Once we have a graph with shortcuts, a CH search is a special variant of Dijkstra where only higher rank nodes are explored, i.e., we never take an edge uv if $\text{rank}(u) > \text{rank}(v)$. The main idea in our construction is to choose an appropriate ranking, and then define the forward hubs of v as the nodes visited during a contraction-based forward search starting at v. The reverse hubs are defined analogously. These are valid hubs, since the highest rank node in a path is guaranteed to be in both hubs.

The choice of the ranking function is crucial. For our experiments, we ranked nodes in G by running a greedy approximate SP cover, selecting the highest rank node as the one covering most uncovered paths in $\mathcal{P}^*$ and continuing greedily. Specifically, start with a cover $C = \varnothing$ and compute the set of all shortest paths $\mathcal{P}^*$. Take a node $v \notin C$ hitting most paths in $\mathcal{P}^*$, then remove all those paths from $\mathcal{P}^*$, add v to C and iterate. The rank is defined as n for the first node added to C, $n-1$ for the second and so on. To implement the SP cover we follow the algorithm in [3]. A practical hurdle in such an approach is that to compute

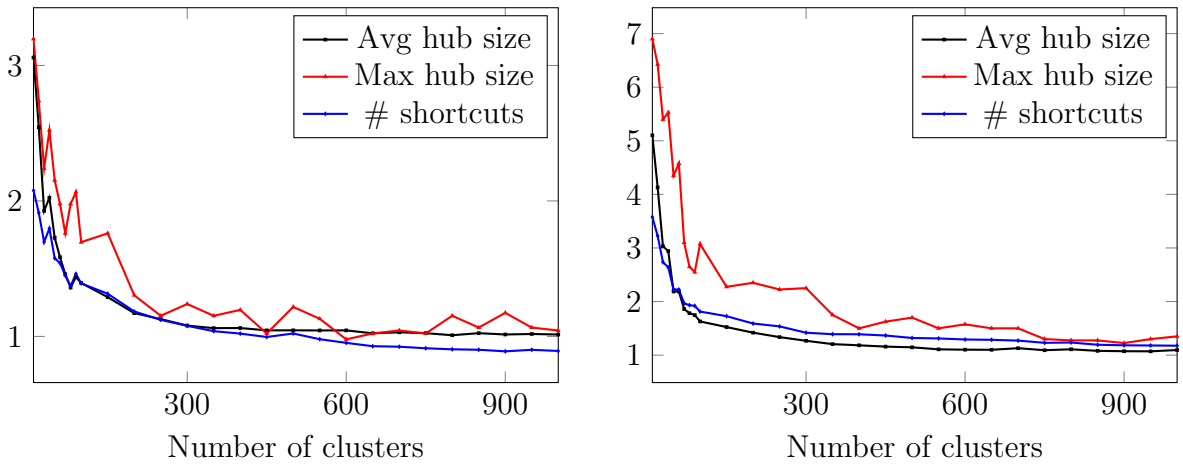


Figure 2.4: Performance of clustering for San Francisco (left) and Luxembourg (right). In the y-axis the quantities are normalized by the cover using all the shortest paths ($k = n$ clusters). For example, the curve for average hub size, say $y_1(k)$, represents that using k clusters the average hub is $y_1(k)$ times bigger than the hubs using all the shortest paths; in the plot we see that, with a few clusters, the size at most doubles and it can be improved by augmenting k. Note that, while the number of short-cuts is an indicator of performance, it is not perfectly correlated with the hub size.

a shortest path cover, a direct approach requires storing all the shortest paths in memory, which, in most mapping applications, is infeasible. To circumvent this, we approximate the shortest-path cover by computing only $k << n$ shortest-path trees and covering these greedily. We use an off-the-shelf clustering method to obtain k cluster centers, from which we compute the shortest paths. As shown in Figure 2.4, the clustering approach provides a very good approximation of the hubs with even a small k. We stress that this is just an easy way to get a ranking; more sophisticated heuristics that decide on-line the next node to contract usually work well in practice [20, 95]. Using another contraction scheme may expedite our algorithms and reduce the hub size.

Depending on the size of our instance, and the specific queries, we work with either the augmented graph G^B or the pruned augmented graph $\tilde{G}^B$. Even though $\tilde{G}^B$ takes time to compute, it can speed up the overall process and yield considerably better hubs. Given a ranking for nodes in G, we contract $\tilde{G}^B$ as follows. Say that V is ordered according to the

ranking, so node 1 is the least important and n the most important. In $\tilde{G}^B$, we first contract the nodes $\langle 1, b\rangle$ for $b = B, \ldots, 0$, then the nodes $\langle 2, b\rangle$ and so on till the nodes $\langle n, b\rangle$ are the last to contract. Finally, when contracting a node v, if u is a predecessor and w a successor of v, we add the short-cut uw only if, by removing v, the distance from u to w is altered, and the new shortest path from u to w is efficient. We can go even further; the short-cut uw is unnecessary if the shortest path is not efficient, even if the distance changes.

To obtain better hubs we prune the results obtained by CH searches. If w is in the forward search of v with distance d, it might be that $\text{dist}(v, w) < d$, this occurs because the search goes only to higher rank nodes and the discovered path is missing some node. When $\text{dist}(v, w) < d$, we can safely remove w from the hub of v, since the highest ranked node in a shortest path will have the correct distance. For frontier queries, we can also prune away a node w if the (v, w)-path has a surplus of budget. The entire process can be summarized in the following steps.

1. Compute the shortest paths in G and use a greedy approach to obtain a cover C
2. Compute the pruned augmented graph $\tilde{G}^B$
3. Contract $\tilde{G}^B$ using the rank induced by C
4. Create hubs $L^+(v), L^-(v)$ using CH
5. Prune the hubs by running HL queries between v and nodes in $L^+(v)$. Run a similar process for $L^-(v)$.

Recall that, for some instances, we skip step 2 and contract G^B instead. Note that in the last step we bootstrap HL to improve it. This works because the fact that some nodes have incorrect distance labels does not impact the correctness of a HL query; a node minimizing the distance is returned and such node must have a correct label.

2.4.2 Experiments

All our experiments were performed on a 64-bit desktop computer with a 3.40GHz Intel Core i7-6700 processor and 16GB RAM running Ubuntu 16.04. Our code is written in Python 2.7. We use the library Networkx for the graph representation and Dijkstra's algorithm. Although all the steps can be parallelized, we did not implement this. Our complete code (implementation and artificial dataset) is publicly available at github.com/albvera/HHL_CSP.

We evaluated the performance of our algorithms with real-world test networks: downtown San Francisco with 2139 nodes and 5697 edges for which real-world travel-time data was available as a Gaussian mixture model [76], and Luxembourg City with 4026 nodes and 9282 edges for which travel-time distributions were synthesized from speed limits [89], as real-world data was unavailable.

In our experiments, we use the mean travel times for our length function and the following cost structure; the top 10% of edges with the highest variance are assigned cost 1 and the rest cost 0. This is a measure of risk, since edges with high variance are prone to cause a delay in the travel time.

Query-time performance

Table 2.1 presents the CSP computation times for different maximum budgets B. For frontier queries, labeled as 'f', the query times are measured as the average of 1000 random s, t. For specific queries, labeled as 's', the times are measured as 1000 random triplets s, t, b. The column for $B = 0$ represents the original graph (without augmentation). As can be seen in the experimental results, our method finds the constrained shortest path solution on average four orders of magnitude faster than running Dijkstra's algorithm on the augmented graph.

B	Prepro [m]	Avg F Size	Avg B Size	Query Dij [ms]	Query HL [ms]
0	1	23	22	10.71	0.005
5-f	5	16	28	80.10	0.02
5-s	5	57	28	31.75	0.01
10-f	9	9	28	168.11	0.03
10-s	10	68	28	56.19	0.01
15-f	12	6	28	237.64	0.03
15-s	16	73	28	77.59	0.01
20-f	17	5	28	342.47	0.03
20-s	20	77	28	100.26	0.01
25-f	22	4	28	460.95	0.03
25-s	25	80	28	126.52	0.01
30-f	26	3	28	569.13	0.03
30-s	31	84	28	152.75	0.01

B	Prepro [m]	Avg F Size	Avg B Size	Query Dij [ms]	Query HL [ms]
0	6	18	18	16.35	0.004
5-f	1	18.0	18.0	175.04	0.02
5-s	21	42.9	18.7	72.03	0.01
10-f	2	18.0	18.0	361.15	0.04
10-s	35	49.3	18.7	102.52	0.01
15-f	3	18.0	18.0	577.89	0.06
15-s	46	53.3	18.7	140.80	0.01
20-f	4	18.0	18.0	821.94	0.07
20-s	60	56.5	18.7	183.11	0.01
25-f	5	18.0	18.0	974.84	0.09
25-s	77	59.5	18.7	227.17	0.01
30-f	7	18	18	1247.72	0.10
30-s	93	62.3	18.7	272.41	0.01

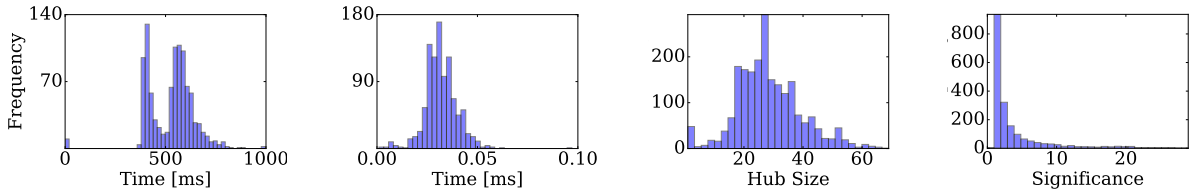
Table 2.1: Experimental results for San Francisco (left) and Luxembourg City (right). Query times are measured with 1000 random s, t pairs for each network and multiple maximum budget levels B. Results on rows $B - f$ correspond to computing the solution frontier for all budgets $b \leq B$ while rows $B - s$ correspond to computing the solution for budget level b.

Preprocessing for frontier queries results in a more compact set of hub labels, since a node $\langle s, b \rangle$ needs to store information for paths with budget exactly equal to b (in case the path is efficient, otherwise it is not stored). On the other hand, for specific queries, $\langle s, b \rangle$ needs to store information for all budgets up to b. The preprocessing time does not include the cover computation, since this is a flat cost of at most the time to pre-process the instance $B = 0$.

Note that preprocessing frontier queries in Luxembourg is faster, despite the network being bigger, this can be explained by the structural properties. For example, in Luxembourg there are more highways and fast roads.

Observe that the *average hub size decreases* in San Francisco for frontier queries, this is because in this instance we use $\tilde{G}$, which prunes away most of the nodes, thus many nodes $\langle v, b \rangle$ are isolated and have empty hubs. The longer preprocessing time for frontier queries can be explained as follows. There are many cases when two nodes are not reachable, to detect this requires Dijkstra to explore the entire graph. In contrast, for specific queries we

add extra edges $\langle s, b\rangle\langle s, b-1\rangle$, hence a reachability test ends, in average, earlier. In the contraction step, we want to remove a node without altering the shortest path, a process that requires many reachability tests.



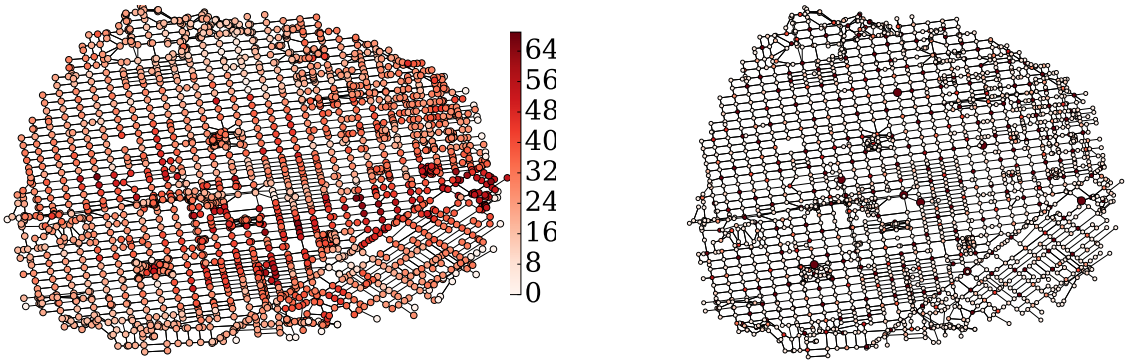
(a) (b) (c) (d)

Figure 2.5: Histogram frontier queries for Dijkstra (a) and HL (b). Times are 1000 random pairs in San Francisco augmented with $B = 25$. Size of reverse hubs (c) and significance (d) for frontier queries in San Francisco augmented with $B = 25$.

Hub sizes and node significance

We focus the analysis on two meaningful quantities. The first is hub size, which is well captured by $|L^-(\langle t, 0\rangle)|$ for $t \in V$. Indeed, for frontier queries the reverse hub is bounding the space requirements; for specific queries the same is true up to a constant factor. For the second quantity, we define the significance of $s \in V$ as the number of hubs containing s, i.e., $\sum_t \sum_b \mathbb{1}_{\{\langle s,b\rangle \in L^-(\langle t,0\rangle)\}}$. Intuitively, a node is highly significant if it belongs to many efficient paths. Figure 2.5 shows a histogram of these metrics.

Fig. 2.6 presents the spatial relationships between the hub size and significance in the San Francisco network. Fig. 2.7 shows the spatial distribution of significance in the Luxembourg network. We observe that highly significant nodes tend to have small hub size. The intuition is simple, if most hubs contain s, then is easier for s to satisfy the cover property with a small hub. Note also that the hub size resembles an harmonic function; nodes are mostly similar to their neighbors.



(a) Heatmap of hub-sizes for SF (b) Heatmap of significance for SF

Figure 2.6: Heat maps for frontier queries in San Francisco augmented with $B = 25$. On the left we see the hub size. Note that the size is not homogeneous, but rather we can observe clusters and neighborhoods tend to be similar. On the right the significance. The most significant nodes have been drawn bigger. The top 3 most significant correspond to Geary Blvd & Gough St, Franklin St & O'Farrel St and Market St & Polk St.

Figure 2.7: Heat map of significance for frontier queries in Luxembourg City augmented with $B = 25$. Notice how the highly significant nodes are in main road crossings.

2.5 Discussion

We introduced a new network primitive, the Constrained Highway Dimension, and used it to parametrize the storage and running time of data structures that support fast CSP queries. Our aim was to study when efficient SP computation yields a similar performance for the harder CSP problem. We derived conditions under which this holds and we can

compare the HD and CHD. For the worst-case setting, the conditions are given by the partial-witness and for average-case by the milder bounded growth. Both conditions have intuitive interpretations in terms of the physical structure of the network.

On the practical side, we validated our findings by developing algorithms that performed four orders of magnitude better on real-world networks, compared to standard techniques. Our work is a first step in bridging the gap between SP and CSP algorithms and we believe our findings are promising for real-world applications.

CHAPTER 3

THE BAYESIAN PROPHET

3.1 Introduction

Everyday life is replete with settings where we have to make decisions while facing uncertainty over future outcomes. Some examples include allocating cloud resources, matching an empty car to a ride-sharing passenger, displaying online ads, selling airline seats, etc. In many of these instances, the underlying arrivals arise from some known generative process. Even when the underlying model is unknown, companies can turn to ever-improving machine learning tools to build predictive models based on past data. This raises a fundamental question in online decision-making: *how can we use predictive models to make good decisions?*

Broadly speaking, an online decision-making problem is defined by a current state and a set of actions, which together determine the next state as well as generate rewards. In Markov decision processes (MDPs), the rewards and state transitions are also affected by some random shock. Optimal policies for such problems are known only in some special cases, when the underlying problem is sufficiently simple, and knowledge of the generative model sufficiently detailed. For many problems of interest, an MDP approach is infeasible due to two reasons: (1) insufficiently detailed models of the generative process of the randomness, and (2) the complexity of computing the optimal policy (the so-called 'curse of dimensionality'). These shortcomings have inspired a long line of work on approximate dynamic programming (ADP).

We focus on two important classes of online decision-making problems: online packing and online matching. In brief, these problems involve a set of d distinct resources, and a principal with some initial budget vector $B \in \mathbb{N}^d$ of these resources, which have to be

allocated among T incoming agents. Each agent has a type comprising of some specific requirements for resources and associated rewards. The exact type becomes known only when the agent arrives. The principal must make irrevocable accept/reject decisions to try and maximize rewards, while obeying the budget constraints.

Online packing and matching problems are fundamental in MDP theory; they have a rich existing literature and widespread applications in many domains. Nevertheless, our work develops new policies for both these problems which admit performance guarantees that are order-wise better than existing approaches. These policies can be stated in classical ADP terms (for example, see Algorithms 2 and 3), but draw inspiration from ideas in Bayesian learning. In particular, our policies can be derived from a meta-algorithm, the Bayes selector (Algorithm 1), which makes use of a black-box *prediction oracle* to obtain statistical information about a chosen offline benchmark, and then acts on this information to make decisions. Such policies are simple to define and implement in practice, and our work provides new tools for bounding their *regret* vìs-a-vìs the offline benchmark. Though we focus on online packing and matching problems, we believe our approach provides a new way for designing and analyzing online decision-making policies using predictive models.

3.1.1 Our Contributions

We believe our contributions in this work are threefold:

1. *Technical*: We present a *new stochastic coupling technique*, which we call the *compensated coupling*, for evaluating the regret of online decision-making policies vis-à-vis offline benchmarks.
2. *Methodological*: Inspired by ideas from Bayesian learning, we propose a class of policies, expressed as the *Bayes Selector*, for general online decision-making problems.

3. *Algorithmic*: For online packing and matching problems, we prove that the Bayes Selector gives expected regret guarantees that are *independent of the size of the state-space*, i.e., constant with respect to the horizon length and budgets.

Organization of the paper: The rest of the paper is organized as follows: First, in Section 3.2, we introduce a general problem, called *Online Allocation*, which includes as special cases the multi-secretary, online packing, online matching, and the more general *choice-over-bundles* settings; we also define the notion of prophet benchmarks, and discuss the shortcoming of prevailing approaches. Next, in Section 3.3, we present our main technical tool, the Compensated Coupling, in the general context of finite-state finite-horizon MDPs. We illustrate the use of our general tool by applying it to the ski-rental problem. In Section 3.3.3 we introduce the Bayes Selector policy, and discuss how the compensated coupling provides a generic recipe for obtaining regret bounds for such a policy. In Sections 3.4 and 3.5, we use these techniques for the online packing and matching problems; we analyse them separately to exploit their structure and obtain stronger results. Finally, in Section 3.6 we analyse the most general problem (Online Allocation). In particular, in Section 3.4, we propose a Bayes Selector policy for online packing and demonstrate the following performance guarantee:

Theorem 3.1.1 (Informal). *For any online packing problem with a finite number of resource types and arrival types, under mild conditions on the arrival process, the Bayes Selector achieves regret which is independent of the horizon T and budgets B (both in expectation and with high probability).*

In more detail, our regret bounds depend on the 'resource matrix' A and the distribution of arriving types, but are independent of T and B. Moreover, the results holds under weak assumptions on the arrival process, including Multinomial and Poisson arrivals, time-dependent processes, and Markovian arrivals. This result generalizes prior and contemporaneous results [94, 78, 37, 108, 10]. We show similar results for matching problems in

Section 3.5.

3.2 Setting and Overview of Results

As we mentioned in the introduction, our contributions in this work are two-fold – (i) we give a technique to analyse the regret of any MDP , and (ii) we apply it to specific problems to obtain constant regret. Our focus in this work is on the subclass of *online packing problems with stochastic inputs*. This is a subclass of the wider class of *finite-horizon online decision-making problems*: given a time horizon $T \in \mathbb{N}$ with discrete time-slots $t = T, T-1, \ldots, 1$, we need to make a decision at each time leading to some cumulative reward. Note that throughout our time index t indicates the *time to go*. We present the details of our technical approach in this more general context whenever possible, indicating additional assumptions when required.

In what follows, we use $[k]$ to indicate the set $\{1, 2, \ldots, k\}$, and denote the (i, j)-th entry of any given matrix A interchangeably by $A_{i,j}$ or $A(i, j)$. We work in an underlying probability space $(\Omega, \mathcal{F}, \mathbb{P})$, and the complement of any event $Q \subseteq \Omega$ is denoted $\bar{Q}$. For any optimization problem (P), we use $v(P)$ to indicate its objective value. If S is a finite set, $|S|$ denotes cardinality. The set $\mathbb{N}$ of naturals includes zero.

3.2.1 Online Allocation Problem

We present a problem, called *Online Allocation*, that encompasses both online matching and online packing. The setup is as follows: There are d distinct resource-types denoted by the set $[d]$, and at time $t = T$, we have an initial availability (budget) vector $B = (B_1, B_2, \ldots, B_d) \in \mathbb{N}^d$. At every time $t = T, T-1, \ldots, 1$, nature draws an arrival with *type* ξ^t from a finite set

of n distinct types $\mathcal{J} = [n]$, via some distribution which is known to the algorithm designer (or principal). We denote $Z(t) = (Z_1(t), Z_2(t), \ldots, Z_n(t)) \in \mathbb{N}^n$ as the cumulative vector of the last t arrivals, where $Z_j(t) \coloneqq \sum_{\tau \leq t} \mathbb{1}_{\{\xi^\tau = j\}}$.

An arrival of type j has associated a family of multisets (bags) $S_j \subseteq 2^{[d]}$, where each $s \in S_j$ is called a *bundle*. We can allocate any bundle $s \in S_j$ to a type j customer, which generates a reward r_{sj} and consumes one unit of each resource $i \in s$. Observe that we do not assume additive valuations, e.g, we do not require $r_{\{1,2\}j} = r_{\{1\}j} + r_{\{2\}j}$.

At each time, the principal must decide whether to allocate a bundle to the request ξ^t (thereby generating the associated reward while consuming the required resources), or reject it (no reward and no resource consumption). Allocating a bundle requires that there is sufficient budget of each resource to cover the request. The principal's aim is to make irrevocable decisions so as to maximize overall rewards.

We present in Section 3.6 results for this general problem. For concreteness, we describe now three particular cases, each of independent interest. We analyse these three cases separately because we can get improved results over the general case.

Multi-Secretary. This is a fundamental one dimensional instance. In this problem, we have $B \in \mathbb{N}$ available positions and want to hire employees with the highest abilities (rewards). There is one resource type ($d = 1$) with budget B; each employee occupies one unit of budget (one position). This is an online allocation problem with $S_j = \{\{1\}\}$ for all j (all candidates want the same resource) and rewards $r_{\{1\}j} = r_j$.

Online Packing. In this multidimensional problem, each request j is associated to one bundle and one reward if allocated. Specifically, we are given a consumption matrix $A \in \mathbb{N}^{d \times n}$, where a_{ij} denotes the units of resource i required to serve the request j and, for each j, there is a reward r_j if served. This is an online allocation problem with one bundle for j:

$S_j = \{\{a_{ij} \text{ copies of } i : i \in [d]\}\}$.

Online Matching. There are d resources, but now each type j wants *any option from the given set* of resources, i.e., type j wants any from A_j instead of all from A_j. The types can be represented by a reward matrix $r \in \mathbb{R}_{\geq 0}^{d \times n}$ and adjacency matrix $A \in \{0,1\}^{d \times n}$; if the arrival is of type $j \in [n]$, we can allocate at most one resource i such that $a_{ij} = 1$, leading to a reward of r_{ij}. This problem can be thought as online bipartite matching, see Section 3.5 for details. It corresponds to an online allocation problem with bundles $S_j = \{\{i\} : i \in [d] \text{ s.t. } a_{ij} = 1\}$ and rewards $r_{\{i\}j} = r_{ij}$.

Arrival Processes: To specify the generative model for the type sequence $\xi^T, \xi^{T-1}, \ldots, \xi^1$, an important subclass is that of *stationary independent* arrivals, which further admits two widely-studied cases:

1. The Multinomial process is defined by a known distribution $p \in \mathbb{R}_{\geq 0}^n$ over $[n]$; at each time, the arrival is of type j with probability p_j, thus $Z(t) \sim \text{Multinomial}(t, p_1, \ldots, p_n)$.

2. The Poisson arrival process is characterized by a known rate vector $\lambda \in \mathbb{R}_{\geq 0}^n$. Arrivals of each class are assumed to be independent such that $Z_j(t) \sim \text{Poisson}(\lambda_j t)$. Note that, although this is a continuous-time process, it can be accommodated in a discrete-time formulation by defining as many periods as arrivals (see Appendix B.2.1 for details).

We assume w.l.o.g. that $p_j > 0$ and $\lambda_j > 0$ for all $j \in [n]$ (if this is not the case for some j, that type never arrives and can be removed from the instance description). More general models allow for non-stationary and/or correlated arrival processes – for example, non-homogeneous Poisson processes, Markovian models (see Example 3.4.12), etc. An important feature of our framework is that it is capable of handling a wide variety of such processes in a unified manner, without requiring extensive information regarding the generative model. We discuss the most general assumptions we make on the arrival process in Section 3.4.2.

3.2.2 The Benchmark

Suppose a given problem is simultaneously solved by two '*agents*', Online and Offline, who are primarily differentiated based on their access to information. Online can only take *non-anticipatory* actions, i.e., use available information only, whereas Offline is allowed to make decisions with full knowledge of future arrivals. This is known in the literature as a *prophet or full-information benchmark*. Denoting the total collected rewards of Offline and Online as v^{off} and v^{on} respectively, we define the *regret* to be the *additive* loss $\text{Reg} := v^{\text{off}} - v^{\text{on}}$. Observe that v^{on} depends on the policy used by Online, the underlying policy will always be clear from context. Our aim is to design policies with low $\mathbb{E}[\text{Reg}]$.

For online packing, the solution to Offline's problem corresponds to solving an integer programming problem. A looser, but more tractable benchmark, is given by an LP relaxation of this policy: given arrivals vector $Z(T)$, we assume Offline solves the following:

$$\begin{aligned} P[Z(T), B]: \quad \max \quad & r'x \\ \text{s.t.} \quad & Ax \leq B \\ & x \leq Z(T) \\ & x \geq 0. \end{aligned} \tag{3.1}$$

The fluid problem must be relinquished: The most common technique for obtaining online packing policies is based on the so-called fluid (a.k.a. deterministic or ex ante) LP benchmark ($P[\mathbb{E}[Z(T)], B]$), where (P) is defined in Eq. (3.1). It is easy to see via Jensen's Inequality that $v(P[\mathbb{E}[Z(T)], B]) \geq \mathbb{E}[v(P[Z(T), B])]$, and hence the fluid LP is an upper bound for any online policy. Although the use of this fluid benchmark is the prevalent tool to bound the regret in online packing problems [103, 94, 78, 108], the following result shows that the approach of using $v(P[\mathbb{E}[Z(T)], B])$ as a benchmark can never lead to a constant expected regret policy, as the fluid benchmark can be far off from the optimal solution in hindsight.

Proposition 3.2.1. *For any online packing problem, if the arrival process satisfies the Central Limit Theorem and the fluid LP is dual degenerate, i.e., the optimal dual variables are not unique, then* $v(P[\mathbb{E}[Z(T)], B]) - \mathbb{E}[v(P[Z(T), B])] = \Omega(\sqrt{T})$.

This gap has been reported in literature, both informally and formally (see [10, 37]). For completeness we provide a proof in Appendix B.1. Note though that this gap does not pose a barrier to showing constant-factor competitive ratio guarantees, i.e. $O(T)$ expected regret; the fluid LP benchmark is widely used for prophet inequalities. In contrast, the gap presents a barrier for obtaining $O(1)$ expected regret bounds. Breaking this barrier thus requires a fundamentally new approach.

3.2.3 Overview of our Results

Our approach can be viewed as a meta-algorithm that uses black-box prediction oracles to make decisions. The quantities estimated by the oracles are related to our offline benchmark and can be interpreted as *probabilities of regretting each particular action in hindsight*. A natural 'Bayesian selection' strategy given such estimators is to adopt the action that is least likely to cause regret in hindsight. This is precisely what we do in Algorithm 1, and hence, we refer to it as the Bayes Selector policy.

We note that Bayesian selection techniques are not new. In fact, they are often used as heuristics in practice. Our work however shows that such policies in fact have excellent performance in such settings – in particular, we show that for matching and packing problems:

1. There are easy to compute estimators (in particular, ones which are based on simple adaptive LP relaxations) that, when used for Algorithm 1, give constant expected

regret for a wide range of distributions (see Theorems 3.4.2, 3.4.6 and 3.5.1).

2. Using other types of estimators for Algorithm 1 yields comparable performance guarantees (see Corollaries 3.4.7 and 3.5.2). This holds, for example, if the estimations are obtained through sampling.

At the core of our analysis is a *novel stochastic coupling technique* for analyzing online policies based on offline (or *prophet*) benchmarks. In particular, unlike traditional approaches to regret analysis, which are based on showing that an online policy tracks a fixed offline policy, our approach is instead based on *forcing* Offline *to follow* Online*'s actions*. We describe this in more detail in the next section.

3.2.4 Related Work

Our work is related to several active areas of research in MDPs and online algorithms. We now briefly survey some of the most relevant connections.

Approximate Dynamic Programming: The complexity of computing optimal MDP solutions can scale with the state space, which often makes it impractical (the so-called 'curse of dimensionality' [93]). This has inspired a long line of work on *approximate dynamic programming* (ADP) [93, 104] to develop lower complexity heuristics. Although these methods often work well in practice, they require careful choice of basis functions, and any bounds are usually in terms of quantities which are difficult to interpret. Our work provides an alternate framework, which is simpler and has interpretable guarantees.

Model Predictive Control: Another popular heuristic for ADP and control which is closer to our paradigm is that of *model predictive control* (or receding horizon control) [88, 25, 43], which is a widely-used heuristic in practice. Recently, MPC techniques have also been

connected with online convex optimization (OCO) [75, 39, 38] to show how prediction oracles can be used for OCO, and applying these policies to problems in power systems and network control. These techniques however generally require continuous controls, and do not handle combinatorial constraints.

Information Relaxation: Parallel to the ADP focus on developing better heuristics, there is a line of work on deriving upper bounds via martingale duality, sometimes referred to as information relaxations [30, 50, 31]. The main idea in these works is to obtain performance bounds for heuristic policies work by defining more refined outer bounds; in particular, this can be done by adding a suitable martingale term to the current reward, in order to penalize 'future information'. Our approach follows a similar construction of bounds via offline benchmarks; however, a critical difference is that instead of using these to analyze a given heuristic, we use the benchmarks directly to derive control policies.

Online Packing and Prophet Inequalities: The majority of work focuses on competitive ratio bounds under worst-case distributions. In particular, there is an extensive literature on the so-called Prophet Inequalities, starting with the pioneering work of [73], to more recent extensions and applications to algorithmic economics [80, 55, 47, 9]. We note that any competitive ratio guarantee implies a $O(T)$ expected regret, in comparison to our $O(1)$ expected regret guarantees – the cost for this, however, is that our results hold under more restrictive assumptions on the inputs. For example, the policy suggested by [55] is static and [10, Theorem 1] shows that any static policy has $\Omega(\sqrt{T})$ expected regret, hence it cannot yield a strong guarantee like ours.

Distribution-agnostic and Adversarial Models: Though we focus only on the case where the input is drawn from a stochastic process, we note that there is a long line of work on online packing with adversarial inputs [35, 36, 79], and also distribution-agnostic approaches [15, 51]. More generally, there is a large body of work on using sublinear expected

regret algorithms for solving online linear and convex programs [7, 68]. The algorithms in these works are incomparable to ours in that, while they cannot get constant expected regret in our setting (stochastic input, finite type space), they provide guarantees under much weaker assumptions.

Regret bounds in Stochastic Online Packing: For these problems, regret is the most meaningful metric to study, see [110] for a discussion, where approximations to the regret are studied. The first work to prove constant expected regret in a context similar to ours is [10], who prove a similar result for the multi-secretary setting with multinomial arrivals; we strengthen their result in Theorem 3.4.2. This result is relevant to a long line of work in applied probability. Some influential works are [94], which provides an asymptotically optimal policy under the diffusion scaling, and [78] who provide a resolving policy with constant expected regret under a certain non-degeneracy condition. In contrast, degeneracy plays no role in the performance of our algorithms. More recently, [37] partially extended the result of [10] for more general packing problems; their guarantee is only valid for i.i.d. Poisson arrival processes and when the system is scaled linearly, i.e., when B is proportional to T (our results and [10] make no such assumption). In Section 3.7, we demonstrate with a numerical study that the Bayes Selector outperforms all the previous policies.

3.3 Compensated Coupling and the Bayes Selector

We introduce our two main technical ideas: 1. the *compensated coupling* technique, and 2. the *Bayes selector* heuristic for online decision-making. We remark that the techniques introduced in this section are valid for a generic MDP. In subsequent sections we apply them to the variants of Online Allocation defined in Section 3.2.1.

3.3.1 MDPs and Offline Benchmarks

The basic MDP setup is as follows: at each time $t = T, T-1, \ldots, 1$ (where t represents the time-to-go), based on previous decisions, the system *state* is one of a set of possible states S^t. Next, nature generates an *arrival* $\xi^t \in \mathcal{J}$, following which we need to choose from a set of available *actions* $\mathcal{U}$. The state updates and rewards are determined via a transition function $\mathcal{T} : \mathcal{U} \times S^t \times \mathcal{J} \to S^t$ and a reward function $\mathcal{R} : \mathcal{U} \times S^t \times \mathcal{J} \to \mathbb{R}$: for current state $s \in S^t$, arrival $j \in \mathcal{J}$ and action $a \in \mathcal{U}$, we transition to the state $\mathcal{T}(a, s, j)$ and collect a reward $\mathcal{R}(a, s, j)$. Infeasible actions a for a given state s correspond to $\mathcal{R}(a, s, j) = -\infty$. The sets $\mathcal{U}, S^t, \mathcal{J}$, as well as the measure over arrival process $\{\xi^t : t \in [T]\}$, are known in advance. Finally, though we focus mainly on maximizing rewards, the formalism naturally ports over to cost-minimization.

Recall that we adopt the view that the problem is simultaneously solved by two '*agents*': Online and Offline. Online can only take *non-anticipatory* actions while Offline makes decisions with knowledge of future arrivals. To keep the notation simple, we restrict ourselves to deterministic policies for Offline and Online, thereby implying that the only source of randomness is due to the arrival process (our results can be extended to randomized policies).

A sample path $\omega \in \Omega$ encodes the arrival sequence $\{\xi^t : t \in [T]\}$. In other words, there exists a unique sequence of types that is consistent with ω; whenever we fix ω, the type ξ^t is uniquely identified, but, for notational ease, we do not write $\xi^t[\omega]$. For a given sample-path $\omega \in \Omega$ and *time t to go*, Offline's value function is specified via the *deterministic* Bellman equations

$$v^{\text{off}}(t, s)[\omega] \coloneqq \max_{a \in \mathcal{U}} \{\mathcal{R}(a, s, \xi^t) + v^{\text{off}}(t-1, \mathcal{T}(a, s, \xi^t))[\omega]\}, \tag{3.2}$$

with boundary condition $v^{\text{off}}(0, s)[\omega] = 0$ for all $s \in S^t$. The notation $v^{\text{off}}(t, s)[\omega]$ is used to emphasise that, given sample-path ω, Offline's value function is a deterministic function

of t and s.

We require that the DP formulation in Eq. (3.2) is well defined. For simplicity, we enforce this with the following assumption: there are some constants $c_1, c_2 \geq 0$ such that $-c_1 \leq \max_{a \in \mathcal{U}} \mathcal{R}(a, s, j) \leq c_2$ for all $s \in S^t, j \in \mathcal{J}$. In other words, every state has a feasible action and the maximum reward is uniformly bounded and attained. The spaces $S^t, \mathcal{J}, \mathcal{U}$ and functions $\mathcal{T}, \mathcal{R}$ are otherwise arbitrary. We enforce this assumption for clarity of exposition, but we observe that it can be further generalized, c.f. [23, Volume II, Appendix A].

On the other hand, ONLINE chooses actions based on *policy* π^{on} defined as follows:

Definition 3.3.1 (Online Policy). An online policy π^{on} is any collection of functions $\{\pi^{\text{on}}(t, s, j) : t \in [T], s \in S^t, j \in \mathcal{J}\}$ such that, if at time t the current state is s and a type j arrives, then ONLINE chooses the action $\pi^{\text{on}}(t, s, j) \in \mathcal{U}$. The function $\pi^{\text{on}}(t, \cdot, \cdot)$ can depend only on $\{\xi^T, \cdots, \xi^t\}$, i.e., on the randomness observed at periods $\tau \geq t$ (the history).

Let us denote $\{S^t : t \in [T]\}$ as ONLINE's state over time, i.e., the stochastic process $S^t \in S^t$ that results from following a given policy π^{on}. We can write ONLINE's accrued value, for a given policy π^{on}, as

$$v^{\text{on}}(t, S^t)[\omega] \coloneqq \sum_{\tau \leq t} \mathcal{R}(\pi^{\text{on}}(\tau, S^\tau, \xi^\tau), S^\tau, \xi^\tau)[\omega]$$

For notational ease, we omit explicit indexing of v^{on} on policy π^{on}.

On any sample-path ω, we can define the *regret* of an online policy to be the *additive* loss incurred by ONLINE using π^{on} w.r.t. OFFLINE, i.e.,

$$\text{REG}[\omega] \coloneqq v^{\text{off}}(T, S^T)[\omega] - v^{\text{on}}(T, S^T)[\omega]$$

Remark 3.3.2 (Regret is Agnostic of Offline Algorithm). Our previous definition of REG depends only on the online policy π^{on}, but it does not depend on the policy (or

algorithm) used by OFFLINE as long as it is optimal. For example, in the case where there are multiple maximizers in the Bellman Eq. (3.2), different tie-breaking rules for OFFLINE yield different algorithms, but all of them are optimal and have the same optimal value v^{off}.

3.3.2 Compensated Coupling

At a high-level, the compensated-coupling is a sample path-wise charging scheme, wherein we try to couple the trajectory of a given policy to a sequence of offline policies. Given any non-anticipatory policy (played by ONLINE), the technique works by making OFFLINE *follow* ONLINE – formally, we couple the actions of OFFLINE to those of ONLINE, while compensating OFFLINE to preserve its collected value along every sample-path.

Example 3.3.3. Consider the multi-secretary problem with budget $B = 1$ and three arriving types $\mathcal{J} = \{1, 2, 3\}$ with $r_1 > r_2 > r_3$. The state space in this problem is $S^t = \mathbb{N}$ and the action space is $\mathcal{U} = \{\text{accept}, \text{reject}\}$. Suppose for $T = 4$ the arrivals on a given sample-path are $(\xi^4, \xi^3, \xi^2, \xi^1) = (1, 2, 1, 3)$. Note that OFFLINE will accept exactly one arrival of type 1, but is indifferent to which arrival. While analyzing ONLINE, we have the freedom to choose a benchmark by specifying the tie-breaking rule for OFFLINE– for example, we can compare ONLINE to an OFFLINE agent who chooses to *front-load* the decision by accepting the arrival at $t = 4$ (i.e., as early in the sequence as possible) or *back-load* it by accepting the arrival at $t = 2$. In conclusion, for this sample path, the following two sequences of actions are optimal for OFFLINE: (accept,reject,reject,reject) and (reject,reject,accept,reject).

Suppose instead that we choose to reject the first arrival ($t = 4$), and then want OFFLINE to accept the type-2 arrival at $t = 3$ – this would lead to a decrease in OFFLINE's final reward. The crucial observation is that we can still *incentivize* OFFLINE to accept arrival type 2 by offering a *compensation* (i.e., additional reward) of $r_1 - r_2$ for doing so. The basic idea

behind the compensated coupling is to generalize this argument. We want OFFLINE to take ONLINE's action, hence we couple the states of OFFLINE and ONLINE with the use of compensations.

Let us start now with a general problem. Given sample-path $\omega \in \Omega$ with arrivals $\{\xi^t[\omega] : t \in [T]\}$, recall $v^{\text{off}}(t,s)[\omega]$ denotes OFFLINE's value starting from state s with t periods to go. $v^{\text{off}}(t,s)[\omega]$ obeys the Bellman Eq. (3.2). The following definition is about the actions satisfying said Bellman Equations.

Definition 3.3.4 (Satisfying Action). Fix $\omega \in \Omega$. For any given state s and time t, we say OFFLINE is *satisfied with an action* a at (s,t) if a is a maximizer in the Bellman equation, i.e.,

$$a \in \text{argmax}_{\hat{a} \in \mathcal{U}} \left\{ \mathcal{R}(\hat{a}, s, \xi^t) + v^{\text{off}}(t-1, \mathcal{T}(\hat{a}, s, \xi^t))[\omega] \right\}.$$

Observe that a may be satisfying for a sample path ω and not for some other ω'; once the sample path is fixed, satisfying actions are unequivocally identified.

Example 3.3.5. Consider the multi-secretary problem with $T = 5$, initial budget $B = 2$, types $\mathcal{J} = \{1,2,3\}$ with $r_1 > r_2 > r_3$, and a particular sequence of arrivals $(\xi^5, \xi^4, \xi^3, \xi^2, \xi^1) = (2,3,1,2,3)$. The optimal value of OFFLINE is $r_1 + r_2$, and this is achieved by accepting the sole type-1 arrival as well as any one out of the two type-2 arrivals. At time $t = 5$, OFFLINE is satisfied either accepting or rejecting ξ^5. Further, at $t = 3$, for any budget $b > 0$ the only satisfying action is to accept.

With the notion of satisfying actions, we can create a coupling as illustrated in Fig. 3.1. Although OFFLINE may be satisfied with multiple actions (see above example and Remark 3.3.2), its value remains unchanged under any satisfying action, i.e., any tie-breaking rule. We define a valid policy π^{off} for OFFLINE to be any *anticipatory functional* such that, for every $\omega \in \Omega$, we have a different mapping to actions. Formally, for every $\omega \in \Omega$,

$\pi^{\text{off}}[\omega] : [T] \times S^t \times \mathcal{J} \to \mathcal{U}$ is a function satisfying the optimality principle:

$$v^{\text{off}}(t,s)[\omega] = v^{\text{off}}(t-1, \mathcal{T}(\pi^{\text{off}}(t,s,\xi^t)[\omega], s, \xi^t))[\omega] + \mathcal{R}(\pi^{\text{off}}(t,s,\xi^t)[\omega], s, \xi^t), \quad \forall t \in [T], s \in S^t.$$

Next, we quantify by how much we need to compensate Offline when Online's action is not satisfying, as follows

Definition 3.3.6 (Marginal Compensation). For action $a \in \mathcal{U}$, time $t \in [T]$ and state $s \in S^t$, we denote the random variable $\bar{R}$ and scalar $\bar{r}$

$$\bar{R}(t,a,s) \coloneqq v^{\text{off}}(t,s) - [v^{\text{off}}(t-1, \mathcal{T}(a,s,\xi^t)) + \mathcal{R}(a,s,\xi^t)]$$
$$\bar{r}(a,j) \coloneqq \max\big\{\bar{R}(t,a,s)[\omega] : t \in [T], s \in S^t, \omega \in \Omega \text{ s.t. } \xi^t[\omega] = j\big\}.$$

The random variable $\bar{R}$ captures exactly how much we need to compensate Offline, while $\bar{r}(a,j)$ provides a uniform (over s,t) bound on the compensation required when Online errs on an arrival of type j by choosing an action a. Though there are several ways of bounding $\bar{R}(t,a,s)$, we choose $\bar{r}(a,j)$ as it is clean, expressive, and admits good bounds in many problems as the next example shows.

Example 3.3.7. For online packing problems define $r_\varphi \coloneqq \max_{j\in[n]} r_j$ as the maximum reward over all classes. The state space in this problem is $S^t = \mathbb{N}^d$ and the actions space is $\mathcal{U} = \{\text{accept}, \text{reject}\}$. Also, for simplicity, we assume that all resource requirements are binary, i.e., $a_{ij} \in \{0,1\} \,\forall\, i \in [d], j \in [n]$. For a given sample-path $\omega \in \Omega$ and any given budget $b \in \mathbb{N}^d$, if Offline decides to accept the arrival at t, we can instead make it reject the arrival while still earning a greater or equal reward by paying a compensation of r_φ. On the other hand, note that Offline can at most extract r_φ in the future for every resource ξ^t uses; hence on sample-paths where Offline wants to reject ξ^t, it can be made to accept ξ^t instead with a compensation of $||A_{\xi^t}||_1 r_\varphi \leq d r_\varphi$. In conclusion, we have $r_j \leq \bar{r}(a,j) \leq d r_\varphi$.

Recall that S^t denotes the random process of ONLINE's state. Additionally, we denote $R^{\text{on}}(t, S^t)[\omega] \coloneqq \mathcal{R}(\pi^{\text{on}}(t, S^t, \theta^t), S^t, \xi^t)$ as the reward collected by ONLINE at time t, and hence $v^{\text{on}}(T, S^T)[\omega] = \sum_{t\in[T]} R^{\text{on}}(t, S^t)[\omega]$.

The final step is to fix OFFLINE's policy to be one which 'follows ONLINE' as closely as possible. For this, given a policy π^{on}, on any sample-path ω we set $\pi^{\text{off}}(t, s, \xi^t)[\omega] = \pi^{\text{on}}(t, s, \xi^t)[\omega]$ if $\pi^{\text{on}}(t, s, \xi^t)[\omega]$ is satisfying, and otherwise set $\pi^{\text{off}}(t, s, \xi^t)[\omega]$ to an arbitrary satisfying action. In other words, we start with any valid policy π^{off} and, for every $\omega \in \Omega$, we modify it as described to obtain another valid policy. Abusing notation, we still call π^{off} this modified policy. Recall that this modification does not change the regret guarantees (see Remark 3.3.2).

Definition 3.3.8 (Disagreement Set). For any state s and time t, and any action $a \in \mathcal{U}$, we define the *disagreement set* $Q(t, a, s)$ to be the set of sample-paths where a is not satisfying for OFFLINE, i.e.,

$$Q(t, a, s) \coloneqq \left\{\omega \in \Omega : v^{\text{off}}(t, s)[\omega] > \mathcal{R}(a, s, \xi^t) + v^{\text{off}}(t-1, \mathcal{T}(a, s, \xi^t))[\omega]\right\}.$$

Finally, let $Q(t, s) \subseteq \Omega$ be the event when OFFLINE cannot follow ONLINE, i.e., $Q(t, s) \coloneqq Q(t, \pi^{\text{on}}(t, s, \xi^t), s)$. Note that $Q(t, s)$ depends on π^{on}, but we omit the indexing since π^{on} is clear from context. *Only under $Q(t, s)$ we need to compensate* OFFLINE, hence we obtain the following.

Lemma 3.3.9 (Compensated Coupling). *For any online decision-making problem, fix any* ONLINE *policy π^{on} with resulting state process S^t. Then we have:*

$$\text{REG}[\omega] = \sum_{t\in[T]} \bar{R}(t, \pi^{on}(t, S^t, \xi^t), S^t)[\omega] \cdot \mathbb{1}_{Q(t,S^t)}[\omega],$$

and thus $\mathbb{E}[\text{REG}] \le \max_{a\in\mathcal{U}, j\in\mathcal{J}} \{\bar{r}(a, j)\} \cdot \sum_{t\in[T]} \mathbb{E}[\mathbb{P}[Q(t, S^t)|S^t]]$.

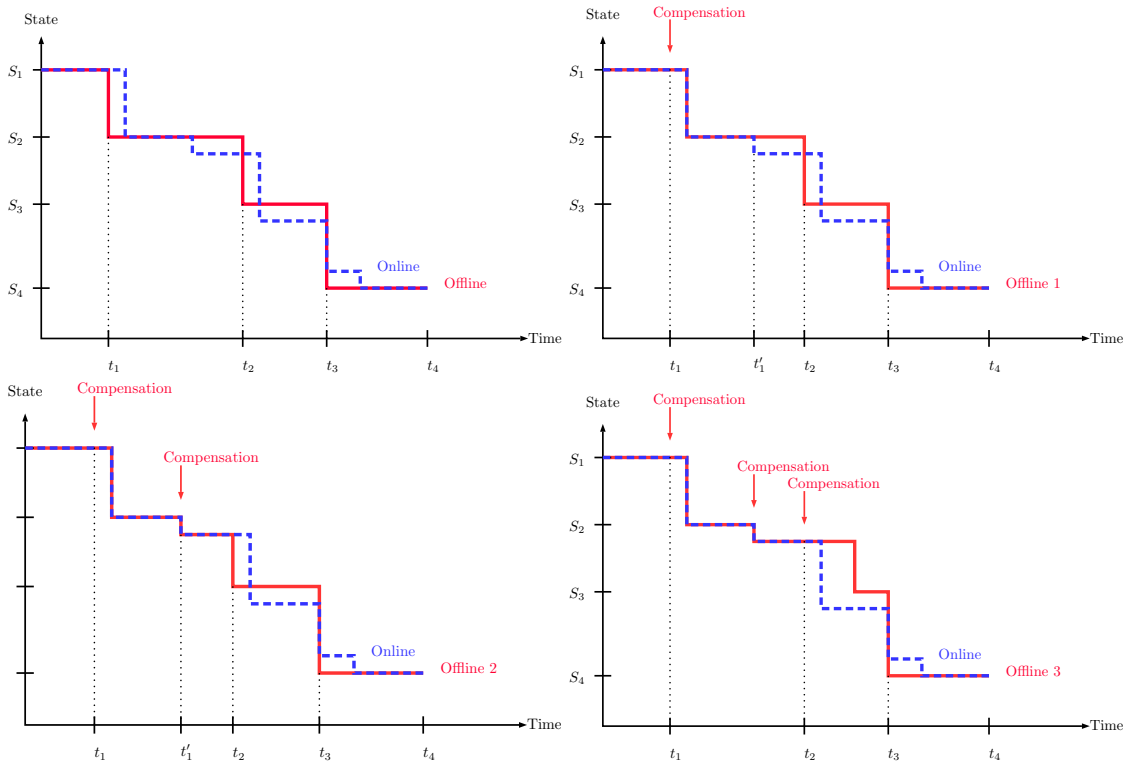


Figure 3.1: The top-left image shows the traditional approach to regret analysis, wherein one considers a fixed offline policy (which here corresponds to a fixed trajectory characterized by accept decisions at $t_1, t_2, t_3, \ldots$) and tries to bound the loss due to "Online oscillating around Offline". In contrast, the compensated coupling approach compares Online to an Offline policy which changes over time. This leads to a sequence of offline trajectories (top-right, bottom-left, and bottom-right), each "agreeing" more with Online. In particular, Offline is not satisfied with Online's action at t_1 (leading to divergent trajectories in the top-left figure), but is made to follow Online by paying a compensation (top-right), resulting in a new Offline trajectory, and a new disagreement at $t'_1 \in (t_1, t_2)$. This coupling process is repeated at time t'_1 (bottom-left), and then at t_2 (bottom-right), each time leading to a new future trajectory for Offline. Coupling the two processes helps simplify the analysis as we now need to study a single trajectory (that of Online), as opposed to all potential Offline trajectories.

PROOF. We stress that, throughout, S^t denotes ONLINE's state. We claim that, for every time t,

$$v^{\text{off}}(t, S^t)[\omega] - v^{\text{off}}(t-1, S^{t-1})[\omega] = R^{\text{on}}(t, S^t)[\omega] + \bar{R}(t, \pi^{\text{on}}(t, S^t, \xi^t), S^t)[\omega] \mathbb{1}_{Q(t,S^t)}[\omega]. \quad (3.3)$$

To see this, let $a = \pi^{\text{on}}(t, S^t, \xi^t)$. If OFFLINE is satisfied taking action a in state S^t, then $v^{\text{off}}(t, S^t)[\omega] - v^{\text{off}}(t-1, S^{t-1})[\omega] = R^{\text{on}}(t, S^t)[\omega]$. On the other hand, if OFFLINE is not satisfied taking action a, then by the definition of marginal compensation (Definition 3.3.6) we have, $v^{\text{off}}(t, S^t)[\omega] - v^{\text{off}}(t-1, \mathcal{T}(a, S^t, \xi^t))[\omega] = \bar{R}(t, a, S^t)[\omega] + \mathcal{R}(a, S^t, \xi^t)$. Since by definition $\mathcal{T}(a, S^t, \xi^t) = S^{t-1}$ and $\mathcal{R}(a, S^t, \xi^t) = R^{\text{on}}(t, S^t)$, we obtain Eq. (3.3). Finally, our first result follows by telescoping the summands and the second by linearity of expectation. □

We list a series of remarks.

- Lemma 3.3.9 is a sample-path property that makes no reference to the arrival process. Though we use it primarily for analyzing MDPs, it can also be used for adversarial settings. We do not further explore this, but believe it is a promising avenue.
- For stochastic arrivals, the regret depends on $\mathbb{E}\left[\sum_{t\in[T]} \mathbb{P}[Q(t, S^t)]\right]$; it follows that, if the disagreement probabilities are summable over all t, then the expected regret is constant. In Sections 3.4 and 3.5 we show how to bound $\mathbb{P}[Q(t, S^t)]$ for different problems.
- The first part of Lemma 3.3.9 provides a distributional characterization of the regret in terms of a weighted sum of Bernoulli variables. This allows us to get high-probability bounds in Section 3.4.3.
- Lemma 3.3.9 gives a tractable way of bounding the regret which does not require either reasoning about the past decisions of ONLINE, or the complicated process OFFLINE may follow. In particular, it suffices to bound $\mathbb{P}[Q(t, S^t)]$, i.e., the probability that, *given* state S^t at time t, OFFLINE loses optimality in trying to follow ONLINE.

- As mentioned before, Lemma 3.3.9 extends to the full generality of MDPs. Indeed, from [72, chapter 11], it follows that any MDP with random transitions and random rewards can be simulated by the family of MDPs we study here; since the inputs ξ^t are allowed to be random, we can define random transitions and rewards based on ξ^t, see [72] for further details.

Lemma 3.3.9 thus gives a generic tool for obtaining regret bounds against the offline optimum for any online policy. Note also that the compensated coupling argument generalises to settings where the transition and reward functions are time dependent, the policies are random, etc. The compensated coupling also suggests a natural greedy policy, which we define next.

Comparison to traditional approaches. As discussed in related work, there are two main approaches. First, the fluid (or ex ante) benchmark, which can be understood as competing against a fixed value. This is the prevailing technique in competitive analysis [9] and the online packing literature [78, 94, 108]. We showed in Proposition 3.2.1 that such an approach cannot yield better than $O(\sqrt{T})$ regret bounds, while we prove $O(1)$. Second, the traditional sample path approach, which competes against the random trajectory of Offline (as illustrated in Fig. 3.1), is based on showing that Online is "close" to a fixed trajectory. This approach is capable of obtaining strong $O(1)$ guarantees [10], but it is highly involved since it necessitates a complete characterisation of Offline's trajectory. The benefit of our approach is abstracting away from the characterization of Offline's trajectory and focusing only on Online's (which is the one that the algorithm controls), while yielding strong $O(1)$ guarantees.

The next example illustrates the Compensated Coupling in a different setting. We consider the Ski Rental problem, which is a well studied minimum cost covering (not packing) problem.

Example 3.3.10 (Ski Rental). Given T days for skiing, each day we decide whether to buy skis for b dollars or to rent them for 1 dollar. The snow may melt any day and we have a distribution over the period we may be able to ski, i.e., there is snow during the first $X \in [T]$ periods and we know the distribution of X. Our aim is to explicitly write the regret of a particular policy (stated later) using the Compensated Coupling.

The optimal offline solution is trivial: if $X < b$, Offline rents every day, otherwise ($X \geq b$) Offline buys the first day. In other words, Offline either buys the first day (which has cost b), or rents every day, with cost X. Since Offline knows X, he picks the minimum.

We map this problem to our framework as follows. The state space is $S^t = \{\text{skis}, \text{no-skis}\}$, where 'skis' means we own the skis. Arrivals are signals $\xi^t \in \{0, 1\}$, where 1 means there is snow and 0 that the season is over. The arrival sequence is always of the form $(\xi^T, \ldots, \xi^1) = (1, \ldots, 1, 0, \ldots, 0)$, where $X = \sum_{t \in [T]} \xi^t$ by definition. Finally, rewards are -1 per day if we rent and $-b$ when we buy.

The compensations are as follows. If the state is 'skis' or if $\xi^t = 0$ (season is over), then no compensation is needed, because we know that Offline does nothing w.p. 1 (either he owns the skis or the problem ended). The only case where Online and Offline may disagree is when $\xi^t = 1$ (can ski today) and the state is 'no-skis'.

Let us denote X^t as the number of remaining skiing days (including t) and say we observe $\xi^t = 1$ at time t. Offline is not satisfied renting if $X^t > b$; forcing him to rent in this event requires a compensation of 1. On the other hand, Offline is not satisfied buying if $X^t < b$; forcing him needs a compensation of $b - X^t$. Consider the following policy π^{on}: for fixed $\tau \geq 0$, rent the first τ days and buy on day $\tau + 1$ contingent on seeing snow all these days, i.e., contingent on $\xi^t = 1$ for all $t = T, \ldots, T - \tau$. The Compensated Coupling (Lemma 3.3.9)

allows us to write

$$\textsc{Reg} = \sum_{t\in[T]} \bar{R}(t, \pi^{\text{on}}(t, S^t, \xi^t), S^t)[\omega] \cdot \mathbb{1}_{Q(t,S^t)}[\omega] = \sum_{t=T-\tau+1}^{T} \mathbb{1}_{\{X^t>b\}} + (b - X^{T-\tau})\mathbb{1}_{\{1\le X^{T-\tau}<b\}}.$$

The first term corresponds to the disagreement of the first τ days (pay one dollar each day t such that $X^t > b$), whereas the second is the disagreement of day $\tau + 1$. This is an example where the compensated coupling yields an intuitive way of writing the regret. Furthermore, since the expression is exact, we can take expectations and optimize over τ to get the optimal policy (for example, see [11]).

3.3.3 The Bayes Selector Policy

Using the formalism defined in the previous sections, let $q(t, a, s) \coloneqq \mathbb{P}[Q(t, a, s)]$ be the *disagreement probability* of action a at time t in state s (i.e., the probability that a is not a satisfying action).

For any t, s, ξ^t, suppose we have an *oracle* that gives us $q(t, a, s)$ for every feasible action a. Given oracle access to $q(t, a, s)$ (or more generally, over-estimates $\hat{q}$ of q), a natural greedy policy suggested by Lemma 3.3.9 is that of choosing action a that minimizes the probability of disagreement. This is similar in spirit to the *Bayes selector* (i.e., hard thresholding) in statistical learning. Algorithm 1 formalizes the use of this idea in online decision-making. The results below are essentially agnostic of how we obtain this oracle.

From Lemma 3.3.9, we immediately have the following:

Corollary 3.3.11 (Regret Of Bayes Selector). *Consider Algorithm 1 with over-estimates* $\hat{q}(t, a, s) \ge \mathbb{P}[Q(t, a, s)]\,\forall\,(t, a, s)$. *If* A^t *denotes the policy's action at time* t, *then*

$$\mathbb{E}[\textsc{Reg}] \le \max_{a\in\mathcal{U}, j\in\mathcal{J}} \bar{r}(a, j) \cdot \sum_{t\in[T]} \mathbb{E}[\hat{q}(t, A^t, S^t)].$$

Algorithm 1 Bayes Selector

Input: Access to over-estimates $\hat{q}(t,a,s)$ of the disagreement probabilities, i.e. $\hat{q}(t,a,s) \geq q(t,a,s)$

Output: Sequence of decisions for ONLINE.

Set S^T as the given initial state
for $t = T, \ldots, 1$ **do**
    Observe the arriving type ξ^t .
    Take an action minimizing disagreement, i.e., $A^t \in \operatorname{argmin}\{\hat{q}(t,a,S^t) : a \in \mathcal{U}\}$.
    Update state $S^{t-1} \leftarrow \mathcal{T}(A^t, S^t, \xi^t)$.

The next result states that, if we can bound the estimation error uniformly over states and actions, then the guarantee of the algorithm increases additively on the error (not multiplicatively, as one may suspect). In more detail, our next result is agnostic of the oracle used to obtain the the estimators $\hat{q}$. Examples of estimation procedures to obtain $\hat{q}$ include: simulation, function approximation, neural networks, etc. Regardless of how $\hat{q}$ is obtained, we can give a regret guarantee based only on the accuracy of the estimators. The following result follows as a special case of Corollary 3.3.11; we state it to emphasise that $\hat{q}$ can be estimated with some error.

Corollary 3.3.12 (Bayes Selector with Imperfect Estimators). *Assume we have estimators $\hat{q}(t,a,s)$ of the probabilities $q(t,a,s)$ such that $|q(t,a,s) - \hat{q}(t,a,s)| \leq \Delta^t$ for all t,a,s. If we run Algorithm 1 with over-estimates $\hat{q}(t,a,s) + \Delta^t$, and A^t denotes the policy's action at time t, then*

$$\mathbb{E}[\text{REG}] \leq \max_{a \in \mathcal{U}, j \in \mathcal{J}} \bar{r}(a,j) \cdot \sum_{t \in [T]} (\mathbb{E}[\hat{q}(t, A^t, S^t)] + \Delta^t).$$

Observe that, the total error induced due to estimation is a constant if, e.g., we can guarantee $\Delta^t = 1/t^2$ or $\Delta^t = 1/(T-t)^2$.

It is natural to consider a more sophisticated version of Algorithm 1, wherein we make decisions not only based on disagreement probabilities, but also take into account marginal compensations, i.e., the marginal loss of each decision. While Algorithm 1 is enough to

obtain constant regret bounds in the problems we consider, we note that such an extension is possible and can be found in Appendix B.2.2.

Remark 3.3.13. We discussed in Section 3.3 that the Compensated Coupling extends to the case where transitions and rewards can be random. By the same argument, the Bayes Selector (Algorithm 1) and its guarantees (Corollaries 3.3.11 and 3.3.12) extend too. Notice that OFFLINE here is a prophet who has full knowledge of all the randomness (arrivals, transitions, and rewards).

3.4 Regret Bounds for Online Packing

We now show that for the online packing problem, the Bayes Selector achieves an expected regret which is *independent of the number of arrivals T and the initial budgets B*; in Section 3.5, we extend this to matching problems.

In more detail, we prove that the *dynamic fluid relaxation* (P_t) in Eq. (3.4) provides a good estimator for the disagreement probabilities $q(t, a, s)$, and moreover, that the Bayes Selector based on these statistics reduces to a simple *re-solve and threshold* policy.

In this setting, the state space corresponds to resource availability, hence $S^t = \mathbb{N}^d$; there are two possible actions, accept or reject, hence $|\mathcal{U}| = 2$; finally, transitions correspond to the natural budget reductions given by the matrix A.

Recall that $Z(t) \in \mathbb{N}^n$ denotes the cumulative arrivals in the last t periods and $B^t \in \mathbb{N}^d$ denotes ONLINE's budget at time t. Given knowledge of $Z(t)$ and state B^t, we define the

ex-post relaxation ($P_t^\star$) and *fluid* relaxation (P_t) as follows.

$$\begin{aligned} (P_t^\star)\ \max\quad & r'x \\ \text{s.t.}\quad & Ax \le B^t \\ & x \le Z(t) \\ & x \ge 0. \end{aligned} \qquad\qquad \begin{aligned} (P_t)\ \max\quad & r'x \\ \text{s.t.}\quad & Ax \le B^t \\ & x \le \mathbb{E}[Z(t)] \\ & x \ge 0. \end{aligned} \tag{3.4}$$

Remark 3.4.1. OFFLINE solves ($P_t^\star$) in Eq. (3.4), while ONLINE solves (P_t). Both problems depend on ONLINE's budget at t; this is a crucial technical point and can only be accomplished due to the coupling we have developed.

Let X^t be a solution of (P_t) and $X^{\star t}$ a solution of ($P_t^\star$). Uniqueness of solutions is not required, see Proposition 3.4.8, and $X^{\star t}$ is for the analysis only. Our policy is detailed in Algorithm 2.

Algorithm 2 Fluid Bayes Selector

Input: Access to solutions X^t of (P_t) and resource matrix A.
Output: Sequence of decisions for ONLINE.

1: Set B^T as the given initial budget levels
2: **for** $t = T, \dots, 1$ **do**
3: Observe arrival $\xi^t = j$ and accept iff $X_j^t \ge \mathbb{E}[Z_j(t)]/2$ and it is feasible, i.e., $A_j \le B^t$.
4: Update $B^{t-1} \leftarrow B^t - A_j$ if accept, and $B^{t-1} \leftarrow B^t$ if reject.

Intuitively, we 'front-load' (accept as early as possible) classes j such that $X_j^t \ge \mathbb{E}[Z_j(t)]/2$ and back-load the rest (delay as much as possible). If OFFLINE is satisfied accepting a front-loaded class (resp. rejecting a back-loaded class), he will do so. Accepting class j is therefore an error if OFFLINE, given the same budget as ONLINE, picks no future arrivals of that class (i.e., $X_j^{\star t} < 1$). On the other hand, rejecting j is an error if $X_j^{\star t} > Z_j(t) - 1$. We summarize this as follows:

1. Incorrect rejection: if $X_j^t < \frac{\mathbb{E}[Z_j(t)]}{2}$ and $X_j^{\star t} > Z_j(t) - 1$.
2. Incorrect acceptance: if $X_j^t \ge \frac{\mathbb{E}[Z_j(t)]}{2}$ and $X_j^{\star t} < 1$.

Observe that a compensation is paid only when the fluid solution is far off from the correct stochastic solution. In the proofs of Theorems 3.4.2 and 3.4.6 we formalize the fact that, since X^t estimates $X^{\star t}$, such an event is highly unlikely – this along with the compensated coupling provides our desired regret guarantees.

Disagreement probabilities and the Fluid Bayes Selector. The Bayes Selector (Algorithm 1) runs with over-estimates $\hat{q}(t, a, b)$ and, at each time, picks the minimum. On the other hand, Algorithm 2 is presented as the "simplified version", in the sense that the decision rule is the one that minimizes suitable over-estimates $\hat{q}$. More importantly, we prove the following properties

$$\operatorname{argmin}\{\hat{q}_j(t, \text{accept}, b), \hat{q}_j(t, \text{reject}, b)\} = \begin{cases} \text{accept} & \text{if } X_j^t \geq \mathbb{E}[Z_j(t)]/2 \\ \text{reject} & \text{if } X_j^t < \mathbb{E}[Z_j(t)]/2 \end{cases}, \tag{3.5}$$

$$\min\{\hat{q}_j(t, \text{accept}, b), \hat{q}_j(t, \text{reject}, b)\} \leq c_1 e^{-c_2 t}. \tag{3.6}$$

In other words, the property in Eq. (3.5) shows that Algorithm 2 is a Bayes Selector, while the property in Eq. (3.6) yields the desired constant regret bound in virtue of the Compensated Coupling and Corollary 3.3.11. We give explicit expressions for the values $\hat{q}$ and constants c_1, c_2, see e.g. Eq. (B.2).

The robustness of the Bayes Selector. The probability minimizing disagreement can be uniformly bounded over all budgets $b \in \mathbb{N}^d$, i.e., the exponential bound in Eq. (3.6) does not depend on b. This property has the following consequence: *since the Fluid Bayes Selector has strong performance, many other Bayes Selector algorithms (using different $\hat{q}$) do too.* In other words, the design of algorithms based on the Bayes Selector is robust and does not depend on "fine tunning" of the parameters $\hat{q}$. We make this precise in Corollaries 3.4.3 and 3.4.7 and uncover the same phenomenon for matching problems, see Corollary 3.5.2.

We need some additional notation before presenting our results. Let $\mathbb{E}_j[\cdot]$ ($\mathbb{P}_j[\cdot]$) be the

expectation (probability) conditioned on the arrival at time t being of type j, i.e., $\mathbb{P}_j[\cdot] = \mathbb{P}[\cdot|\xi^t = j]$. We denote $r_\varphi \coloneqq \max_{j\in[n]} r_j$ and $p_{\min} \coloneqq \min_{j\in[n]} p_j$.

3.4.1 Special Case: Multi-Secretary with Multinomial Arrivals

Before we proceed to the general case, we state the result for the Multi-Secretary problem. We present this result separately because in this one dimensional problem we can obtain a better and explicit constant. The proofs of Theorem 3.4.2 and Corollary 3.4.3 below can be found in Appendix B.2.3.

Theorem 3.4.2. *The expected regret of the Fluid Bayes Selector (Algorithm 2) for the multi-secretary problem with multinomial arrivals is at most* $r_\varphi \sum_{j>1} 2/p_j \leq 2(n-1)r_\varphi/p_{\min}$.

This recovers the best-known expected regret bound for this problem shown in a recent work [10]. However, while the result in [10] depends on a complex martingale argument, our proof is much more succinct, and provides explicit and stronger guarantees; in particular, in Section 3.4.3, we provide concentration bounds for the regret.

Moreover, Theorem 3.4.2, along with Corollary 3.3.12, provides a critical intermediate step for characterizing the performance of Algorithm 1 for the multi-secretary problem.

Corollary 3.4.3. *For the multi-secretary problem with multinomial arrivals, the expected regret of the Bayes Selector (Algorithm 1) with any imperfect estimators* $\hat{q}$ *is at most* $2r_\varphi\left(\sum_{j>1} 1/p_j + \sum_{t\in[T]} \Delta^t\right)$, *where* Δ^t *is the accuracy defined by* $|q(t,a,b) - \hat{q}(t,a,b)| \leq \Delta^t$ *for all* $t \in [T], a \in \mathcal{U}, b \in \mathbb{N}$.

Observe that, if Δ^t is summable, e.g., $\Delta^t = 1/t^2$ or $\Delta^t = 1/(T-t)^2$, then Corollary 3.4.3 implies constant expected regret for all these types of estimators we can use in Algorithm 1.

3.4.2 Online Packing with General Arrivals

We consider now the case $d > 1$ and arrival processes other than Multinomial. We assume the following condition on the process $Z(t)$, which we refer to as *all time deviation*.

Definition 3.4.4 (All Time Deviation). Let μ be a given norm in $\mathbb{R}^n$ and $\kappa \in \mathbb{R}^n_{\geq 0}$ a constant parameter. An n dimensional process $Z(t)$ satisfies the all time deviation bound w.r.t. μ and κ if, for all $j \in [n]$, there are constants $c_j \geq 0$ and naturals τ_j such that

$$\mathbb{P}\left[||Z(t) - \mathbb{E}(Z(t))||_\mu \geq \frac{\mathbb{E}[Z_j(t)]}{2\kappa_j}\right] \leq \frac{c_j}{t^2} \quad \forall t > \tau_j. \tag{3.7}$$

We remark that we do not need exponential tails, as it is common to assume, but rather a simple quadratic tail. Additionally, some common tail bounds are valid only for large enough samples; the parameters τ_j capture this technical aspect. In this section we will use the definition with κ_j the same entry for all j, thus denoted simply by $\kappa > 0$. In Section 3.5 we require the definition with the more general form.

Example 3.4.5 (Multinomial and Poisson tails). In these examples we actually have the stronger exponential tails, so we do not elaborate on the constant c_j.

For multinomial arrivals, [52, Lemma 3] guarantees

$$\mathbb{P}[||Z(t) - \mathbb{E}(Z(t))||_1 > t\varepsilon] \leq e^{-t\varepsilon^2/25}, \forall 0 < \varepsilon < 1, t \geq \frac{\varepsilon^2 n}{20}. \tag{3.8}$$

By setting $\varepsilon = p_j/2\kappa$, we conclude that Definition 3.4.4 is satisfied with constants $\tau_j = (p_j/2\kappa)^2 n/20$.

For Poisson arrivals, from the proof of [52, Lemma 3], $\mathbb{P}(|X - \lambda| \geq \varepsilon\lambda) \leq 2e^{-\lambda\varepsilon^2/4}$ is valid for $X \sim \text{Poisson}(\lambda)$ and any $\varepsilon > 0$. Using this, we can simply take $\tau_j = 0$.

In the remainder of this subsection we generalize our ideas to prove the following.

Theorem 3.4.6. *Assume the arrival process $(Z(t) : t \in [T])$ satisfies the conditions in Eq. (3.7). The expected regret of the Fluid Bayes Selector (Algorithm 2) for online packing is at most $dr_\varphi M$, where M is independent of T and B. Specifically, for $\kappa = \kappa(A)$ we have*

1. *For Multinomial arrivals: $M \leq 103\kappa^2 \sum_{j\in[n]} 1/p_j$.*
2. *For general distributions satisfying Eq. (3.7): $M \leq \sum_{j\in[n]} p_j(2c_j + \max\{\tau_j, \tilde{\tau}_j\})$, where p_j is an upper bound on $\mathbb{P}[\xi^t = j]$ and $\tilde{\tau}_j$ is such that $\mathbb{E}[Z_j(\tilde{\tau}_j)] \geq 2$, i.e., it is large enough.*

The constant $\kappa(A)$ is given by Proposition 3.4.8 below. Just as before, Theorem 3.4.6, along with Corollary 3.3.12, provides a performance guarantee for Algorithm 1. We state the corollary without proof, since it is identical to that of Corollary 3.4.3.

Corollary 3.4.7. *For the online packing problem, if the arrival process satisfies the conditions in Eq. (3.7), the expected regret of the Bayes Selector (Algorithm 1) with any imperfect estimators $\hat{q}$ is at most $dr_\varphi(M + 2\sum_{t\in[T]} \Delta^t)$, where M is as in Theorem 3.4.6 and Δ^t is the accuracy defined by $|q(t, a, b) - \hat{q}(t, a, b)| \leq \Delta^t$ for all $t \in [T], a \in \mathcal{U}, b \in \mathbb{N}^d$.*

To prove Theorem 3.4.6, we need to quantify how the change in the right-hand side of an LP impacts optimal solutions. Indeed, as stated in Eq. (3.4), the solutions X^t and $X^{\star t}$ correspond to perturbed right-hand sides ($\mathbb{E}[Z(t)]$ and $Z(t)$ respectively). The following proposition implies that small changes in the arrivals vector do not change the solution by much and it is based on a more general result from [84, Theorem 2.4].

Proposition 3.4.8 (LP Lipschitz Property). *Given $b \in \mathbb{R}^d$, and any norm $||\cdot||_\mu$ in $\mathbb{R}^n$, consider the following LP*

$$P(y) \quad \max\{r'x : Ax \leq b, 0 \leq x \leq y, y \in \mathbb{R}^n_{\geq 0}\}.$$

Then $\exists$ constant $\kappa = \kappa_\mu(A)$ such that, for any $y, \hat{y} \in \mathbb{R}^n_{\geq 0}$ and any solution x to $P(y)$, there exists a solution $\hat{x}$ solving $P(\hat{y})$ such that $||x - \hat{x}||_\infty \leq \kappa ||y - \hat{y}||_\mu$.

Proof of Theorem 3.4.6. Recall the two conditions derived from our decision rule: (1) Incorrect rejection of j means $X_j^t < \mathbb{E}[Z_j(t)]/2$ and $X_j^{\star t} > Z_j(t) - 1$. (2) Incorrect acceptance of j means $X_j^t \geq \mathbb{E}[Z_j(t)]/2$ and $X_j^{\star t} < 1$. We have to additionally account for feasibility, i.e., we can only accept a request j if $B_i^t \geq a_{ij}$ for all $i \in [d]$. In case there are not enough resources, our decision rule is feasible if either $X_j^t < \mathbb{E}[Z_j(t)]/2$ (reject) or $X_j^t \geq 1$ (since X^t is feasible for (P_t)). Only in the case $X_j^t \geq \mathbb{E}[Z_j(t)]/2$ and $X_j^t < 1$ we need to disregard our decision rule and are forced to reject; under such a condition we must pay a compensation of r_j. Observe that this condition is never met if $\mathbb{E}[Z_j(t)] \geq 2$, i.e., it is vacuous for $t \geq \tilde{\tau}_j$.

The disagreement sets (Definition 3.3.8) are thus $Q(t, b) = \{\omega \in \Omega : \text{either } (1), (2) \text{ or } t < \tilde{\tau}_j\}$, where (1) and (2) are the previous conditions. Now we can upper bound the probability of paying a compensation as follows. Call E_j the event $\{\omega \in \Omega : ||Z(t) - \mathbb{E}(Z(t))||_1 \leq \mathbb{E}[Z_j(t)]/2\kappa\}$. In this event, Proposition 3.4.8 implies $|X_j^t - X_j^{\star t}| \leq \mathbb{E}[Z_j(t)]/2$, hence conditions (1) and (2) do not happen when E_j occurs, i.e., $\mathbb{P}_j[Q(t, b)|E_j] \leq \mathbb{1}_{\{t < \tilde{\tau}_j\}}$. Observe that $\mathbb{P}[\bar{E}_j] \leq f_j(t) + \mathbb{1}_{\{t < \tau_j\}}$, where $f_j(t) = c_j/t^2$ for general processes satisfying Eq. (3.7) and $f_j(t) = e^{-t(p_j/2\kappa)^2/25}$ for the Multinomial process (see Eq. (3.8)). Finally,

$$q_j(t, B^t) \leq \mathbb{P}[\bar{E}_j] + \mathbb{P}_j[Q(t, B^t)|E_j] \leq \mathbb{P}[\bar{E}_j] + \mathbb{1}_{\{t < \tilde{\tau}_j\}} \leq f_j(t) + \mathbb{1}_{\{t < \tau_j \text{ or } t < \tilde{\tau}_j\}}. \tag{3.9}$$

Summing up over time, we get

$$\sum_{t \in [T]} q(t, B^t) \leq \sum_{j \in [n]} p_j \left(\sum_{t \in [T]} f_j(t) + \max\{\tau_j, \tilde{\tau}_j\} \right)$$

Since $\sum_{t \in [T]} 1/t^2 \leq \pi^2/6 \leq 2$, this finishes the proof for general processes. For the case of Multinomial arrivals, we can be more refined. Indeed, $\tilde{\tau}_j$ is defined by $\mathbb{E}[Z_j(\tilde{\tau}_j)] \geq 2$, i.e., $\tilde{\tau}_j \geq 2/p_j$ and $\tau_j = (p_j/2\kappa)^2 n/20$ (see Eq. (3.8)). From the previous equation, with the

stronger exponential bound $f_j(t) = e^{-t(p_j/2\kappa)^2/25}$ we get

$$\sum_{t\in[T]} q(t, B^t) \le \sum_{j\in[n]} p_j \left(\frac{25}{(p_j/2\kappa)^2} + \max\{(p_j/2\kappa)^2 n/20, 2/p_j\} \right)$$
$$\le 100\kappa^2 \sum_{j\in[n]} \frac{1}{p_j} + 3n.$$

Since $n \le \sum_{j\in[n]} \frac{1}{p_j}$, we arrive a the desired bound. The result follows via the compensated coupling (Lemma 3.3.9) and Corollary 3.3.11. □

Remark 3.4.9. In the multi-secretary problem it is easy to conclude $\kappa(A) = 1$, thus this analysis recovers the same bound up to absolute constants (namely 103 vs 2). The larger constant comes exclusively from the larger constants in the tail bounds of Multinomial compared to Binomial r.v.

Remark 3.4.10. More refined bounds on M can be obtained by not bounding $\mathbb{P}[\xi^t = j] \le p_j$, but rather by $\mathbb{P}[\xi^t = j] \le p_j(t)$. For example, a time-varying version of a Multinomial process easily fits in our framework and the proof does not change.

Remark 3.4.11. The theorem holds even under Markovian correlations (see Example 3.4.12 below), where the distribution of $Z(t-1)$ depends on ξ^t. It is interesting that in this case it is impossible to run the optimal policy for even moderate instance sizes, since the state space is huge, while the Bayes Selector still offers bounded expected regret.

We now give two examples of other arrival processes that satisfy the All Time Deviation (Definition 3.4.4). The proofs of the bounds are short, but we relegate them to Appendix B.2.4. We emphasise that Example 3.4.13 below has quadratic tails (instead of exponential), hence we term it heavy tailed.

Example 3.4.12 (Markovian Arrival Processes). We consider the case where ξ^t is drawn from an ergodic Markov chain. Let $P \in \mathbb{R}^{n\times n}_{\ge 0}$ be the corresponding matrix of transition probabilities. The process unfolds as follows: at time $t = T$ an arrival $\xi^T \in [n]$ is drawn

according to an arbitrary distribution, then for $t = T, \dots, 2$ we have $\mathbb{P}[\xi^{t-1} = j | \xi^t] = P_{\xi^t j}$. Let $\nu \in \mathbb{R}^n_{\geq 0}$ be the stationary distribution. *We do not require long-run or other usual stationary assumptions; the process is still over a finite horizon T.* This process satisfies All Time Deviation with exponential tails. Specifically, with the norm $\mu = ||\cdot||_\infty$, for some constants c_j, c' that depend on P only we have

$$\mathbb{P}[||Z(t) - \nu t||_\infty \geq \nu_j t / 2\kappa_j] \leq nc' e^{-c_j t}, \quad \forall t \in [T], j \in [n]. \tag{3.10}$$

Example 3.4.13 (Heavy Tailed Poisson Arrivals). We consider the case where the arrival process is governed by independent time varying Poisson processes with arrival rates $\lambda_j(t) > 0$, which we assume for simplicity have finitely many discontinuity points (so that all the expectations are well defined). Under the following conditions, the process satisfies the All Time Deviation with *quadratic tails* and norm $\mu = ||\cdot||_\infty$.

$$\max_{j,k \in [n]} \max_{s \in [0,t]} \frac{\lambda_j(s)}{\lambda_k(s)} \leq g(t) \quad \forall t \geq 0 \tag{3.11}$$

$$\min_{j \in [n]} \min_{s \in [0,t]} \lambda_j(s) \geq g(t) f(t) \frac{\log(t)}{t}, \quad \text{where} \quad \lim_{t \to \infty} f(t) = \infty. \tag{3.12}$$

In other words, we require $f(t) = \omega(1)$ and $g(t)$ is any function. Intuitively, Eq. (3.11) guarantees that no type j "overwhelms" all other types; observe that, when the rates are constant, this is trivially satisfied with $g(t)$ constant. On the other hand, Eq. (3.12) controls the minimum arrival rate, which can be as small as $\omega(\log(t)/t)$. Observe that our conditions allow for the intensity to increase closer to the end ($t = 0$), i.e., we incorporate the case where customers are more likely to arrive closer to the deadline.

3.4.3 High-Probability Regret Bounds

We have proved that $\mathbb{E}[\textsc{Reg}]$ is constant for packing problems. One may worry that this is not enough because, since it is a random variable, $\textsc{Reg}$ may still realize to a large value. We

present a bound for the distribution of REG showing that it has light tails.

Proposition 3.4.14. *For packing problems, there are constants τ and c_j for $j \in [n]$, depending on A, p and the distribution of Z only, such that*

1. *For Multinomial or Poisson arrivals: $\forall x > \tau$, $\mathbb{P}[\text{REG} > x] \leq \sum_j p_j e^{-c_j x / r_\varphi} / c_j$.*
2. *For general distributions satisfying Eq. (3.7): $\forall x > \tau$, $\mathbb{P}[\text{REG} > x] \leq \frac{r_\varphi}{x} \sum_j p_j c_j$.*

The proof is based on the following simple lemma. The idea is to first bound the disagreements of our algorithm, as defined in Section 3.3.3. The total number of disagreements is a sum of dependent Bernoulli variables, which we bound next.

Lemma 3.4.15. *Let $\{X^t : t \in [T]\}$ be a sequence of dependent r.v. such that $X^t \sim$ Bernoulli(p_t) and let $\{q_t : t \in [T]\}$ be numbers such that $q_t \geq p_t$. If we define $D \coloneqq \sum_{t=1}^T X^t$, then*

$$\mathbb{P}[D \geq d] \leq \sum_{t=d}^{T} q_t$$

PROOF. Fix $d \in [T]$ and observe that

$$\{\omega \in \Omega : D \geq d\} \subseteq \{\omega \in \Omega : \exists t \geq d, X^t = 1\}.$$

Indeed, if the condition $(\exists t \geq d, X^t = 1)$ fails, then at most $d - 1$ variables X^t can be one.

Finally, a union bound shows $\mathbb{P}[D \geq d] \leq \sum_{t \geq d} \mathbb{P}[X^t = 1]$. Since $q_t \geq p_t$, the proof is complete. □

PROOF OF PROPOSITION 3.4.14. As described in the previous subsections, we can write $\text{REG} \leq r_\varphi D$, with D the number of disagreements. Additionally, D is a sum of T Bernoulli r.v. X^t, each with parameter bounded by q_t.

In the case of Multinomial and Poisson r.v., as described in Section 3.4.2, we have exponential bounds $q_t \leq \sum_{j\in[n]} p_j e^{-c_j t}$ for $t \geq \tau = \max_{j\in[n]} \tau_j$. We conclude invoking Lemma 3.4.15 and upper bounding $\sum_{t=x+1}^{T} e^{-c_j t} \leq e^{-c_j x}/c_j$.

For general distributions, as described in Section 3.4, we have the bounds $q_t \leq \sum_{j\in[n]} p_j \frac{c_j}{t^2}$ for $t \geq \tau$. Using Lemma 3.4.15 and bounding $\sum_{t=x+1}^{T} t^{-2} \leq 1/x$ finishes the proof. □

3.5 Regret Bounds for Online Matching

We turn to an alternate setting, where each incoming arrival corresponds to a *unit-demand* buyer – in other words, each arrival wants a unit of a single resource, but has different valuations for different resources. This is essentially equivalent to the online bipartite matching problem with edge weights (weights correspond to rewards) where there can be multiple copies of each node.

As before, we are given a matrix $A \in \{0,1\}^{d\times n}$ characterizing the demand for resources, which can be interpreted as the adjacency matrix in the online matching problem. Define $S_j \coloneqq \{i \in [d] : a_{ij} = 1\}$. If we allocate any resource $i \in S_j$ to a customer type j, we obtain a reward of r_{ij}, whereas allocating $i \notin S_j$ has no reward. We can allocate at most one item to each customer.

Given resource availability $B \in \mathbb{N}^d$ and total arrivals $Z \in \mathbb{N}^n$, we can formulate Offline's problem as follows, where the variable x_{ij} denotes the number of items i allocated

to customers type j.

$$\begin{aligned} (P[Z,B]) \quad \max \quad & \textstyle\sum_{i,j} x_{ij} r_{ij} a_{ij} \\ \text{s.t.} \quad & \textstyle\sum_j x_{ij} \leq B_i \quad \forall i \in [d] \\ & \textstyle\sum_{i\in[d]} x_{ij} \leq Z_j \quad \forall j \in [n] \\ & \mathbf{x} \geq 0. \end{aligned} \tag{3.13}$$

We assume that the process $Z(t)$ satisfies the all time deviation bound (see Definition 3.4.4) w.r.t. the one-norm and parameters $\kappa_j = (|S_j| + 1)/2$. This condition can be restated as follows. For every $j \in [n]$, there are constants $c_j \geq 0$ and naturals τ_j such that

$$\mathbb{P}\left[||Z(t) - \mathbb{E}(Z(t))||_1 \geq \frac{\mathbb{E}[Z_j(t)]}{|S_j|+1}\right] \leq \frac{c_j}{t^2} \quad \forall t > \tau_j. \tag{3.14}$$

We now state the main result of this section, which is based on an instantiation of the Bayes Selector. As before, the theorem readily implies performance guarantees for Algorithm 1, which we state without proof, since it is identical to that of Corollary 3.4.3.

Theorem 3.5.1. *For the online matching problem, if the arrival process satisfies the conditions in Eq. (3.14), then the expected regret of the Fluid Bayes Selector (Algorithm 3) is at most* $r_\varphi \sum_{j\in[n]} p_j(c_j + \tau_j)$*, where* p_j *is an upper bound on* $\mathbb{P}[\xi^t = j]$.

Corollary 3.5.2. *For the online matching problem, if the arrival process satisfies the conditions in Eq. (3.14), then the expected regret of the Bayes Selector (Algorithm 1) with any imperfect estimators* $\hat{q}$ *is at most* $r_\varphi(M + 2\sum_{t\in[T]} \Delta^t)$*. The constant* $M = \sum_{j\in[n]} p_j(c_j + \tau_j)$ *is as in Theorem 3.5.1 and* Δ^t *is the accuracy defined by* $|q(t,a,s) - \hat{q}(t,a,s)| \leq \Delta^t$.

3.5.1 Algorithm and Analysis

We start from the LP in Eq. (3.13), then add a fictitious item $d+1$ which no customer wants with initial budget $B_{d+1}^T = T$; now all customers are matched, but, if we match a customer

to $d+1$, there is no reward. Using the Compensated Coupling, we can write two coupled optimization problems, $(P_t^\star)$ for Offline and (P_t) for Online as follows.

$$
\begin{array}{lrll}
(P_t^\star)\max & \sum_{i\in[d],j\in[n]} x_{ij} r_{ij} a_{ij} & & \\
\text{s.t.} & \sum_{j\in[n]} x_{ij} & \le B_i^t & \forall i \in [d+1] \\
& \sum_{i\in[d+1]} x_{ij} & = Z_j(t) & \forall j \in [n] \\
& \mathbf{x} & \ge 0. &
\end{array}
\qquad
\begin{array}{lrll}
(P_t)\max & \sum_{i\in[d],j\in[n]} x_{ij} r_{ij} a_{ij} & & \\
\text{s.t.} & \sum_{j\in[n]} x_{ij} & \le B_i^t & \forall i \in [d+1] \\
& \sum_{i\in[d+1]} x_{ij} & = \mathbb{E}[Z_j(t)] & \forall j \in [n] \\
& \mathbf{x} & \ge 0. &
\end{array}
\tag{3.15}
$$

Recall that B^t represents Online's budget with t periods to go. We solve (P_t) in Eq. (3.15) and obtain an optimizer X^t. If $\xi^t = j$, let $K \in \operatorname{argmax}\{X^t_{i,j} : i \in [d+1]\}$ be the maximal entry, breaking ties arbitrarily, then match j to K. The resulting policy is presented in Algorithm 3. Observe that, matching a customer to $K = d+1$ (fictitious resource) is equivalent to rejecting him.

Algorithm 3 Fluid Bayes Selector For Online Matching

Input: Access to solutions X^t of (P_t) in Eq. (3.15).
Output: Sequence of decisions for Online.

1: Set B^T as the given initial budget levels
2: **for** $t = T, \ldots, 1$ **do**
3: Observe arrival $\xi^t = j$ and let $K \leftarrow \operatorname{argmax}\{X^t_{ij} : i \in [d+1]\}$, breaking ties arbitrarily.
4: Match ξ^t to K.
5: Update $B_i^{t-1} \leftarrow B_i^t$ for $i \neq K$ and $B_K^{t-1} \leftarrow B_K^t - 1$.

Disagreement Sets. At each time t, matching $\xi^t = j$ to K requires a compensation only if Offline never matches a type j to K, i.e., $X^{\star t}_{K,j} < 1$. On the other hand, Algorithm 3 picks K to be the largest component, hence we should have $X^t_{K,j} >> 1$ (precisely, $X^t_{K,j} \ge \frac{\mathbb{E}[Z_j(t)]}{d+1}$). More formally, the constraint $\sum_{i\in[d+1]} x_{ij} = \mathbb{E}[Z_j(t)]$ in Eq. (3.15) and the definition of S_j imply $X^t_{K,j} \ge \mathbb{E}[Z_j(t)]/(|S_j|+1)$. We conclude that, if matching to K is not satisfying (see Definition 3.3.4), it must be that $||X^t - X^{\star t}||_\infty > \mathbb{E}[Z_j(t)]/(|S_j|+1)$. Proposition 3.5.3 below characterizes exactly this deviation.

Observe that, in Eqs. (3.13) and (3.15), the matrix A appears only on the objective

function; this is not the usual LP formulation for this problem, but it allows us to obtain the following result. We remark that not only we have a Lipschitz property, but the Lipschitz constant is exactly 1. We present the proof of Proposition 3.5.3 in Appendix B.2.5.

Proposition 3.5.3 (Lipschitz Property for Matching). *Take any* $z^1, z^2 \in \mathbb{R}^d_{\geq 0}$ *and* $b \in \mathbb{R}^d_{\geq 0}$. *If* $\mathbf{x}^1$ *is a solution of* $P[z^1, b]$, *then there exists* $\mathbf{x}^2$ *solving* $P[z^2, b]$ *such that* $||\mathbf{x}^1 - \mathbf{x}^2||_\infty \leq ||z^1 - z^2||_1$.

From here, the proof of Theorem 3.5.1 is applying the Compensated Coupling (Lemma 3.3.9) and Corollary 3.3.11 in the same way as we did in Section 3.4, hence we omit it.

3.5.2 Online Stochastic Matching

A classical problem that fits naturally into the above framework is that of online bipartite matching problem with stochastic inputs [86]. The reader unfamiliar with the problem can find the details of the setup in Appendix B.2.6. For this setting, the bound obtained via compensated coupling surprisingly holds with equality:

Lemma 3.5.4. *For the stochastic online bipartite matching, given an online policy, if* U^t *denotes the node matched at time* t *by* ONLINE *and* S^t *the available nodes, then*

$$v^{off} - v^{on} = \sum_{t \in [T]} \mathbb{1}_{Q(t, U^t, S^t)}.$$

Based on this, it is tempting to conjecture that the Bayes Selector does in fact lead to an optimal policy for this setting. This however is not the case, although showing this is surprisingly subtle; in Appendix B.2.6, we discuss this in more detail. Moreover, it is known that this problem cannot admit an expected regret that has better than linear scaling with T

(in particular, [86] proves a constant upper bound on the competitive ratio for this setting). That said, the strength of the above bound suggests that the Bayes selector may have strong approximation guarantees – showing this remains an open problem.

3.6 Online Allocation

We now give the algorithm and analysis for the most general problem defined in Section 3.2.1. As before, let us introduce a fictitious resource $i = d+1$ with initial capacity $B_{d+1} = T$, zero rewards ($r_{\{d+1\}j} = 0$ for all $j \in [n]$) and such that $\{d+1\} \in S_j$ for all $j \in [n]$. Now we can assume w.l.o.g. that each customer gets assigned a bundle. Finally, for a bundle s, we denote $a_{is} \in \mathbb{N}$ the number of times the resource i appears in s (recall that bundles are multisets).

Given resource availability $B \in \mathbb{N}^{d+1}$ and total arrivals $Z \in \mathbb{N}^n$, we can formulate the coupled problems for Offline and Online as follows, where the variable x_{sj} denotes the number of times a bundle $s \in S_j$ is allocated to a type j.

$$
\begin{array}{lrll}
(P_t^\star)\max & \sum_{j\in[n],s\in S_j} x_{sj} r_{sj} & & \\
\text{s.t.} & \sum_{j\in[n],s\in S_j} a_{is} x_{sj} & \le B_i^t & \forall i \in [d+1] \\
& \sum_{s\in S_j} x_{sj} & = Z_j(t) & \forall j \in [n] \\
& \mathbf{x} & \ge 0. &
\end{array}
\qquad
\begin{array}{lrll}
(P_t^\star)\max & \sum_{j\in[n],s\in S_j} x_{sj} r_{sj} & & \\
\text{s.t.} & \sum_{j\in[n],s\in S_j} a_{is} x_{sj} & \le B_i^t & \forall i \in [d+1] \\
& \sum_{s\in S_j} x_{sj} & = \mathbb{E}[Z_j(t)] & \forall j \in [n] \\
& \mathbf{x} & \ge 0. &
\end{array}
\tag{3.16}
$$

We assume that the process $Z(t)$ satisfies the all time deviation bound (see Definition 3.4.4) w.r.t. some norm μ and parameters $\kappa_j = (d+1)\kappa$, where $\kappa = \kappa_\mu(A)$ depends only on A and μ. This condition can be restated as follows. For every $j \in [n]$, there are constants $c_j \ge 0$ and naturals τ_j such that

$$
\mathbb{P}\left[||Z(t) - \mathbb{E}(Z(t))||_\mu \ge \frac{\mathbb{E}[Z_j(t)]}{\kappa(|S_j|+1)}\right] \le \frac{c_j}{t^2} \quad \forall t > \tau_j. \tag{3.17}
$$

We present the resulting policy in Algorithm 4 with its guarantee in Theorem 3.6.1. We remark that the constant κ depends only on the constraint matrix defining the LP in Eq. (3.16), i.e., it does depend on the choices of bundles S_j, but it is independent of T and B.

Algorithm 4 Fluid Bayes Selector For Online Allocation

Input: Access to solutions X^t of (P_t) in Eq. (3.16).
Output: Sequence of decisions for ONLINE.

1: Set B^T as the given initial budget levels
2: **for** $t = T, \dots, 1$ **do**
3: Observe arrival $\xi^t = j$ and let $K \leftarrow \text{argmax}\{X^t_{sj} : s \in S_j\}$, breaking ties arbitrarily.
4: If it is not feasible to assign bundle K, then reject. Otherwise assign K to ξ^t.
5: Update $B^{t-1}_i \leftarrow B^t_i$ for $i \notin K$ and $B^{t-1}_i \leftarrow B^t_i - a_{iK}$ for $i \in K$.

Theorem 3.6.1. *For the Online Allocation problem, there exists a constant κ that depends on $(S_j : j \in [n])$ only such that, if the arrival process satisfies the conditions in Eq. (3.17), then the expected regret of the Fluid Bayes Selector (Algorithm 4) is at most $r_\varphi \sum_{j\in[n]} p_j(c_j + \tau_j)$, where p_j is an upper bound on $\mathbb{P}[\xi^t = j]$.*

The proof of Theorem 3.6.1 is analogous to that of Theorem 3.4.6, hence we omit it and provide here only the key steps. Recall that, for request j, since we include the fictitious item, there are are $|S_j| + 1$ possible bundles. Crucially, incorrect allocation of s to j necessitates $X^t_{sj} \geq \mathbb{E}[Z_j(t)]/(|S_j| + 1)$ (because Algorithm 4 takes the maximum entry) and $X^{\star t}_j < 1$ (OFFLINE never allocates s to j). By the Lipschitz property of LPs (see Proposition 3.4.8), this event requires a large deviation of $Z(t)$ w.r.t. its mean, which has low probability. More formally, the disagreement sets (Definition 3.3.8) are $Q(t, b) = \{\omega \in \Omega : X^t_{sj} \geq \mathbb{E}[Z_j(t)]/(|S_j| + 1) \text{ and } X^{\star t}_j < 1\}$. By the Lipschitz property, $Q(t, b) \subseteq \{\omega \in \Omega : ||Z(t) - \mathbb{E}(Z(t))||_\mu \geq \frac{\mathbb{E}[Z_j(t)]}{\kappa(|S_j|+1)}\}$. The probability of this last event is bounded by Eq. (3.17), hence the Compensated Coupling concludes the proof.

3.7 Numerical Experiments

The theoretical results we have presented, together with known lower bounds for previous algorithms, show that our approach outperforms existing heuristics for online packing and matching problems. We now re-emphasize these results via simulation with synthetic data, which demonstrates both the suboptimality of existing heuristics (in terms of expected regret which scales with T), as well as the fact that the Bayes selector has constant expected regret.

We run experiments for both online packing and online matching with multinomial arrivals. For each problem we consider two instances, i.e., two sets of parameters (r, A, p), then we scale each instance to obtain a family of ever larger systems. For each scaling we run 100 simulations. In conclusion, we run four sets of parameters (two for packing and two for matching), each scaled to generate many systems. The code for all the algorithms can be found in https://github.com/albvera/bayes_selector.

3.7.1 Online Packing

	Type j					
	1	2	3	4	5	6
Resource $i = 1$	1	1	0	0	1	1
Resource $i = 2$	0	0	1	1	1	1
p_j	0.2	0.2	0.2	0.2	0.1	0.1
r_j	10	6	10	5	9	8

Table 3.1: Parameters for the first online packing instance. Coordinates (i, j) represent the consumption A_{ij}.

We compare the Bayes Selector against three policies: (i) Static Randomized (SR) is the first known policy with regret guarantees, it is based on solving the fluid LP once and using the solution as a randomized acceptance rule [103]. (ii) Re-solve and Randomize is based on re-solving the fluid LP at each time and using the solution as a randomized acceptance rule

[78]. (iii) Infrequent Re-solve with Thresholding (IRT) is based on re-solving the fluid LP at carefully chosen times, specifically at times $\{T^{(5/6)^u} : u = 0, 1, ..., \log\log(T)/\log(6/5)\}$, then either randomize or threshold depending on the value of the solution [37].

Our first instance has $d = 2$ resources and $n = 6$ customer types. Types $j \in \{1, 2\}$ require one unit of resource $i = 1$, types $j \in \{3, 4\}$ require one unit of $i = 2$, and types $j \in \{5, 6\}$ require one unit of each resource. All the parameters are presented in Table 3.1. We consider a base system with capacities $B_1 = B_2 = 40$ and horizon $T = 200$. The base system is chosen such that the problem is near dual-degenerate (which is the regime where heuristics based on the fluid benchmark are known to have poor performance; see Proposition 3.2.1). Finally, for a scaling $k \in \mathbb{N}$, the k-th system has capacities kB and horizon $(k + k^{0.7})T$. We remark that traditionally the horizon is scaled as kT, but we chose this slightly different scaling to emphasise that our result does not depend on the specific way the system is scaled.

The results for the first instance are summarized in Fig. 3.2, where we also present a log-log plot which allows better to appreciate how the regret grows. Static Randomized has the worst performance in our study; indeed, we do not include it in the plot since it is orders of magnitude higher. We note that not only the Bayes Selector outperforms previous methods, but the regret is very small (both in average and sample-path wise), specially in comparison with the overall reward which grows linearly with k, i.e. $v^{\text{off}} = \Omega(k)$ (in expectation and w.h.p.).

The second instance has $n = 15$ customer types and $d = 20$ resources, the parameters were generated randomly. We take a base system with horizon $T = 50$ and capacities $B_i = 10$ for all $i \in [20]$, then the k-th system has horizon kT and capacities kB. The performance of different algorithms is presented in Fig. 3.3. We notice that this instance is not degenerate and we are scaling linearly, hence all the algorithms except Static Randomize (which we again omit from the plots) are known to achieve constant regret. Nevertheless, we observe

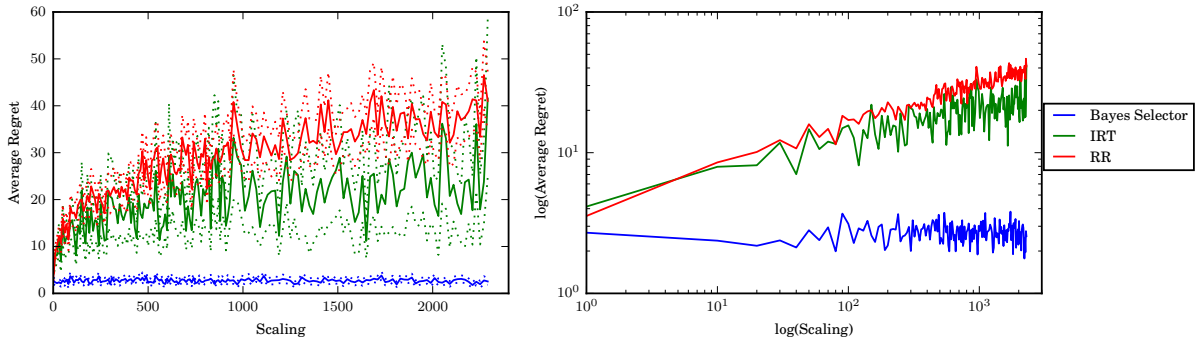


Figure 3.2: Average regret of different policies for online packing in the first instance. We present a plot on the left and a log-log plot on the right. We run the Bayes Selector, Infrequent Re-solve with Thresholding (IRT) [37], Re-solve and Randomize (RR) [78], and Static Randomized (SR) [103] (this last one is not reported because its high regret distorts the figures). The plot shows the regret incurred by the policies versus the offline optimum, for different scalings. Dotted lines represent 90% confidence intervals.

that the Bayes Selector has the best performance by a large margin.

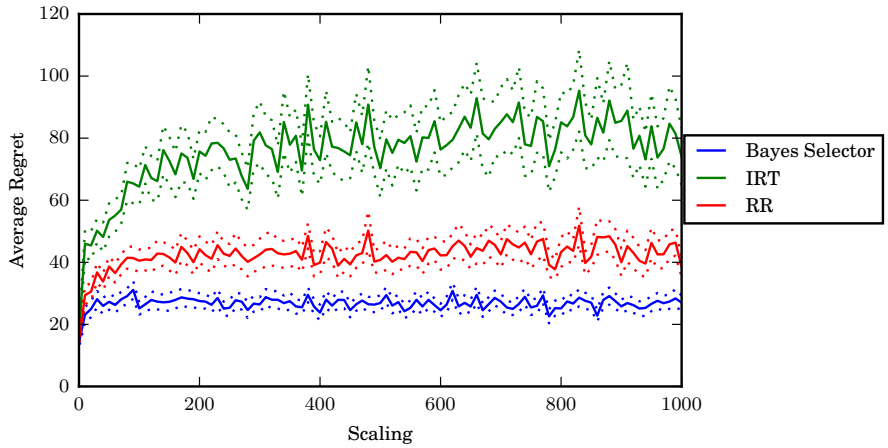


Figure 3.3: Average regret of different policies for online packing in the second instance. We run the Bayes Selector, Infrequent Re-solve with Thresholding (IRT) [37], Re-solve and Randomize (RR) [78], and Static Randomized (SR) [103] (this last one is not reported because its high regret distorts the figures). The plot shows the regret incurred by the policies versus the offline optimum, for different scalings. Dotted lines represent 90% confidence intervals.

3.7.2 Online Matching

As we mentioned in Section 3.5, our problem corresponds to stochastic matching with edge weights. There has been previous work studying constant factor approximations for worst-case distributions. In particular, the state of the art is a 0.705 competitive ratio [32], while a previous algorithm achieved a 0.667 competitive ratio [71]. Both algorithms are impractical since they require a sampling procedure over poly($T \cdot \max_{i\in[d]} B_i$) many matchings. To the best of our knowledge, the best guarantee of a practical algorithm is a $1 - 1/e \approx 0.63$ competitive ratio and is achieved by the base algorithm in [71] (that when built upon achieves the 0.667 guarantee). We therefore benchmark against this algorithm, which we call "Competitive".

Competitive is based on solving a big LP once (it has $\Omega(T \cdot \max_{i\in[d]} B_i)$ variables) and using the solution as a probabilistic allocation rule. We also compare against a contemporaneous algorithm, called Marginal Allocation, that is based on bid-prices [105]. Marginal Allocation uses approximate dynamic programming to obtain the marginal benefit of a matching, then uses this marginal value as a bid price so that, if the reward exceeds it, then we match the request. We give further details for both Marginal Allocation and Competitive in Appendix B.3.

The first instance we consider has $d = 2$ resources and $n = 6$ customer types. The specific parameters are presented in Table 3.2, where reward $r_{ij} = 0$ implies that type j cannot be matched to that resource i, i.e., $A_{ij} = 0$. We consider a base system with horizon $T = 20$ and capacities $B = (4, 5)'$ and then scale it linearly so that the k-th system has horizon kT and capacities kB. Our second instance has $d = 6$ resources and $n = 10$ customer types, the specific parameters are presented in Table B.1 in Appendix B.3. We consider a base system with horizon $T = 200$ and capacities $B = (40, 50, 40, 30, 20, 40)'$ and then scale it linearly so that the k-th system has horizon kT and capacities kB.

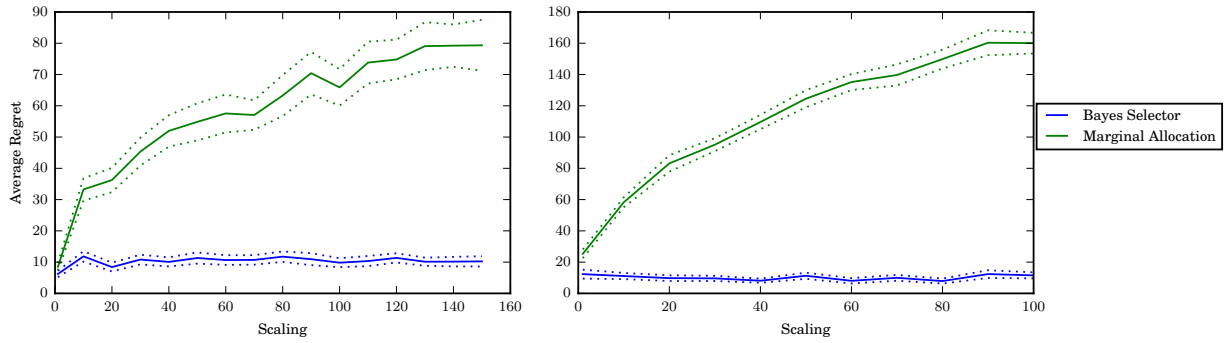


Figure 3.4: Average regret of different policies for online matching. First instance on the left and second on the right. We run the Bayes Selector, Marginal Allocation [105], and Competitive [71] (this last one is not reported because its high regret distorts the figures). The plot shows the regret incurred by the policies versus the offline optimum, for different scalings.

	Type j					
	1	2	3	4	5	6
Resource 1	10	6	0	0	9	8
Resource 2	0	0	5	10	20	20
p_j	0.2	0.2	0.2	0.2	0.1	0.1

Table 3.2: Parameters used for the first online matching instance. Coordinates (i, j) represent the reward r_{ij} and $r_{ij} = 0$ implies that it is not possible to match i to j.

The results are presented in Fig. 3.4. We do not include the regret of Competitive, since it is so high that it distorts the plots (it starts at 80 times the regret of the other algorithms and then grows linearly with k). We can confirm that the Bayes Selector has constant regret and, additionally, offers the best performance. Marginal Allocation offers a much better performance than Competitive, but its regret still grows and seems to scale as $\Omega(\sqrt{T})$.

3.8 Discussion

We reiterate that our contributions in this paper are both to develop new online policies that achieve constant regret for a large class of online allocation problems, and also, a new technique for analyzing online decision-making heuristics.

Our work herein has developed a new technical tool—the Compensated Coupling—for analyzing online decision making policies with respect to offline benchmarks. In short, the main insight is that, through the use of compensations, we can couple Offline's state to that of Online on every sample path. This simplifies the analysis of online policies since, in contrast to existing approaches, we do not need to track the complicated offline process.

Next, we presented a general class of problems, called Online Allocation, where different customers request different bundles of resources. This problem captures, among others, Online Packing and Online Matching. For all of these problems we present a tractable policy, the Bayes Selector, based on re-solving an LP, that achieves constant regret.

Our analysis is based on the Compensated Coupling and, thanks to its versatility, we can accommodate a large class of arrival processes including: correlated processes, heavy tailed, and the classical Poisson and Multinomial (i.i.d.).

Although we instantiate the Bayes Selector for Online Allocation, we defined it for general MDPs; we hope this policy is useful for other types of problems too. We remark two properties of the Bayes Selector: (i) it works on interpretable quantities, namely the estimation of disagreement probabilities $\hat{q}$, and (ii) it is amenable to simulation, since $\hat{q}$ can be estimated by running offline trajectories. We therefore think that a promising avenue for further research is to apply this policy to other problems using modern estimation techniques.

The assumption of finite types of customers is well founded in revenue management problems, but there are settings where the number of types could be very large or even continuous. Based on reported numerical results [10], the Bayes Selector appears to have good performance even in this setting. An interesting problem is to obtain parametric guarantees (not worst case) in the case where the number of types is very large or continuous.

CHAPTER 4

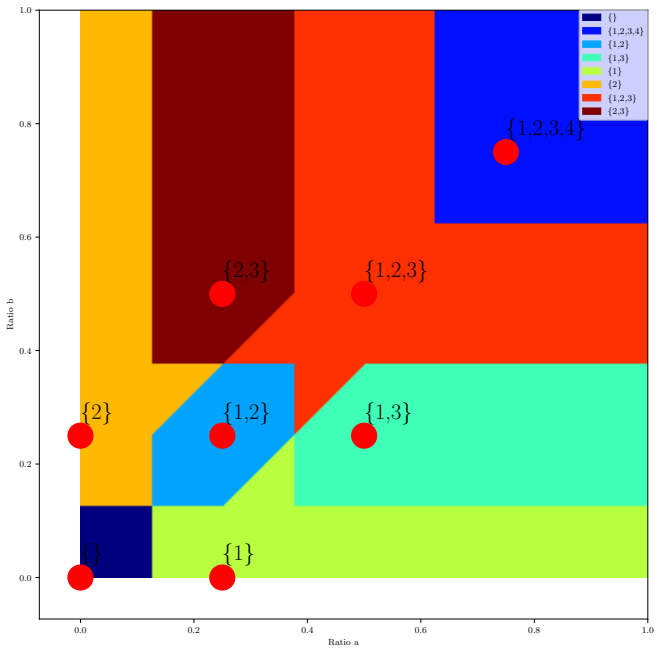
GEOMETRY AND ROBUSTNESS OF CONSTANT REGRET

4.1 Introduction

We study a family of dynamic resource allocation problems. Requests of multiple types arrive over a finite horizon of T periods. If served, depending on its type, a request consumes a set of resources and generates a reward. There is an inventory of resources available at time 0 and additional units can arrive over time (restock). In general, a request can be either impatient — if not served immediately upon arrival it is lost — or patient, in which case the controller can choose to place them in queue. The controller's trade-off is to use "correctly" its resource capacity to maximize the total reward collected over the finite horizon.

We refer to this large family of problems as *Resource Allocation Networks* or RAN. If the controller could solve the problem in an *offline* fashion, she would wait until the end of the horizon and, given the realization of the random arrivals, choose the best allocation of resources. The reward of the offline (or hindsight) decision maker is obviously an upper bound on any online algorithm. *In this paper, we obtain several performance guarantees and robustness results compared to the offline upper bound.*

The algorithm we study, which we refer to as BudgetRatio, is based on re-solving a *packing* linear program at each time. Specifically, let $(p_j)_{j\in\mathcal{J}}$ be the arrival probabilities of requests $\mathcal{J}$ and $(p_i)_{i\in\mathcal{R}}$ the restock probabilities of resources $\mathcal{R}$. If at time t the inventory of different resources is $I^t \in \mathbb{N}^{\mathcal{R}}$, then we obtain $\bar{y}^t$ as the solution to $\mathrm{LP}(\frac{1}{T-t}I^t + p_{\mathcal{R}}, p_{\mathcal{J}})$ (see Fig. 4.1). Intuitively, the inventory plus expected restock is $I^t + (T-t)p_{\mathcal{R}}$ and we divide it into per-period expected available inventory. Similarly, we divide the expected arrivals of requests into per-period arrivals.



$$\begin{aligned} \max \quad & v'y \\ \text{s.t.} \quad & Ay \leq R^t := \tfrac{1}{T-t} I^t + p_{\mathcal{R}} \\ & y \leq p_{\mathcal{J}} \\ & y \geq 0. \end{aligned}$$

Figure 4.1: Action regions for a problem with two resources and four request types. The plot is in the space of ratios that represent the per-period resource availability $R_i = \frac{1}{T-t} I^t + p_i$ for $i \in \{a, b\}$. Each point corresponds to a pair of budget states (R_a, R_b). When solving the LP, we obtain the set of request types $\mathcal{K} = \{j : \bar{y}_j \geq \eta_j\}$ that should be accepted at that inventory level. Each color on the plot corresponds to a different such set. For example, the light-blue region shaped like a rhombus corresponds to $\mathcal{K} = \{1, 2\}$. The specific parameters are $p_{\mathcal{J}} = (1/4, 1/4, 1/4, 1/4)$, $p_{\mathcal{R}} = 0$, $v = (4, 4, 5, 1)$, and $A = (1, 0, 1, 1; 0, 1, 1, 1)$.

The number $\bar{y}_j^t$ is a proxy for the number of type-j requests that we want to accept: an inventory-dependent "score" of type j. BudgetRatio accepts requests with sufficiently large score, i.e., such that $\bar{y}_j^t \geq \eta_j$ for some thresholds η_j specified later. Observe that the scores $\bar{y}^t$ depend on the random *budget-ratio* process $R^t := \frac{1}{T-t} I^t + p_{\mathcal{R}}$. This random process, evolving in the space of scaled resources, drives our analysis, see Figure 4.1.

Methodology: a geometric view of re-solving policies. The problems we consider cannot be solved optimally due to the so-called "curse of dimensionality". This fact alone justifies the pursuit of policies that are simple to implement, adapt, and scale according to the problem instance. Linear-programming based algorithms have been introduced to overcome this challenge.

Our motivation in this paper is not purely algorithmic. We uncover the fundamental structure of the online stochastic packing problem and its optimal solution by taking a stochastic-process view of the dynamics. Our proof method captures the geometric nature of the problem and the dynamics of the budget consumption as a *stochastic-process* in the space $\mathbb{R}_+^{\mathcal{R}}$ of budget ratios. Our analysis makes explicit how BudgetRatio interacts with the geometry of the packing LP.

The thresholding of the decision $\bar{y}^t$ divides the space of resource-budgets $\mathbb{R}_+^{\mathcal{R}}$ into mutually exclusive *action* regions. When the ratio is in a given region all requests of a subset of types are accepted and all other are rejected, see Fig. 4.1.

To achieve bounded regret, the online policy must act in a way that is consistent with the optimal, unknown, offline basis. The policy must perform, what we call, *basic allocations* (see Definition 4.3.1). Thresholding—which underlies Figure 4.1— guarantees that, notwithstanding the unrevealed offline basis, the algorithm performs basic allocations. To establish this we must (i) develop a *generalizable mathematical description* of Figure 4.1 and (ii) study the dynamic of the stochastic process R^t inside and between the actions regions in this figure.

Since our geometric view exposes the structure of the problem, we can use it to study other popular algorithms. Indeed, we can explain the "failure" of the natural randomized re-solving algorithm and also give an appealing interpretation of BudgetRatio as bid-price control. In the context of online packing (a.k.a network revenue management), the standard bid-price algorithm solves the packing the LP and accepts a request if its reward exceeds a bid-price that captures the *opportunity cost* of serving the request. The bid-price is the sum of the shadow prices of the requested resources. To achieve bounded regret, BudgetRatio is more careful: it is equivalently formulated as a bid-price control where the bid-price is obtained from a maximum over several shadow prices; see Section 4.8.

We study the robustness of BudgetRatio along multiple dimensions:

Model ingredients: the effect of queues and restock. In the baseline setting of online packing (or network revenue management), requests are impatient (leave if not immediately served), and inventory is not restocked, i.e., there is only initial inventory that is gradually depleted. Our geometric view of the problem allows to elicit the effect patient customers and restock on (i) the very feasibility of achieving bounded regret—characterizing when such regret is or is not achievable and, (ii) when feasible, on achieving said regret with the *same* algorithm.

We show that the offline benchmark is generally too ambitious. Indeed, in the absence of (what we call) a "slow restock" assumption, the offline upper bound is not generally achievable. Under the slow restock assumption, BudgetRatio achieves bounded regret. The thresholds, however, must be suitably perturbed by a quantity that captures the restock probability.

With patient requests, it seems reasonable that utilizing queued requests can increase the rewards. Instead, we prove that—with slow restock—constant regret for total reward can be achieved without interacting with the queues. In particular, this shows the robustness of the algorithm w.r.t. the behaviour of patient customers (e.g., infinitely patient, geometric patience, etc).

The performance metric. The algorithm's threshold can be tuned to achieve nested objectives: to minimize—in first order—the holding cost over the horizon subject to near optimal rewards. In other words, we consider the disutility of holding requests or inventory as well as the rewards. We show that, within the family of *all* policies that are nearly optimal for reward, BudgetRatio—with carefully tuned thresholds—achieves the minimal holding costs in first order.

This result hinges to a great extent on the understanding of the geometry of the problem and the dynamics of the remaining inventory as a function of the thresholds that are used.

Parameter misspecification. Our geometric analysis uncovers the sensitivity of the optimality gap to errors in the forecasting of the demand and/or the rewards. Specifically, we study the case where the true parameters (values and probabilities) are (v, p) and we run the algorithm with $(\tilde{v}, \tilde{p})$ instead. We quantify how accurate $(\tilde{v}, \tilde{p})$ need to be so that BudgetRatio remains near optimal.

Put simply, as long as the centroids remain unchanged, the collection of action regions in Fig. 4.1 are stable under perturbations of the parameters. It is known that, in one dimension, $\tilde{v}$ needs to be accurate enough to deduce the ranking of the requests (see Chapter 5). Surprisingly, the centroids provide a generalization of the inherently one-dimensional notion of ranking, thus allowing us to understand the multidimensional problem. Our robustness guarantees easily yield the optimal regret guarantees in the learning setting (bandits with knapsacks).

4.2 Setting and Overview of Results

A decision maker must allocate resources to requests over a horizon of T periods. There is a set of resources $\mathcal{R} = \{1, \ldots, d\}$ and, at time $t = 0$, there is an initial inventory I_i^0 for each $i \in \mathcal{R}$. Additionally, at each time $t \in [T]$, a unit of resource i arrives with probability p_i; we call this the *restock* probability. We let $(Z_i^t : i \in \mathcal{R})$ be the accumulated restock over the time interval $[1, t]$. The controller cannot consume more than $I_i^0 + Z_i^t$ units of resource i by time t.

There is a set $\mathcal{J} = \{1, \ldots, n\}$ of possible requests, each request $j \in \mathcal{J}$ generates a reward

v_j and requires some resources; resource consumption is encoded in a matrix $A \in \{0,1\}^{d\times n}$, where $A_{ij} = 1$ means that type j requires one unit of resource i. We identify resources $\mathcal{R}$ and requests $\mathcal{J}$ with natural numbers for simplicity, but they correspond to different entities, i.e., $\mathcal{J} \cap \mathcal{R} = \varnothing$. At each time $t \in [T]$, a request j arrives with probability p_j independently of the past with $\sum_{j\in\mathcal{J}} p_j = 1$. We let $(Z_j^t : j \in \mathcal{J})$ be the accumulated arrivals over $[1, t]$.

Resources accumulate, i.e., if not used by time t, they are available at $t+1$. We allow for two kinds of requests: patient and impatient. If requests are impatient, they must be served when they arrive or not at all. In the classical NRM setting all requests are impatient. In contrast, patient requests queue when not served at their arrival time. The controller knows in advance which types $j \in \mathcal{J}$ are patient and which are not. This modelling flexibility allows us to capture, in one framework, both online packing and assembly. Decisions are final: if a request j is served, the resources are consumed. Similarly, an impatient request that is rejected is lost forever.

We let V^t be the reward brought by the request arriving at time t. Thus, $V^1, V^2, \ldots, V^T$ are i.i.d with $\mathbb{P}[V^t = v_j] = p_j$, $j \in \mathcal{J}$. The inventory on hand at time $t \in [T]$ is denoted by $I^t = (I_1^t, \ldots, I_d^t)'$. Say the arrival is of type j, i.e., $V^t = v_j$, then the selection process unfolds as follows. If $I^t \not\geq A_j$ (the available inventory is insufficient), then the request must be queued. On the other hand, if the request is feasible ($A_j \leq I^t$), then it may be served, thereby generating a reward of v_j and decreasing the inventory to $I^t - A_j$, or it may be queued. Queued requests corresponding to previous patient arrivals, may be served at any period after their arrival.

No online policy can do better than the offline, full information, counterpart in which all values are presented in advance. Allowing this offline to use fractional allocations gives a further upper bound. This fractional offline controller is our benchmark. The expected total reward of the offline problem is given by the expectation of the following linear program,

where $Z_{\mathcal{R}}$ are resource arrivals (restock) and $Z_{\mathcal{J}}$ are request arrivals:

$$V^*_{off}(T, I^0) := \mathbb{E}\left[\begin{array}{lll} \max & v'y & \\ \text{s.t.} & Ay & \leq I^0 + Z^T_{\mathcal{R}} \\ & y & \leq Z^T_{\mathcal{J}} \\ & y & \in \mathbb{R}^n_{\geq 0} \end{array}\right]. \tag{4.1}$$

We make the following assumption throughout:

Assumption 4.2.1 (Slow Restock). For each request j and each resource i with $A_{ij} = 1$, we have $p_i < p_j$, i.e., the restock is bounded by the demand.

If Assumption 4.2.1 fails, then the regret generally grows proportionally to $\sqrt{T}$. To show this, it suffices to consider a single type of impatient requests arriving at rate p, and a single resource also arriving at rate p. This corresponds to a *critically loaded* queue.

Lemma 4.2.2. *There exists a RAN that violates Assumption 4.2.1 and such that, for some $c > 0$,*

$$V^*_{off}(T) - V^*_{on}(T) = c\sqrt{T} + o(\sqrt{T}), \text{ as } T \to \infty,$$

where V^*_{on} is the optimal policy. Hence, no policy can achieve $o(\sqrt{T})$ regret.

The proof is in Appendix C.1. A slow restock assumption is thus necessary. When it fails, offline is no longer a useful benchmark for performance measurement.

4.2.1 Main Results

Theorem 4.2.3 (Constant Regret). *Assuming slow restock,* BudgetRatio *(Algorithm 5) has a uniformly bounded regret: there exists a constant M such that*

$$V^*_{off}(T, I^0) - V_{on}(T, I^0) \leq M,$$

where $V_{on}(T, I^0)$ is the total reward of BudgetRatio. *The constant M may depend on (p, v, A), but it is independent of (T, I^0). Furthermore,* BudgetRatio *ignores the queued requests, hence it is robust to the modelling choice of patient customers.*

Remark 4.2.4 (modelling patience). We assume for simplicity that customers are either impatient or have infinite patience. Our main result (Theorem 4.2.3) gives something stronger: the regret guarantee does not depend on the patience model, be it deterministic (departure after D periods), geometric (departure with probability θ in each period) or any other.

Slow restock includes as a special case the online packing setting (where there is no restock, i.e., $p_{\mathcal{R}} = 0$). The fact that, in the absence of restock, the ability to queue request does not change the regret guarantee can be deduced from earlier work. What is interesting here is that even in the presence of restock (and even with no initial inventory), it is not valuable to use queued requests at a later point when, possibly, more resources become available.

Robustness results. In Fig. 4.1, each action region corresponds to the set of types accepted when the budget ratio is in this region. The red circle corresponds to the budget (the resource capacity) required to serve in expectation the corresponding types. The point $(0.25, 0.25)$ is the *centroid budget* for the *centroid* corresponding to the set $\{1, 2\}$; see Section 4.4 for formal definitions.

As is clear in Figure 4.1, these red circles anchor the geometry of the action regions. It thus makes sense that robustness can be specified in terms of these points.

For two centroids $\mathcal{K}, \mathcal{K}'$, we measure the *separation* of their budgets under the distribution p:

$$d_p^{\min}(\mathcal{K}, \mathcal{K}') := \min_{i \in [d]} \left| (r_{\mathcal{K}}(p) - r_{\mathcal{K}'}(p))_i \right|.$$

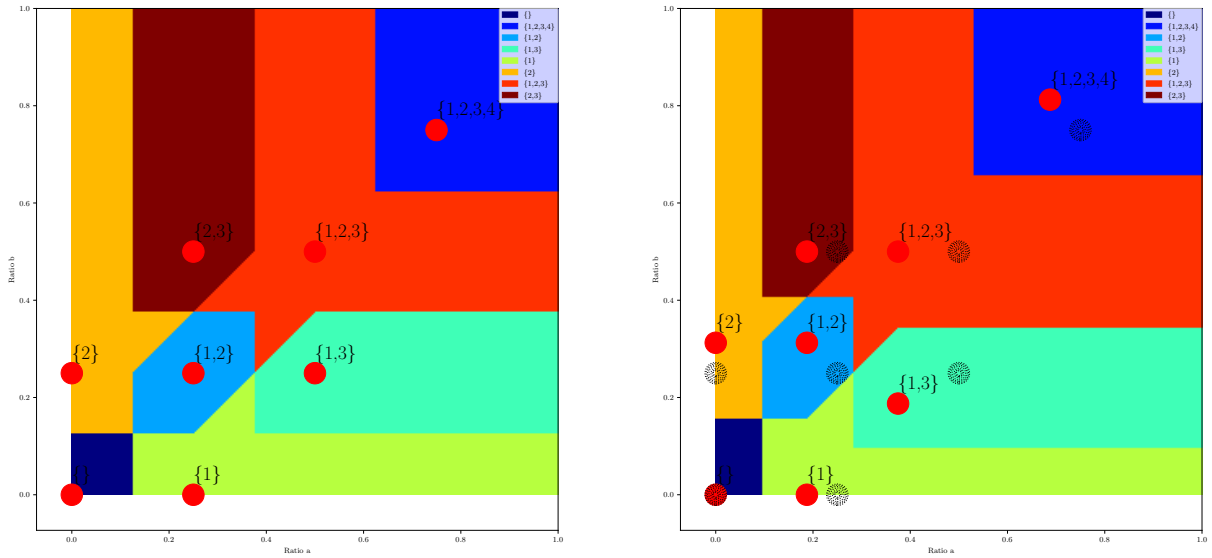


Figure 4.2: Action regions with true and misspecified probabilities (p and $\tilde{p}$). On the left we have the areas with the true values p and on the right we have a perturbation $\tilde{p}_j = p_j - \frac{1}{16}$ for $j = 1, 3$ and $\tilde{p}_j = p_j + \frac{1}{16}$ for $j = 2, 4$. We observe that the structure of centroids and neighbourhoods is maintained, but the areas have slightly different shapes due to the fact that the centroid budgets change fro $r_{\mathcal{K}}(p)$ to $r_{\mathcal{K}}(\tilde{p})$. It is crucial that the centroid budgets still lie in the interior of the regions: on the right the true centroid budget are the dashed circles.

In Figure 4.1 we can deduce $\delta = 1/4$. The key insight, illustrated in Figure 4.2, is that the true centroid budget remain in the interior of the action regions of the algorithm with the perturbed "wrong" probabilities.

Proposition 4.2.5 (demand mis-specification). *Let p be the true underlying distribution and let $\delta = \min_{\mathcal{K} \neq \mathcal{K}'} d_p^{\min}(\mathcal{K}, \mathcal{K}')$. Then, for any distribution $\widetilde{p}$ such that*

$$\max_{\mathcal{K}} ||r_{\mathcal{K}}(p) - r_{\mathcal{K}}(\widetilde{p})||_\infty \leq \frac{\delta}{4},$$

BudgetRatio*, with the LP solved each period with $\widetilde{p}$ replacing p, produces bounded regret in the sense of Theorem 4.2.3. The condition is satisfied, in particular, if $||p - \tilde{p}||_\infty \leq \frac{\delta}{4n}$.*

In Figure 4.1, $d_p^{min} \equiv \frac{1}{4}$ so that it is enough to have an estimation of the centroid budgets $r_{\mathcal{K}}(p)$ with accuracy $\frac{\delta}{4} = \frac{1}{16}$. This, notice, is a constant accuracy, i.e., does not need to

improve with the horizon T.

The next result, concerning the robustness to misspecification of the reward vector v, hinges on a notion of strict complementarity (see Section 4.7.2 for details). In Figure 4.1, at each point (R_a, R_b) in the space, the optimal basis for the packing LP is different. In the following, we require strict complementarity for all those optimal bases.

Proposition 4.2.6 (value misspecification). *Let v be the true values and let δ be such that all the bases are δ-complementary. There exists a constant $c \geq 1$ such that, for any estimation $\tilde{v}$ satisfying*

$$||v - \tilde{v}||_\infty \leq \frac{\delta}{c(d+2)},$$

BudgetRatio*, with the LP solved each period with $\tilde{v}$ replacing v, produces bounded regret in the sense of Theorem 4.2.3. Furthermore $c \leq \max_{\mathcal{B}} ||\mathcal{B}^{-1}||_\infty$.*

In the example of Figure 4.1, we can take $\delta = c = 1$ (computed numerically), so that the robustness region is $||v - \tilde{v}||_\infty \leq \frac{1}{4}$. It is important that c, δ do not depend on p or the horizon, only on (v, A).

Consider next the setting where the holding of inventory (or of customers, if they cannot be rejected) is costly. In that case, there is a tradeoff between maximizing rewards and consuming inventory (or emptying queues) as soon as possible.

We show that, with suitable tuning of the thresholds, among all policies with regret $o(T)$, BudgetRatio has the optimal holding cost scaling. Specifically we consider two cases for holding cost: (i) each period a request type j is not served, we incur a cost $c_j > 0$ and (ii) each period a resource i is in stock we incur a cost $h_i > 0$. See Section 4.6 for details and illustrative graphics. The holding cost of a policy is

$$\mathcal{C}^\pi(T, I^0) := \mathbb{E}^\pi_{I^0}\left[\sum_{t=1}^{T} c \cdot Q^t\right] \quad \text{or} \quad \mathcal{C}^\pi(T, I^0) := \mathbb{E}^\pi_{I^0}\left[\sum_{t=1}^{T} h \cdot I^t\right].$$

Proposition 4.2.7 (dual objectives). *Suppose that the deterministic relaxation* $\mathrm{LP}(\mathbb{E}[R^0], D)$ *has a unique solution* $\bar{y}$ *and that* $\bar{y}_j < p_j$ *for at least one* $j \in \mathcal{J}$. *Then,* $\widehat{\pi}$ =BudgetRatio *with threshold* $\alpha^T = T^{-1/4}$ *achieves simultaneously (1) constant regret for reward maximization and (2) asymptotic optimality for cost minimization, i.e.,*

$$\liminf_{T\uparrow\infty} \frac{\mathcal{C}^{\pi}(T, I^0)}{\mathcal{C}^{\widehat{\pi}}(T, I^0)} \geq 1 \quad \textit{for any policy } \pi \textit{ with regret } o(T).$$

The meaning of bounded regret. Theorem 4.2.3, states that the *additive (not multiplicative) gap* in collected reward between our algorithm (BudgetRatio) and the optimal algorithm (that requires the solution of a dynamic program), is bounded by a constant that does not depend on length of the horizon T or the initial inventory I^0. The proof establishes that BudgetRatio makes at most M (constant) mistakes relative to the optimal decision maker. It makes at most M mistakes over an horizon of $T = 10$ periods, and also at most M mistakes if the horizon is $T = 10^6$. Since

$$\text{Regret } = (\text{ Number of errors }) \cdot (\text{ Maximal cost of a single error}),$$

a finite number of mistakes translates into bounded regret.

The (multiplicative) approximation factor is then $1 - O(\frac{1}{\min_i\{T, I_i^0\}})$. In percentage terms the error becomes negligible as the horizon length grows.

An alternative notion for algorithm evaluation is that of competitive ratio: an α approximation algorithm is guaranteed to achieve an α fraction of the optimal reward. Bounded regret is not universally a stronger optimality notion than an α competitive ratio. In regimes where the cost of a single mistake is catastrophic, e.g., small T or small I^0, a competitive algorithm may be preferable.

For most operational setting (network revenue management, inventory management etc.),

regret is a more reasonable notion for algorithm evaluation.

4.2.2 Algorithm

Our analysis is driven by the concept of future budget, which corresponds to inventory at hand plus *expected* future restock. Similarly, future demand corresponds to current queue plus expected future requests. Throughout we denote $p_{\mathcal{R}}$ as the vector of restock probabilities and $p_{\mathcal{J}}$ the vector of request arrival probabilities.

Definition 4.2.8 (Budget Ratio). If the inventory at hand at time t is I^t, then the ratio at t is

$$R^t \coloneqq \frac{1}{T-t}(I^t + \mathbb{E}[Z_{\mathcal{R}}^T - Z_{\mathcal{R}}^t]) = \frac{1}{T-t} I^t + p_{\mathcal{R}}, \qquad t \in [1, T].$$

The ratio at $t = 0$ is defined by the random variables (without expectation) $R^0 \coloneqq \frac{1}{T}(I^0 + Z_{\mathcal{R}}^T)$. The demand at time $t = 0$ is defined by $D^0 \coloneqq \frac{1}{T} Z_{\mathcal{J}}^T$.

The deterministic relaxation, in Eq. (4.2) below, is obtain by considering a parametric view of the offline LP in Eq. (4.1). Our policy, presented in Algorithm 5, re-solves the deterministic relaxation and thresholds its solution accordingly.

$$\begin{aligned} \text{LP}(R, D) \qquad \max \quad & v'x \\ \text{s.t.} \quad & Ay \leq R, \\ & y \leq D, \\ & y \in \mathbb{R}^n_{\geq 0}. \end{aligned} \tag{4.2}$$

Remark 4.2.9 (The Aggressiveness Parameter α). The parameter α is used to tune the algorithm. Note that the smaller the value of α, the more easily requests are accepted (see step 8 of Algorithm 5). The main result (Theorem 4.2.3) is obtained by setting $\alpha = 1/2$ and we set it to this value up to Section 4.6, where we will use the flexibility to obtain

Algorithm 5 Budget Ratio Policy

Input: Aggressiveness parameter $\alpha \in (0,1)$

Set thresholds: for $j \in \mathcal{J}$, let $\delta_j := \max_{i:A_{ij}=1} p_i$ and set $\bar{p}_j \leftarrow p_j + \delta_j \mathbb{1}_{\{\delta_j > \alpha p_j\}}$.
for $t = 1, \dots, T$ **do**
    If a request $j \in \mathcal{J}$ arrived, $Q_j^t \leftarrow Q_j^t + 1$. If a resource $i \in \mathcal{R}$ arrived, $I_i^t \leftarrow I_i^t + 1$.
    Set $R^t \leftarrow \frac{1}{T-t} I^t + p_{\mathcal{R}}$.
    Solve $\mathrm{LP}(R^t, p_{\mathcal{J}})$ to obtain the optimal decision variables $\bar{y}$.
    for requests $j \in \mathcal{J}$ in decreasing order of $\bar{y}_j / \bar{p}_j$ **do**
        if $Q_j^t = 0$ or $I^t \not\geq A_j$: not feasible to serve j
        else if $\bar{y}_j \geq \alpha \bar{p}_j$: accept a request j. Update $Q_j^t \leftarrow Q_j^t - 1$, $I^t \leftarrow I^t - A_j$.
    Deplete the queues of all impatient requests: $Q_j^t \leftarrow 0$ if $j \in \mathcal{J}$ is impatient.
    Update for next period $I^{t+1} \leftarrow I^t$ and $Q^{t+1} \leftarrow Q^t$

robustness results. Furthermore, from our constructions it is clear (a posteriori), that α can be set to any fixed constant, hence in practice it can be a useful tuning parameter to improve the performance. In fact, this parameter can also shrink with T (more details in Section 4.6).

Remark 4.2.10 (the effect of restock: $\bar{p}$ vs. p). In the online packing setting (no restock), we have $\bar{p} = p$. In the presence of restock, its rate $p_{\mathcal{R}}$ must be taken into account. To build intuition, take two types j, j' and suppose that they have identical solutions $\frac{1}{2} < \bar{y}_j / p_j = \bar{y}_{j'} / p_{j'} < 1$. A priori it is not clear which type should be prioritized, but if, for example, $0 = \delta_{j'} < \delta_j$, then the inventory availability for j' will not improve in the future while that for j will. Thus, j' is more valuable because of its relative scarcity. This is captured by setting $\bar{p}_{j'} = p_{j'}$ and $\bar{p}_j = p_j + \delta_j$.

Final Setup Details. Let $\mathcal{F}_0$ denote the trivial σ-field. For $t \in [T]$, let $\mathcal{F}_t = \sigma\{Z^\tau : \tau = 1, \dots, t\}$ be the σ-field generated by the random arrivals (of resource and requests). An online policy π can be expressed with binary random variables $(\sigma_j^{\pi,t} : j \in \mathcal{J})$ such that $\sigma_j^{\pi,t} = 1$ means that a request type j is served at time t. Observe that, if $V^t = v_j$ and $\sigma_j^{\pi,t} = 0$, then the arriving request is queued; for online adapted policies $\sigma^{\pi,t}$ must be $\mathcal{F}_t$-measurable. Let

$$Y_j^{\pi,t} := \sum_{\tau \in [t]} \sigma_j^{\pi,\tau},$$

be the total number of type-j requests accepted over $[1, t]$. A policy is feasible if (1) the total consumption of resource i does not exceed its initial inventory I_i^0 plus the restock and (2) the total acceptance does not exceed arrivals, i.e.

$$\begin{aligned} AY^{\pi,t} &\leq I^0 + Z_{\mathcal{R}}^t, \ t \in [T], \\ Y^{\pi,t} &\leq Z_{\mathcal{J}}^t, \ t \in [T] \\ \sigma_j^{\pi,t} &\leq \mathbb{1}_{\{V^t = v_j\}}, \ t \in [T], j \in \mathcal{J}, j \text{ impatient.} \end{aligned} \tag{4.3}$$

Let Π be the set of feasible online policies, i.e., $\mathcal{F}_t$-adapted and satisfying Eq. (4.3). For $\pi \in \Pi$, the total reward of an online policy π is then given by

$$V^\pi(T, I^0) = \mathbb{E}\left[\sum_{t \in [T]} v' \sigma^{\pi,t}\right].$$

For each (T, I^0), the goal of the decision maker is to maximize the expected value:

$$V^*(T, I^0) = \max_{\pi \in \Pi} V^\pi(T, I^0).$$

Solving for V^* directly is infeasible for most realistically-sized problems. To prove optimality guarantees, we therefore compare BudgetRatio against the offline benchmark previously defined.

Additional Notation. Given a subset $\mathcal{K} \subseteq \mathcal{J}$ we let $A_{\mathcal{K}}$ be the submatrix of A that only columns in the index set $\mathcal{K}$ (but has all rows). We similarly define sub-vectors: if x is a column vector, $x_{\mathcal{K}}$ is a subvector with the indices in the set $\mathcal{K}$. We use $\mathbf{e}$ for the vector of ones with dimension that will be clear from the context. We use $\mathbf{e}_{\mathcal{K}}$ for a vector that has 1 for $j \in \mathcal{K}$ and 0 otherwise.

For real numbers x, y, ϵ, we write $x = y \pm \epsilon$ if $|x - y| \leq \epsilon$. Finally, following standard notation, for a subset $D \subseteq \mathbb{R}^d$ and a point $x \in \mathbb{R}^d$ we let $d(x, D) = \inf_{y \in D} \|x - y\|$ be the

distance of x from the set D. We adopt the convention that the maximum over the empty set is zero, i.e., $\max \varnothing = 0$. We use throughout M to be a constant —that can depend only on (A, p, v), but it is independent of (T, I^0)— whose value can change from one line to the next.

4.2.3 Related Work

Online Packing (Network Revenue Management). Our work follows up on several bounded regret results for variants of the models we consider here. Theorem 4.2.3 covers, as particular cases, the one dimensional problem (also called multi-secretary) studied in [10], and the multi-dimensional version studied in Chapter 5. We observe that [10] uses a geometric view, like we do, but theirs is ad-hoc for one dimension. The generalization of one dimensional concepts requires new ideas (centroids, bases, cones, etc) and allows us to go significantly beyond current results and identifying the key ingredients that characterize the robustness limits of this algorithm. Due to its applicability, this is a particularly active line of research. Indeed, there are other recent algorithms that achieve constant regret in limited settings (when horizon and inventory are scaled proportionally) [37] and under nondegenarcy conditions [78]. More recently, in Chapter 5 we studied a larger family of resource allocation problems, including pricing, but they limit to study only the regret in different classes of problems and provide no robustness results except for the perturbation of v in one dimension, which we generalize in Section 4.7.2.

Our alternative geometric view adds to the earlier line of work both in terms of results and in terms of clarity. Indeed, we support the expansion of the models to the addition of queues and inventory arrivals (restock) and we provide an explicit study of robustness to parameter perturbations. In terms of clarity, we provide an alternative explanation grounded in a stochastic processes.

Bandits and robustness to parameter estimation. A traditional approach to optimization under uncertainty is to model the possible parameter perturbation with uncertainty sets and obtain minmax guarantees, see [24] for a survey on robust optimization. We take a similar stance in that we give explicit conditions on the parameters of the model such that our regret guarantees hold. This is also closely related with the bandits literature, where certain parameters are learned as the process evolves, see [33] for more on this topic. An important class of problems is that of bandits with knapsacks [109], where there is learning under some budgets constraints. This is closely related to our models, since we have limited inventory. A further refinement is that of contextual bandits with knapsacks [5], where arrivals present a type before the controller makes decisions. The results in previous papers imply $O(\sqrt{T})$ regret bounds in our setting, but we get a stronger $O(\log T)$, which is the optimal regret scaling. Irrespective of the view we take (robustness vs learning), we obtain our guarantees by appealing to our notion of centroids and, furthermore, our understanding is grounded in duality and geometry, hence it makes transparent the separation conditions. To emphasize the last point, we note that Chapter 5 also has robustness results for the one-dimensional problem ($d = 1$), but they use a separation condition based on sorting requests by reward (i.e., ranking), which is an intrinsically one-dimensional notion; centroids generalize this condition to multiple dimensions.

Two-sided arrivals and assembly (assemble-to-order). Arrivals of inventory capture assembly networks with fixed production rates. In assembly models, orders arrive (and wait in queue if patient) to be assembled by using relevant components (resources in our case). See [101] for a survey of managing these systems and related applications. There is a long line of work in obtaining performance guarantees; for example, [92] gives an asymptotically optimal policy for holding cost minimization under a high demand assumption (related to our slow restock assumption). In contrast, we focus on finite time guarantees (not asymptotic) for reward maximization.

Paremetric linear programming. Our parametric analysis of the packing linear program is an instance of this larger literature, where the objective is to understand optimization programs as the parameters move, see [64] for a survey. In our case, several parameters are perturbed simultaneously, a.k.a. multiparametric linear programming; see e.g. [26, 22] which, like us, use this parametric analysis in support of optimal control problems, specifically model predictive control. Instead of developing algorithms for for parametric linear programs, our analysis requires the characterization of the geometry of the problem. This is made feasible by the special structure of the packing LP.

Drift analysis. Much of the analysis centers on the dynamics of the process R^t. We argue that, when close to certain boundaries, the ratio process R^t drifts further towards the boundary. Such Lyapunov/drift methods are common in the analysis of stochastic models to establish positive recurrence of underlying Markov processes. In the context of online control, there are similarities to queueing theory where max-weight policies—based on re-solving local optimization problems—lead to the attraction to a subset of the state space; see e.g. [56, 83].

4.3 Overview of Our Approach

It is useful to think of the online policy as building dynamically an approximate solution for a random linear system. Indeed, the offline linear system requires information of future arrivals, hence it is not revealed till time T, whereas an online policy must commit to solutions in a dynamic fashion. Below we make precise this connection to linear system precise and spell out conditions on approximate (dynamic) solutions that guarantee constant regret.

Offline representation. Introducing slack variables, we rewrite the offline LP Eq. (4.1), $\{Ay \leq I^0, y \leq Z_{\mathcal{J}}^T\}$ in standard form $\{Ay + s = I^0, y + u = Z_{\mathcal{J}}^T\}$, where $s \in \mathbb{R}^d_{\geq 0}$ is the

surplus of resource and $u \in \mathbb{R}^n_{\geq 0}$ is the amount of unmet demand. Augmenting the matrix A to $\bar{A}$, and naturally assigning zero value to slack variables, we have the following standard form LP

$$V^*_{off}(T, I^0) = \mathbb{E}\left[\max\left\{v'y : \bar{A}\begin{pmatrix} y \\ u \\ s \end{pmatrix} = C\right\}\right], \quad \text{where} \quad C \coloneqq \begin{pmatrix} I^0 + Z^T_{\mathcal{R}} \\ Z^T_{\mathcal{J}} \end{pmatrix}. \tag{4.4}$$

The random vector $C \in \mathbb{R}^{d+n}_{\geq 0}$ is the *maximum consumption* of offline. Given a basis $\mathcal{B}$ (columns of $\bar{A}$) for the LP in Eq. (4.4), the optimal solution satisfies $\mathcal{B}x_{\mathcal{B}} = C$, where $x = (y, u, s)$ stands for all the variables. Hence, the realized (random) value of offline can be written as

$$\sum_{\mathcal{B}} v'_{\mathcal{B}} y_{\mathcal{B}} \mathbb{1}_{\{\mathcal{B} \text{ is optimal}\}} = \sum_{\mathcal{B}} v'_{\mathcal{B}} \mathcal{B}^{-1} C \mathbb{1}_{\{\mathcal{B} \text{ is optimal}\}}. \tag{4.5}$$

We abuse notation in the usual way, where $\mathcal{B}$ denotes both the indices of basic columns and the sub matrix $\bar{A}_{\mathcal{B}}$. Note that the optimality of a basis depends on the random vector C.

Online construction of the offline linear system. As expressed in Eq. (4.5), if the optimal offline basis is $\mathcal{B}$, then the offline actions correspond to the unique solution of the system $\mathcal{B}x_{\mathcal{B}} = C$, where $x = (y, u, s)$ stands for all the variables. Our first insight is that *any online policy* π tries to build an approximate solution to the previous system. Furthermore, the quality of the solution depends on how long the policy π "operates" in the basis $\mathcal{B}$. We make this precise in Proposition 4.3.3 below, where we show that the following property characterizes the regret of any policy.

Definition 4.3.1 (Basic Allocation). Let π be any online policy and $\mathcal{B}$ the optimal offline basis (which is only revealed at T). We say that π performs *basic allocation* at $t \in [T]$ if it only serves requests j such that $y_j \in \mathcal{B}$ (request variable is basic) and it only queues arriving requests such that $u_j \in \mathcal{B}$ (unmet variable is basic).

Intuitively, as long as the policy π is performing basic allocations, it is "operating" in the optimal basis, hence, if τ^π is the last time when π performed basic allocation, then the regret is incurred in the remaining $T - \tau^\pi$ periods. We need the following notion to completely characterize the regret.

Definition 4.3.2 (Wastage). Let π be any online policy and $\mathcal{B}$ the optimal offline basis. Let S_i^t be the surplus of resource $i \in [d]$ at time t when using the policy π, i.e., $S^t = I^0 + Z_{\mathcal{R}}^t - AY^{\pi,t}$. The wastage of π at t is $W^{\pi,t} \coloneqq \max\{S_i^t : \text{surplus variable } s_i \text{ is non basic}\} = \max\{S_i^t : s_i \notin \mathcal{B}, i \in [d]\}$.

Intuitively, if $s_i \notin \mathcal{B}$, then resource i has no slack, i.e., it is completely utilized in the offline solution. Hence, the wastage exactly captures how much inventory is left by the online policy that should have been completely used. Now we show that both the regret and quality of the online system, i.e., the approximation to $\mathcal{B}x_{\mathcal{B}} = C$, are determined by this time τ^π and its wastage.

Proposition 4.3.3. *Let $\mathcal{B}$ be the optimal basis for the offline problem in Eq. (4.4) and denote $J^t \in \mathcal{J}$ the t-th arriving request. For any online policy π define the time*

$$\begin{aligned}\tau^\pi &\coloneqq \min\{t \le T : \textit{the policy does not perform basic allocation at } t\} - 1 \\ &= \min\{t \le T : (\sigma_j^{\pi,t} = 1 \textit{ and } y_j \notin \mathcal{B} \textit{ some } j) \textit{ or } (\sigma_j^{\pi,t} = 0 \textit{ and } u_j \notin \mathcal{B} \textit{ for } j = J^t)\} - 1.\end{aligned}$$

Then the expected regret of π is at most $M\mathbb{E}[T - \tau^\pi + W^{\pi,\tau^\pi}]$, where M is a constant independent of (T, I^0), but that may depend on (A, v), and $W^{\pi,t}$ is the wastage (see Definition 4.3.2).

Proof. Throughout the proof, the policy π is fixed and we omit it from the notation. Let Y_j^t, U_j^t be the number of type-j requests accepted and queueed/rejected by the policy over the interval $[1, t]$. Similarly, C^t denotes the maximum consumption in $[1, t]$, i.e., $C^t \coloneqq \begin{pmatrix} I^0 + Z_{\mathcal{R}}^t \\ Z_{\mathcal{J}}^t \end{pmatrix}$ and the surplus is defined as $S^t \coloneqq I^0 + Z_{\mathcal{R}}^t - AY^t \in \mathbb{R}_{\ge 0}^d$. By definition of

surplus, we have the inventory equation $\bar{A}X^t = C^t$, where $X^t = (Y^t, S^t, U^t)$ stands for all the variables. Let us divide the matrix $\bar{A}$ into basic and non-basic columns as $\bar{A} = [\mathcal{B}, \mathcal{B}^c]$. We claim that,

$$\mathcal{B}\begin{pmatrix} Y^t \\ U^t \\ S^t \end{pmatrix}_{\mathcal{B}} + \mathcal{B}^c \begin{pmatrix} 0 \\ 0 \\ S^t \end{pmatrix}_{\mathcal{B}^c} = C^t \quad \text{and} \quad C - C^t = Z^T - Z^t, \quad \forall t \leq \tau^\pi. \tag{4.6}$$

Indeed, the first equation follows by the decomposition $\bar{A} = [\mathcal{B}, \mathcal{B}^c]$ and the fact that, up to time τ^π, the only non-zero variables Y^t, U^t are in the basis $\mathcal{B}$. The second equation is by definition of C and C^t. Recall that the offline variables $x = (y, u, s)$ are the solution to the offline system, i.e., $x_{\mathcal{B}} = \mathcal{B}^{-1}C$. From Eq. (4.6) we have

$$\begin{pmatrix} y \\ u \\ s \end{pmatrix}_{\mathcal{B}} - \begin{pmatrix} Y^t \\ U^t \\ S^t \end{pmatrix}_{\mathcal{B}} = \mathcal{B}^{-1}(Z^T - Z^t) + \mathcal{B}^{-1}\mathcal{B}^c \begin{pmatrix} 0 \\ 0 \\ S^t \end{pmatrix}_{\mathcal{B}^c} \quad \forall t \leq \tau^\pi. \tag{4.7}$$

Since Y is an increasing process $Y^T \geq Y^t$ for all t so that

$$\textsc{Reg} = (v'_{\mathcal{B}} y_{\mathcal{B}} - v' Y^T) \leq (v'_{\mathcal{B}} y_{\mathcal{B}} - v' Y^t) \leq v'_{\mathcal{B}}(y_{\mathcal{B}} - Y^t_{\mathcal{B}}).$$

We can bound the last expression using Eq. (4.7). Indeed, since there is at most one arrival per period, $||Z^T - Z^t||_\infty \leq T - t$, and the surplus is bounded by definition as $||S^t_{\mathcal{B}^c}||_\infty = W^{\pi,t}$. Finally, we can take the worst case over $\mathcal{B}$ in Eq. (4.7) and conclude the result by setting $t = \tau^\pi$. □

If we are able to prove $\mathbb{E}[T - \tau^\pi + W^{\pi,\tau^\pi}] = \mathcal{O}(1)$, where π is our BR algorithm, we would have proved our main result (Theorem 4.2.3) in virtue of Proposition 4.3.3. Therefore, we now focus on bounding the distribution of τ^π and its associated wastage.

Remark 4.3.4 (On randomized policies and (conventional) bid-price controls). From our definition of basic allocation (Definition 4.3.1) and its associated guarantee in

Proposition 4.3.3, we see that it is essential for a policy to correctly allocate w.r.t. the offline basis $\mathcal{B}$. Randomized policies do not respect this [78], hence their sub-optimal performance. This can be easily visualized in the one-dimensioanl case; see Figure 4.3. Suppose that types are ordered in decreasing order of their rewards. In this illustration the initial ratio R^0 is slightly below $\bar{Z}_1^T + \bar{Z}_2^T$ so that offline will take all of type 1 and most of type 2 but none of type 3. However, since $\bar{Z}_1^T + \bar{Z}_2^T$ is a (small) perturbation of $p_1 + p_2$ we can have that R^0 is greater than $p_1 + p_2$. In this case, the randomized policy will (with some small probability) accept an arriving type 3 early in the horizon. In doing so, it will perform a *non*-basic allocation. In contrast, BudgetRatio will not take any such type 3, because it needs the condition $R^0 \geq p_1 + p_2 + p_3/2$ to do so. Informally, the thresholding adds a confidence interval.

Another popular heuristic is bid-price control, wherein one computes the shadow price vector λ associated to resources and accepts a request if its reward v_j exceeds the sum of prices of requested resources, i.e., if $v_j \geq \sum_{i\in[d]} a_{ij}\lambda_i$. The same issue persists in this case, i.e., the policy performs non-basic allocations. This is evident again in the one-dimensional example; when $R^0 \in (p_1 + p_2, p_1 + p_2 + p_3)$ the shadow price of the (single) resource is v_3. Thus, we would accept type 3 request although this is not a basic allocation.

In Section 4.8 we re-visit bid prices and show that BudgetRatio can be interpreted as a bid-price control where instead of taking the duals under one basis, we have to take a maximum over all bases in a centroid.

Overview of the analysis in one dimension. Let us consider in some detail the one dimensional packing problem, also known as the multi-secretary [10], which can be stated as follows. There are I^0 positions to be filled and candidates arrive one at a time with abilities (values) $V^1, \ldots, V^T$, the goal is to select at most I^0 candidates to maximize the total value.

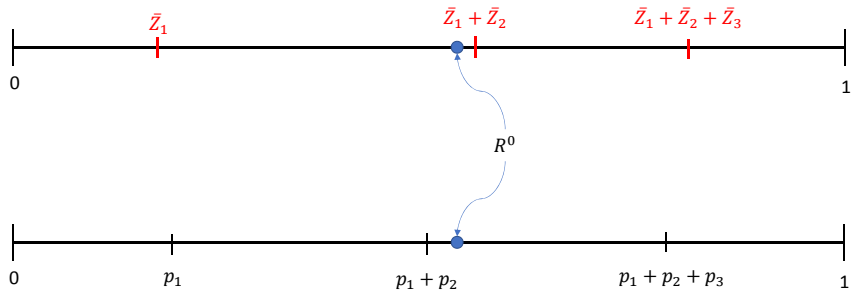


Figure 4.3: Why randomized policies do not maintain basic allocations.

In our notation, there is here a single resource ($d = |\mathcal{R}| = 1$), no restock ($p_{\mathcal{R}} = 0$, so that $\mathbb{E}[R^0] = I^0$), each request consumes one item ($A = \mathbf{e}'$), and all requests are impatient. The deterministic relaxation has $n+1$ constraints, one for each of the demand constraints and a single budget constraint:

$$\begin{aligned} \mathrm{LP}(R, p) \qquad \max \quad & v'y \\ \text{s.t.} \quad & \mathbf{e}'y \leq \mathbb{E}[R^0] \\ & y \leq p_{\mathcal{J}} \\ & y \geq 0. \end{aligned} \tag{4.8}$$

We assume w.l.o.g. that types are labelled in decreasing order of rewards, i.e., $v_1 > v_2 > \ldots > v_n$, and let $\bar{F}_i := \sum_{j=1}^{i} p_j$ be the survival function at v_i. The deterministic relaxation in Eq. (4.8) has a simple greedy solution: in increasing order of k, set $\bar{y}_k = p_k$ as long as $\bar{F}_k \leq \mathbb{E}[R^0]$. Letting $i_0 = \max\{k : \bar{F}_k \leq R\}$, finally set $\bar{y}_{i_0+1} = \mathbb{E}[R^0] - \bar{F}_{i_0}$.

Centroids. The concept of *centroid* is our main definition and can be described as follows. If the expected ratio were exactly $\mathbb{E}[R^0] = \bar{F}_j$, then the deterministic relaxation (4.8) takes all types $\mathcal{K} = [j]$ (and only those types). In other words, for this choice of right-hand side (i.e., budget), the problem $\mathrm{LP}(\bar{F}_j, p)$ is such that all variables $y_1, \ldots, y_j$ are completely saturated and all other variables are zero. The sets $\mathcal{K}$ with this property (and their later generalization to multiple dimensions, see Definition 4.4.6) are *centroids*. The set $\mathcal{K} = [j]$

is optimal when the budget is $r_{\mathcal{K}} = \bar{F}_j$, hence we refer to $r_{\mathcal{K}}$ as the *centroid's budget*; see Fig. 4.4 for an illustration.

It is clear that the centroids *do not depend on* p. In other words, regardless of the distribution, the LP "takes" all requests $[j]$ before taking any request of type $j+1$. Both the deterministic relaxation $\mathrm{LP}(\mathbb{E}[R^0], p)$ and the offline problem $\mathrm{LP}(R^0, D^0)$ follow the same nested rule. We show that this concept generalizes in multiple dimensions, i.e., there are sets of requests $\mathcal{K} \subseteq \mathcal{J}$ that are always prioritized in some part of the space independent of the demand p. In conclusion, centroids elicit a geometric view of the problem—a useful summary of the matrix A and the reward vector v—that does not depend on the demand.

Action sets: The centroid neighborhood. The thresholding of the algorithm—that we accept type j request only if $\bar{y}_j/p_j \geq \frac{1}{2}$ can be though of as building a confidence interval—a neighborhood—around the centroid's budget. The neighborhood of the centroid $\{1, 2\}$ is the interval $[\bar{F}(v_3) - \frac{f_2}{2}, \bar{F}(v_3) + \frac{f_3}{2}]$ "centered" at exactly the centroid budget $\bar{F}(v_3) = f_1 + f_2$. As long as $B^t = I^t/(T-t)$ is in this interval the algorithm accepts only (and all) arriving requests of types $\{1, 2\}$. The algorithm will start accepting type-3 requests if B^t exceeds the right threshold $\bar{F}(v_3) + f_3/2$. It will drop some type-2 requests if it goes below the left threshold $\bar{F}(v_3) - f_2/2$.

Oracle Containment. Proposition 4.3.3 makes clear that a good online policy (with large time τ^π) should have "almost" oracle access to the offline basis $\mathcal{B}$. It's decisions must be consistent with (the apriori) unknown $\mathcal{B}$. We will formalize this intuition by showing that if the budget-ratio process R^t is contained in an appropriate region, then we have this oracle access.

Let us suppose that $R^0 = \bar{F}(v_2)$, hence in the deterministic relaxation there is exactly the amount of budget to take all requests of types 1 and 2 and nothing else (see Fig. 4.4).

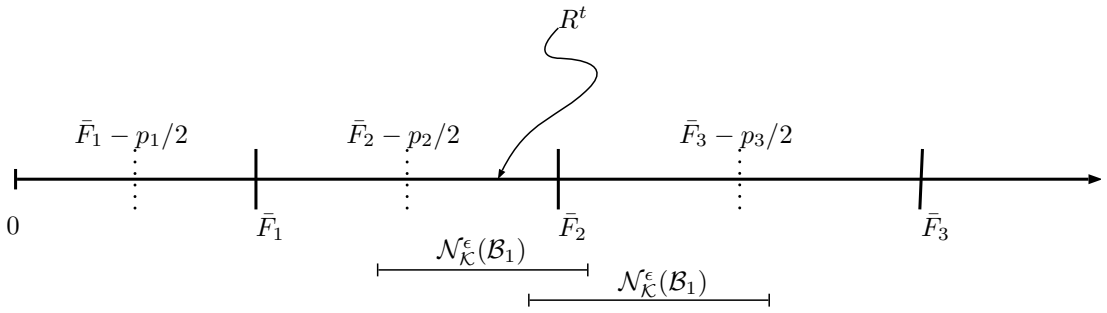


Figure 4.4: The position of the ratio R^t with respect to the centroid budgets $r_{\{1,..,i\}}(p) = \bar{F}_i$ determines the actions of the policy. At time t, the policy accepts a type i iff $R^t \geq \bar{F}_i - p_i/2$. The oracle containment property implies that if the realization Z^T is such that offline accepts only types $\{1, 2\}$, then $R^t \in \mathcal{N}^\varepsilon(\mathcal{B}_1)$; conversely, if Z^T is such that offline does accept type-3, then $R^t \in \mathcal{N}^\varepsilon(\mathcal{B}_2)$. In conclusion, R^t evolves in the "correct" region $\mathcal{N}^\varepsilon(\mathcal{B}_1)$ or $\mathcal{N}^\varepsilon(\mathcal{B}_2)$, this guarantees that the policy accepts only requests in the optimal offline basis.

There are two possible optimal bases at R^0. One, $\mathcal{B}_1$, has variables $\{y_1, y_2\}$ and the surplus variable s_2 (as a degenerate zero-valued variable) and the other, $\mathcal{B}_2$, has y_3 instead of s_2, but also as a degenerate zero-valued variable.

Consider Fig. 4.4. We will prove that, if offline takes only types $\{1, 2\}$, i.e., if $I^0 - (Z_1^T + Z_2^T) \leq 0$ (offline selects basis $\mathcal{B}_1$), then $R^t \in \mathcal{N}^\varepsilon(\mathcal{B}_1)$ for all $t \in [1, \tau]$ where τ is a large stopping time, i.e., $\mathbb{E}[T - \tau] \leq M$. Observe that, as long as $R^t \in \mathcal{N}^\varepsilon(\mathcal{B}_1)$ (see Fig. 4.4), the policy only accepts requests $\{1, 2\}$ which is consistent with $\mathcal{B}_1$, hence the policy performs basic allocation. On the other hand, if offline does accept type 3, i.e., $I^0 - (Z_1^T + Z_2^T) \geq 0$ (offline selects basis $\mathcal{B}_2$), then (we will prove) $R^t \in \mathcal{N}^\varepsilon(\mathcal{B}_2)$ for $t \in [1, \tau]$ which again implies that the policy performs basic allocation. Finally, because the process $R^t = \frac{1}{T-t} I^t$ remains contained in this bounded region, we have $R^{\tau^\pi} \leq M$, hence $I^{\tau^\pi} \leq M(T - \tau^\pi)$, which proves a bounded wastage, hence Proposition 4.3.3 yields bounded regret.

Beyond one dimension. We will formalize the notion of centroid and actions sets; see Section 4.4.2. This requires a careful parametric analysis of the packing LP. We will show that the action sets are, in turn, composed of convex subsets each corresponding to an optimal

basis (see Lemma 4.4.10). These convex sets are parametrized by the distribution and hence apply to the deterministic relaxation and to the offline (random) LP. Second, we we will set the ground for the proof of oracle containment property in multiple dimensions. This builds on both a sticky boundary phenomenon 4.5.2, that shows that the residual budget process remains in one action set and a cone-containment that—via convex separation theorems and the construction of suitable random walks—establishes that depending on offline's optimal basis, the budget process—controlled online by BudgetRatio—remains constrained in the convex set where its actions are consistent with offline's: they are basic allocations.

The geometric structure, and hence the stochastic analysis that builds on it—convex sets, basic cones and so on—is trivial in the one dimensional case.

4.3.1 Towards the General Analysis: An Example

For visualization purposes, it is useful to fix a two dimensional example ($d = 2$ resources). A two dimensional problem is sufficiently rich to capture key characteristics and, at the same time, it is sufficiently simple to allow for informative visuals of the problem's geometry. We consider a traditional packing problem (network revenue management), i.e., with impatient requests and no resource arrival.

We denote the resources by a, b with initial inventory I_a and I_b. There are four customer types $\{1, 2, 3, 4\}$ with the following matrix consumption

$$A = \begin{array}{c|cccc} & 1 & 2 & 3 & 4 \\ \hline a & 1 & 0 & 1 & 1 \\ b & 0 & 1 & 1 & 1 \end{array}$$

We use the reward vector $v = (4, 4, 5, 1)$ and probabilities $p_j = 1/4$ for all j. Observe that

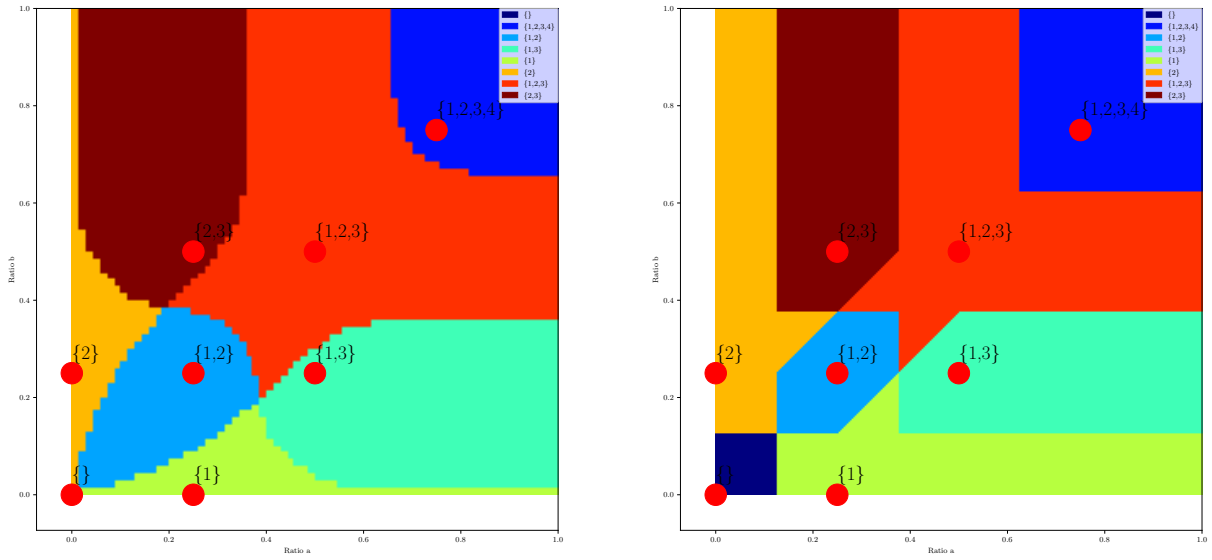


Figure 4.5: (LEFT) The action regions of the optimal policy (computed via DP) with 70 periods to go. When the vector of ratios (R_a, R_b) is in a color coded region, the policy accepts a fixed set of requests and rejects all others. For example, the light blue region in the bottom left corresponds to accepting $\{1, 2\}$ and rejecting all other requests. (RIGHT) The action regions for BudgetRatio. The color coded regions have the same interpretation.

type-1 and type-2 have high reward for resources a and b, respectively, while type-3 has a higher reward, but uses both resources, making it less desirable. Type-4 is clearly the least desirable, using both resources with low reward.

Fig. 4.5 captures the action-regions of BudgetRatio. For future reference we will label this as the *base example*.

Final setup details. Throughout we assume, without loss of generality, that $I_i \leq T$ for all $i \in \mathcal{R}$. If $I_i > T$, this resource is non-binding and we can reduce the problem to one with $d - 1$ resources. Additionally, we randomly perturb the rewards as follows. For every $j \in \mathcal{J}$, we add an independent $U(0, \frac{1}{T})$ to v_j. This perturbation induces only a $\mathcal{O}(1)$ error while guaranteeing that all our objects (optimal solutions, optimal bases, etc) are uniquely defined.

4.4 Parametric Structure of Packing Problems

In this section we give a geometric characterization of packing problems. Our results here are for a large family of linear programs and not specialized to an algorithm. In Section 4.5 we use the results here to analyse BudgetRatio.

The performance of a policy can be characterized by how long it is performing basic allocations (see Proposition 4.3.3), i.e. those consistent with offline's optimal basis. Therefore, we need to uncover the parametric structure of the packing LP and understand the properties of optimal bases.

Let us write the packing LP in standard form. Our key descriptor is the budget ratio $R^t = \frac{1}{T-t} I^t + p_{\mathcal{R}}$. Let $\bar{A}$ be the augmented matrix

$$\bar{A} = \begin{bmatrix} A & 0 & I^d \\ I^n & I^n & 0 \end{bmatrix},$$

where I^n is the identity matrix of dimension $n \times n$. For any $R \in \mathbb{R}^n_{\geq 0}$ and $D \in \mathbb{R}^d_{\geq 0}$, we re-write the LP relaxation as

$$\max \left\{ v'y : \bar{A} \begin{pmatrix} y \\ u \\ s \end{pmatrix} = \begin{pmatrix} R \\ D \end{pmatrix}, \ (y, u, s) \geq 0 \right\}, \qquad (\text{LP}(R, D))$$

where $(y, u, s)' \in \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^d$ is the decision vector. The variables $y \in \mathbb{R}^n$ represent the amount of requests served, henceforth *request variables*; $u \in \mathbb{R}^n$ correspond to the amount of unmet requests, henceforth *unmet variables*; and $s \in \mathbb{R}^d$ are the surplus of resource, henceforth *surplus variables*.

We use the general notation $\mathcal{B}$ to denote a basis of (LP(R, D)) as well as to denote the $(d+n) \times (d+n)$ sub-matrix of $\bar{A}$ corresponding to the variables in the basis $\mathcal{B}$; $\mathcal{B}^c$ denotes

the non-basic columns. Let $\bar{v} = (v, 0, 0) \in \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^d$ be the extended value vector, where we naturally assign zero value to variables u and s.

We are interested in the parametric structure of the LP (LP(R, D))—in how its solution changes with changes to the right-hand sides (R, D). BudgetRatio solves in each period $LP(R^t, p_{\mathcal{J}})$. Changes in R will be relevant for us as a result of the movement of R^t. Offline solves $LP(R^0, D^0)$ and hence so that it useful to understand how the solution depends on R^0 and D^0.

The first result states that there is a fixed—independent of the right-hand-side (R, D)—set of relevant bases we should consider. This will allow us henceforth, and w.l.o.g., to consider only bases that satisfy the conditions (i) and (ii) in Lemma 4.4.1.

Lemma 4.4.1. *Let $\mathcal{B}$ be a basis and let $\lambda = (\mathcal{B}^{-1})'\bar{v}_{\mathcal{B}}$ be the dual variables associated to $\mathcal{B}$. Assume (i) $\lambda \geq 0$ and (ii) $\bar{A}'\lambda \geq \bar{v}$. Then, for any (R, D), $\mathcal{B}$ is optimal for (LP(R, D)) if $\mathcal{B}^{-1}\begin{pmatrix} R \\ D \end{pmatrix} \geq 0$. Conversely, for any right-hand side (R, D), there is an optimal basis that satisfies (i) and (ii).*

Proof. The dual problem of LP(R, D) is

$$\min\{(R, D)'\lambda : \bar{A}'\lambda \geq \bar{v}, \lambda \geq 0\}.$$

For any basis $\mathcal{B}$, the associated dual variables are $\lambda = (\mathcal{B}^{-1})'\bar{v}_{\mathcal{B}}$, hence conditions (i) and (ii) imply that λ is dual-feasible. The associated primal variables are $(y, u, s)'_{\mathcal{B}} = \mathcal{B}^{-1}\begin{pmatrix} R \\ D \end{pmatrix}$. Finally, the primal and dual objectives coincide, so we conclude the optimality of $\mathcal{B}$ by Weak Duality provided that the solution is primal feasible, i.e., $(y, u, s)'_{\mathcal{B}} = \mathcal{B}^{-1}\begin{pmatrix} R \\ D \end{pmatrix} \geq 0$. Conversely, for any (R, D), if we run the Simplex Algorithm, we can find a basis $\mathcal{B}$ with an associated feasible dual solution. □

4.4.1 Action Regions and Exit Times

For ease of exposition we strengthen Assumption 4.2.1 to require that $p_i < p_j/2$ (rather than $p_i < p_j$) all i, j such that $A_{ij} = 1$ $p_i < p_j/2$. At the end of this section we will comment how the analysis extends to $p_i < p_j$. With this strengthened assumption $\bar{p}_j = p_j$ (see the first step of Algorithm 5) so that we threshold at at $p_j/2$.

For a set $\mathcal{K} \subseteq \mathcal{J}$ and a demand vector $D \in \mathbb{R}^n$, we can define the *action region* for $\mathcal{K}$ as the set of ratios $\mathcal{N}_{\mathcal{K}}(D) \subseteq \mathbb{R}^d$ where the algorithm serves *exclusively* requests in $\mathcal{K}$, i.e., all requests $j \in \mathcal{K}$ are served and $j \notin \mathcal{K}$ are queued:

$$\begin{aligned}\mathcal{N}_{\mathcal{K}}(D) &:= \left\{R \in \mathbb{R}^d : \text{ the algorithm serves exclusively requests } \mathcal{K} \text{ when } (R^t, p) = (R, D)\right\} \\ &= \bigcup_{\mathcal{B}} \left\{ R \in \mathbb{R}^d : \mathcal{B} \text{ optimal, } y_{\mathcal{K}} = \left(\mathcal{B}^{-1}\begin{pmatrix} R \\ D \end{pmatrix}\right)_{\mathcal{K}} \geq \frac{1}{2} D_{\mathcal{K}},\ y_{\mathcal{K}^c} = \left(\mathcal{B}^{-1}\begin{pmatrix} R \\ D \end{pmatrix}\right)_{\mathcal{K}^c} < \frac{1}{2} D_{\mathcal{K}^c} \right\}.\end{aligned} \tag{4.9}$$

The equality holds because the algorithm serves a request j iff $y_j \geq D_j/2$. We will use this construction with two distinct values of D: $D = p_{\mathcal{J}}$ and $D = \frac{1}{T} Z^T$. For some $\mathcal{K} \subseteq \mathcal{J}$ the set $\mathcal{N}_{\mathcal{K}}(D)$ might be empty (the algorithm never "prioritizes" the set $\mathcal{K}$ of items).

Lemma 4.4.2. *For $\mathcal{K} \subseteq \mathcal{J}$, the policy serves exclusively requests in $\mathcal{K}$ iff $R^t \in \mathcal{N}_{\mathcal{K}}(p_{\mathcal{J}})$. Furthermore, for a constant M that depends on (A, p) only, whenever $t \leq T - M$ and $R^t \in \mathcal{N}_{\mathcal{K}}(p_{\mathcal{J}})$, there is enough inventory to serve every $j \in \mathcal{K}$.*

Proof. The first part follows by definition of $\mathcal{N}_{\mathcal{K}}(p_{\mathcal{J}})$. For the second part we claim that, if $R^t \in \mathcal{N}_{\mathcal{K}}(p_{\mathcal{J}})$, then $I_i^t \geq |\{j \in \mathcal{K} : A_{ij} = 1\}|$, which proves that the inventory is enough to serve all requests in $\mathcal{K}$. Fix $j \in \mathcal{K}$ and $i \in [d]$ such that $A_{ij} = 1$. The fact that (y, u, s) solves $\mathrm{LP}(R^t, p_{\mathcal{J}})$ implies $Ay \leq \frac{1}{T-t} I^t + p_{\mathcal{R}}$. Since $j \in \mathcal{K}$ it must be that $y_j \geq \frac{1}{2}\bar{p}_j$, hence $I_i^t \geq (T-t)(\frac{1}{2}\bar{p}_j - p_i)$. Since the resources used by j have slow restock (Assumption 4.2.1), $\frac{1}{2}\bar{p}_j > p_i$. Taking the constant $M = \frac{|\{j \in \mathcal{K} : A_{ij} = 1\}|}{\bar{p}_j/2 - p_i}$ we obtain the claim for all $t \leq T - M$. □

Remark 4.4.3 (On The Slow Restock Assumption). The proof of Lemma 4.4.2 reveals that we can weaken Assumption 4.2.1. Indeed, the Lemma still holds if we assume instead $p_i < \sum_{j\in\mathcal{K}} a_{ij}p_j$ for every centroid $\mathcal{K}$ (Definition 4.4.6) and every resource i used by some $j \in \mathcal{K}$. In other words, the restock must be slower than the combined demand for it.

We let $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B}) \subseteq \mathbb{R}^d$ be the set of ratios where the algorithm serves exclusively $\mathcal{K}$ and the optimal basis for the relaxation is $\mathcal{B}$.

$$\begin{aligned}\mathcal{N}_{\mathcal{K}}(D,\mathcal{B}) &:= \left\{R \in \mathbb{R}^d : \text{ algorithm uses } \mathcal{B} \text{ and serves exclusively } \mathcal{K} \text{ when } (R^t, p) = (R, D)\right\} \\ &= \left\{R \in \mathbb{R}^d : \mathcal{B} \text{ optimal, } y_{\mathcal{K}} = \left(\mathcal{B}^{-1}\begin{pmatrix} R \\ D\end{pmatrix}\right)_{\mathcal{K}} \geq \frac{1}{2}D_{\mathcal{K}},\ y_{\mathcal{K}^c} = \left(\mathcal{B}^{-1}\begin{pmatrix} R \\ D\end{pmatrix}\right)_{\mathcal{K}^c} < \frac{1}{2}D_{\mathcal{K}^c}\right\}.\end{aligned} \tag{4.10}$$

By definition we have $\mathcal{N}_{\mathcal{K}}(D) = \cup_{\mathcal{B}}\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$. The next result states that (i) the sets $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$ are the "correct resolution" to study the problem and (ii) perturbations of $\mathcal{N}_{\mathcal{K}}(D)$ completely characterize the exit time τ^π which, per Proposition 4.3.3, controls the regret.

Proposition 4.4.4. *Let $\mathcal{B}$ be the optimal offline basis and set $\mathcal{K} \subseteq \mathcal{J}$. For each $\epsilon > 0$ define the time*

$$\tau^{\epsilon,\mathcal{K}} := \min\{t \leq T : d(R^t, \mathcal{N}_{\mathcal{K}}(D, \mathcal{B})) > \epsilon\}.$$

Then, there exists a constant $\epsilon > 0$ such that $\tau^{\epsilon,\mathcal{K}} \leq \tau^\pi$, where π is the policy given by Algorithm 5 and $\tau^\pi + 1$ is the first time that π does not perform basic allocation as in Proposition 4.3.3.

As long as R^t is close to $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$, then, the algorithm is performing basic allocation. To prove this result we must elicit the structure of the action regions $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$. Before we turn to this task, we state a lower bound on $\tau^{\epsilon,\mathcal{K}}$ from which follows the proof Theorem 4.2.3 in virtue of Proposition 4.3.3. Recall that $\mathbb{E}[R^0] = \frac{1}{T}I^0 + p_{\mathcal{R}}$ and $\mathbb{E}[D^0] = p_{\mathcal{J}}$ hence at time

$t = 0$ we can identify the set $\mathcal{K}$ such that $\mathbb{E}[R^0] \in \mathcal{N}_{\mathcal{K}}(\mathbb{E}[D^0])$; indeed, this set $\mathcal{K}$ is obtained the first time we solve the deterministic relaxation.

Proposition 4.4.5. *Let $\mathcal{K}$ be such that $\mathbb{E}[R^0] \in \mathcal{N}_{\mathcal{K}}(\mathbb{E}[D^0])$ and for $\epsilon > 0$ let $\tau^{\epsilon,\mathcal{K}}$ be as in Proposition 4.4.4. Then, there is a constant M such that $\mathbb{E}[T - \tau^{\epsilon,\mathcal{K}} + W^{\tau^{\epsilon,\mathcal{K}}}] \leq M$, where W^t is the wastage at time t (see Definition 4.3.2).*

Proof of Theorem 4.2.3. By Proposition 4.4.4 we have that $\tau^{\epsilon,\mathcal{K}} \leq \tau^{\pi}$, hence the policy performs basic allocation over the interval $[1, \tau^{\epsilon,\mathcal{K}}]$. By Proposition 4.4.5, the expected wastage and remaining time $T - \tau^{\epsilon,\mathcal{K}}$ are bounded by a constant, hence we can apply Proposition 4.3.3 and conclude. □

It remains to prove Propositions 4.4.4 and 4.4.5. The former is a purely geometric one while the latter require the detailed analysis of the stochastic process and is contained in Theorem 4.5.2 and 4.5.5 further below. Both propositions are stated in terms, and require the analysis, of the action regions $\mathcal{N}_{\mathcal{K}}(D)$ and their subsets $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$. The next two subsections provide the geometric characterization of these regions.

4.4.2 Centroids and Neighborhoods

In this subsection we formally define the packing *centroids*. In Section 4.4.3 we give a characterization of $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$ based on centroids.

Definition 4.4.6 (Centroids). A subset $\mathcal{K} \subseteq \mathcal{J}$ is a centroid if, for some $D \in \mathbb{R}^n_{>0}$, there exists a solution (y, u, s) to $\mathrm{LP}(A_{\mathcal{K}}D, D)$ such that $u_{\mathcal{K}} = 0$ (no request in $\mathcal{K}$ is unmet) and $y_{\mathcal{K}^c} = 0$ (no request in $\mathcal{K}^c$ is served). If $\mathcal{K}$ is a centroid, we call $r_{\mathcal{K}}(D) \coloneqq A_{\mathcal{K}}D_{\mathcal{K}}$ the centroid budget.

Intuitively, a set $\mathcal{K}$ is a centroid if, given the exact budget required in expectation for all requests $\mathcal{K}$—this is $A_{\mathcal{K}}p_{\mathcal{K}}$—it is optimal in the deterministic relaxation to serve all request $\mathcal{K}$ and no others. In the one dimensional setting of Section 4.3 the centroids are the sets $[i]$ for $i = 1, 2, \ldots, n$ and their corresponding budgets are $r_{\mathcal{K}}(p) = r_{[i]}(p) = \bar{F}(v_i)$. Surprisingly, the characterization of centroids does not depend on D, but only on the matrix A and the values v. In particular, sets $\mathcal{K}$ is a centroid under either the theoretical distribution ($D = p$) and the empirical distribution ($D = \frac{1}{T}Z^T$).

Fig. 4.5 has (in red) the centroids for our base example and the location of the centroid budgets $r_{\mathcal{K}}(p)$ in $\mathbb{R}^2_+$ and their neighborhood.

The LP $LP(R, p)$ has multiple optimal bases at $R = r_{\mathcal{K}}(p) = A_{\mathcal{K}}p_{\mathcal{K}}$ and they are all degenerate: the solution (y, u, s) at $r_{\mathcal{K}}$ is, per Definition 4.4.6, $y_{\mathcal{K}} = p_{\mathcal{K}}$, $u_{\mathcal{K}^c} = p_{\mathcal{K}^c}$. All other variables are zero. Thus, only n of the basic variables are strictly positive, whereas the dimension of the right-hand side is $n + d$. There must be d zero-valued basic variables.

Definition 4.4.7 (Zero Valued Basic Variables). Fix a centroid $\mathcal{K}$ for some $\hat{p}$ as in Definition 4.4.6 and let $\mathcal{B}$ be a basis that is optimal at $r_{\mathcal{K}}(\hat{p})$, i.e., optimal for $\text{LP}(r_{\mathcal{K}}(\hat{p}), \hat{p})$ with (y, u, s) the associated solution. Define the sets of basic variables

$$K^+ \coloneqq \{j \in \mathcal{J} : y_j \in \mathcal{B}, y_j = 0\},\ K^- \coloneqq \{j \in \mathcal{J} : u_j \in \mathcal{B}, u_j = 0\},\ K^0 \coloneqq \{i \in \mathcal{R} : s_i \in \mathcal{B}, s_i = 0\}.$$

When either the basis $\mathcal{B}$ or the centroid $\mathcal{K}$ are not clear from context, we write, e.g., $K^+(\mathcal{B})$ or $K^+(\mathcal{B}, \mathcal{K})$ to avoid ambiguity.

The next result shows that both the definition of centroids (Definition 4.4.6) and zero-valued variables (Definition 4.4.7) are independent of the distribution $\hat{p}$.

Lemma 4.4.8. *Let $\mathcal{K}$ be a centroid for some $\hat{p} \in \mathbb{R}^n_{>0}$ as in Definition 4.4.6. Then, for any $\tilde{p} \in \mathbb{R}^n_{>0}$, the same property holds for $\tilde{p}$, i.e., $\text{LP}(A_{\mathcal{K}}\tilde{p}_{\mathcal{K}}, \tilde{p})$ has the solution $u_{\mathcal{K}} = 0$ and*

$y_{\mathcal{K}^c} = 0$. *Similarly, the sets of zero-valued basic variables in Definition 4.4.7 are the same under* $\hat{p}$ *and* $\tilde{p}$.

Proof. Let $\mathcal{B}$ be the optimal basis of $\mathrm{LP}(A_{\mathcal{K}}\hat{p}, \hat{p})$. We will prove that $\mathcal{B}$ is also optimal for $\mathrm{LP}(A_{\mathcal{K}}\tilde{p}, \tilde{p})$ and has an associated solution $(y, u, s) = (\tilde{p}_{\mathcal{K}}, \tilde{p}_{\mathcal{K}^c}, 0)$, which shows the result.

Since $\mathcal{B}$ has the basic variables $y_{\mathcal{K}}$ and $u_{\mathcal{K}^c}$, by inspection we have the following:

$$\mathcal{B}\begin{pmatrix} \tilde{p}_{\mathcal{K}} \\ \tilde{p}_{\mathcal{K}^c} \\ 0 \end{pmatrix} = \begin{pmatrix} A_{\mathcal{K}}\tilde{p}_{\mathcal{K}} \\ \tilde{p} \end{pmatrix} \Longrightarrow \mathcal{B}^{-1}\begin{pmatrix} A_{\mathcal{K}}\tilde{p}_{\mathcal{K}} \\ \tilde{p} \end{pmatrix} \geq 0.$$

By Lemma 4.4.1 it follows that $\mathcal{B}$ is optimal for the right-hand side $(A_{\mathcal{K}}\tilde{p}_{\mathcal{K}}, \tilde{p})$ and the associated solution is indeed $(y, u, s) = (\tilde{p}_{\mathcal{K}}, \tilde{p}_{\mathcal{K}^c}, 0)$. Finally, it is clear from the structure of the solution (y, u, s) that the set of zero-valued basic variables is the same under $\hat{p}$ and $\tilde{p}$. □

Finally, we define a relation between centroids. In Section 4.4.3 we prove that $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$, which determines the desired exit time, can be characterized in terms of the centroid $\mathcal{K}$ and its neighbors.

Definition 4.4.9 (Neighbors). Let $\mathcal{K}$ be a centroid. If the basis $\mathcal{B}$ is optimal at the centroid budget $r_{\mathcal{K}}$, we say that $\mathcal{B}$ is associated to $\mathcal{K}$. Another centroid $\mathcal{K}'$ is a neighbor of $\mathcal{K}$ if there is a basis $\mathcal{B}$ that is associated to both $\mathcal{K}$ and $\mathcal{K}'$, i.e., both centroids share an optimal basis. We will say that $\mathcal{K}'$ is a neighbor of $(\mathcal{K}, \mathcal{B})$.

Like the centroids themselves, the relation of "neighborhood" does not depend on the distribution $\hat{p}$. Once we fix $\mathcal{K}$ with associated basis $\mathcal{B}$, we can obtain neighbors of $\mathcal{K}$ based on the zero-valued basic variables (see Definition 4.4.7).

4.4.3 Cones and Characterization of Action Sets

Recall that we need to study the exit time in Proposition 4.4.4, which depends on the distance to the action set $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$. Here we take the first step in understanding this distance, since we can write $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$ as appropriate linear combinations. We present the proof of the following result in the appendix.

Lemma 4.4.10 (Characterization of $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$ and neighbors). *Fix a centroid $\mathcal{K}$ with associated basis $\mathcal{B}$. Let (K^+, K^-, K^0) be the zero-valued basic variables (Definition 4.4.7). Then,*

1. *The basis $\mathcal{B}$ is optimal for any right-hand side (R, D) of the form*

$$R = r_{\mathcal{K}}(D) + \alpha(A_{\kappa^+} D_{\kappa^+} - A_{\kappa^-} D_{\kappa^-}) + b,$$

   where $\kappa^+ \subseteq K^+, \kappa^- \subseteq K^-, \alpha \in [0, 1]$, and $b \in \mathbb{R}^d_{\geq 0}$ is zero for components not in K^0, i.e., $b_i = 0$ for $i \notin K^0$. In particular, the set $\mathcal{K} \cup \kappa^+ \backslash \kappa^-$ is a centroid it is a neighbor of $\mathcal{K}$.

2. *The basis $\mathcal{B}$ is optimal for (R, D) if and only if R is of the form*

$$R = r_{\mathcal{K}}(D) + \sum_{\kappa^+ \subseteq K^+, \kappa^- \subseteq K^-} \alpha_{(\kappa^+, \kappa^-)} (A_{\kappa^+} D_{\kappa^+} - A_{\kappa^-} D_{\kappa^-}) + b, \tag{4.11}$$

   where b is as before, $\alpha \geq 0$, and $\sum_{\kappa^+ \subseteq K^+, \kappa^- \subseteq K^-} \alpha_{(\kappa^+, \kappa^-)} = 1$.

3. *$R \in \mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$ if and only if*

$$R - r_{\mathcal{K}}(D) = A_{K^+} x_{K^+} - A_{K^-} x_{K^-} + b,$$

   where $x_j \in [0, D_j/2]$ for $j \in K^+ \cup K^-$ and b is as before.

In Figure 4.6 (RIGHT) we plot three of the neighbors of the centroid $\{1, 2\}$. For the direction $(\kappa^+, \kappa^-) = (\{3\}, \{2\})$ the neighboring centroid is $\{1, 3\}$. In moving from $\mathcal{K} = \{1, 2\}$

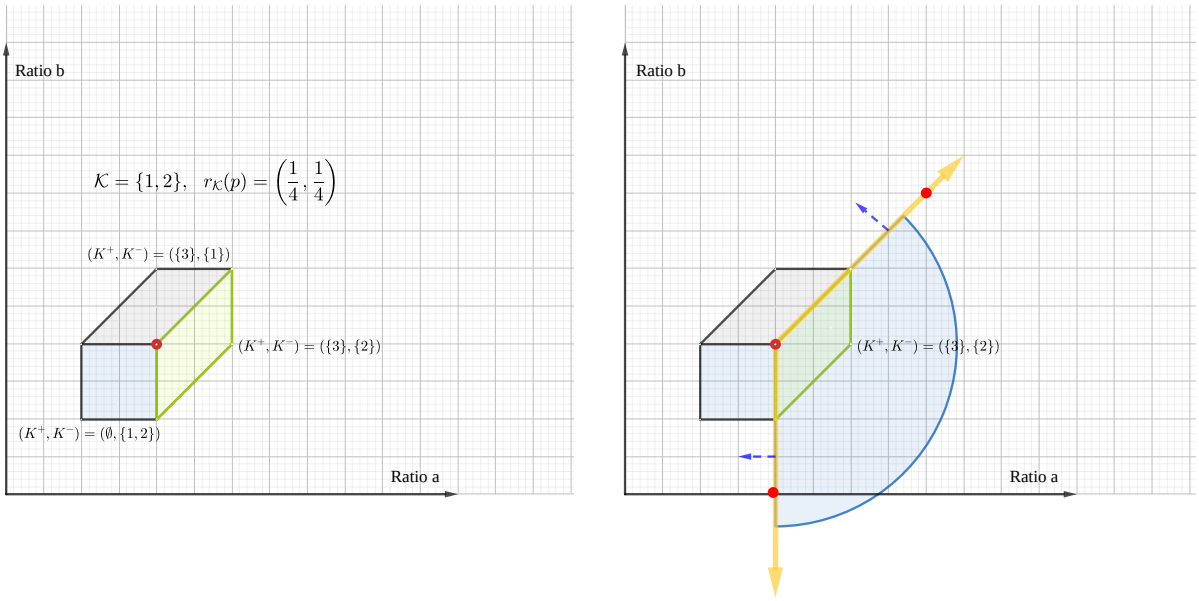


Figure 4.6: Geometric properties in the base example for the centroid $\{1, 2\}$ whose budget is $r = (1/4, 1/4)$: (LEFT) The extreme points and convex subsets, and (RIGHT) the cone, with orange boundaries, corresponding to $(K^+, K^-) = (\{3\}, \{2\})$. The dashed vectors are the outer normals, $\psi_1 = (-1, 1)'$ and $\psi_2 = (-1, 0)'$.

to $\mathcal{K} = \{1, 3\}$ the request variable y_2 and the slack u_3 leave the basis, while y_3 and u_2 enter the basis.

In Fig. 4.6, we zoom-in on the centroid $\mathcal{K} = \{1, 2\}$. One optimal basis at this centroid has $K^+ = \{3\}$ and $K^- = \{2\}$. The neighboring centroids with $\kappa^+ \in K^+$ and $\kappa^- \subseteq K^-$ are $\{1, 3\}$, $\{1\}$, and $\{1, 2, 3\}$. The set $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$ is the convex hull of the mid-points of the lines leading to those neighbors and corresponds to the yellow region.

The following will also be central to the analysis.

Definition 4.4.11 (Basic Cone). Let $\mathcal{K}$ be a centroid with associated basis $\mathcal{B}$ and (K^+, K^-) be the zero-valued basic variables (Definition 4.4.7). We define

$$\text{cone}(\mathcal{K}, \mathcal{B}) = \{y \in \mathbb{R}^n : y = A_{K^+} x_{K^+} - A_{K^-} x_{K^-}, \text{ for some } x \geq 0\}.$$

Observe that, from Lemma 4.4.10 (item 3), we know that $R \in \mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$ if and only if $R - r_{\mathcal{K}}(D) = A_{K^+} x_{K^+} - A_{K^-} x_{K^-} + b$ with $x_j \in [0, D_j/2]$. Our definition of cone drops the upper bound $D_j/2$ and is therefore independent of D; as can be seen in Definition 4.4.11, it depends only on $(\mathcal{K}, \mathcal{B})$.

Instead of bounding directly the time $\tau^{\epsilon,\mathcal{K}}$ (see Proposition 4.4.4), we will consider the minimum of two times: the exit time from the cone $\text{cone}(\mathcal{K}, \mathcal{B})$ and the exit time from the neighborhood $\mathcal{N}_{\mathcal{K}}(D)$. The intersection of these is the set $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$.

Lemma 4.4.12. *Let $\mathcal{K}$ be a centroid with basis $\mathcal{B}$. Then, $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B}) = \mathcal{N}_{\mathcal{K}}(D) \cap (r_{\mathcal{K}}(D) + \text{cone}(\mathcal{K}, \mathcal{B}))$.*

Proof. If $R \in \mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$ then in particular $R \in \mathcal{N}_{\mathcal{K}}(D) = \cup_{\mathcal{B}} \mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$ and from Lemma 4.4.10 (item 3) we have $R - r_{\mathcal{K}}(D) = A_{K^+} x_{K^+} - A_{K^-} x_{K^-} + b$, hence $R \in r_{\mathcal{K}}(D) + \text{cone}(\mathcal{K}, \mathcal{B})$.

If $\mathcal{N}_{\mathcal{K}}(D) \cap (r_{\mathcal{K}}(D) + \text{cone}(\mathcal{K}, \mathcal{B}))$, then necessarily $R = r_{\mathcal{K}}(D) + A_{K^+} x_{K^+} - A_{K^-} x_{K^-}$ (since it is in the cone) and $x_j \leq D_j/2$, since otherwise, by Lemma 4.4.10 (item 3), the associated solution would not have $y_{\mathcal{K}} > \frac{1}{2} D_{\mathcal{K}}$ and $y_{\mathcal{K}^c} < \frac{1}{2} D_{\mathcal{K}^c}$. □

The properties of the outwards normals to the cone will be central to the proof of oracle containment (see Theorem 4.5.5). Informally speaking, in that proof we will study, as a stochastic process, the angle between these normals and the resource consumption process. When accepting at type-j request, the consumption is the vector A_j. Figure 4.6(RIGHT) illustrates these vectors.

Lemma 4.4.13. *Fix a centroid $\mathcal{K}$ with associated basis $\mathcal{B}$. The vectors characterizing $\text{cone}(\mathcal{K}, \mathcal{B})$ (i.e., such that $\max_l \psi_l' x \leq 0$) have the following properties: For each $\kappa^+ \subseteq K^+$ and $\kappa^- \subseteq K^-$ with $|\kappa^+| + |\kappa^-| = 1$, $\psi[\kappa^+, \kappa^-]' A_{\kappa^+} = 0$ or $\psi[\kappa^+, \kappa^-]' A_{\kappa^-} = 0$. Also, $\psi[\kappa^+, \kappa^-]' A_j < 0$ for all $j \in K^+(\mathcal{B}), j \notin \kappa^+$ and $\psi[\kappa^+, \kappa^-]' A_j > 0$ for all $j \in K^-(\mathcal{B}), j \notin \kappa^-$.*

Additionally, for any other basis $\bar{\mathcal{B}} \neq \mathcal{B}$ *associated to* $\mathcal{K}$, $\psi[\kappa^+, \kappa^-]' A_j > 0$ *for all* $j \in (\kappa^+)^c \cup K^+(\bar{\mathcal{B}})$ *and* $\psi[\kappa^+, \kappa^-]' A_j < 0$ *for any* $j \in (\kappa^-)^c \cup K^-(\bar{\mathcal{B}})$.

In the one-dimensional case, all this structure is not needed. A basis has either K^+ or K^- (but never both) as well as $|K^+| + |K^-| = 1$. Thus, the second part ($\psi[\kappa^+, \kappa^-]' A_j < 0$ for $j \in K^+(\mathcal{B}), j \notin \kappa^+$) is moot. The vectors ψ are just the number 0.

The next lemma confirms that the online policy takes only requests only types in the set $\mathcal{K} \cup K^+$ when in a suitably small neighborhood of the subset $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$. That is, it performs only basic allocations for the basis $\mathcal{B}$.

Henceforth, we fix

$$\epsilon := \frac{1}{4} \min\{p_k : k \in \mathcal{R} \cup \mathcal{J}, p_k > 0\}. \tag{4.12}$$

Lemma 4.4.14 (Optimal Bases and Budget-Ratio Actions). *There exist constants* M_1, M_2 *such that, if* $d(R^t, \mathcal{N}_{\mathcal{K}}(\mathcal{B}, D)) \leq \frac{\epsilon}{M_1}$, *then the policy performs basic allocations at* t*: it serves only (but not necessarily all) requests in* $\mathcal{K} \cup K^+$ *and it queues only requests in* $\mathcal{K}^c \cup K^-$. *Moreover,* $I_i^t \leq M_2(T - t)$ *for all* $i \notin K^0(\mathcal{B})$.

Proof. Let $(\bar{y}, \bar{u}, \bar{s})$ be the solution to $\mathrm{LP}(R^t, D)$ and define

$$\mathcal{Y} = \{(y, u, s) : \exists R \in \mathcal{N}_{\mathcal{K}}(\mathcal{B}, D) \text{ s.t. } (y, u, s) \text{ solves } \mathrm{LP}(R, D)\}.$$

By assumption $d_\infty(R^t, \mathcal{N}_{\mathcal{K}}(\mathcal{B}, D)) \leq \frac{\epsilon}{M_1}$. By the Lipschitz continuity of the LP solution [46, Theorem 5], we can choose M_1 large enough (depending on A) such that $d_\infty((\bar{y}, \bar{u}, \bar{s}), \mathcal{Y}) \leq \epsilon$. Let $(y^0, u^0, s^0) \in \mathcal{Y}$ be such that $d_\infty((\bar{y}, \bar{u}, \bar{s}), (y^0, u^0, s^0)) \leq \epsilon$. Since for all $j \in \mathcal{K} \backslash K^-$ we have that $y_j^0 = D_j$ then we also have that $\bar{y}_j \geq D_j - \epsilon \geq D_j/2$ so that all these items are taken. Also, for any $j \notin \mathcal{K} \cup K^+$, we have that $u_j^0 = D_j$ so that $\bar{u}_j^0 \geq D_j - \epsilon$ and hence $\bar{y}_j^0 < D_j/2$ so these requests are queued. Finally, $s_i^0 = 0$ for all $i \notin K^0$, hence $\bar{s}_i \leq \epsilon$ for all such i, which implies $R_i^t \leq Ay + \epsilon$ and using $y \leq p$ we get the result. □

We need to characterize when the basis $\mathcal{B}$ is offline optimal. Recall that the empirical demand distribution is $D^0 = \frac{1}{T} Z^T_{\mathcal{J}}$ and the empirical budget ratio is $R^0 = \frac{1}{T}(I^0 + Z^T_{\mathcal{R}})$. The one dimensional case can be useful here. The optimal offline basis has type variables $1, 2$ (and not 3) if $R \in (\bar{Z}^T_1, \bar{Z}^T_1 + \bar{Z}^T_2]$. This is the intersection of the cone $r_{\{1\}}(\bar{Z}^T) + [0, \infty)$ (i.e., $R - r_{\{1\}} \in [0, \infty)$) with the set $(0, \bar{Z}^T_1 + \bar{Z}^T_2]$. Notice that if $\mathbb{E}[R^0] \in [p_1/2, p_1 + p_2/2]$ then, on the event that

$$\mathcal{A}^\epsilon := \{\|Z - p\|_\infty \leq \epsilon := \frac{1}{4} \min\{p_1, \ldots, p_{\mathcal{J}}\},$$

$R^0 \in (0, \bar{Z}^T_1 + \bar{Z}^T_2]$. More generally, we have the following simple result. We present the proof in the appendix.

Lemma 4.4.15. *Let $\mathcal{K}$ be the centroid such that $\mathbb{E}[R^0] \in \mathcal{N}_{\mathcal{K}}(\mathbb{E}[D^0])$ and let*

$$\mathcal{M}(\mathcal{B}) := \big\{\omega \in \Omega : R^0 - r_{\mathcal{K}}(D^0) \in \text{cone}(\mathcal{K}, \mathcal{B})\big\}.$$

Then, on the event $\mathcal{M}(\mathcal{B}) \cap \mathcal{A}^\epsilon$, $\mathcal{B}$ is an optimal offline basis. In particular, $\mathcal{B}$ has the following variables: $\{j : y_j \in \mathcal{B}\} \subseteq \mathcal{K} \cup K^+(\mathcal{B})$, $\{j : u_j \in \mathcal{B}\} \subseteq \mathcal{K}^c \cup K^-$, and $\{i : s_i \in \mathcal{B}\} \subseteq K^0$.

Recall that $\tau^{\epsilon,\mathcal{K}} = \min\{t \leq T : d_\infty(R^t, \mathcal{N}_{\mathcal{K}}(\mathcal{B}, D)) > \epsilon\}$ is the first time when the process gets too far from the action region. Also recall that all we are left to prove is Propositions 4.4.4 and 4.4.5. We are already in shape to prove the former.

Proof of Proposition 4.4.4. By Lemma 4.4.14, if $\mathcal{B}$ is the offline basis and $d(R^t, \mathcal{N}_{\mathcal{K}}(\mathcal{B}, D)) \leq \epsilon$, the policy performs basic allocation at t. By Lemma 4.4.15, on this event the basis $\mathcal{B}$ is offline optimal. □

4.5 Analysis of our Algorithm

We apply all the geometric characterizations obtained in Section 4.4 to BudgetRatio. Recall that we need to bound the time $\tau^{\epsilon,\mathcal{K}} = \min\{t \leq T : d_\infty(R^t, \mathcal{N}_{\mathcal{K}}(\mathcal{B}, D)) > \epsilon\}$. We

introduce two auxiliary exit times and relate them to $\tau^{\epsilon,\mathcal{K}}$ in Lemma 4.5.1 below:

$$\tau^{\epsilon,\mathcal{K}}_{\text{region}} := \inf\big\{t \le T : d(R^t, \mathcal{N}_{\mathcal{K}}(D)) > \epsilon\big\} \tag{4.13}$$

$$\tau^{\epsilon,\mathcal{B}}_{\text{cone}} := \inf\{t \le T : \max_l \psi_l'(R^t - r_{\mathcal{K}}(D)) > \epsilon\} \tag{4.14}$$

where the vectors $\psi_l \equiv \psi_l[\mathcal{B}]$ are as in Lemma 4.4.13. By definition, the action region $\mathcal{N}_{\mathcal{K}}(\mathcal{B}, D)$ is the set where (i) the basis $\mathcal{B}$ is optimal and (ii) requests $\mathcal{K}$ are served exclusively (see Lemma 4.4.12). The time in Eq. (4.13) relates to condition (i) and the time in Eq. (4.14) relates to condition (ii) in virtue of Lemma 4.4.13. This is formalized in the following result.

Lemma 4.5.1 (Exit Times). *Let $\mathcal{K}$ be a centroid with associated basis $\mathcal{B}$ and fix $\epsilon > 0$. There exists $\epsilon' > 0$, that depends on (ϵ, A, v) only, such that, for any $R \in \mathbb{R}^d$, if $d(R, \mathcal{N}_{\mathcal{K}}(D)) \le \epsilon'$ and $\max_l \psi_l'(R - r_{\mathcal{K}}(D)) \le \epsilon'$, then $d(R, \mathcal{N}_{\mathcal{K}}(\mathcal{B}, D)) \le \epsilon$. Consequently, $\tau^{\epsilon,\mathcal{K}} \ge \tau^{\epsilon',\mathcal{K}}_{region} \wedge \tau^{\epsilon',\mathcal{B}}_{\text{cone}}$.*

Additionally, if for some κ^+, κ^- with $|\kappa^+| + |\kappa^-| = 1$ we have $\psi[\kappa^+, \kappa^-]'(R - r_{\mathcal{K}}(D)) \le \epsilon$ and $d(R, \mathcal{N}_{\mathcal{K}}(D)) \le \epsilon$, then $R \in \mathcal{N}_{\mathcal{K}^0}(D)$ where $\mathcal{K}^0 = \mathcal{K} \cup \kappa^+ \backslash \kappa^-$.

Theorem 4.5.2 (Sticky Boundaries). *Let $\mathcal{K}$ be the centroid such that $\mathbb{E}[R^0] \in \mathcal{N}_{\mathcal{K}}(\mathbb{E}[D^0])$ and $\tau^{\epsilon,\mathcal{K}}_{region}$ as in Eq. (4.13). Then,*

$$\mathbb{P}[T - \tau^{\epsilon,\mathcal{K}}_{region} > \ell] \le \theta_1 e^{-\theta_2 \ell},$$

where $\theta_1, \theta_2 > 0$ do not depend on (T, I^0) but could possibly depend on p, A, v.

Remark 4.5.3 (sticky boundaries). The proof of this theorem contains also the implication that, once close to the boundary, the process R^t stays there. Formally, let

$$\tau^0_\partial = \inf\{t \ge 0 : d(R^t, \partial\mathcal{N}_{\mathcal{K}}(D)) \le \epsilon\} \wedge T, \text{ and } \tau^1_\partial = \inf\{t \ge \tau^0_\partial : d(R^t, \partial\mathcal{N}_{\mathcal{K}}(D) \ge 2\epsilon\} \wedge T.$$

Then, $\mathbb{P}\{T - \tau^1_\partial \ge \ell\} \le \theta_1 e^{-\theta\ell}$.

Define the set of request variables consistent with the action region $\mathcal{N}_{\mathcal{K}}(D)$:

$$\mathcal{Y}(\mathcal{K}, D) := \{y : \exists R \in \mathcal{N}_{\mathcal{K}}(D) \text{ s. t. for some } (u, s), (y, u, s) \text{ solves LP}(R, D)\}.$$

Lemma 4.5.4. *Fix $\mathcal{K}$ and a neighbor $\mathcal{K}^0 = \mathcal{K} \cup \kappa^+ \backslash \kappa^-$. Let $R \in \mathcal{N}_{\mathcal{K}^0}(D)$ and (y, u, s) be the solution to* $\mathrm{LP}(R, D)$. *Let*

$$(\theta_{\mathcal{K}}(y, D))_j = \begin{cases} y_j & \text{if } j \notin \kappa^+ \cup \kappa^- \\ D_j/2 & \text{if } j \in \kappa^+ \cup \kappa^-. \end{cases}$$

Then, the following holds:

1. *$\theta_{\mathcal{K}}(y, D) \in$ closure$(\mathcal{Y}(\mathcal{K}, D))$ and $(y - \theta_{\mathcal{K}}(y, D))_j = 0$ for all $j \notin \kappa^+ \cup \kappa^-$.*
2. *If y is the optimal request variable for* $\mathrm{LP}(R, D)$ *with optimal basis $\bar{\mathcal{B}}$ and $\mathcal{B}$ is adjacent, i.e., such that $\kappa^+ \subseteq K^+(\mathcal{B}) \cap K^+(\bar{\mathcal{B}})$ and $\kappa^- \subseteq K^-(\mathcal{B}) \cap K^-(\bar{\mathcal{B}})$, then $\left(\mathcal{B}^{-1} \begin{pmatrix} R \\ D \end{pmatrix}\right)_j =$ y_j for $j \in \kappa^+ \cup \kappa^-$.*

Proof of Theorem 4.5.2. Let $S^t \in \mathbb{R}^d$ be the surplus of $\mathrm{LP}(R^t, D)$ at time t, i.e., the value of the surplus variable s. Observe that the request variable y is exactly the same for both problems $\mathrm{LP}(R^t, D)$ and $\mathrm{LP}(R^t - S^t, D)$, hence, for any centroid $\mathcal{K}$, $R^t \in \mathcal{N}_{\mathcal{K}}(D)$ iff $R^t - S^t \in \mathcal{N}_{\mathcal{K}}(D)$. In conclusion, we can assume that R^t has zero surplus. We will show that

$$\mathbb{P}\left[\sup_{t \in [1, T-\ell]} d(R^t, \mathcal{N}_{\mathcal{K}}(D)) > \epsilon\right] \leq \theta_1 e^{-\theta_2 \ell},$$

which is exactly the statement of the theorem.

To simplify notation we will write $\theta^t = \theta_{\mathcal{K}}(y^t, D)$, where $\theta_{\mathcal{K}}(y^t, D)$ is as in Lemma 4.5.4 and $\delta^t = y^t - \theta^t$. We define the following Lyapunov function

$$g^t := d(y^t, \mathcal{Y}(\mathcal{K}, D)) = \|y^t - \theta^t\|^2 = \|\delta^t\|^2.$$

We claim that, whenever $g^t \leq \epsilon^2/nd$, we have $d(R^t, \mathcal{N}_{\mathcal{K}}(D)) \leq \epsilon^2$. Indeed, if $g^t \leq \epsilon^2/nd$, then by Cauchy-Schwarz $|Ay^t - A\theta^t|_i^2 = (a_i'(y^t - \theta^t))^2 \leq \epsilon^2/d$. Finally, we observe that $A\theta^t \in \mathcal{N}_{\mathcal{K}}(D)$ and $Ay^t = R^t$ (since R^t has zero surplus), hence $||Ay^t - A\theta^t||^2 \leq \epsilon^2$ implies $d(R^t, \mathcal{N}_{\mathcal{K}}(D)) \leq \epsilon^2$.

Let us set $\varepsilon^2 := \epsilon^2/nd$. We conclude that, as long as $g^t \le \varepsilon^2$, we can guarantee the desired condition $d(R^t, \mathcal{N}_{\mathcal{K}}(D)) \le \epsilon^2$. We argue the following drift condition: for some constant M

$$\mathbb{E}[g^{t+1} - g^t | \mathcal{F}_t] \le -\frac{M}{T-t}, \text{ whenever } g^t \in [\varepsilon/2, \varepsilon]. \tag{4.15}$$

Assuming Eq. (4.15), concentration arguments as in [10, Theorem 3] show that $\mathbb{P}[\max_{t\in[1,T-\ell]} g^t > \varepsilon^2] \le \theta_1 e^{-\theta_2(T-\ell)}$ for some constants (θ_1, θ_2) that depend on M only, which proves the theorem.

The remainder of the proof is devoted to obtain Eq. (4.15). If $R^t \in \mathcal{N}_{\mathcal{K}^0}(D)$ with $\mathcal{K}^0 = \mathcal{K} \cup \kappa^+ \setminus \kappa^-$, then using Lemma 4.5.4 (item 1) we obtain

$$\begin{aligned}\mathbb{E}[g^{t+1} - g^t | \mathcal{F}_t] &= \mathbb{E}[\|\delta^{t+1} - \delta^t\|^2 | \mathcal{F}_t] + 2\mathbb{E}[(\delta^{t+1} - \delta^t)'\delta^t | \mathcal{F}_t] \\ &= \mathbb{E}[\|\delta^{t+1} - \delta^t\|^2 | \mathcal{F}_t] + 2\mathbb{E}[(\delta^{t+1} - \delta^t)'_\kappa (\delta^t)_\kappa | \mathcal{F}_t],\end{aligned}$$

where the sub-index κ denotes the components in $\kappa^+ \cup \kappa^-$. Our aim is to prove $\mathbb{E}[\|\delta^{t+1} - \delta^t\|^2 | \mathcal{F}_t] = O(\frac{1}{(T-t)^2})$ and $\mathbb{E}[(\delta^{t+1} - \delta^t)'_\kappa (\delta^t)_\kappa | \mathcal{F}_t] \le \frac{-M}{T-t}$, which together would imply Eq. (4.15). Hence, we divide the proof in two parts: obtaining the linear bound $\mathbb{E}[(\delta^{t+1} - \delta^t)'_\kappa (\delta^t)_\kappa | \mathcal{F}_t] \le \frac{-M}{T-t}$ and the quadratic bound $\mathbb{E}[\|\delta^{t+1} - \delta^t\|^2 | \mathcal{F}_t] = O(\frac{1}{(T-t)^2})$. We remark that the linear bound is the challenging part and the quadratic is purely algebraic. We start with a fact about the different neighborhoods that the process R^t visits during its evolution.

Main property of visited neighbors. A vector $y \in \mathcal{Y}(\mathcal{K}, D)$ has, by definition, $y_j \ge D_j/2$ for $j \in \mathcal{K}$ and $y_j < D_j/2$ otherwise. Hence, if $g^t \le \varepsilon^2$, we have

$$y^t_j \ge D_j/2 - \varepsilon \ \forall j \in \mathcal{K} \quad \text{and} \quad y^t_j \le D_j/2 + \varepsilon \ \forall j \notin \mathcal{K}. \tag{4.16}$$

Let $\tilde{\mathcal{K}}$ be such that $R^{t+1} \in \mathcal{N}_{\tilde{\mathcal{K}}}(D)$, where $\tilde{\mathcal{K}} = \mathcal{K} \cup \tilde{\kappa}^+ \setminus \tilde{\kappa}^-$. We claim that

$$R^{t+1}_i - R^t_i = O\left(\frac{1}{T-t}\right) \forall i \in [d] \quad \text{and} \quad \kappa^+ \cup \kappa^- = \tilde{\kappa}^+ \cup \tilde{\kappa}^-. \tag{4.17}$$

The first fact in Eq. (4.17) is straightforward since we are taking ratios and we can assume zero surplus. Assume by way of contradiction that the second fact in Eq. (4.17) fails. Take $j \in \kappa^+ \cup \kappa^-$, then Eq. (4.16) implies $y_j^t = \frac{1}{2}D_j \pm \varepsilon$. If $j \notin \tilde{\kappa}^+ \cup \tilde{\kappa}^-$, then $y_j^{t+1} \in \{0, D_j\}$. We conclude $|y_j^t - y_j^{t+1}| > M$, but we know that solutions to the LP are Lipschitz continuous (see [46, Theorem 5]), hence $|y_j^t - y_j^{t+1}| \leq M||R^t - R^{t+1}|| = O(\frac{1}{T-t})$, so we arrive at a contradiction. The case $j \notin \kappa^+ \cup \kappa^-$ and $j \in \tilde{\kappa}^+ \cup \tilde{\kappa}^-$ is completely analogous.

Linear bound. We claim that, if $R^t \in \mathcal{N}_{\mathcal{K}^0}(D)$ and $g^t \leq \varepsilon^2/2$, then

$$\begin{aligned} \mathbb{E}[\delta_j^{t+1} - \delta_j^t | \mathcal{F}_t] &\leq \frac{-M}{T-t} \quad j \in \kappa^+, \\ \mathbb{E}[\delta_j^{t+1} - \delta_j^t | \mathcal{F}_t] &\geq \frac{M}{T-t} \quad j \in \kappa^-. \end{aligned} \tag{4.18}$$

Assuming Eq. (4.18), if $g^t \geq \varepsilon^2/2$, then there exists $j \in \kappa^+$ s.t. $\delta_j \geq \varepsilon/\sqrt{2}$ or some $j \in \kappa^-$ s.t. $\delta_j \leq -\varepsilon/\sqrt{2}$, hence from (4.18) we can conclude our desired linear bound

$$\mathbb{E}[(\delta^{t+1} - \delta^t)'_\kappa (\delta^t)_\kappa | \mathcal{F}_t] \leq \frac{-M}{T-t}.$$

Now let us prove (4.18). Let $\mathcal{K}^0$ be such that $R^t \in \mathcal{N}_{\mathcal{K}^0}(D^t)$. Recall that σ_j^t is the indicator that a request j is served at time t. Since only requests in $\mathcal{K}^0$ may be served, we have the identity $\mathbb{E}[I^{t+1}] = p_{\mathcal{R}} + I^t - \mathbb{E}[A_{\mathcal{K}^0}\sigma_{\mathcal{K}^0}^t]$, which implies

$$\mathbb{E}[R^{t+1} - R^t] = \frac{1}{(T-t-1)}\big(R^t - A_{\mathcal{K}^0}\mathbb{E}[\sigma_{\mathcal{K}^0}^t]\big).$$

If $\mathcal{B}$ is the optimal basis for $\mathrm{LP}(R^t, D)$ and $\bar{\mathcal{B}}$ is optimal for $\mathrm{LP}(R^{t+1}, D^{t+1})$, then we can write the previous equation in vector form (adding n components)

$$\mathbb{E}\left[\bar{\mathcal{B}}\begin{pmatrix} y^{t+1} \\ 0 \\ u^{t+1} \end{pmatrix} - \mathcal{B}\begin{pmatrix} y^t \\ 0 \\ u^t \end{pmatrix} \middle| \mathcal{F}_t\right] = \frac{1}{T-t-1}\mathcal{B}\begin{pmatrix} y^t - \mathbb{E}[\sigma_{\mathcal{K}^0}^t] \\ 0 \\ x \end{pmatrix},$$

where x is set so that the equation is satisfied in the n new components. Now we can multiply this equation by $\mathcal{B}^{-1}$ and use Lemma 4.5.4 (item 2) to conclude that, in components $j \in \kappa^+ \cup \kappa^-$ we have

$$\mathbb{E}[(y^{t+1} - y^t)_\kappa | \mathcal{F}_t] = \frac{1}{T-t-1}(y^t - \mathbb{E}[\sigma^t_{\mathcal{K}^0}])_\kappa. \tag{4.19}$$

For $j \in \kappa^+$ we use Lemma 4.4.2 to conclude that j is served whenever possible (the request does not use resources low on inventory), hence $\mathbb{E}[\sigma^t_j] = \mathbb{E}[\mathbb{1}_{\{J^t=j \text{ or } Q^t_j>0\}}] \geq D_j$. We know $\theta^t_j = \frac{1}{2}D_j$ and $\theta^{t+1}_j = \frac{1}{2}D_j$ (since $\tilde{\kappa} = \kappa$ according to Eq. (4.17)). From Eq. (4.19),

$$\mathbb{E}[\delta^{t+1}_j - \delta^t_j | \mathcal{F}_t] = \frac{1}{T-t-1}(y^t_j - \mathbb{E}[\sigma^t_j]) \leq \frac{1}{T-t-1}(-D_j/2 + \varepsilon),$$

where we used $y^t_j \leq D^t_j/2 + \varepsilon$ for $j \in \kappa^+$ (see Eq. (4.16)), and the fact $\mathbb{E}[\sigma^t_j] \geq D^t_j$. This gives Eq. (4.18) in the case for $j \in \kappa^+$.

On the other hand, for $j \in \kappa^-$, since $j \notin \mathcal{K}^0$, from Eq. (4.19) we have

$$\mathbb{E}[\delta^{t+1}_j - \delta^t_j | \mathcal{F}_t] = \frac{1}{T-t-1}y^t_j \geq \frac{1}{T-t-1}(D_j/2 - \varepsilon),$$

where we used $y^t_j \geq D_j/2 - \varepsilon$ (see Eq. (4.16)). This concludes the proof of Eq. (4.18).

Quadratic bound. Finally, we prove $||\delta^{t+1} - \delta^t|| = O(\frac{1}{T-t})$. By Lemma 4.5.4 (item 1) we have $\delta^t = (y^t - \frac{1}{2}D)_\kappa$ and $\delta^{t+1} = (y^{t+1} - \frac{1}{2}D)_{\tilde{\kappa}}$. Using Eq. (4.17), we have $\delta^{t+1} = (y^{t+1} - \frac{1}{2}D)_\kappa$. It follows

$$||\delta^{t+1} - \delta^t|| \leq ||(y^t - y^{t+1})_\kappa||.$$

The term $||(y^t - y^{t+1})_\kappa||$ is bounded by the Lipschitz continuity just as in the proof of Eq. (4.17). □

Theorem 4.5.5. *Let $\mathcal{K}$ be the centroid such that $\mathbb{E}[R^0] \in \mathcal{N}_\mathcal{K}$ is such that $\max_l \psi'_l(\mathbb{E}[R^0] - r_\mathcal{K}) \leq \epsilon/2$ and let $\tau^{\epsilon',\mathcal{B}}_{\text{cone}}$ be as in Eq. (4.14). Then,*

$$\mathbb{P}[T - \tau^{\epsilon',\mathcal{B}}_{\text{cone}} > \ell, R^0 - A_\mathcal{K}\bar{Z}^T_\mathcal{K} \in \text{cone}(\mathcal{N}_\mathcal{K}(\mathcal{B}, D))] \leq \theta_1 e^{-\theta_2 \ell},$$

for some constants θ_1, θ_2 that do not depend on (T, I^0).

Lemma 4.5.6. *Fix $\mathcal{K}$. Let $\mathcal{K}_0$ and $\mathcal{K}_1$ be two centroid neighbors of $\mathcal{K}$. There exists $\epsilon', \delta > 0$ such that: if $\exists R$ such that $d(R, \mathcal{N}_{\mathcal{K}}(D)), d(R, \mathcal{N}_{\mathcal{K}_0}(D)), d(R, \mathcal{N}_{\mathcal{K}_1}(D)) \leq \epsilon'$ then*

1. *$\mathcal{K}$,$\mathcal{K}_0$, and $\mathcal{K}_1$ share a basis $\mathcal{B}$: $\mathcal{K}_0 = \mathcal{K} \cup \kappa_0^+ \backslash \kappa_0^-$ and $\mathcal{K}_1 = \mathcal{K} \cup \kappa_1^+ \backslash \kappa_1^-$ with $\kappa_0^+, \kappa_1^+ \subseteq K^+(\mathcal{B})$ and $\kappa_0^-, \kappa_1^- \subseteq K^-(\mathcal{B})$.*
2. *R is in the strict interior of $\text{cone}(\mathcal{K}, \mathcal{B})$, i.e., for the vectors $\{\psi_l\}$ characterizing $\text{cone}(\mathcal{K}, \mathcal{B})$ as in Lemma 4.4.13, we have $\max_\ell \psi_l'(R - r_{\mathcal{K}}(D)) \leq -\delta$.*

Proof. Let us write $\mathcal{K}_0 = \mathcal{K} \cup \kappa_0^+ \backslash \kappa_0^-$ and $\mathcal{K}_1 = \mathcal{K} \cup \kappa_1^+ \backslash \kappa_1^-$. Then, by the Lipschitz continuity of the LP, if y solves $\text{LP}(R, D)$, we must have that $y_j \geq D_j/2 - \epsilon$ for all $j \in \kappa_0^+ \cup \kappa_1^0$ as well as $y_j \geq D_j/2 - \epsilon$ for all $j \in \kappa_0^- \cup \kappa_1^-$. We must have $y_j \leq \epsilon$ for all j outside these sets. Thus, R must be of the form

$$R = r_{\mathcal{K}}(D) + \frac{1}{2} A_{\kappa_0^+ \cup \kappa_1^+} D_{\kappa_0^+ \cup \kappa_1^+} - \frac{1}{2} A_{\kappa_0^- \cup \kappa_1^-} D_{\kappa_0^- \cup \kappa_1^-} \pm M\epsilon.$$

For the first item, assume by way of contradiction that the bases are different. That is, $\mathcal{K}_0$ is generated (from $\mathcal{K}$) by a basis $\mathcal{B}_0$ and $\mathcal{K}_1$ is generated by a basis $\mathcal{B}_1 \neq \mathcal{B}_0$. Let $\{\psi_l^0\}$ be the vectors characterizing $\text{cone}(\mathcal{K}, \mathcal{B}_0)$. Then, we must have that $\max_l (\psi_l^0)'(\frac{1}{2} A_{\kappa_0^+ \cup \kappa_1^+} D_{\kappa_0^+ \cup \kappa_1^+} - \frac{1}{2} A_{\kappa_0^- \cup \kappa_1^-} D_{\kappa_0^- \cup \kappa_1^-}) \leq M\epsilon$. On the other hand, we claim that, if there is $j \in \kappa_1^+$ or $j \in \kappa_1^-$ such that $j \notin K^+(\mathcal{B}_0) \cup K^-(\mathcal{B}_0)$ then we would have $\max_l (\psi_l^0)'(\frac{1}{2} A_{\kappa_0^+ \cup \kappa_1^+} D_{\kappa_0^+ \cup \kappa_1^+} - \frac{1}{2} A_{\kappa_0^- \cup \kappa_1^-} D_{\kappa_0^- \cup \kappa_1^-}) \geq \zeta \min_j D_j$. The two previous inequalities are a contradiction.

To see the claim, recall from Lemma 4.4.13 that the vectors defining $\text{cone}(\mathcal{K}, \mathcal{B})$, for some generic $\mathcal{B}$, correspond to neighboring centroids with $|\kappa^+| + |\kappa^-| = 1$ and are such that

$$\psi[\kappa^+, \kappa^-]' A_j < 0, \ j \in K^+(\mathcal{B}), j \neq \kappa^+ \quad \text{and} \quad \psi[\kappa^+, \kappa^-]' A_j > 0, \ j \in K^-(\mathcal{B}), j \neq \kappa^-. \tag{4.20}$$

Fix such a vector $\psi[\kappa^+,\kappa^-]$ for $\text{cone}(\mathcal{K},\mathcal{B}^0)$. From Lemma 4.4.13 we also have $\psi[\kappa^+,\kappa^-]'A_j > 0$ for all $j \in (\kappa^+)^c \cup \kappa_1^+$ and $\psi[\kappa^+,\kappa^-]'A_j < 0$ for any $j \in (\kappa^-)^c \cup \kappa_1^-$. This last fact together with Eq. (4.20) imply $\psi[\kappa^+,\kappa^-]'(\frac{1}{2}A_{\kappa_0^+\cup\kappa_1^+}D_{\kappa_0^+\cup\kappa_1^+}) > \zeta \min_j D_j$ and $\psi[\kappa^+,\kappa^-]'(\frac{1}{2}A_{\kappa_0^-\cup\kappa_1^-}D_{\kappa_0^-\cup\kappa_1^-}) < -\zeta \min_j D_j$, from here the claim follows.

For the second item we can assume $\mathcal{B}_0 = \mathcal{B}_1 = \mathcal{B}$. Fix a vector $\psi[\kappa^+,\kappa^-]$ defining $\text{cone}(\mathcal{K},\mathcal{B})$. Then, it must be that either $\kappa^+ \neq \kappa_0^+ \cup \kappa_1^+$ or $\kappa^- \neq \kappa_0^- \cup \kappa_1^-$ because $\mathcal{K}_0 \neq \mathcal{K}_1$. Thus, from Eq. (4.20), $\psi[\kappa^+,\kappa^-]'A_{\kappa_0^+\cup\kappa_1^+}D_{\kappa_0^+\cup\kappa_1^+} \leq -\zeta \min_j D_j$. Similarly, $\psi[\kappa^+,\kappa^-]'(-A_{\kappa_0^-\cup\kappa_1^-}D_{\kappa_0^-\cup\kappa_1^-}) \leq -\zeta \min_j D_j$. In turn, we have that $\psi[\kappa^+,\kappa^-]'(R - r_{\mathcal{K}}(D)) \leq -2\zeta \min_j D_j + M\epsilon$. This is negative for all ϵ sufficiently small. □

Proof of Theorem 4.5.5. Throughout, $\mathcal{K}$ and the basis $\mathcal{B}$ are fixed. Also, we write $r_{\mathcal{K}} = r_{\mathcal{K}}(p)$. Recall that whenever $\mathbb{E}[R^0] \in \mathcal{N}_{\mathcal{K}}(p)$, we have the following bound from Theorem 4.5.2

$$\mathbb{P}[T - \tau_{\text{region}}^{\epsilon,\mathcal{K}} > \ell] \leq \theta_1 e^{-\theta_2 \ell} \quad \text{where} \quad \tau_{\text{region}}^{\epsilon,\mathcal{K}} = \inf\{t \leq T : d(R^t, \mathcal{N}_{\mathcal{K}}(p)) \geq \epsilon\}.$$

For each ℓ, define the event $\Omega^\ell := \{T - \tau_{\text{region}}^{\epsilon,\mathcal{K}} \leq \ell\}$ and observe that, by the previous, $\mathbb{P}[(\Omega^\ell)^c]$ is exponentially small. The event of interest is $\mathcal{D} := \{T - \tau_{\text{cone}}^{\epsilon,\mathcal{B}} > \ell, \mathcal{M}(\mathcal{B})\}$ and hence it suffices to bound $\mathbb{P}[\mathcal{D}, \Omega^\ell]$.

Outline of the proof. We are left to bound the measure of $\mathcal{D} \cap \Omega^\ell$. To do so, we consider two cases. First, we assume that at most one action region other than $\mathcal{N}_{\mathcal{K}}(D)$ was visited; this corresponds to the case where the process starts close to the boundary of the cone and it is therefore the challenging case. Second, we assume that more action regions were visited; we show that this only happens when the process is in the strict interior of the cone and has a straightforward analysis.

First case (boundary). We assume that over the interval $[1, \tau_{\text{cone}}^{\epsilon,\mathcal{B}}]$ only two action regions are visited, $\mathcal{N}_{\mathcal{K}}(p)$ and $\mathcal{N}_{\mathcal{K}^0}(p)$ for some neighbor $\mathcal{K}^0$. We will bound the measure of the

desired event as follows. We define a process S^t with zero-mean increments and the following properties:

$$S^1 \le T\epsilon/2 \text{ and } S^T \le 0 \text{ a.s.} \quad \text{and} \quad \tau^{\epsilon,\mathcal{B}}_{\text{cone}} < T-\ell \implies S^t > \epsilon(T-t) \text{ for some } t \in [1, T-\ell].$$

In other words, we define a process with negative drift, that starts bounded and ends negative (initial and terminal conditions above), and it must grow larger than a linear target in order to have $\tau^{\epsilon,\mathcal{B}}_{\text{cone}} < T-\ell$. Standard concentration arguments show that this happens with exponentially low probability. We make this precise after we define the process S^t.

Let l^0 be such that $\psi'_{l^0}(R^t - r_{\mathcal{K}}) > \epsilon$ at time $t = \tau^{\epsilon,\mathcal{B}}_{\text{cone}}$. In other words, the normal vector ψ_{l^0} is the one that makes the condition $\max_l \psi'_l(R^t - r_{\mathcal{K}}) \le \epsilon$ fail. The normal ψ_{l^0} is defined by some κ^+, κ^- such that $|\kappa^+ \cup \kappa^-| = 1$ (see Lemma 4.4.13). Since only two action regions are visited, we claim that the other action region corresponds to the neighbor $\mathcal{K}^0 = \mathcal{K} \cup \kappa^+ \setminus \kappa^-$. Indeed, the segment $L = \{\alpha r_{\mathcal{K}} + (1-\alpha) r_{\mathcal{K}^0} : \alpha \in [0,1]\}$ is completely contained in $\mathcal{N}_{\mathcal{K}}(p) \cap \mathcal{N}_{\mathcal{K}^0}(p)$ whenever $|\kappa^+| + |\kappa^-| = 1$, see Lemma 4.4.10. Furthermore, ϵ is small enough so that $L \pm \epsilon \subseteq \mathcal{N}_{\mathcal{K}}(p) \cap \mathcal{N}_{\mathcal{K}^0}(p)$, which proves the claim.

We have the inventory equation $I^s = I^0 + Z^s_{\mathcal{R}} - AY^s$. Since $(T-s)R^s = I^s + (T-s)p_{\mathcal{R}}$ and $(T-s)r_{\mathcal{K}} = (T-s)A_{\mathcal{K}}p_{\mathcal{K}}$,

$$\begin{aligned} I^s - (T-s)r_{\mathcal{K}} &= I^0 + Z^s_{\mathcal{R}} - AY^s - Tr_{\mathcal{K}} + sA_{\mathcal{K}}p_{\mathcal{K}} \\ (T-s)(R^s - r_{\mathcal{K}}) &= T(\mathbb{E}[R^0] - r_{\mathcal{K}}) + \widehat{Z}^s_{\mathcal{R}} - AY^s + sA_{\mathcal{K}}p_{\mathcal{K}} \end{aligned} \tag{4.21}$$

where we have defined the centred process $\widehat{Z}^t_{\mathcal{R}} \coloneqq Z^t_{\mathcal{R}} - tp_{\mathcal{R}}$. Over the interval $[1, \tau^{\epsilon,\mathcal{B}}_{cone}]$ the only requests accepted correspond to $\mathcal{K} \cup \kappa^+$, hence $Y^s = Y^s_{\mathcal{K}} + Y^s_{\kappa^+}$. Additionally, all of the requests in $\mathcal{K} \setminus \kappa^-$ are accepted, hence $Y^s_{\mathcal{K}\setminus\kappa^-} = Z^s_{\mathcal{K}\setminus\kappa^-}$. Finally, by Lemma 4.4.13 we know that ψ_{l_0} is orthogonal to the columns of A corresponding to κ^+ and κ^-, hence we have the following identities

$$\psi'_{l_0} AY^s = \psi'_{l_0} A_{\mathcal{K}\setminus\kappa^-} Z^s_{\mathcal{K}\setminus\kappa^-} \quad \text{and} \quad \psi'_{l_0} A_{\mathcal{K}} p_{\mathcal{K}} = \psi'_{l_0} A_{\mathcal{K}\setminus\kappa^-} p_{\mathcal{K}\setminus\kappa^-}.$$

Defining the centred process $\widehat{Z}^t_{\mathcal{J}} \coloneqq Z^t_{\mathcal{J}} - tp_{\mathcal{J}}$ and using the previous identities together with Eq. (4.21) we have

$$(T-s)\psi'_{l^0}(R^s - r_{\mathcal{K}}) = T\psi'_{l^0}(\mathbb{E}[R^0] - r_{\mathcal{K}}) + \psi'_{l^0}(\widehat{Z}^s_{\mathcal{R}} - A_{\mathcal{K}\setminus\kappa^-}\widehat{Z}^s_{\mathcal{K}\setminus\kappa^-}) \quad \forall s \leq \tau^{\epsilon,\mathcal{B}}_{cone}.$$

From the previous equation, we can define the following process

$$S^t \coloneqq T\psi'_{l^0}(\mathbb{E}[R^0] - r_{\mathcal{K}}) + \psi'_{l^0}(\widehat{Z}^t_{\mathcal{R}} - A_{\mathcal{K}\setminus\kappa^-}\widehat{Z}^t_{\mathcal{K}\setminus\kappa^-}), \quad t \in [1,T].$$

We have proved the following:

$$\tau^{\epsilon,\mathcal{B}}_{\text{cone}} < T-\ell \iff S^t > (T-t)\epsilon \quad \text{for some } t \in [0, T-\ell]. \tag{4.22}$$

The process S^t has zero-mean increments. Furthermore, $S^0 = T\psi'_{l^0}(R^{\tau^\epsilon} - r_{\mathcal{K}}) \leq T\epsilon/2$ by assumption. The remainder of the proof is showing $S^T \leq 0$. Using that ψ_{l^0} is orthogonal to the columns A_{κ^-},

$$\begin{aligned} S^T &= T\psi'_{l^0}(\mathbb{E}[R^0] - r_{\mathcal{K}}) + \psi'_{l^0}(\widehat{Z}^T_{\mathcal{R}} - A_{\mathcal{K}}\widehat{Z}^T_{\mathcal{K}}) \\ &= \psi'_{l^0}(I^0 + Z^T_{\mathcal{R}} - A_{\mathcal{K}}Z^T_{\mathcal{K}}), \end{aligned}$$

In the event $\mathcal{M}(\mathcal{B})$ we have $I^0 + Z^T_{\mathcal{R}} - A_{\mathcal{K}}Z^T_{\mathcal{K}} \in \text{cone}(\mathcal{K},\mathcal{B})$, hence we conclude $S^T \leq 0$.

From Eq. (4.22), we deduce the inequality

$$\mathbb{P}[\mathcal{D},\Omega^\ell] = \mathbb{P}\big[T - \tau^{\epsilon,\mathcal{B}}_{\text{cone}} > \ell, \mathcal{M}(\mathcal{B}), \Omega^\ell\big] \leq \mathbb{P}\left[\bigcup_{t\in[T-\ell]} \big\{S^t \geq \epsilon(T-t)\big\}, \max_l S^T_l \leq 0\right] \leq \theta_1 e^{-\theta_2 \ell},$$

for some $\theta_1, \theta_2 > 0$. The final bound follows from the analysis of a random walk crossing a positive moving threshold conditional on being negative at the end of the horizon. This is formally proved in Lemma C.2.1 in the appendix.

Second case (strict interior). Assume that over the interval $[1, \tau^{\epsilon,\mathcal{B}}_{\text{cone}}]$ the process R^t visits three or more action regions. This case corresponds to where the process is in the strict interior of the cone. Indeed, by Theorem 4.5.2, the process R^t remains close to each action

region visited. Formally, $d(R^t, \mathcal{N}_{\mathcal{K}^k}(p)) \leq \epsilon$ for each $\mathcal{K}^k$ visited over the interval $[1, T-\ell]$ with probability $\theta_1 e^{-\theta_2(T-\ell)}$. By Lemma 4.5.6, it must be that R^t is far from the boundary of the cone over the interval $[1, T-\ell]$, which completes the proof. □

Proof of Proposition 4.4.5. By Lemma 4.5.1 we have that the policy performs basic allocation up to time $\tau^{\epsilon',\mathcal{K}}_{\text{region}} \wedge \tau^{\epsilon',\mathcal{B}}_{\text{cone}}$. From Lemma 4.4.14, the wastage is bounded by $M(T - \tau^{\epsilon',\mathcal{K}}_{\text{region}} \wedge \tau^{\epsilon',\mathcal{B}}_{\text{cone}})$. Finally, from Theorems 4.5.2 and 4.5.5 we obtain that $\mathbb{E}[T - \tau^{\epsilon',\mathcal{K}}_{\text{region}} \wedge \tau^{\epsilon',\mathcal{B}}_{\text{cone}}] \leq M$. □

4.6 A Tale of Two Objectives

The usual two metrics for measuring performance of algorithms are (1) high reward and (2) small cost. These metrics are oftentimes at odds and it is a modeling choice to pursue one over the other. We just proved that BudgetRatio is near-optimal in the sense of reward maximization, now we prove that it can, at the same time, be cost minimizing.

In other words, we prove that, with suitably tuned parameters, BudgetRatio maintains bounded regret while achieving—in a competitive ratio sense— the least holding cost among *all policies* π that have regret $o(T)$.

If we identify requests with customers, then it is natural to assign holding costs to capture the dissatisfaction of requests when queued. We also consider the case where inventory has a holding cost, we show in Lemma 4.6.1 that inventory cost can be transformed to queue costs.

Let $c \in \mathbb{R}^n_{>0}$ be a vector of holding-cost coefficients: each period a request type j is not

served, we incur a cost c_j. The holding cost of a policy is

$$\mathcal{C}_c^\pi(T, I^0) \coloneqq \mathbb{E}_{I^0}^\pi \left[\sum_{t=1}^{T} c \cdot Q^t \right],$$

In this section we assume that there is no restock and we adopt the following standard asymptotic framework [78]: given a fixed inventory $\bar{I}^0$, we consider a sequence of instances indexed by the horizon T such that the T-th instance has initial inventory $I_T^0 = T\bar{I}^0$.

The main result of this section is that we can tune BudgetRatio to obtain constant regret and at the same time fluid optimality w.r.t. the previous cost. In order to do this, we simply modify the threshold used by the algorithm. Specifically, in step 8 of Algorithm 5, a request is accepted if $y_j \geq \frac{1}{2}\bar{p}_j$. We modify this rule to: accept whenever $y_j \geq \alpha^T p_j$, where now α^T is a tuning parameter. With this simple modification we obtain the following result:

Proposition 4.2.7. *Suppose that the deterministic relaxation* $\mathrm{LP}(\mathbb{E}[R^0], D)$ *has a unique solution* $\bar{y}$ *and that* $\bar{y}_j < p_j$ *for at least one* $j \in \mathcal{J}$*. Then, Algorithm 5 with tuning parameter* $\alpha^T = T^{-1/4}$ *achieves simultaneously (1) constant regret for reward maximization and (2) asymptotic optimality for cost minimization, i.e.,*

$$\liminf_{T\uparrow\infty} \frac{\mathbb{E}^\pi[\sum_{t=1}^T c \cdot Q^t]}{\mathbb{E}^{\widehat{\pi}}[\sum_{t=1}^T c \cdot Q^t]} \geq 1 \quad \textit{for any policy } \pi \textit{ with regret } o(T).$$

We focus on holding cost of requests, but all of our results apply to the case where inventory has a holding cost too, as shown in the next result.

Lemma 4.6.1. *Consider the inventory holding cost minimization, i.e., for some coefficients* $h \in \mathbb{R}^d_{>0}$ *we set the cost of a policy as* $\sum_{t=1}^T h \cdot I^t$*. This objective can be transformed to a request holding cost minimization with* $c_j \coloneqq A_j' h$*.*

Proof. We have the inventory equation $I^t = I^0 - AY^t$, where $Y^t \in \mathbb{N}^n$ represents the total amount of each request served over the interval $[1, t]$. Furthermore, we have $Q^t = Z^t - Y^t$.

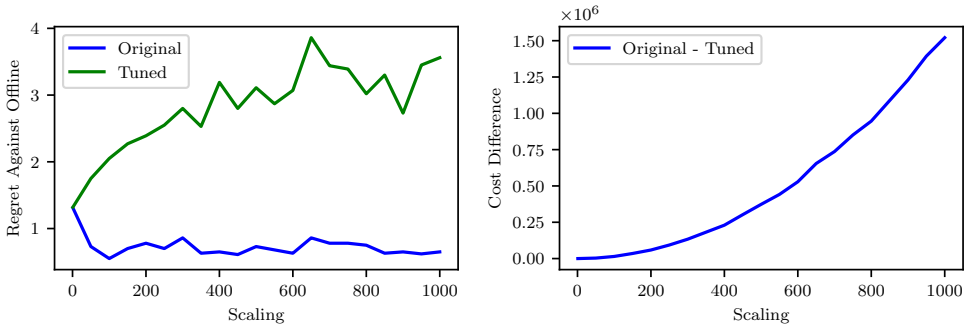


Figure 4.7: (LEFT) Regret of the two algorithms, we observe that the tuned version has slightly higher regret, this is due to its aggressiveness (it accepts more frequently). (RIGHT) Difference in cost of the two algorithms, we observe that the difference diverges as $\Omega(T^2)$, i.e., the tuned algorithm has a superior performance by orders of magnitude with respect to the cost metric (observe the scale is 10^6).

These two equations together imply $I^t = I^0 - A(Z^t - Q^t)$, thus

$$\sum_{t=1}^{T} h \cdot I^t = Th \cdot I^0 - \sum_{t=1}^{T} h'AZ^t + \sum_{t=1}^{T} h'AQ^t.$$

In terms of optimization, the first two terms are constant (cannot be affected by the controller), hence minimizing inventory cost is equivalent to minimizing queue cost. □

4.6.1 Numerical Demonstration

Before we give the proof of Proposition 4.2.7, we present a numerical study that clearly shows the different behaviour of the tuned algorithm. Let us consider our guiding example (see Section 4.3.1), which has $n = 6$ types and $d = 2$ resources. We assign costs $c_j = 1$, starting inventories $I_a = 9, I_b = 11$, and a horizon of $T = 20$ periods. We scale this system by a factor k just as described at the beginning of this section.

In Fig. 4.7 we present the performance of the two algorithms (original and tuned). We observe that the tuned algorithm has slightly higher regret, but it has a cost that scales at a slower rate. Indeed, the difference in cost between the algorithms is $\Omega(T^2)$. In Fig. 4.8 we

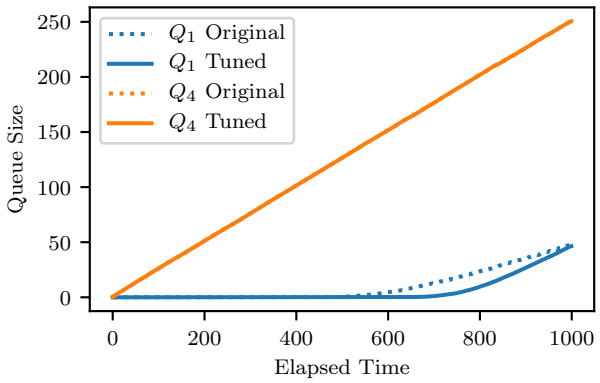


Figure 4.8: Queues maintained by the two algorithms. Queues for types $j = 2$ and $j = 3$ are maintained at zero by both algorithms, hence we do not plot them. We observe that the queue for $j = 4$ (least desirable request) grows at the same linear rate for both algorithms, this is because the reward maximizing policy completely rejects this type. Finally, the queue for type $j = 1$ presents a different behaviour; it grows at a slower rate for the tuned algorithm, hence accumulating less cost.

present the queues maintained by the two algorithms, where we can see that, for the type $j =$, they grow at a different rate, thus explaining the difference in cost.

4.6.2 Analysis of the Tuned Algorithm

To prove Proposition 4.2.7, we define a lower bound $C(c, \bar{y}, T)$ for the holding cost of any policy with regret $o(T)$, then show that, for any policy π with regret $o(T)$, we have:

$$\liminf_{T\uparrow\infty} \frac{\mathbb{E}^{\pi}[\sum_{t=1}^{T} c \cdot Q^t]}{C(c, \bar{y}, T)} \geq 1. \tag{4.23}$$

Finally, we show that BudgetRatio achieves this bound:

$$\lim_{T\uparrow\infty} \frac{\mathbb{E}^{\hat{\pi}}[\sum_{t=1}^{T} c \cdot Q^t]}{C(c, \bar{y}, T)} = 1. \tag{4.24}$$

The lower bound. We now proceed to construct the bound $C(c, \bar{y}, T)$. Let us recall the

deterministic linear relaxation at time 0

$$\begin{aligned} \mathrm{LP}(\mathbb{E}[R^0], D) \qquad \max \quad & v'x \\ \text{s.t.} \quad & Ay \leq \mathbb{E}[R^0], \\ & y \leq \mathbb{E}[D^0], \\ & y \in \mathbb{R}^n_{\geq 0}. \end{aligned}$$

Let $\bar{y}$ be the LP's solution. Recall that $\bar{y}_j$ represents the fraction of requests j accepted, hence $\bar{y}_j T$ is a first-order proxy for number requests j that the optimal policy should accept. Indeed, let us denote by $\tilde{Y}_j^T$ the number of requests j accepted by the offline policy. It follows from the Lipschitz continuity of LPs that $\tilde{Y}_j^T = \bar{y}_j T + o(T)$, hence any policy π that has sub-linear regret has $\mathbb{E}^\pi[Y_j^T] = \bar{y}_j T + o(T)$. Let $\delta_{j,\pi}^s = \mathbb{E}^\pi[Y_j^s]$ and $q_{j,\pi}^s = \mathbb{E}^\pi[Q_j^s]$ then $q_{j,\pi}^s = p_j s - \delta_{j,\pi}^s$. The following LP is our lower bound for the (constrained) holding-cost problem

$$\begin{aligned} C(c, \bar{y}, T) := \quad \min \quad & \textstyle\sum_{t=1}^T c \cdot q^t \\ \text{s.t.} \quad & q_j^t = p_j t - \textstyle\sum_{s=1}^t x_j^s && t \in [T],\ j \in \mathcal{J} \\ & \textstyle\sum_{s=1}^t x_j^s \leq p_j t && t \in [T],\ j \in \mathcal{J} \\ & \textstyle\sum_{t=1}^T x_j^t = \bar{y}_j T && j \in \mathcal{J} \\ & x_j^t \geq 0 && t \in [T],\ j \in \mathcal{J}. \end{aligned} \tag{4.25}$$

This problem has the following interpretation: the variables x_j^t represent how much of type-j is accepted at time t, hence the first constraint is the fluid queue (expected arrivals minus acceptance). The second constraint guarantees that the queues remain positive. Finally, the third constraint imposes that the overall acceptance matches the deterministic relaxation.

Now we are ready to prove Proposition 4.2.7, we separate the proof in two parts (cost and reward) and present the bound on the cost first.

Changes in the geometry. The centroid sets remain unchanged as they do not depend

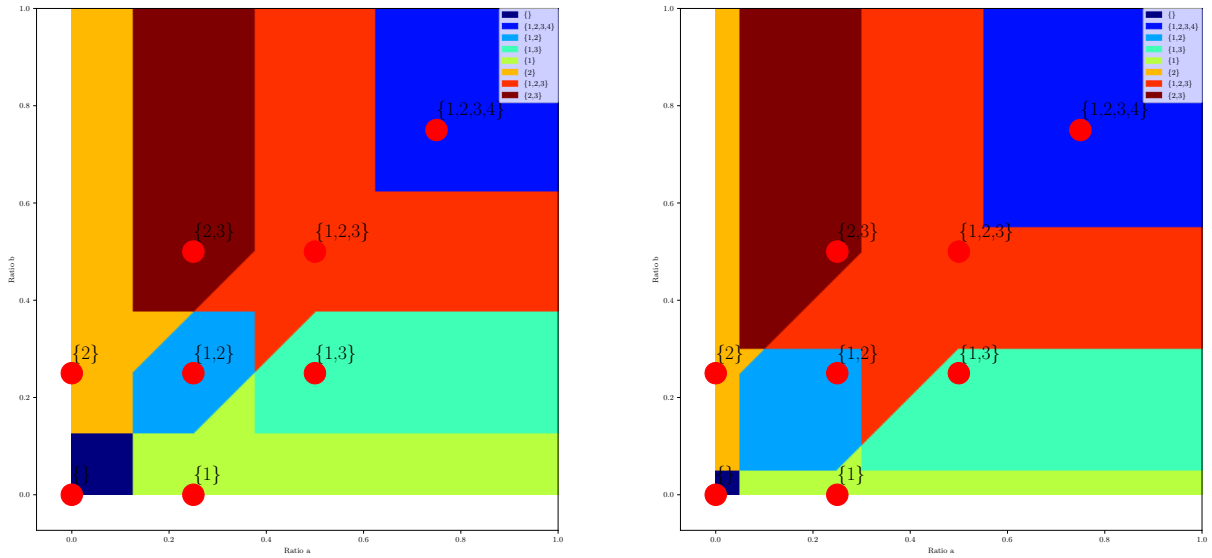


Figure 4.9: Action regions with different thresholds. On the left we use 0.5 and on the right 0.2. We observe that the structure of centroids and neighbourhoods is maintained. The only difference introduced by the threshold is in the shape of the regions. Recall that, the smaller the threshold, the more aggressive the policy (serves more frequently). Observe that, for example, the region corresponding to $\{1, 2, 3, 4\}$ (top-right) is larger under the threshold 0.2, i.e., request 4 is served in a larger part of the space, which is explained by the extra aggressiveness of a smaller threshold.

on the choice of the threshold. Similarly, the optimal bases at a centroid and the centroid neighbors also remain the same. By using the threshold α^T instead of $1/2$, the action regions $\mathcal{N}_{\mathcal{K}}(D)$ and $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$ get shifted, see Fig. 4.9. Specifically, in Lemma 4.4.10 (item 3), we replace the statement with:

> $R \in \mathcal{N}_{\mathcal{K}}(\mathcal{B}, D)$ if and only if $R = r_{\mathcal{K}}(D) + A_{K^+} x_{K^+} - A_{K^-} x_{K^-} + b$ where $x_j \in [0, \alpha^T D_j]$ for all $j \in K^+$ and $x_j \in [0, (1 - \alpha^T) D_j]$ for all $j \in K^-$.

As before, $\mathcal{N}_{\mathcal{K}}(D) = \cup_{\mathcal{B}} \mathcal{N}_{\mathcal{K}}(\mathcal{B}, D)$, where the union is over all bases optimal at the centroid $\mathcal{K}$. See Fig. 4.10 for an more detailed illustration of this "shifting".

Proof of Proposition 4.2.7 (Item 1, Reward Maximization). The proof of our

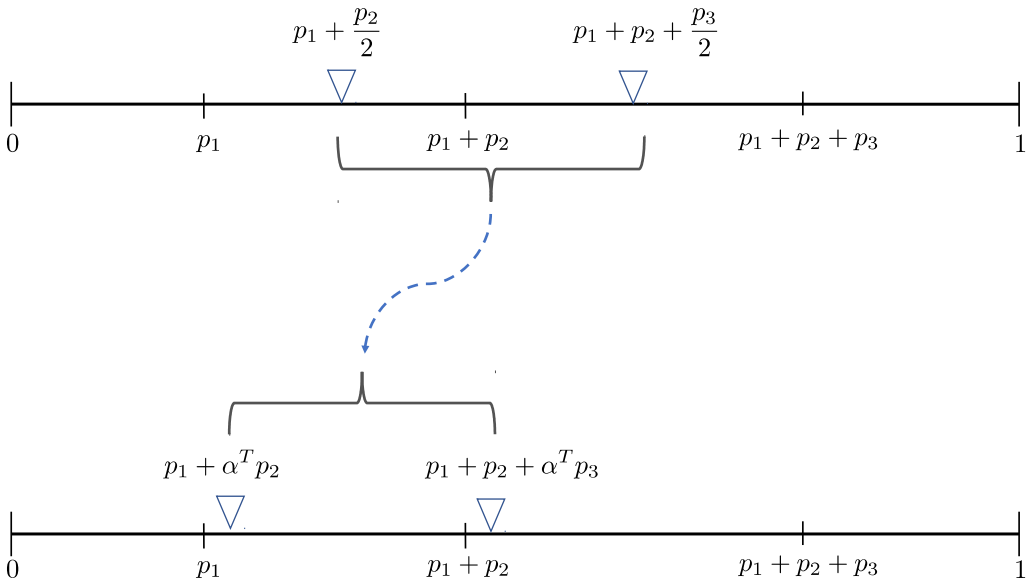


Figure 4.10: Changing the threshold from $1/2$ to α^T shifts the centroid neighborhood. For example, the set $\mathcal{N}_{\mathcal{K}}(\mathcal{B}, D)$ corresponding to $\mathcal{K} = \{1, 2\}$ changes from $[p_1 + p_2, p_1 + p_2 + p_3/2]$ to $[p_1 + p_2, p_1 + p_2 + \alpha^T p_3]$. Observe that making the parameter α smaller makes the algorithm more aggressive. Indeed, the centroids remain the same, but the boundary from $[k]$ to $[k+1]$ is now easier to cross (to start accepting more requests), because all the midpoints shift to left.

main result (bounded regret) remains mostly unchanged. We limit the argument here to identifying steps in the proof of Theorem 4.5.2 and Theorem 4.5.5 where the threshold α^T plays a role and prove that those steps remain unchanged. For the remaining proofs we replace the small constants ϵ, ϵ' with their analogues $\epsilon_T = \alpha^T \epsilon$ and $\epsilon'_T = \alpha^T \epsilon'$. In particular, the set $\mathcal{A}^{\epsilon}$ is replaced by the set $\mathcal{A}^{\epsilon_T}$. Notice that concentration inequalities still guarantee that $T\mathbb{P}[(A^{\epsilon_T})^c] \to 0$ as $T \to \infty$.

The proof of Theorem 4.5.2. In the statement of Lemma 4.5.4 (used in the proof of the theorem), the vector $\theta_{\mathcal{K}}(y, D)$ is defined by $(\theta_{\mathcal{K}}(y, D))_j = \alpha^T D_j$ for $j \in \kappa^+$ and $(\theta_{\mathcal{K}}(y, D))_j = (1 - \alpha^T) D_j$ instead of $(\theta_{\mathcal{K}}(y, D))_j = D_j/2$ for $j \in \kappa^+ \cup \kappa^-$.

In the drift bound in Eq. (4.18), we now have instead the conditions

$$\mathbb{E}[\delta_j^{t+1} - \delta_j^t | \mathcal{F}_t] \leq \frac{-\alpha^T M}{T-t} \quad j \in \kappa^+,$$
$$\mathbb{E}[\delta_j^{t+1} - \delta_j^t | \mathcal{F}_t] \geq \frac{\alpha^T M}{T-t} \quad j \in \kappa^-.$$

which then implies an updated version of the Lyapunov drift in Eq. (4.15):

$$\mathbb{E}[g^{t+1} - g^t | \mathcal{F}_t] \leq -\frac{-\xi^T}{T-t}, \text{ whenever } g^t \in [\varepsilon/2, \varepsilon],$$

where $\xi^T = M\alpha^T$ for some constant M. The concentration argument in [10, Theorem 3] continues to hold for $\epsilon_T = \epsilon\alpha^T$ and ξ^T whenever ϵ^T, ξ^T shrink slower than $1/\sqrt{T}$.

The proof of Theorem 4.5.5. This proof remains mostly unchanged with the exception of replacing ϵ with ϵ_T as discussed before. We observe that the random walk in Lemma C.2.1 continues to produce an exponentially decaying tail as long as ϵ_T shrinks slower than $1/\sqrt{T}$. □

Proof of Proposition 4.2.7 (Item 2, Cost Minimization). We start with the lower bound, i.e., Eq. (4.23). Fix a policy π and recall $q^s := \mathbb{E}^\pi[Q^s]$, $\delta_j^s = \mathbb{E}^\pi[Y_j^s]$, and $x_j^t = \mathbb{E}[Y_j^t - Y_j^{t-1}] \geq 0$ for $t \in [T]$ (with $Y^0 = 0$). The expected queue length satisfies $q_j^s = p_j s - \delta_j^s$. Also, $q_j^t, x_j^t \geq 0$ for all $t \in [T]$. Because the queue must be positive we also have $\sum_{s=1}^t x_j^s \leq p_j t$.

In turn, $\mathbb{E}^\pi[\sum_{t=1}^T c \cdot Q^t] \geq \sum_{t=1}^T c \cdot \bar{q}^t$, for any pair $\bar{q}, \bar{x} \geq 0$ that satisfies these constraints and has $\bar{\delta}^T = \sum_{s=1}^T \bar{x}_j^s = \delta^T$ (same total number of acceptances). Thus, we must have that

$$\mathbb{E}^\pi \left[\sum_{t=1}^T c \cdot Q^t \right] \geq C(c, \delta^T/T, T).$$

We make two claims: (i) $||\tilde{Y}^T - \bar{y}T||_\infty = o(T)$ and (ii) $||\delta^T - \bar{y}T||_\infty = o(T)$, where $\tilde{Y}^T$ is the offline allocation and, recall, $\delta^T = \mathbb{E}^\pi[Y^T]$. The first claim follows directly from the Lipschitz continuity of LPs [84]. For claim (ii), from (i) and the fact that π has $o(T)$ regret,

we conclude $v'\delta^T = v'\bar{y}T + o(T)$. Take a convergent subsequence of δ^T/T with limit $\tilde{y}$. By our previous equation, $v'\tilde{y} = v'\bar{y}$ and $\tilde{y}$ is feasible for $LP(\mathbb{E}[R^0], D)$, hence $\tilde{y} = \bar{y}$ by our uniqueness assumption, proving claim (ii).

Under the assumption that $\bar{y}_j < p_j$ for at least one j, we trivially have that $C(c, \bar{y}, T) \geq \Gamma T^2$ for some $\Gamma > 0$. By the Lipschitz continuity of LP in the right hand side, we have that $||q - \bar{q}|| = o(T)$ so that $C(c, \bar{y}, T) = C(c, \delta^T/T, T) + o(T^2)$. In particular,

$$\liminf_{T\uparrow\infty} \frac{\mathbb{E}^\pi[\sum_{t=1}^T c \cdot Q^t]}{C(c, \bar{y}, T)} = \liminf_{T\uparrow\infty} \frac{\mathbb{E}^\pi[\sum_{t=1}^T c \cdot Q^t]}{C(c, \delta_j^T/T, T)} \frac{C(c, \delta_j^T/T, T)}{C(c, \bar{y}, T)} \geq 1,$$

which corresponds to the lower bound Eq. (4.23).

We now prove that BudgetRatio achieves the lower bound, i.e., Eq. (4.24). We start with a simple observation about the structure of optimal solution to $C(c, \bar{y}, T)$. For every j such that $\bar{y}_j = p_j$, we have $\bar{q}_j^t = 0$ for all $t \in [T]$. For all j with $\bar{y}_j < p_j$ we let $t_j^0 := \bar{y}_j T/p_j < T$. Then, $x_j^s = p_j$ for all $s \leq \lfloor t_j^0 \rfloor$; $x_j^{\lfloor t_j^0+1 \rfloor} = \bar{y}_j T - p_j t_j^0$ and $x_j^t = 0$ for all $t > \lfloor t_j^0 \rfloor + 1$. That is, a type j is fully served up to some time t_j^0 and then not served at all.

We claim that for each j, there exists a time τ_j such that:

(I) $\mathbb{E}[Y_j^{\tau_j}] = p_j \mathbb{E}[\tau_j] + o(T)$: most requests j are taken over the interval $[1, \tau_j]$.

(II) $\mathbb{E}[|\tau_j - t_j^0|] = o(T)$: τ_j is close to the deterministic time t_j^0

(III) $\mathbb{E}[Y_j^T - Y_j^{\tau_j}] = o(T)$: few requests j are taken after τ_j.

These properties are sufficient to prove Eq. (4.24). Indeed, (I) and (II) combined guarantee that $\mathbb{E}[Y_j^{t_j^0}] = p_j t_j^0 + o(T)$. Since $\mathbb{E}[Y_j^t] \leq p_j t$ for all t, we must have that $\mathbb{E}[Y_j^t] = p_j t + o(T)$ for all $t \leq t_j^0$ so that $\mathbb{E}[Q_j^t] = o(T)$ for all $t \leq t_j^0$. By (III), $\mathbb{E}[Y_j^T - Y_j^{t_j^0}] = o(T)$ so that $\mathbb{E}[Q_j^t] = p_j(t - t_j^0) + o(T)$ for all $t \geq t_j^0$. In conclusion, for all $t \geq 0$, $\mathbb{E}[Q^t] - \bar{q}^t = o(T)$ so that

$$\mathbb{E}^{\hat{\pi}}\left[\sum_{t=1}^T c \cdot Q^t\right] = C(c, \bar{y}, T) + o(T^2).$$

This then guarantees that

$$\lim_{T\uparrow\infty} \frac{\mathbb{E}^{\widehat{\pi}}[\sum_{t=1}^{T} c \cdot Q^t]}{C(c, \bar{y}, T)} = 1.$$

Now we prove properties (I)-(III) for BudgetRatio. We proceed in two cases: $j \in \mathcal{K}$ and $j \notin \mathcal{K}$.

Case 1: $j \in \mathcal{K}$. Let us define τ_j as the first time when the algorithm rejects j, i.e., the first time the solution y to LP(R^t, p) has $y_j < \alpha^T p_j$. Let us also define $\tau = \tau^{\epsilon'_T, \mathcal{K}} \wedge \tau^{\epsilon'_T, \mathcal{B}}$ to be the time when the process completely escapes the action region and the cone. From Theorems 4.5.2 and 4.5.5 we have the properties $\mathbb{E}[T - \tau] = O(1)$ and $\mathbb{E}[Y^T - Y^\tau] = O(1)$. On the other hand, from our main result Theorem 4.2.3 we have $\mathbb{E}[\tilde{Y}^T - Y^T] = O(1)$. Recall that up to time τ we perform basic allocation, hence if we denote $\mathcal{B}$ the optimal offline basis, we have the inventory equation $I^t = I^0 - \mathcal{B}Y^t$ for all $t \leq \tau$. Using the inventory equation for times τ_j and τ we have

$$\mathcal{B}(Y^\tau - Y^{\tau_j}) = (T - \tau)(R^{\tau_j} - R^\tau) + (\tau - \tau_j)R^{\tau_j}.$$

From this equation we claim $Y_j^\tau - Y_j^{\tau_j} = o(T)$. To see this, recall that the solution y to the LP is $\mathcal{B}^{-1}R^t$ and by definition of τ_j we have $y_j \leq \alpha^T p_j$, hence the second term is $(\tau - \tau_j)R^{\tau_j} = o(T)$. The first term is bounded by Theorem 4.5.2 (see also Remark 4.5.3), where we proved that the process R^t remains close to any boundary it hits, i.e., from Theorem 4.5.2 we deduce $||R^{\tau_j} - R^\tau|| \leq \epsilon^T = \epsilon\alpha^T$, finishing the proof of our claim.

From here properties (I)-(III) are straightforward algebraic checks. For (I), since all requests j are accepted up to τ_j, $\mathbb{E}[Y^{\tau_j}] = \mathbb{E}[Z^{\tau_j}] = p_j\mathbb{E}[\tau_j]$. For (III), since $\mathbb{E}[Y^T - Y^\tau] = O(1)$ (Theorems 4.5.2 and 4.5.5) and we just proved the claim $Y_j^\tau - Y_j^{\tau_j} = o(T)$, the property holds. For (II), since $\tilde{Y}^T - Y^\tau = O(1)$ and $\tilde{Y}_j^T - T\bar{y}_j = o(T)$, from our claim we obtain $Y_j^{\tau_j} - T\bar{y}_j = o(T)$. Since, by definition, $t_j^0 = \bar{y}_j T/p_j$ and $\mathbb{E}[Y_j^{\tau_j}] = p_j\mathbb{E}[\tau_j]$, we get the result.

Case 2: $j \notin \mathcal{K}$. Set $\tau_j = 0$ and notice that, in this case, $\bar{y}_j < \alpha^T p_j$, which implies

$\tilde{Y}_j^T = T\bar{y}_j + o(T) = o(T)$. By our main result $Y^T - \tilde{Y}^T = O(1)$, hence $Y^T = o(T)$. Finally, since $t_j^0 = \bar{y}_j T/p_j = o(T)$, all properties (I)-(III) follow directly. □

4.7 Robustness with respect to primitives

4.7.1 Demand pertrubations

Suppose that, instead of knowing the distribution p exactly, we have an estimate $\widetilde{p} \neq p$. In this section, we quantify how close $\widetilde{p}$ needs to be to p so that the algorithm using the estimate $\widetilde{p}$ still has constant regret. Crucially, the measure of closeness follows directly from our geometric analysis and it is captured by centroid budgets.

For two centroids $\mathcal{K}, \mathcal{K}'$, we define their separation as

$$d_p^{\min}(\mathcal{K}, \mathcal{K}') := \min_{i \in [d]} \left| (r_{\mathcal{K}}(p) - r_{\mathcal{K}'}(p))_i \right|.$$

Intuitively, $d_p^{\min}$ measures the separation of the "true centroid budgets", i.e., centroid budgets under the true distribution p. We present an illustration of the action regions for p and $\tilde{p}$ in Fig. 4.2.

PROPOSITION 4.2.5. *Let p be the true underlying distribution and let $\delta = \min_{\mathcal{K} \neq \mathcal{K}'} d_p^{\min}(\mathcal{K}, \mathcal{K}')$. Then, for any distribution $\widetilde{p}$ such that*

$$\max_{\mathcal{K}} ||r_{\mathcal{K}}(p) - r_{\mathcal{K}}(\widetilde{p})||_\infty \leq \frac{\delta}{4},$$

BUDGETRATIO*, with the LP solved each period with $\widetilde{p}$ replacing p, produces bounded regret in the sense of Theorem 4.2.3.*

PROOF. Observe that all of our constructions $\mathcal{N}_{\mathcal{K}}(D)$ and $\mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$ are for arbitrary D. On the other hand, cones do not depend on the probability distribution. With the previous

facts, and under our assumptions, it can be verified that all of our results hold with $D = \widetilde{p}$. Indeed, we have that $r_{\mathcal{K}}(p) \in \mathcal{N}_{\mathcal{K}}(\widetilde{p})$ and the proof Theorem 4.5.2 goes through with the region being $\mathcal{N}_{\mathcal{K}}(\widetilde{p})$ (instead of $\mathcal{N}_{\mathcal{K}}(p)$). Similarly, the proof of Theorem 4.5.5 remains the same. □

Example 4.7.1 (One Dimension). The separation condition in Proposition 4.2.5 is rather intuitive in the one dimensional case. Recall that, in this case, centroids are of the form $[j]$ so that

$$\delta = \min_{j=1,\dots,n-1} e'_{[i+1]} p_{[i+1]} - e'_{[i]} p_{[i]} = \min_j p_j,$$

where we define $[0] := \varnothing$. Then, the condition of Proposition 4.2.5 requires that the estimation $\widetilde{p}$ is such that

$$\left| \sum_{k=1}^{j} p_j - \widetilde{p}_j \right| \leq \frac{\min_k p_k}{4} \quad \forall j \in [n].$$

In particular, this is satisfied if the estimation is such that $||p - \widetilde{p}||_\infty \leq \frac{\min_k p_k}{4n}$, but the condition weaker than this last requirement.

Learning Setting (Contextual Bandits with Knapsacks). Our robustness result directly implies $O(\log T)$ regret for the setting where p needs to be learned online. In this problem, requests types are usually called contexts, and the controller needs to make decisions while learning the correct probabilities. Precisely, p is unknown, but the controller observes the context $j \in [n]$ at each time, hence he can estimate it by the empirical averages $\hat{p}_j^t = \frac{1}{t} \sum_{\tau=1}^{t} \mathbb{1}_{\{J^\tau = j\}}$, where J^τ is the random arrival at τ. We get the following result.

Corollary 4.7.2 (Learning Demand Distribution). *Assume the separation condition in Proposition 4.2.5. Then, we can obtain $O(\log T)$ regret in the case that the probabilities p are unknown.*

Proof. Fix the constant $\epsilon = \frac{\delta}{4n}$. If we have an initial exploration phase of length $c \log T$, for some $c = c(\epsilon) > 0$, where all requests are rejected, then the empirical $\hat{p}^t$ is s.t. $||p - \hat{p}^t|| \leq \epsilon$

for all $t \geq c \log T$ with probability $\frac{1}{T}$. After time $c \log T$ we run BudgetRatio and obtain constant regret in the remaining periods in virtue of Proposition 4.2.5. □

4.7.2 Reward perturbations

We turn to the setting where the values v are not known, i.e., the reward v_j for accepting a request j is estimated by $\tilde{v}_j \neq v_j$. We quantify how good our estimation needs to be so that BudgetRatio using the estimates $\tilde{v}$ maintains constant regret.

In this case, it is known that a separation is required for the one-dimensional case (see also Example 4.7.5 below), but their condition is based on ranking the requests by reward, which is inherently one-dimensional. We show that our geometric analysis directly informs the necessary condition for multiple dimensions in terms of stability of centroids. Our condition makes centroids invariant to small perturbations of v by enforcing something stronger than complementary slackness.

Reminder of complementary slackness. In our setting, there are d constraints of the form $Ay + s = R$, which correspond to resources, and n constraints of the form $y + u = D$, corresponding to demand. Therefore, we have $d + n$ dual variables and an optimal primal-dual pair $(y, \lambda) \in \mathbb{R}^n \times \mathbb{R}^{d+n}$ satisfies the following complementarity: $s_i > 0 \Rightarrow \lambda_i = 0$, $u_j > 0 \Rightarrow \lambda_j = 0$, and, from the dual constraints, $([A'|I^n]\lambda)_j > v_j \Rightarrow y_j = 0$. There is the possibility that, for y_j non-basic (in particular $y_j = 0$), we have $([A'|I^n]\lambda)_j = v_j$, i.e., complementarity is not strict. This creates a fundamental problem, since it is not possible to distinguish whether or not y_j is basic (see Example 4.7.4).

We now give our definition of separation which allows us to establish the robustness result below. Recall that we associate to v the extended value vector $\bar{v} \coloneqq (v, 0, 0) \in \mathbb{R}^n \times \mathbb{R}^n \times \mathbb{R}^d$,

where the zeros correspond to unmet and surplus variables.

Definition 4.7.3 (δ-Complementarity). Let $\mathcal{B}$ be a basis and λ the dual variables associated to $(\mathcal{B}, v)$, i.e., $\lambda = (\mathcal{B}^{-1})'\bar{v}_{\mathcal{B}}$. We say that $\mathcal{B}$ is δ-complementary if (i) $\lambda_i \geq \delta$ for i s.t. $s_i \notin \mathcal{B}$, (ii) $\lambda_j \geq \delta$ for j s.t. $u_j \notin \mathcal{B}$, and (iii) $(\bar{A}'\lambda)_j \geq v_j + \delta$ for j s.t. $y_j \notin \mathcal{B}$.

PROPOSITION 4.2.6. *Let v be the true values let δ be such that all the bases are δ-complementary. Then, there exists a constant $c \geq 1$ such that, for any estimation $\tilde{v}$ that satisfies*

$$||v - \tilde{v}||_\infty \leq \frac{\delta}{c(d+2)},$$

BUDGETRATIO*, with the LP solved each period with $\tilde{v}$ replacing v, produces bounded regret in the sense of Theorem 4.2.3. Furthermore $c \leq \max_{\mathcal{B}} ||\mathcal{B}^{-1}||_\infty$.*

PROOF. Fix a basis $\mathcal{B}$ associated to some centroid $\mathcal{K}$ under values v. Let λ be the dual variables associated to $(\mathcal{B}, v)$. Recall that, by Lemma 4.4.1, we know that $\lambda \geq 0$ and $\bar{A}'\lambda \geq v$. We first prove that $\mathcal{B}$ is also an optimal basis associated to $\mathcal{K}$ under values $\tilde{v}$. In virtue of Lemma 4.4.1, it suffices to show that the dual variables $\tilde{\lambda}$ associated to $(\mathcal{B}, \tilde{v})$ satisfy (i) $\tilde{\lambda} \geq 0$ and (ii) $\bar{A}'\tilde{\lambda} \geq \tilde{v}$.

Using complementary slackness, the $d+n$ dual variables are the solution to the following $d+n$ linear equations:

$$\begin{aligned} ([A'|I^n]\tilde{\lambda})_j &= \tilde{v}_j && j \text{ s.t. } y_j \in \mathcal{B} \\ \tilde{\lambda}_j &= 0 && j \text{ s.t. } u_j \in \mathcal{B} \\ \tilde{\lambda}_i &= 0 && i \text{ s.t. } s_i \in \mathcal{B}. \end{aligned}$$

Observe that the left-hand side of the equations is invariant (depends on $A, \mathcal{B}$ only). We thus can write $H\tilde{\lambda} = \tilde{h}$ and note that the same system is satisfied with λ, i.e., $H\lambda = h$, with $(h - \tilde{h})_j = (v_j - \tilde{v}_j)\mathbb{1}_{\{y_j \in \mathcal{B}\}}$. Finally, by the Lipschitz continuity of linear systems [84],

we have that $||\lambda - \tilde{\lambda}|| \leq c||h - \tilde{h}||$. It remains to conclude the properties (i) $\tilde{\lambda} \geq 0$ and (ii) $\bar{A}'\tilde{\lambda} \geq \tilde{v}$.

For (i), from the system $H\tilde{\lambda} = h$ it follows that we only need to study the case $u_j \notin \mathcal{B}$ or $s_i \notin \mathcal{B}$. In this case, by δ-complementarity, we are guaranteed $\tilde{\lambda} \geq \delta - c||h - \tilde{h}||_\infty \geq 0$, as desired. For (ii), by the same reasoning, we need to study only the dual constraints $([A'|I^n]\tilde{\lambda})_j \geq \tilde{v}_j$ for $y_j \notin \mathcal{B}$. Let us denote η the j-th row of $[A'|I^n]$. By δ-complementarity, we have

$$([A'|I^n]\tilde{\lambda})_j = \eta\tilde{\lambda} \geq \delta + v_j + \eta(\tilde{\lambda} - \lambda) \geq v_j + \delta - (||A_j||_1 + 1)c||v - \tilde{v}||_\infty \geq 0.$$

We have proved that all centroids and their bases remain the same under $\tilde{v}$. We conclude now the proof by observing that all of our constructions depend on identifying the centroids only and the values v are used nowhere else, thus none of our proofs change. □

Example 4.7.4 (One Dimension). We show that, in the case $d = 1$, δ-complementarity reduces to the following natural condition: $v_j \geq \delta$ for all $j \in [n]$ and $|v_j - v_{j'}| \geq \delta$ for all $j \neq j'$. In other words, δ-complementarity captures exactly the condition needed to distinguish among requests.

The primal and dual problems are as follows, where λ_0 denotes the resource multiplier and λ_j the demand multipliers for $j \in [n]$:

$$\begin{array}{lrl} \max & v'y & \\ \text{s.t.} & \sum_j y_j + s & = R \\ & y + u & = D \\ & y_j, u_j, s & \geq 0 \end{array} \qquad \begin{array}{lrll} \min & R\lambda_0 + \sum_j D_j\lambda_j & & \\ \text{s.t.} & \lambda_0 + \lambda_j & \geq v_j & \forall j \in [n] \\ & \lambda_0, \lambda_j & \geq 0. & \end{array}$$

Consider a basis of the form $\mathcal{B} = \{y_j : j = 1, \ldots, k+1\} \cup \{u_j : j = k+1, \ldots, n\}$, i.e., all requests $j \in [k]$ are completely served, $j = k+1$ is partially served, and all other requests are completely rejected. By inspection, the dual variables associated to this basis are as follows: $\lambda_0 = v_{k+1}$, $\lambda_j = v_j - v_{k+1}$ for $j \in [k]$ and $\lambda_j = 0$ for $j > k$.

Finally, we can verify conditions (i)-(iii) of Definition 4.7.3. For (i), we need $v_{k+1} \geq \delta$. For (ii), we need $v_j - v_{k+1} \geq \delta$ for $j \in [k]$. Finally, for (iii) $v_{k+1} \geq v_j + \delta$ for $j = k+2, \ldots, n$. This shows that indeed the proposed condition guarantees δ-complementarity.

Example 4.7.5 (Necessity of Separation). Consider the case $d = 1$ with two patient requests types. The optimal policy waits till the end and chooses just one type, the one with maximum reward, and allocates all the inventory to that type. Assume in this context that rewards are not observed, but all we have is access to the estimation $\tilde{v}$. If $\tilde{v}_1 < \tilde{v}_2$, in the absence of separation, it can be that $v_1 > v_2$ or $v_1 < v_2$, so that the optimal policy cannot obtain constant regret even in this simple case.

Learning Setting (Contextual Bandits with Knapsacks). Our robustness result directly implies $O(\log T)$ regret for the setting where v needs to be learned online. In this problem, the controller needs to make decisions while learning the correct values v. Precisely, we assume that type-j draws a reward $V_j \sim F_j$, where F_j is some unknown distribution, and $v_j = \mathbb{E}[V_j]$ represents the true expectation. At each time, the controller observes the context $j \in [n]$ and, if he serves the request, he observes a realization of V_j, thus he can estimate v with the empirical averages $\hat{v}_j^t$. We get the following result.

Corollary 4.7.6 (Learning Reward Distribution). *Assume the separation condition in Proposition 4.2.6. Further, assume that the distributions F_j are sub-gaussian, but otherwise unknown and arbitrary. Then, we can obtain $O(\log T)$ regret in the case that the expectations v are unknown.*

Proof. Fix the constant $\epsilon = \frac{\delta}{c(d+2)}$. If we have an initial exploration phase of length $c' \log T$, for some $c' = c'(\epsilon) > 0$, where all requests are accepted, then the empirical $\hat{v}^t$ is s.t. $||v - \hat{v}^t|| \leq \epsilon$ for all $t \geq c \log T$ with probability $\frac{1}{T}$. After time $c \log T$ we run BudgetRatio and obtain constant regret in the remaining periods in virtue of Proposition 4.2.6. □

4.8 Interpretation of BudgetRatio as a max-bid-price control

Bid-price heuristics are popular due to their intuitive interpretation. There are several examples where these policies achieve good provable performance; see [102] for asymptotic results and [28] for a broader overview of bid-prices in the context of revenue management.

In a setting with multiple resources, a bid-price policy is described as follows: at time t compute a vector $\lambda^t \in \mathbb{R}^d_{\geq 0}$ of resource prices, then reject an arrival type j iff its reward is below the combined price of requested resources, i.e., reject j iff $v_j < A_j'\lambda^t$.

We show below that an enhanced version of bid-price achieves constant regret. The main insight is that *we need to consider the maximum of several prices.*

Recall that we say a basis $\mathcal{B}$ is associated to a centroid $\mathcal{K}$ if $\mathcal{B}$ is optimal at the centroid budget (Definition 4.4.9) and the dual variables associated to a basis $\mathcal{B}$ are $\lambda = (\mathcal{B}^{-1})'\bar{v}_{\mathcal{B}}$ (Definition 4.7.3). To define bid-prices, we identify the centroid $\mathcal{K}$ such that $R^t \in \mathcal{N}_{\mathcal{K}}(p)$, then we define

$$\Lambda^t(R^t) \coloneqq \{\lambda : \lambda \text{ are duals associated to } \mathcal{B} \text{ for some basis } \mathcal{B} \text{ associated to } \mathcal{K}\}.$$

At each time t, we compute this set of prices $\Lambda^t(R^t) \subseteq \mathbb{R}^d_{\geq 0}$ and use the following decision rule:

> Accept a type-j request if $v_j \geq \max_{\lambda \in \Lambda^t(R^t)} A_j'\lambda$ and $B^t \leq A_j$. Reject the request otherwise.

In Fig. 4.11 we show the action regions and their interpretation as max-bid-price.

Proposition 4.8.1. *Assume that all the bases are δ-complementary (Definition 4.7.3) for some $\delta > 0$. Then,* BudgetRatio *is equivalent to max-bid-price.*

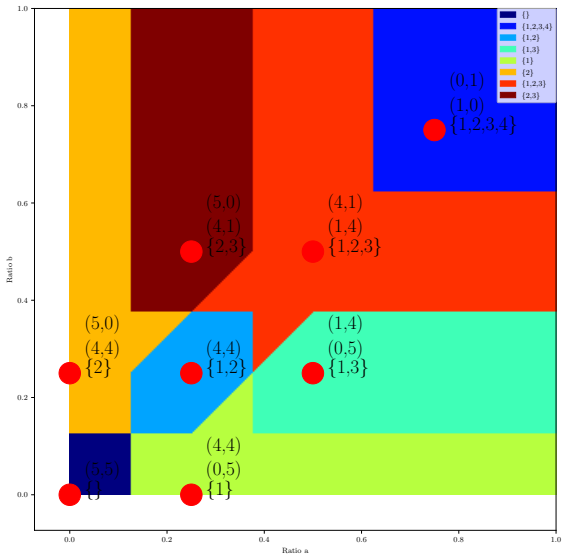


Figure 4.11: Action regions with the associated bid-prices. Every centroid $\mathcal{K} \neq \varnothing, \{1,2\}$ has two vectors of shadow prices with components (λ_a, λ_b). For example, if $R^t \in \mathcal{N}_{\{1\}}$ (bottom region), then the decision rule is to reject if $v_j < (4,4)'A_j$ or $v_j < (0,5)'A_j$.

PROOF. We prove that the decision rule with this set of prices is the same as BUDGETRATIO. We divide the analysis into $j \in \mathcal{K}$ (acceptance) and $j \notin \mathcal{K}$ (rejection). The case $j \in \mathcal{K}$ follows easily since, at the centroid budget $r_{\mathcal{K}}$, $y_j = p_j > 0$ for all bases $\mathcal{B}$ associated to $\mathcal{K}$, so by complementary slackness $v_j \geq A_j'\lambda$ for any such associated dual variable.

We are left to prove the case $j \notin \mathcal{K}$. First, we claim that some basis $\mathcal{B}$ is such that $y_j \notin \mathcal{B}$. Assuming this claim, δ-complementarity yields $v_j < A_j'\lambda - \delta$, so that j fails the test with the λ associated to this basis. All that remains is proving the claim.

Assume by contradiction that $j \notin \mathcal{K}$ is such that $y_j \in \mathcal{B}$ for all bases associated to $\mathcal{K}$. Since $\mathcal{K} \neq \varnothing$, we have $r_{\mathcal{K}} \neq 0$. Recall that $\mathcal{N}_{\mathcal{K}}(D) = \cup_{\mathcal{B}} \mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$, i.e., $\mathcal{N}_{\mathcal{K}}(D)$ is divided into finitely many regions. Therefore, we can take a direction η such that $r_{\mathcal{K}} \pm \eta$ is in the interior of these regions (if $r_{\mathcal{K}} + \eta$ lies in a boundary, a perturbation will be in the interior because there are finitely many boundaries).

In conclusion, for some bases $\mathcal{B}, \mathcal{B}'$ we have $r_{\mathcal{K}} + \eta \in \text{interior}(\mathcal{N}_{\mathcal{K}}(\mathcal{B}, D))$ and $r_{\mathcal{K}} - \eta \in$

interior($\mathcal{N}_{\mathcal{K}}(\mathcal{B}', D)$). Observe that, in the interior of a region, all the basic variables are strictly positive. Since $y_j \in \mathcal{B}, \mathcal{B}'$, it must be that the solutions y^+ and y^- at ratios $r_{\mathcal{K}} + \eta$ and $r_{\mathcal{K}} - \eta$ have $y_j^+, y_j^- > 0$. This is a contradiction with the fact $y_j = 0$ at the centroid budget $r_{\mathcal{K}}$. □

The way we construct Λ^t may suggest that we need to pre-compute all bases associated to all centroids, but this is not the case. Indeed, it suffices to pre-compute the duals for the initial centroid (i.e., such that $R^0 \in \mathcal{N}_{\mathcal{K}}(p)$ and) and its immediate neighbors; by our analysis, up to $\tau_{region}^{\epsilon',\mathcal{K}} \wedge \tau_{cone}^{\epsilon',\mathcal{B}}$ the process R^t remains in these sets.

4.9 Discussion

We developed a geometric understanding of the structure of a large family of linear programs. This understanding allows us to prove optimality guarantees for a family of resource allocation problems, which we call RAN and includes, in particular, online packing and assembly.

Our fundamental definition is that of centroids, which correspond to subsets of requests that should be fully served. For each demand D, a centroid $\mathcal{K}$ has associated a region $\mathcal{N}_{\mathcal{K}}(D)$. The key to the analysis is to understand how the process of ratios R^t evolves relative to these regions $\mathcal{N}_{\mathcal{K}}(D)$ and show that it is attracted to the correct region (that used by the offline controller).

Our geometric understanding allows us not only to prove constant regret, but also to obtain the following robustness results:

1. Queues: BudgetRatio is resilient to the dynamics of the queues. Indeed, they can

be ignored and still have a near optimal performance, even though the optimal policy may interact in a complicated way with the queues.

2. Cost minimization: we showed that a simple tuning of BudgetRatio achieves simultaneously near optimality with respect to reward maximization and cost minimization.

3. Demand estimation: we showed that, if the arrival probabilities p are estimated within a constant error, BudgetRatio still achieves constant regret.

4. Reward estimation: we showed that, if the values v, corresponding to the reward of serving requests, are estimated within a constant error, BudgetRatio achieves constant regret. Crucially, using the notion of centroids, we give a straightforward generalization to multiple dimensions that was not possible with previous techniques.

CHAPTER 5

BELLMAN INEQUALITIES

5.1 Introduction

Online decision-making under uncertainty is widely studied across a variety of fields, including operations research, control, and computer science. A canonical framework for such problems is that of Markov decision processes (MDP), with associated use of stochastic dynamic programming for designing policies. In complex settings, however, such approaches suffer from the known curse-of-dimensionality; moreover, they also fail to provide insights into structural properties of the problem: the performance of heuristics, dependence on distributional information, etc.

The above challenges have inspired an alternate approach to MDPs based on the use of *benchmarks* (such as offline/prophet and fluid relaxations). These are proxies for the value function that provide bounds for the optimal policy, and guide the design of heuristics. The performance of any policy can be quantified by its additive loss, or *regret*, relative to the benchmark; this consequently also bounds the additive optimality gap, i.e., performance against the optimal policy.

In this work, we develop *new benchmark-driven policies for online resource allocation problems* – settings where a finite set of resources are dynamically allocated to arriving requests, with associated constraints and rewards/costs. Our baseline setting is the online stochastic knapsack problem (henceforth `OnlineKnapsack`): a controller has an initial resource budget, and requests arrive sequentially over a horizon. Each arriving request has a random type corresponding to a resource requirement-reward pair. Item types are generated from some known stochastic process, and are revealed upon arrival; the controller must then

decide whether to accept/reject each request, in order to maximize collected rewards while satisfying budget constraints.

We consider three variants of this basic setting: (1) online probing, (2) online contextual bandits with knapsacks, and (3) dynamic pricing. The formal models for these settings are presented in Section 5.2; each setting augments the baseline `OnlineKnapsack` with additional constraints/controls, and the problems are widely-studied in their own right. Instead of solving each problem in an ad-hoc manner, however, our policies are derived from a single underlying framework. In particular, our results can be summarized as follows:

Meta-theorem *Given an online resource allocation problem, we identify an offline benchmark and specify a simple online policy – based on solving a tractable optimization problem in each period – that achieves a constant regret compared to this benchmark, (and hence, bounded additive gap versus the optimal policy).*

In more detail, our approach is based on adaptively constructing a benchmark that has privileged (but not necessarily full) information about future randomness. This can be viewed as an online primal-dual method, wherein our benchmark serves as a dual solution, which we then use to construct a feasible online policy. The centrepiece of our approach are the *Bellman Inequalities*, which decompose the regret of an online policy into two distinct terms: The first, which we call the *Bellman Loss*, arises from computational considerations, specifically, from requiring that the benchmark is tractable (instead of a dynamic program, which is usually intractable); The second, which we call the *Information Loss*, accounts for the unpredictability of the sample path, i.e., the random trajectory. Our policies trade off these two losses to produce strong performance guarantees.

Our framework is flexible in allowing benchmarks with varying degrees of coarseness. To understand why this is important, consider two common benchmarks for the dynamic

pricing problem, wherein each request has a random valuation for an item and we must post prices for our inventory. One common benchmark, known as the *offline* or *prophet*, is based on a controller that has *full information* of all the randomness; it is easy to show that no online policy can get better than $\Omega(T)$ regret against this benchmark where T is the decision, see Example 5.3.2. An alternate benchmark, known as the *ex ante* or *fluid*, corresponds to replacing all random quantities with their expected values; again here, no online policy can get better than $\Omega(\sqrt{T})$ regret against this benchmark [10]. Our approach allows us to identify benchmarks which have $O(1)$ regret for all our settings.

Prophet and fluid benchmarks are also widely used in adversarial models of online allocation, leading to algorithms with worst-case guarantees. In contrast, our work focuses on stochastic inputs, and consequently obtains much stronger guarantees. In particular, our guarantees are parametric and depend explicitly on the distribution and the *primitives*, i.e., constant parameters defining the instance.

5.2 Setting and Overview of Results

5.2.1 Problem Settings and Results

We illustrate our framework by developing low-regret algorithms for the following problems:

Online Stochastic Knapsack. This serves as a baseline for our other problems. The controller has an initial resource budget B, and items arrive sequentially over T periods. Each item has a random type j which corresponds to a *known* resource requirement (or 'weight') w_j and a *random* reward R_j. In period $t = T, T-1, \ldots, 1$ (where t denotes the *periods to go*), we assume the arriving type is drawn from a finite set $[n]$ from some known

distribution $\mathbf{p} = (p_1, \ldots, p_n)$. At the start of each period, the controller observes the type of the arriving item, and must decide to accept or reject the item. The expected reward from selecting a type-j item is $\mathbb{E}[R_j]$ so that maximizing the expected collected rewards is equivalent (in terms of optimal actions) to a setting where type-j items bring a deterministic reward $r_j = \mathbb{E}[R_j]$.

Online Probing. The controller – who knows the type of an arriving request, but not the realization of the reward R_j – now has the option to probe each request to observe the realization, and then accept/reject the item based on the revealed reward. The controller can also choose to accept the item without probing; in this case, the decision must be based on the distribution of R_j only. In addition to the resource budget B, the controller has an additional a probing budget B_p that limits the number of arrivals the controller can probe. This introduces a trade-off between depleting the resource budget B and the probing budget B_p. We assume in this setting that R_j has finite support $\{r_{jk}\}_{k\in[m]}$ of size m, and define $q_{jk} \coloneqq \mathbb{P}[R_j = r_{jk}]$ for $k \in [m]$.

Dynamic Pricing. The controller has an initial inventory $B \in \mathbb{N}^d$ for d different resources. The arrival in each period t corresponds to a customer who wants a specific subset of resources and has some private valuation R^t for the allocation. Specifically, there are n different types of customers with consumption encoded in a matrix $A \in \{0,1\}^{d\times n}$. Upon arrival, we observe the type $j \in [n]$ and then post a price (fare) f from the finite set $\{f_{j1}, \ldots, f_{jm}\}$. After the price is posted, the customer draws $R^t \sim F_j$ and purchases iff $R^t > f$. The valuation functions $(F_j : j \in [n])$ are known, but otherwise arbitrary.

Online Contextual Bandits with Knapsacks. We return to the `OnlineKnapsack` setting, with horizon T, capacity B, and finite types $[n]$ where items of type $j \in [n]$ have weight w_j and random reward R_j. Now however the controller is unaware of the distribution of R_j, and must learn $\mathbb{E}[R_j]$ from observations. At every period t, the controller observes the type

j (i.e., context) of the arriving item and must decide to accept or reject the item based on the observations of rewards up to time t. We consider two feedback models: *full feedback* where the controller observes R_j regardless of whether the item is accepted or rejected, and *censored feedback* where the controller only observes rewards of accepted items. For this setting, we also assume that the rewards R_j have sub-Gaussian tails [27, Section 2.3].

Benchmarks and guarantees. The framework we develop, which we denote as RABBI (Resolve and Act Based on the Bellman Inequalities, see Section 5.3.2) is based on comparing two 'controllers': OFFLINE, who takes the optimal action given access to future information, and a non-anticipative controller ONLINE who tries to follow OFFLINE.

Both ONLINE and OFFLINE start in an initial state S^T, i.e., the same initial level of budget. We denote v^{off} as the expected total reward collected by OFFLINE acting optimally given its information structure; this is given by a Bellman equation that takes into consideration the extra information. In contrast, ONLINE uses some non-anticipative policy π that maps the current state to an action, resulting in a total expected reward v_π^{on}. We denote by π_R the online policy produced by our RABBI framework. The expected regret relative to an offline benchmark is defined as

$$\mathbb{E}[\text{REG}] := v^{\text{off}} - v_{\pi_R}^{\text{on}} \geq \max_\pi v_\pi^{\text{on}} - v_{\pi_R}^{\text{on}}$$

The last inequality, which follows from the fact that $v_\pi^{\text{on}} \leq v^{\text{off}}$ for any pair of benchmark and online policies, emphasizes that the regret is a bound on the *additive gap w.r.t. the best online policy*.

For all the above problems, we use the RABBI framework to identify a benchmark (fully specified in each section) and obtain the following guarantees. For the `OnlineKnapsack`, our result is a re-statement of a theorem proved in [10] and Chapter 3; we recover it here to build intuition for our general framework.

Theorem 5.2.1 (Baseline Setting [10] and Chapter 5). *For known reward distributions with finite mean,* RABBI *obtains regret that depends only on the primitives* $(n, \mathbf{p}, \mathbf{r}, \mathbf{w})$*, but is independent of the horizon length* T *and resource budget* B*.*

In the case where rewards are deterministic, the benchmark used in Theorem 5.2.1 for the baseline `OnlineKnapsack` is the natural full-information prophet. In the remaining settings (pricing, probing, and bandits), however, the full-information benchmark is too loose to get constant regret policies, i.e., no online policy can get close to that benchmark, see Example 5.3.2. This is where our framework helps in guiding the choice of the right benchmark. In particular, in the remaining settings, we get the following results.

Theorem 5.2.2 (Probing). *For reward distributions with finite support of size* m*,* RABBI *obtains regret that depends only on* $(n, m, \mathbf{q}, \mathbf{p}, \mathbf{r})$*, but is independent of the horizon length* T*, resource budget* B *and probing budget* B_p*.*

Theorem 5.2.3 (Dynamic Pricing). *For any reward distributions* $(F_j : j \in [n])$ *and prices* $\mathbf{f}$*,* RABBI *obtains regret that depends only on* $(A, \mathbf{f}, F_1, \ldots, F_n)$*, but is independent of the horizon length* T *and initial budget levels* $B \in \mathbb{N}^d$*.*

We define a separation parameter $\delta = \min_{j \neq j'} |\mathbb{E}[R_j]/w_j - \mathbb{E}[R_{j'}]/w_{j'}|$ for the bandits setting; this is only for our bounds, and it is not known to the algorithm.

Theorem 5.2.4 (Contextual Bandits with Knapsacks). *Assuming the reward distributions are sub-Gaussian, in the full feedback setting,* RABBI *obtains regret that depends only on the primitives* $(n, \mathbf{p}, \mathbf{r}, \mathbf{w}, \delta)$ *and is independent of the horizon length* T *and knapsack capacity* B*.*

The last result can also be used as a black-box for the censored feedback setting to get an $O(\log T)$ regret guarantee. The statement of the result (Corollary 5.6.4) requires some building blocks that we introduce in Section 5.6.5.

We provide explicit regret bounds for each of the above results in terms of the problem primitives. Note however that the algorithms in Theorems 5.2.2 to 5.2.4 enjoy *constant-regret* guarantees – in particular, the algorithms are near optimal (up to additive constants) in any regime where T, B are $\omega(1)$, keeping other problem primitives fixed.

5.2.2 Overview of our Framework

We develop our framework in the full generality of MDPs in Section 5.3. To give an overview and gain insight into the general version, we use the baseline `OnlineKnapsack` as a warm-up, using our framework to recover the results in [10]. A schema for the framework is provided in Fig. 5.1.

In the `OnlineKnapsack` problem, at any time-to-go t, let $Z_j(t) \in \mathbb{N}$ denote the (random) number of type-j arrivals in the remaining t periods. Recall rewards of type-j arrivals have expected value $r_j := \mathbb{E}[R_j]$. We define Offline to be a controller that knows $Z(t)$ for all t in advance. The total reward collected by Offline can be written as an integer programming problem

$$V(t, b|Z(t)) = \max_{\mathbf{x_a} \in \mathbb{N}^n} \{\mathbf{r}'\mathbf{x} : \mathbf{w}'\mathbf{x_a} \leq b, \mathbf{x_a} \leq Z(t)\} = \max_{\mathbf{x_a}, \mathbf{x_r} \in \mathbb{N}^n} \{\mathbf{r}'\mathbf{x_a} : \mathbf{w}'\mathbf{x_a} \leq b, \mathbf{x_a} + \mathbf{x_r} = Z(t)\}. \tag{5.1}$$

We use the notation $|Z(t)$ to emphasise that, given the knowledge of $Z(t)$, Offline computes the value for any (t, b). For every j, the variables $x_{\mathtt{a},j}, x_{\mathtt{r},j}$ represent *action summaries*: the number of type-j arrivals accepted and rejected, respectively. This function V is Offline's value (see Fig. 5.1).

Offline's value and policy can also be represented via Bellman equations. Specifically, at time-to-go t, assuming the controller has budget b at the start of the period, and the

arriving type is ξ, the value function and optimal actions are given by the Bellman equation

$$V(t,b|Z(t)) = \max\Big\{\Big[r_{\xi^t} + V\Big(t-1, b-w_{\xi^t}|Z(t-1)\Big)\Big]\mathbb{1}_{\{w_{\xi^t}\le b\}}, V\Big(t-1,b|Z(t-1)\Big)\Big\}, \quad \forall t,b,\xi^t.$$

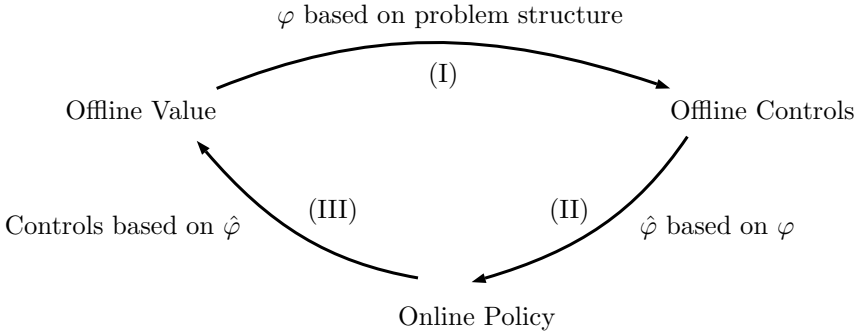


Figure 5.1: In our framework, we first define OFFLINE's value function by specifying its access to future information. Next, we identify a relaxation φ for OFFLINE's value under this same information structure (step I), such that φ approximates OFFLINE's policy in a tractable way. Next, we introduce a non-anticipative proxy $\hat{\varphi}$ which serves as an estimate for φ, and use it to design online controls (step II). Finally, the resulting online policy is evaluated against OFFLINE's value (step III).

Next consider the linear programming relaxation for $V(t,b)$

$$\varphi(t,b|Z(t)) \coloneqq \max_{\mathbf{x_a},\mathbf{x_r}\ge 0}\{\mathbf{r}'\mathbf{x_a} : \mathbf{w}'\mathbf{x_a} \le b, \mathbf{x_a}+\mathbf{x_r} = Z(t)\},$$

It is clear that φ approximates V up to an integrality gap. However, φ does not obey a Bellman equation. To circumvent this, we introduce the notion of *Bellman Inequalities*, wherein we require that φ satisfies Bellman-like conditions for 'most' sample paths. Formally, for some random variables L_B, we want φ to satisfy

$$\varphi(t,b|Z(t)) \le \max\Big\{\Big[r_{\xi^t} + \varphi\Big(t-1, b-w_{\xi^t}|Z(t-1)\Big)\Big]\mathbb{1}_{\{w_{\xi^t}\le b\}}, \varphi\Big(t-1,b|Z(t-1)\Big)\Big\} + L_B(t,b).$$

Note that, if $\mathbb{E}[L_B(t,b)]$ is small, with expectation taken over $Z(t)$, then φ 'almost' satisfies the Bellman equations. We henceforth refer to φ as a *relaxed value* for V and L_B the Bellman Loss. The term L_B is related to the Bellman error used in ADP, except that we require only an upper bound on it, since the exact Bellman error is also intractable.

The main advantage of this φ is its tractability. On the other hand, using φ as a benchmark to design an online policy can induce an error because it is not OFFLINE's value function; this is quantified by the *Bellman Loss*, as it arises due to violations of the Bellman equations.

Establishing that actions derived from φ are nearly optimal for OFFLINE accomplishes step (I) of our framework, as illustrated in Fig. 5.1. For step (II), we want to design an online policy that tries to guess OFFLINE's actions by estimating φ based on the current information. A natural estimate is obtained by taking expectations over future randomness to get

$$\hat{\varphi}(t, b) \coloneqq \max_{\mathbf{y}_{\mathtt{a}}, \mathbf{y}_{\mathtt{r}} \geq 0} \{\mathbf{r}' \mathbf{y}_{\mathtt{a}} : \mathbf{w}' \mathbf{y}_{\mathtt{a}} \leq b, \mathbf{y}_{\mathtt{a}} + \mathbf{y}_{\mathtt{r}} = \mathbb{E}[Z(t)]\}.$$

We want to emphasise that $\hat{\varphi}$ *does not approximate* V *or* φ up to a constant error (see Chapter 3), hence $\hat{\varphi}$ should not be interpreted a as a value function approximation; however, $\hat{\varphi}$ can be used as a predictor for the action taken by OFFLINE. Specifically, at time t with current budget b, in RABBI we first compute $\hat{\varphi}(t, b)$ and then interpret the solution $\mathbf{y}$ as scores for each action (here, accept/reject). We show that taking the action with the highest score (i.e., action $\operatorname{argmax}_{u \in \{\mathtt{a}, \mathtt{r}\}} \{y_{\xi^t, u}\}$) guarantees that that ONLINE and OFFLINE play the same action with high probability. Whenever OFFLINE and ONLINE play different actions, then we incur a loss; we refer to this as the *information loss*, as it quantifies how having less information impacts ONLINE's actions. This process of using $\hat{\varphi}$ to derive actions is represented as step III in Fig. 5.1.

The above three-step process forms a template for our general framework in Section 5.3.2. For all the four problems introduced in Section 5.2.1 (Theorems 5.2.1 to 5.2.4), our approach produces similar policies, wherein we choose an OFFLINE benchmark, identify a relaxed value φ which takes the form of an optimization problem, and create an online policy based on an estimate $\hat{\varphi}$. Due to this structure, we refer to our framework as RABBI (*Re-solve and Act*

Based on Bellman Inequalities).

In all the problems we consider, *we decompose the regret into two distinct sources*: the Bellman Loss arising from using the relaxation φ instead of OFFLINE's value-function, and the Information Loss arising from estimating φ using $\hat{\varphi}$. The former is a loss arising from the need for computational tractability, while the latter is inherent to the randomness in the problem.

Challenges for a general framework. Constant regret policies for the baseline `OnlineKnapsack` were obtained in [10], and generalized to multiple dimensions (i.e., the network revenue management/online packing problem) in Chapter 3; moreover, these builds on a long line of work [94, 78] demonstrating constant regret guarantees in restricted settings. In particular, [10] propose a simple online algorithm based on re-solving a fluid LP, that attains constant regret.

The techniques developed in the above papers are tailored to the baseline `OnlineKnapsack`, and have two fundamental shortcomings that prevent them from addressing general problems:

- Based on full information: previous techniques rely on an achievable full information benchmark, but this excludes most MDPs. Indeed, for probing/pricing/learning settings, *no algorithm can have constant regret compared to the full information benchmark* (see Example 5.3.2).
- Based on simple characterisations: previous papers exploit the explicit form of value functions, e.g., the optimization problem in Eq. (5.1). Most MDPs do not have such a closed form solution, but are recursive by nature, making previous work inapplicable in general settings.

In this paper, we offer a framework whose *generality* allows us to obtain bounded regret algorithms (and guarantees) for several canonical resource allocation problems. We resolve the previous two shortcomings in an structured way, via a flexible notion of information, while eliciting two important notions that guide the design and analysis of algorithms: we decompose the regret into Information Loss (capturing the unpredictability of the inputs) and Bellman Loss (due to our limited computational power).

5.2.3 Related Work

Our work uses benchmarks to bound the performance of the optimal online algorithm, and also for developing good heuristics. This has commonalities with two closely related approaches:

Prophet Inequalities: A well-known framework to obtain performance guarantees is to compare against a full information agent, or "prophet". This line of work focuses on competitive-ratio bounds, see [80, 55, 47] for overviews of the area. In particular, [47] obtains a multiplicative guarantee for dynamic posted pricing with a single item under worst case distribution. In contrast, we obtain an additive guarantee for multiple items in a parametric setting.

Information Relaxations. MDP information-relaxations, in particular, the framework developed in [29, 18], are a fairly general way to create benchmarks, based on endowing Offline with additional information, while forcing him to 'pay a penalty' for using this information. [29, 18] use such relaxations in a *dual-fitting* approach, to construct performance bounds for standard greedy algorithms for a variety of problems. In contrast, our framework is similar to a *primal-dual* approach: we do not need to identify appropriate penalties, but rather, adaptively construct our relaxations, and derive controls directly from them. We

compare the two approaches in more detail in Appendix D.3.

We test our framework on `OnlineKnapsack` variants which are widely studied in the literature.

Probing. This corresponds to a family of problems where the controller has the option to "pay for information", leading to a trade-off between making a bad choice and obtaining better information. Approximation algorithms have been developed for *offline* probing problems, both under constraints on the probed items [69], as well as with costs for probing [100], with applications to Bayesian auction design and in kidney-exchange. Another line of work pursues tractable *non-adaptive* algorithms for this problem to achieve (multiplicative) constant bounds [70]. In terms of *online adaptive* algorithms, [41] introduces an algorithm that has bounded competitive ratio (hence linear regret) in an adversarial setting.

Dynamic posted pricing. This is a canonical problem in operations management, with a vast literature; we refer the reader to [103] for an overview. Much of this literature focuses on asymptotically optimal policies in regimes where the inventory B and/or the horizon T grow large. In the regime where both B and T are scaled together by a factor of k, there are known algorithms with regret that scales as $O(\sqrt{k})$ or $O(\log(k))$, depending on assumptions on the primitives (e.g., smoothness of the demand with price) [77]. There is also vast literature on pricing when the demand function is not known and has to be learned [40]. We assume that the demand distribution is known and prices are chosen from a discrete price menu. We make no further assumptions on the primitives. There has also been work on worst-case guarantees, see [13] for an overview, where they prove $O((B \log T)^{2/3})$ regret with T periods and inventory B for adversarial input, as opposed to our $O(1)$ guarantee with stochastic input.

Knapsack with learning. Multi-armed bandit problems have been well studied, we refer

to [34, 33] for an overview. Bandit problems with combinatorial constraints on the arms that can be pulled are known as Bandits With Knapsacks [16]. The generalization wherein arms arrive online is known as Contextual Bandits With Knapsacks [17, 5]. Results in this bandit literature study worst-case distributions. We, in contrast, pursue parametric regret bounds, namely, bounds that explicitly depend on the (unknown) discrete distribution. Closest to our work is [109], where an algorithm based on UCB (upper confidence bound) is developed and shown to achieve $O(\sqrt{T})$ regret. The algorithm we develop strengthens their result to a logarithmic regret.

Online Packing. There is a long line of work related to what we call baseline `OnlineKnapsack`. This problem and its generalizations to multiple dimensions is also known as Network Revenue Management. An influential work in this line is [78], where they give a policy with constant expected regret under a certain non-degeneracy condition. We note that the benchmark used by [78] is the ex ante (or fluid), hence the condition they require is fundamental, because this benchmark is known to be loose (see Chapter 3). More recently, [37] partially extended the result of [10] for more general packing problems; however their guarantee is only valid for i.i.d. Poisson arrival processes and when the system is scaled linearly, i.e., when B is proportional to T. In contrast, [10] (one dimension) and Chapter 3 (multiple dimensions) prove constant regret with no assumption on the scaling, while providing better theoretical bounds, and outperforming other algorithms in practice.

5.3 Approximate Control Policies via the Bellman Inequalities

In this section, we describe our general framework. Before proceeding, we introduce some notation: Given an optimization problem (P), we denote its optimal value by P. We have an underlying probability space $(\Omega, \Sigma, \mathbb{P})$. For an event $\mathcal{B} \subseteq \Omega$, we denote by $\mathcal{B}^c$ its complement.

We use boldface letters to indicate vector-valued variables (e.g. $\mathbf{p}, \mathbf{w}$, etc.), and capital letters to denote matrices and/or random variables. When using LP formulations with decision variables $\mathbf{x}$, we use interchangeably $x_{ij} = x(i,j)$ to denote the (i,j) component of $\mathbf{x}$.

5.3.1 Offline Benchmarks and Bellman Inequalities

We consider an online decision-making problem with state space $\mathcal{S}$ and action space $\mathcal{U}$, evolving over periods $t = T, T-1, \ldots, 1$; here T denotes the horizon, and t is the time to-go. In any period t, the controller first observes a random input $\xi^t \in \Xi$, following which it must choose an action $u \in \mathcal{U}$. Given system-state $s \in \mathcal{S}$ at the beginning of period t, a random input $\xi \in \Xi$ and action $u \in \mathcal{U}$ result in a reward $\mathcal{R}(s, \xi, u)$, and transition to the next state $\mathcal{T}(s, \xi, u)$. Note that both reward and next state are random variables whose realizations are determined for every u given ξ. This assumption is for ease of exposition only; our results also hold when rewards or transitions are random given ξ, see Appendix D.4 for details.

The feasible actions for state s and input ξ correspond to the set $\{u \in \mathcal{U} : \mathcal{R}(s, \xi, u) > -\infty\}$. We assume that this feasible set is non-empty for all $s \in \mathcal{S}, \xi \in \Xi$, and also, that the maximum reward is bounded, i.e., $\sup_{s \in \mathcal{S}, \xi \in \Xi, u \in \mathcal{U}} \mathcal{R}(s, \xi, u) < \infty$.

The MDP described above induces a natural filtration $\mathcal{F}$, with $\mathcal{F}_t = \sigma(\{\xi^\tau : \tau \geq t\})$; a non-anticipative policy is one which is adapted to $\mathcal{F}_t$. We allow Offline to use a *richer* information filtration $\mathcal{G}$, where $\mathcal{G}_t \supseteq \mathcal{F}_t$. Note that since t denotes the time-to-go, we have $\mathcal{G}_{t-1} \supseteq \mathcal{G}_t$. Henceforth, to keep track of the information structure, we use the notation $f(\cdot|\mathcal{G}_t)$ to clarify that a function f is measurable with respect to the sigma-algebra $\mathcal{G}_t$.

Given any filtration $\mathcal{G}$, Offline is assumed to play the optimal policy adapted to $\mathcal{G}$,

hence OFFLINE's value function is given by the following Bellman equation:

$$V(t, s, \xi^t|\mathcal{G}_t) = \max_{u\in\mathcal{U}}\{\mathcal{R}(s, \xi^t, u) + \mathbb{E}[V(t-1, \mathcal{T}(s, \xi^t, u), \xi^{t-1}|\mathcal{G}_{t-1})|\mathcal{G}_t]\}, \tag{5.2}$$

with the boundary condition $V(0, \cdot) = 0$. We denote the expected value as $v^{\text{off}} \coloneqq \mathbb{E}[V(T, S^T, \xi^T|\mathcal{G}_T)]$. Note that v^{off} is an upper bound on the performance of the optimal non-anticipative policy.

We present a specific class of filtration (generated by augmenting the canonical filtration) that suffice for our applications (see Fig. 5.2 for an illustration of the definition).

Definition 5.3.1 (Canonical augmented filtration). Let $G_\Theta \coloneqq (G_\theta : \theta \in \Theta)$ be a set of random variables. The canonical filtration w.r.t. G_Θ is

$$\mathcal{G}_t = \sigma(\{\xi^l : l \geq t\} \cup G_\Theta) \supseteq \mathcal{F}_t. \tag{5.3}$$

The richest augmented filtration is the *full information* filtration, wherein $\mathcal{G}_t = \mathcal{F}_1$ for all t, i.e., the canonical filtration with $G_\Theta = (\xi^t : t \in [T])$. As $\mathcal{G}_t$ gets coarser, the difference in performance between OFFLINE and ONLINE decreases. Indeed, when $\mathcal{G} = \mathcal{F}$, then Eq. (5.2) reduces to the Bellman equation for the value-function of an optimal non-anticipative policy:

$$V(t, s, \xi^t) = \max_{u\in\mathcal{U}}\{\mathcal{R}(s, \xi^t, u) + \mathbb{E}[V(t-1, \mathcal{T}(s, \xi^t, u), \xi^{t-1})]\}, \quad V(0, \cdot, \cdot) = 0, \tag{5.4}$$

where the expectation is taken with respect to the next period's input ξ^{t-1}. Since ξ^t is included in all the filtrations we consider, we henceforth use the shorthand $V(t, s|\mathcal{G}_t)$ for $V(t, s, \xi^t|\mathcal{G}_t)$.

Example 5.3.2 (Full Information is Too Loose). Consider the dynamic pricing instance with $n = d = 1$ (selling multiple copies of a product to homogeneous customers) with prices $\mathbf{f} = (1, 2)$. The valuation distribution is $\mathbb{P}[R^t = 1 + \varepsilon] = p$ and $\mathbb{P}[R^t = 2 + \varepsilon] = 1 - p$. When $B = T$, the optimal policy is to always post the price that maximizes $f \cdot \mathbb{P}[R^t > f]$.

If we consider $p \geq 1/2$, then the optimal policy (DP) always posts price $f = 1$ and has an expected reward of T. On the other hand, full information extracts the realized valuation, i.e., it posts the exactly $R^t - \varepsilon \in \{1, 2\}$ at each time, hence $v^{\text{off}} = \sum_t \mathbb{E}[R^t - \varepsilon] = T(2-p)$. We conclude that the regret against full information must grow as $\Omega(T)$.

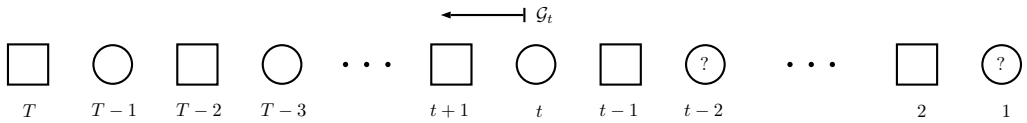


Figure 5.2: Illustration of Definition 5.3.1. In online probing (see Section 5.4), arrivals first reveal their public type, then the controller chooses an action (accept/probe/reject), and then the private type (true reward) is revealed. Squares (resp. circles) represent public (resp. private) information. One possible filtration $\mathcal{G}$ is to reveal all public types, i.e. $G_\Theta = (\xi^\theta : \xi^\theta \text{ is a public type})$. At time t, Offline knows all the information thus far (to the left and including t), plus the future squares. Given this filtration, Offline has to take expectation w.r.t. the future circles.

We are now ready to introduce the notion of relaxed value φ and Bellman Inequalities. Intuitively, φ is "almost" defined by a dynamic-programming recursion; quantitatively, whenever φ does not satisfy the Bellman equation, we incur an additional loss L_B, which we denote the Bellman loss.

Definition 5.3.3 (Bellman Inequalities). The family of r.v. $\{\varphi(t,s)\}_{t,s}$ satisfies the Bellman Inequalities w.r.t. filtration $\mathcal{G}$ and r.v. $\{L_B(t,s)\}_{t,s}$ if $\varphi(t,\cdot)$ and $L_B(t,\cdot)$ are $\mathcal{G}_t$-measurable for all t and the following conditions hold:

1. Initial ordering: $\mathbb{E}[V(T, S^T)|\mathcal{G}_T] \leq \varphi(T, S^T|\mathcal{G}_T)$.
2. Monotonicity: $\forall s \in \mathcal{S}, t \in [T]$,

$$\varphi(t, s|\mathcal{G}_t) \leq \max_{u \in \mathcal{U}} \{\mathcal{R}(s, \xi^t, u) + \mathbb{E}[\varphi(t-1, \mathcal{T}(s, \xi^t, u)|\mathcal{G}_{t-1})|\mathcal{G}_t]\} + L_B(t, s). \tag{5.5}$$

3. Terminal Condition: $\varphi(0, s) = 0 \, \forall \, s \in \mathcal{S}$

We refer to φ and L_B as the *relaxed value* and *Bellman loss* pair with respect to $\mathcal{G}$. We use the notation $|\mathcal{G}_t$ when writing $\varphi(t, s|\mathcal{G}_t)$ to remind the reader that we need the information contained in $\mathcal{G}_t$ to evaluate $\varphi(t, s)$

Given any φ, monotonicity holds trivially with $L_B = \varphi$, but in this case, of course, the performance guarantee will be poor. On the other hand, if we require $L_B = 0$, then φ would be the value function, which is intractable in general. A good choice of φ balances the loss and computational tractability. The crux of our approach is to identify relaxations φ with small Bellman Loss.

A special case is when the Bellman Loss is zero w.h.p., i.e., when we can use an indicator:

Definition 5.3.4 (Exclusion Sets). If we can write the Bellman Loss as $L_B(t, s) = r_\varphi \mathbb{1}_{\mathcal{B}(t,s)}$ for some constant $r_\varphi > 0$ and events $\mathcal{B}(t, s) \subseteq \Omega$, we refer to $\mathcal{B}(t, s)$ as the *exclusion sets*.

If the Bellman Loss can be defined with exclusion sets, then from Definition 5.3.3 (monotonicity) we obtain the condition $\varphi(t, s|\mathcal{G}_t) \leq \max_{u\in\mathcal{U}}\{\mathcal{R}(s, \xi^t, u) + \mathbb{E}[\varphi(t - 1, \mathcal{T}(s, \xi^t, u)|\mathcal{G}_{t-1})|\mathcal{G}_t]\}$, i.e., monotonicity is satisfied for all realizations $\omega \in \Omega$ except for those in the exclusion set $\mathcal{B}(t, s)$.

To build intuition, we specify the Bellman Inequalities for our baseline `OnlineKnapsack`. For this end, we need the following LP lemma.

Lemma 5.3.5. *Consider the standard-form LP* $(P[\mathbf{d}]) : \max\{\mathbf{r}'\mathbf{x} : M\mathbf{x} = \mathbf{d}, \mathbf{x} \geq 0\}$ *, where* $M \in \mathbb{R}^{m\times n}$ *is an arbitrary constraint matrix. If* $\bar{\mathbf{x}}$ *solves* $(P[\mathbf{d}])$ *and* $\bar{x}_j \geq 1$ *for some* j*, then* $P[\mathbf{d}] = r_j + P[\mathbf{d} - M_j]$.

Proof. By assumption, the optimal value of $(P[\mathbf{d}])$ remains unchanged if we add the inequality $x_j \geq 1$. Therefore we have $P[\mathbf{d}] = \max\{\mathbf{r}'(\mathbf{x} + \mathbf{e}_j) : M(\mathbf{x} + \mathbf{e}_j) = \mathbf{d}, \mathbf{x} \geq 0\}$. □

Lemma 5.3.5 allows to divide $P[\mathbf{d}]$ in two summands: the immediate reward r_j and the future reward $P[\mathbf{d} - M_j]$, which has the flavour of dynamic programming we need.

Example 5.3.6 (Bellman Loss For Baseline Setting). For the baseline `OnlineKnapsack`, discussed in Section 5.2.2, we chose the full information filtration $\mathcal{G}_t = \mathcal{F}_1$ for all t so that $\varphi(t, b|\mathcal{G}_t) \coloneqq \max_{\mathbf{x}\geq 0}\{\mathbf{r}'\mathbf{x}_\mathtt{a} : \mathbf{w}'\mathbf{x}_\mathtt{a} \leq b, \mathbf{x}_\mathtt{a} + \mathbf{x}_\mathtt{r} = Z(t)\}$. We define the exclusion sets as

$$\mathcal{B}(t, b) = \{\omega \in \Omega : \nexists \mathbf{x} \text{ solving } \varphi(t, b) \text{ s.t. } x(\mathtt{a}, \xi^t) \geq 1 \text{ or } x(\mathtt{r}, \xi^t) \geq 1\}.$$

By Lemma 5.3.5, outside the exclusion sets $\mathcal{B}(t, b)$, monotonicity holds with zero Bellman Loss, i.e.,

$$\varphi(t, s|\mathcal{G}_t) \leq \max_{u\in\mathcal{U}}\{\mathcal{R}(s, \xi^t, u) + \mathbb{E}[\varphi(t-1, \mathcal{T}(s, \xi^t, u)|\mathcal{G}_{t-1})|\mathcal{G}_t]\} \quad \forall \omega \notin \mathcal{B}(t, s).$$

Let us define the maximum loss as

$$r_\varphi = \max_{t,s,u:\mathcal{R}(s,\xi^t,u)>-\infty}\{\varphi(t, s|\mathcal{G}_t) - (\mathcal{R}(s, \xi^t, u) + \mathbb{E}[\varphi(t-1, \mathcal{T}(s, \xi^t, u)|\mathcal{G}_{t-1})|\mathcal{G}_t])\}$$

In words, the maximum violation of the Bellman equation is bounded by r_φ. For our choice of φ, since the optimal solution is to sort items by "bang for the buck" ratios r_j/w_j, it is easily verified that $r_\varphi \leq \max_{j,i}\{w_i r_j/w_j - r_i\}$, which depends on the primitives only. In conclusion, we can see that Definition 5.3.3 is satisfied with the Bellman Loss $L_B(t, b) = r_\varphi \mathbb{1}_{\mathcal{B}(t,b)}$.

Before proceeding to define online policies based on φ, we require another important definition. Note that though the RHS of Eq. (5.5) is defined in terms of an 'optimal' action, this action need not be unique, and indeed the inequality can be satisfied by multiple actions. For given φ and L_B, we define the set of satisfying actions:

Definition 5.3.7 (Satisfying actions). Given a filtration $\mathcal{G}$ and relaxed value φ, we say that u is *a satisfying action* for state s at time t if

$$\varphi(t, s|\mathcal{G}_t) \leq \mathcal{R}(s, \xi^t, u) + \mathbb{E}[\varphi(t-1, \mathcal{T}(s, \xi^t, u)|\mathcal{G}_{t-1})|\mathcal{G}_t] + L_B(t, s). \tag{5.6}$$

At any time t and state $s \in \mathcal{S}$, any action in $\operatorname{argmax}_{u\in\mathcal{U}}\{\mathcal{R}(s,\xi^t,u) + \mathbb{E}[\varphi(t-1,\mathcal{T}(s,\xi^t,u)|\mathcal{G}_{t-1})|\mathcal{G}_t]\}$ is always a satisfying action (see monotonicity in Definition 5.3.3); moreover, to identify a satisfying action, we must know $\mathcal{G}_t$. We now have the following proposition.

Proposition 5.3.8. *Consider a relaxation φ and Bellman loss L_B that satisfy the Bellman inequalities w.r.t. filtration $\mathcal{G}$. Let $(S^t, t \in [T])$ denote the state trajectory under a policy that, at time t, takes* any *satisfying action $U^t = U^t(S^t|\mathcal{G}_t)$. Then,*

$$\mathbb{E}\big[V(T,S^T|\mathcal{G}_T)\big] - \mathbb{E}\left[\sum_{t=1}^{T}\mathcal{R}(S^t,\xi^t,U^t)\right] \le \mathbb{E}\left[\sum_{t=1}^{T} L_B(t,S^t|\mathcal{G}_t)\right].$$

Proof. From the monotonicity condition in the Bellman inequalities (Definition 5.3.3), and the definition of a satisfying action (Definition 3.3.4), we have, for all time t, that

$$\varphi(t,S^t|\mathcal{G}_t) \le \mathbb{E}[\mathcal{R}(S^t,\xi^t,U^t) + \varphi(t-1,S^{t-1}|\mathcal{G}_{t-1}) + L_B(t,S^t|\mathcal{G}_t)|\mathcal{G}_t].$$

Iterate the above inequality over t to obtain

$$\varphi(T,S^T|\mathcal{G}_T) \le \sum_{t=1}^{T}\mathbb{E}[\mathcal{R}(S^t,\xi^t,U^t) + L_B(t,S^t|\mathcal{G}_t)|\mathcal{G}_t].$$

Finally, the initial ordering condition gives $\mathbb{E}[V(T,S^T)|\mathcal{G}_T] \le \varphi(T,S^T|\mathcal{G}_T)$, hence taking expectations in the above equation yields the result. □

Proposition 5.3.8 shows that a policy that always plays a satisfying action U^t *approximates* the performance of Offline up to an additive gap given by $\mathbb{E}\left[\sum_{t=1}^{T} L_B(t,S^t|\mathcal{G}_t)\right]$, which we henceforth refer to as the *total Bellman loss*. More importantly, the proposition suggests a goal when designing an online policy: Online should try to track Offline by 'guessing' and playing a satisfying action U^t in each period. We next illustrate how Online can generate such guesses, i.e., how Online can identify actions that are likely to be satisfying.

5.3.2 From Relaxations to Online Policies

Suppose we are given an augmented canonical filtration $\mathcal{G}_t = \sigma(\{\xi^l : l \geq t\} \cup G_\Theta)$, and assume that the relaxed value φ can be represented as a function of the random variables $\{\xi^l : l \geq t\} \cup G_\Theta$ as $\varphi(t, s|\mathcal{G}_t) = \varphi(t, s; f_t(\xi^T, \ldots, \xi^t, G_\Theta))$. In particular, we henceforth focus on a special case where φ is expressed as the solution of an optimization problem:

$$\varphi(t, s; f_t(\xi^T, \ldots, \xi^t, G_\Theta)) = \max_{\mathbf{x} \in \mathbb{R}^{\mathcal{U} \times \Xi}} \{h_t(\mathbf{x}; s, f_t(\xi^T, \ldots, \xi^t, G_\Theta)) : g_t(\mathbf{x}; s, f_t(\xi^T, \ldots, \xi^t, G_\Theta)) \leq 0\}. \tag{5.7}$$

The decision variables give *action summaries*: for a given state s and time-to-go t, $x_{u,\xi}$ represents the number of times action u is taken for an input ξ in the remaining periods. We can also interpret $x_{u,\xi}$ as a *score* for action u when the input ξ is presented.

To designing a non-anticipative policy, note that a natural 'projection' of $\varphi(t, s|\mathcal{G}_t)$ on the filtration $\mathcal{F}$ is given via the following optimization problem

$$\hat{\varphi}(t, s|\mathcal{F}_t) = \varphi(t, s; \mathbb{E}[f_t(\xi^T, \ldots, \xi^t, G_\Theta)|\mathcal{F}_t]) = \max_{\mathbf{y} \in \mathbb{R}^{\mathcal{U} \times \Xi}} \{h_t(\mathbf{y}; s, \mathbb{E}[f_t|\mathcal{F}_t]) : g_t(\mathbf{y}; s, \mathbb{E}[f_t|\mathcal{F}_t]) \leq 0\}. \tag{5.8}$$

The solution of this optimization problem gives action summaries (or scores) $\mathbf{y}$; the main idea of the RABBI algorithm is to play the action with the highest score.

RABBI (Re-solve and Act Based on Bellman Inequalities)

Input: Access to functions f_t such that $\varphi(t, s|\mathcal{G}_t) = \varphi(t, s; f_t(\xi^T, \ldots, \xi^t, G_\Theta))$.
Output: Sequence of decisions $\hat{U}^t$ for ONLINE.

1: Set S^T as the given initial state
2: **for** $t = T, \ldots, 1$ **do**
3: Compute $\hat{\varphi}(t, S^t) = \varphi(t, S^t; \mathbb{E}[f_t(\xi^T, \ldots, \xi^t, G_\Theta)|\mathcal{F}_t])$ with associated scores $\mathbf{y} = \{y_{u,\xi}\}_{u \in \mathcal{U}, \xi \in \Xi}$
4: Given input ξ^t, choose the action $\hat{U}^t$ with the highest score y_{u,ξ^t}
5: Collect reward $\mathcal{R}(S^t, \xi^t, \hat{U}^t)$; update state $S^{t-1} \leftarrow \mathcal{T}(S^t, \xi^t, \hat{U}^t)$

Before computing the regret of the RABBI policy, we need an additional definition

Definition 5.3.9 (Maximum Loss). For a given relaxation φ, define the maximum loss as

$$r_\varphi := \max_{t,s,u:\mathcal{R}(s,\xi^t,u)>-\infty} \{\varphi(t,s|\mathcal{G}_t) - (\mathcal{R}(s,\xi^t,u) + \mathbb{E}[\varphi(t-1,\mathcal{T}(s,\xi^t,u)|\mathcal{G}_{t-1})|\mathcal{G}_t])\}$$

Theorem 5.3.10. *Let* OFFLINE *be defined by the filtration $\mathcal{G}_t$ as in Definition 5.3.1. Assume the relaxation $\varphi(t,s)$ satisfies the Bellman Inequalities with loss L_B. For $t \in [T], s \in \mathcal{S}$, let $\mathcal{Q}(t,s) \subseteq \Omega$ be the event where $\hat{U}^t$, as specified in* RABBI*, is not a satisfying action. If $(S^t, t \in [T])$ is the state trajectory when following the actions derived from* RABBI*, then*

$$\mathbb{E}[\text{REG}] \leq \mathbb{E}\left[\sum_t (r_\varphi \mathbb{1}_{\mathcal{Q}(t,S^t)} + \mathbb{1}_{\mathcal{Q}(t,S^t)^c} L_B(t,S^t))\right] \leq \sum_t (r_\varphi \mathbb{P}[\mathcal{Q}(t,S^t)] + \mathbb{E}[L_B(t,S^t)]).$$

Remark 5.3.11 (Bellman and Information Loss). The bound in Theorem 5.3.10 has two distinct summands. The term $\sum_t \mathbb{P}[\mathcal{Q}(t,S^t)]$ measures how often RABBI takes a non-satisfying action, hence we refer to it as *information loss*. On the other hand, $\sum_t \mathbb{E}[L_B(t,S^t)])$ captures the fact that φ "almost" has a DP representation, which we defined before as the *Bellman loss*.

Remark 5.3.12 (More General Actions). In order to present a precise rule on how to select an action $\hat{U}^t$ in RABBI, we have assumed that φ is based on an optimization problem. Nevertheless, Theorem 5.3.10 holds even if φ has another representation, e.g., it is obtained through ADP techniques. For concreteness, assume φ is obtained by fitting some basis functions and then the action is greedy w.r.t. this representation. The theorem still holds in this case, but it may require additional techniques to bound the measure of $\mathcal{Q}(t,s)$. We do not further explore this route.

Compensated Coupling: The proof of Theorem 5.3.10 is based on the *compensated coupling* approach introduced in Chapter 3. The basic idea is as follows: Suppose we 'simulate' controllers OFFLINE and ONLINE in parallel with identical random inputs $(\xi^t : t \in [T])$, with ONLINE acting before OFFLINE. Moreover, suppose at some time t, both controllers are in

the same state s. Recall that, for any given state s at time t, an action u is satisfying if Offline's value does not decrease when playing u (see Definition 3.3.4). If Online chooses to play a satisfying action, then we can make Offline play the same action, and consequently both move to the same state. On the other hand, if Online chooses an action that is not satisfying, then the two trajectories will separate; we can avoid this however by 'compensating' Offline so that he agrees to take the same action as Online. Specifically, if the maximum loss is bounded by r_φ, increasing the reward earned by Offline by r_φ is enough to make him choose Online's action. As a consequence, *we have that the (compensated)* Offline *and* Online *follow the same actions, and thus their state trajectories are coupled.*

As an example, consider the baseline `OnlineKnapsack` with budget $B = 2$, unit weights ($w_j \equiv 1$), and horizon $T = 5$. Take a realization $\omega \in \Omega$ with rewards $(\xi^5, \xi^4, \xi^3, \xi^2, \xi^1) = (5, 7, 2, 7, 2)$. In particular, three different types arrived in this realization. The sequence of actions $(\mathtt{r}, \mathtt{a}, \mathtt{r}, \mathtt{a}, \mathtt{r})$ is optimal for Offline with a total reward of 14 (selecting the two 7-valued items). Suppose that Online, at period $t = 5$, wishes to accept the item (obtain a reward of 5, and lose one budget unit). Then, Offline is "willing" to follow this action if paid a compensation of 2 (in addition to the collected ξ^5); Offline and Online then start the next period $t = 4$ in the same state (same remaining budget), hence remain coupled.

Proof of Theorem 5.3.10. Denoting Offline's state as $\bar{S}^t$, we have via Proposition 5.3.8 that $\forall\, t$:

$$\varphi(t, \bar{S}^t | \mathcal{G}_t) \leq \mathbb{E}[\mathcal{R}(\bar{S}^t, \xi^t, U^t) + \varphi(t-1, \bar{S}^{t-1} | \mathcal{G}_{t-1}) + L_B(t, \bar{S}^t) | \mathcal{G}_t].$$

Let us assume as the induction hypothesis that $\bar{S}^t = S^t$. This holds for $t = T$ by definition. At any time t and state S^t, if $\hat{U}^t$ is not a satisfying action for Offline, then we have from the definition of the maximum loss (Definition 5.3.9) that:

$$r_\varphi \geq \varphi(t, S^t | \mathcal{G}_t) - \mathcal{R}(S^t, \xi^t, \hat{U}^t) + \mathbb{E}[\varphi(t-1, S^{t-1} | \mathcal{G}_{t-1}) | \mathcal{G}_t]) \quad a.s..$$

Now to make OFFLINE take action $\hat{U}^t$ so as to have the same subsequent state as ONLINE, it is sufficient to compensate OFFLINE with an additional reward of r_φ. Specifically, we have

$$\varphi(t, S^t|\mathcal{G}_t) \leq \mathbb{E}[\mathcal{R}(S^t, \xi^t, \hat{U}^t) + \varphi(t-1, S^{t-1}|\mathcal{G}_{t-1}) + r_\varphi \mathbb{1}_{\mathcal{Q}(t,S^t)} + \mathbb{1}_{\mathcal{Q}(t,S^t)^c} L_B(t, S^t)|\mathcal{G}_t].$$

Finally, as in Proposition 5.3.8, we can iterate over t to obtain

$$\mathbb{E}[\varphi(T, S^T|\mathcal{G}_T)] \leq \mathbb{E}\left[\sum_t \mathcal{R}(S^t, \xi^t, \hat{U}^t) + \sum_t (r_\varphi \mathbb{1}_{\mathcal{Q}(t,S^t)} + \mathbb{1}_{\mathcal{Q}(t,S^t)^c} L_B(t, S^t))\right].$$

The first sum on the right-hand side corresponds exactly to ONLINE's total reward using the RABBI policy. By the initial ordering of Bellman Inequalities (Definition 5.3.3), $\mathbb{E}[V(T, S^T)] \leq \mathbb{E}[\varphi(T, S^T)]$ and we obtain the desired bound. □

In the remaining sections we apply the general framework to each of the examples introduced in Section 5.2.1 and prove the results stated in Section 5.2.2 (Theorems 5.2.2 to 5.2.4).

5.4 Partial Information and Probing

In this setting, each type j has an independent random reward R_j drawn from the set $\{r_{jk} : k \in [m]\}$; the reward is r_{jk} with probability q_{jk}. We assume without loss of generality that $r_{j1} < r_{j2} < \ldots < r_{jm}$ and $r_{jm} > 0$. The parameters $\mathbf{r}$ and $\mathbf{q}$ are known. For ease of exposition, we assume that all items have weight 1; our analysis however extends to general weights w_j. The controller has a resource budget (i.e., knapsack capacity) $B_h \in \mathbb{N}$ and a probing budget $B_p \in \mathbb{N}$. When an arrival is accepted (resp. probed), we reduce B_h (resp. B_p) by one. If an item of type j is probed, then R_j is revealed and the controller can decide whether to accept or reject the item. The controller can also accept an item of type j without probing, in which case the expected reward is $\bar{r}_j := \sum_{k\in[m]} r_{jk} q_{jk}$.

Note that, when either $B_p \geq T$ or $B_p = 0$, this problem reduces to the baseline `OnlineKnapsack`. In particular, when $B_p \geq T$, the controller can probe at every single

period, in which case we are back at the baseline `OnlineKnapsack` with revealed rewards and mn types. If $B_p = 0$, we have the baseline `OnlineKnapsack` with n types and reward equal to the expectation $\bar{r}_j$ for type j.

5.4.1 Probing RABBI

Consider the following LP, parametrized by $(\mathbf{b}, \mathbf{z}) \in \mathbb{N}^2 \times \mathbb{R}^n_{\geq 0}$,

$$
\begin{aligned}
(P[\mathbf{b},\mathbf{z}]) \qquad \max \quad & \textstyle\sum_{j,k} r_{jk} x_{jk\mathtt{a}} + \sum_j \bar{r}_j x_{j\mathtt{a}} & & \\
\text{s.t.} \quad & \textstyle\sum_{j,k} x_{jk\mathtt{a}} + \sum_j x_{j\mathtt{a}} \leq b_h & & \\
& \textstyle\sum_j x_{j\mathtt{p}} \leq b_p & & \\
& x_{j\mathtt{a}} + x_{j\mathtt{p}} + x_{j\mathtt{r}} = z_j & & \forall j \in [n] \\
& x_{jk\mathtt{a}} + x_{jk\mathtt{r}} = q_{jk} x_{j\mathtt{p}} & & \forall j \in [n], k \in [m] \\
& x_{j\mathtt{a}}, x_{j\mathtt{p}}, x_{j\mathtt{r}}, x_{jk\mathtt{a}}, x_{jk\mathtt{r}} \geq 0 & & \forall j \in [n], k \in [m].
\end{aligned}
\tag{5.9}
$$

The decision variables $\mathbf{x} \in \mathbb{R}^{3n+2nm}_{\geq 0}$ have a natural interpretation as action summaries: $x_{j\mathtt{a}}, x_{j\mathtt{r}}, x_{j\mathtt{p}}$ are the total number of future type-j arrivals that are accepted without probing, rejected without probing, and probed respectively; $x_{jk\mathtt{a}}, x_{jk\mathtt{r}}$ are the number of probed future type-j arrivals that are revealed to have reward r_{jk}, and then accepted/rejected respectively. The first constraint requires that the number of accepted items does not exceed the knapsack capacity; the second constraint guarantees that the number of items probed does not exceed the probing budget; the third is a "demand constraint" that guarantees the number of type-j items accepted, probed or rejected equals arrivals of that type. Finally, the last constraint guarantees that a q_{jk} fraction of probed type-j items have sub-type k (i.e., reward r_{jk}).

The LP $P[\mathbf{b}, \mathbf{z}]$ now plays the role of the proxy $\hat{\varphi}$ for the Probing RABBI presented below in Algorithm 7 (with ties broken arbitrarily); we specify the relaxation φ later in Eq. (5.10).

Intuition behind Offline. Online probing is representative of multi-stage problems and

Algorithm 7 Probing RABBI

Input: Access to solutions of $(P[\mathbf{b}, \mathbf{z}])$
Output: Sequence of decisions for ONLINE.

Set $(B_h^T, B_p^T) \leftarrow (B_h, B_p)$ as the given initial state
for period $t = T, \ldots, 1$ **do**
    If $B_h^t = 0$ (no more budget): break.
    Compute X^t, an optimal solution to $(P[B^t, \mathbb{E}[Z(t)]])$
    Observe the arrival, say it is of type j, then take action $\hat{U}^t \in \operatorname{argmax}_{u=\mathtt{a},\mathtt{p},\mathtt{r}}\{X_{j,u}^t\}$.
    If $\hat{U}^t = \mathtt{r}$ or $\hat{U}^t = \mathtt{a}$: collect zero or random R_j, respectively.
    If $\hat{U}^t = \mathtt{p}$: observe the realization, say $R_j = r_{jk}$, then take action $\operatorname{argmax}_{u=\mathtt{a},\mathtt{r}}\{X_{j,k,u}^t\}$
    Update states B^{t-1} accordingly.

it is naturally described by two distinct types of stages: on a *first stage* the type $j \in [n]$ is revealed and the controller may probe which results in a *second stage* where the actual reward or "sub-type" $(j, k) \in [n] \times [m]$ is revealed. OFFLINE is defined as the controller that knows the randomness of all first stages, but not that of second stages. In other words, OFFLINE knows, for each type j, the number $Z_j(T)$ of arrivals, but not the sub-types. Since OFFLINE does not know the sub-type, he still wants to solve a dynamic program to decide whether to probe or not, therefore, since OFFLINE faces randomness (sub-types), there will be a Bellman Loss associated to this (and not only due to the integrality gap).

5.4.2 Dynamic Programming Formulation

It is useful to model each period as consisting of two stages. In this section, we assume each period $t \in \{T, T-1, \ldots, 1\}$ comprises of two stages $\{t, t-1/2\}$, driven by external inputs $\xi^t \in [n]$ and $\xi^{t-1/2} \in [n] \times [m]$. In the first stage t, the controller observes the type of the arriving request $\xi^t = j$; in the second stage $t - 1/2$, the realization of the reward is drawn according to probabilities $\mathbf{q}$. In particular, if the arrival is of type j (revealed in the first stage), then in the second stage the "sub-type" $\xi^{t-1/2} = (j, k)$, associated with reward $r_{j,k}$, is drawn with probability q_{jk}. We augment the state space with a variable $\diamond$ that captures

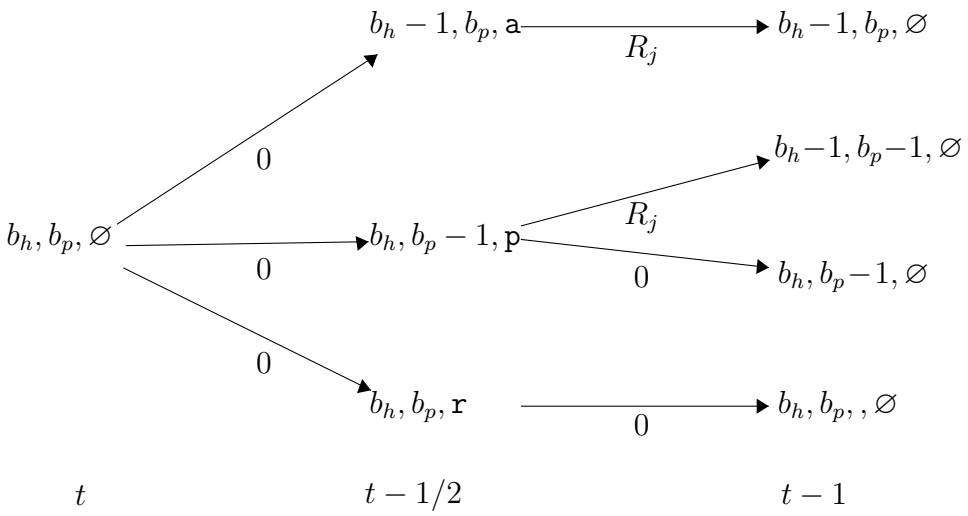


Figure 5.3: Actions/transitions in online probing in periods t, $t-1/2$, and $t-1$, with inputs $\xi^t = j$ and $\xi^{t-1/2} = R^j$. Numbers below the arrows represent the reward of a transition. At t, available actions are $\{\mathtt{a}, \mathtt{p}, \mathtt{r}\}$ (i.e., accept, probe, reject; from top to bottom); at $t - 1/2$, if we chose to probe in the first-stage (i.e., are in the middle state), then available actions are $\{\mathtt{a}, \mathtt{r}\}$.

the first stage decision (i.e., whether we accept/reject without probing or probe). The state space $\mathcal{S}$ of the controlled process is thus $\mathcal{S} = \{(b_h, b_p, \diamond) : b_h, b_p \in \mathbb{N}, \diamond \in \{\mathtt{a}, \mathtt{p}, \mathtt{r}, \varnothing\}\}$, where b_h, b_p are the residual hiring and probing budgets. In first stages we set $\diamond = \varnothing$, and only collect rewards in second stages. Fig. 5.3 displays the actions and state transitions under this reformulation.

5.4.3 Offline Information and Relaxation

The Offline controller knows the public types of *all* arrivals in advance (i.e., it knows $Z_j(t)$, the number of type-j items that will arrive in the last t periods), but does not know the realization of the rewards (sub-types). Formally, Offline is endowed with the canonical filtration given by $\Theta = [T]$ and $G_\theta = \xi^\theta$ (see Definition 5.3.1): with t steps to go, Offline has the information filtration $\mathcal{G}_t = \sigma(\{\xi^t : t \in [T]\} \cup \{\xi^\tau : \tau \geq t\})$.

Solving for Offline's optimal actions may be non-trivial, as it corresponds to a dynamic program where, for each arrival, the controller must decide whether to accept/reject without probing, or probe, and then decide to accept/reject based on the realization of the reward. We circumvent this via an offline relaxation φ built using the LP $P[\mathbf{b}, \mathbf{z}]$ (5.9). Recall that a state is of the form $s = (b_h, b_p, \diamond) = (\mathbf{b}, \diamond)$ with $\diamond \in \{\mathtt{a}, \mathtt{p}, \mathtt{r}, \varnothing\}$. For period t (i.e., first stage, $\diamond = \varnothing$), we define

$$\varphi(t, s|\mathcal{G}_t) = P[\mathbf{b}, Z]$$

For $t-1/2$ (i.e., second stage decisions), we modify φ to incorporate the action $(\mathtt{a}, \mathtt{p}, \mathtt{r})$ taken in the first stage. Overall, our relaxation is defined as follows

$$\varphi(t-1/2, s|\mathcal{G}_t) = \begin{cases} P[\mathbf{b}, Z(t-1)] & \diamond = \mathtt{r} \\ r_{\xi^{t-1/2}} + P[\mathbf{b}, Z(t-1)] & \diamond = \mathtt{a} \\ \max\{r_{\xi^{t-1/2}} + P[\mathbf{b} - \mathbf{e}_h, Z(t-1)], P[\mathbf{b}, Z(t-1)]\} & \diamond = \mathtt{p} \end{cases} \tag{5.10}$$

5.4.4 Bellman Inequalities and Loss

Initial ordering. Lemma 5.4.1 provides the first ingredient for the application of Definition 5.3.3.

Lemma 5.4.1. *For any $b_h, b_p \in \mathbb{N}$, and realization Z of arrivals, $\mathbb{E}[V(T, \mathbf{b}|\mathcal{G}_T)] \leq \mathbb{E}[\varphi(T, (\mathbf{b}, \varnothing)|\mathcal{G}_T)]$.*

Proof. The main idea is showing that any offline policy induces action summaries that satisfy the constraints defining φ. Consider a policy for Offline which determines when to probe, accept or reject. A policy is a mapping $\pi : [T] \times \mathcal{S} \to \mathcal{U}$ such that $\pi(t, s)$ is $\mathcal{G}_t$-measurable for all t, s.

The policy, once fixed, induces a random trajectory determined by the realization of the probed rewards. Arrivals, recall, are known to Offline. Define the following random

variables counting the number of times in which a type j was: probed, denoted $X_{j\mathtt{p}}$; accepted (rejected) without probing, $X_{j\mathtt{a}}$ ($X_{j\mathtt{r}}$); and was accepted (rejected) after the probe resulted in (j,k), $X_{jk\mathtt{a}}$ ($X_{jk\mathtt{r}}$).

Then, we can write $\mathbb{E}[V(T,\mathbf{b}|\mathcal{G}_T)] = \mathbb{E}[\sum_j \bar{r}_j X_{j\mathtt{a}} + \sum_{j,k} r_{jk} X_{jk\mathtt{a}}|\mathcal{G}_T]$, where we use the fact that the policy is $\mathcal{G}_t$-adapted for the term $\bar{r}_j$, i.e., conditional on accepting without probing, the expected reward is $\bar{r}_j$. We conclude,

$$\mathbb{E}[V(t,\mathbf{b}|\mathcal{G}_T)] = \sum_j \bar{r}_j \mathbb{E}[X_{j\mathtt{a}}|\mathcal{G}_T] + \sum_{j,k} r_{jk}\mathbb{E}[X_{jk\mathtt{a}}|\mathcal{G}_T].$$

We claim that the expectation of the random vector X yields a feasible solution to $(P[T,\mathbf{b},Z])$. In turn, $P[T,\mathbf{b},\mathcal{G}_T]$ can only be larger than the value. Indeed, with the exception of the constraint $x_{jk\mathtt{a}} + x_{jk\mathtt{r}} = q_{jk}x_{j\mathtt{p}}$, the random variables satisfy a.s. all the constraints of $(P[T,\mathbf{b},Z])$, hence their expectations do too. Furthermore, since Offline's policy is adapted to $\mathcal{G}$, we obtain $\mathbb{E}[X_{jk\mathtt{a}} + X_{jk\mathtt{r}}|X_{j\mathtt{p}},\mathcal{G}_T] = q_{jk}X_{j\mathtt{p}}$, thus the expected values satisfy the desired constraint. To summarize, $V(T,\mathbf{b}|\mathcal{G}_T)$ equals the value of the feasible solution given by the expectations. □

Monotonicity and satisfying actions. We will require the following LP lemma

Lemma 5.4.2. *Consider the standard-form LP* $(P[\mathbf{d}]) : \max\{\mathbf{r}'\mathbf{x} : M\mathbf{x} = \mathbf{d}, \mathbf{x} \geq 0\}$*, where* $M \in \mathbb{R}^{m\times n}$ *is an arbitrary constraint matrix and* $\mathbf{d} \in \mathbb{R}^m$*. The function* $\mathbf{d} \mapsto P[\mathbf{d}]$ *is concave and therefore, if* X *is a random right-hand side, then* $\mathbb{E}[P[X]] \leq P[\mathbb{E}[X]]$*.*

Proof. The dual problem is $(D[\mathbf{d}]) : \min\{\mathbf{d}'\mathbf{y} : M'\mathbf{y} \geq \mathbf{r}\}$. The function $\mathbf{d} \mapsto D[\mathbf{d}]$ is a minimum of linear functions, therefore concave. □

Before proving that φ satisfies the Bellman Inequalities, we recall the concept of a *satisfying action* in Definition 3.3.4 with our choice of L_B. Given φ, filtration $\mathcal{G}$, state s and

time t, action u is satisfying if $\varphi(t, s|\mathcal{G}_t) \leq \mathcal{R}(s, \xi^t, u) + \mathbb{E}[\varphi(t-1, \mathcal{T}(s, \xi^t, u)|\mathcal{G}_{t-1})|\mathcal{G}_t]$.

Lemma 5.4.3. *Let $\bar{X}$ be a maximizer of $(P[\mathbf{b}, Z(t)])$ for t a first stage and say $\xi^t = i$. Then we have the following implications for satisfying actions*

(1) *If $\bar{X}_{ia} \geq 1$ or $\bar{X}_{ir} \geq 1$ ($\max\{\bar{X}_{ia}, \bar{X}_{ir}\} \geq 1$): OFFLINE is satisfied (in the sense of Definition 3.3.4) accepting or rejecting, respectively, at time t.*

(2) *If $\bar{X}_{ip} \geq 1$ and $\xi^{t-1/2} = (i, k)$ is such that either $\bar{X}_{ika} \geq 1$ or $\bar{X}_{ikr} \geq 1$ ($\bar{X}_{ip} \geq 1$ and $\max\{\bar{X}_{ika}, \bar{X}_{ikr}\} \geq 1$): OFFLINE is (i) satisfied probing at time t and (ii) satisfied accepting if $\bar{X}_{ika} \geq 1$ or rejecting if $\bar{X}_{ikr} \geq 1$ at time $t - 1/2$.*

In conclusion, φ satisfies the Bellman Inequalities with Bellman Loss $L_B(t, \mathbf{b}) = r_\varphi \mathbb{1}_{\mathcal{B}(t,\mathbf{b})}$, where $\mathcal{B}$ are the following exclusion sets:

$$\mathcal{B}(t, \mathbf{b}) = \{\omega \in \Omega : \nexists \bar{X} \text{ solving } (P[\mathbf{b}, Z(t)]) \text{ s.t. (1) or (2) hold}\}.$$

PROOF. We will establish the monotonicity condition (second requirement for Bellman Inequalities in Definition 5.3.3):

$$\varphi(t, s|\mathcal{G}_t) \leq \max_{u \in \mathcal{U}} \{\mathcal{R}(s, \xi^t, u) + \mathbb{E}_{\xi^{t-1/2}}[\varphi(t - 1/2, \mathcal{T}(s, \xi^t, u)|\mathcal{G}_{t-1/2})|\mathcal{G}_t]\} \quad \forall \omega \notin \mathcal{B}(t, s).$$

Observe that, since t is a first stage, the instant reward $\mathcal{R}(s, \xi^t, u)$ is zero. In case (1) of the lemma it is easy to verify the inequality by invoking Lemma 5.3.5.

For case (2) we need to introduce some notation. Let $\mathbf{q}_j \in \mathbb{R}^{n \times m}$ be a vector with value q_{jk} in components (j, k), $k \in [m]$, and zero otherwise (i.e. in components (j', k) with $j' \neq j$). Similarly, let $\mathbf{1}_{(j,k)} \in \mathbb{R}^{n \times m}$ have value 1 in the single component (j, k) and zero otherwise. The following LP is the same as in Eq. (5.9), but has the extra parameter $\mathbf{y}$ that facilitates the analysis; $P[\mathbf{b}, \mathbf{z}] = \bar{P}[\mathbf{b}, \mathbf{z}, \mathbf{0}]$.

$$
\begin{array}{llll}
(\bar{P}[\mathbf{b},\mathbf{z},\mathbf{y}]) & \max & \sum_{j,k} r_{jk}x_{jk\mathtt{a}} + \sum_j \bar{r}_j x_{j\mathtt{a}} & \\
& \text{s.t.} & \sum_{j,k} x_{jk\mathtt{a}} + \sum_j x_{j\mathtt{a}} \leq b_h & \\
& & \sum_j x_{j\mathtt{p}} \leq b_p & \\
& & x_{j\mathtt{a}} + x_{j\mathtt{p}} + x_{j\mathtt{r}} = z_j & \forall j \\
& & x_{jk\mathtt{a}} + x_{jk\mathtt{r}} = q_{jk}x_{j\mathtt{p}} + y_{jk} & \forall j,k \\
& & \mathbf{x} \geq 0. &
\end{array}
\tag{5.11}
$$

In case (2), $\bar{X}_{i\mathtt{p}} \geq 1$ and $\xi^{t-1/2} = (i,k)$ is such that either $\bar{X}_{ik\mathtt{a}} \geq 1$ or $\bar{X}_{ik\mathtt{r}} \geq 1$. By Lemma 5.3.5, we have the following breakdown (depending on the random $\xi^{t-1/2}$)

$$
P[\mathbf{b}, Z(t)] = \bar{P}[\mathbf{b}, Z(t), \mathbf{0}] = r_{\xi^{t-1/2}} I + \bar{P}[\mathbf{b} - \mathbf{e}_p - \mathbf{e}_h I, Z(t-1), \mathbf{q}_{\xi^t} - \mathbf{1}_{\xi^{t-1/2}}], \quad \forall \omega \notin \mathcal{B}(t,b),
$$

where $I \coloneqq \mathbb{1}_{\left\{\bar{X}(\xi^{t-1/2},a)\geq 1\right\}}$ and the vectors $\mathbf{q}, \mathbf{1}$ are evaluated in random components; since by assumption $\bar{X}_{i\mathtt{p}} \geq 1$ under the optimal solution, the optimal value in the optimization problem is the same as the reward obtained "now" ($r_{\xi^{t-1/2}}$) and the residual value after discounting b_p by one.

Taking expectations $\mathbb{E}[\cdot|\mathcal{G}_t]$ and using Lemma 5.4.2 we have

$$
\begin{aligned}
P[\mathbf{b}, Z(t)] &= \mathbb{E}[r_{\xi^{t-1/2}} I + \bar{P}[\mathbf{b} - \mathbf{e}_p - \mathbf{e}_h I, Z(t-1), \mathbf{q}_{\xi^t} - \mathbf{1}_{\xi^{t-1/2}}]|\mathcal{G}_t] \\
&\leq \mathbb{E}[r_{\xi^{t-1/2}} I + \bar{P}[\mathbf{b} - \mathbf{e}_p - \mathbf{e}_h I, Z(t-1), \mathbf{0}]|\mathcal{G}_t] \\
&\leq \mathbb{E}[\max\{r_{\xi^{t-1/2}} + P[\mathbf{b} - \mathbf{e}_p - \mathbf{e}_h, Z(t-1)], P[\mathbf{b} - \mathbf{e}_p, Z(t-1)]\}|\mathcal{G}_t].
\end{aligned}
$$

The last expression is obtained by considering the possible realizations of $I \in \{0,1\}$ and corresponds to the desired inequality. □

5.4.5 Information Loss and Overall Performance Guarantee

Our relaxation φ in Eq. (5.10) is a function of the future arrivals at time t, $Z(t)$, and can be written in the form $\varphi(t,s) = \varphi(t,s;Z(t))$. This coincides with the general formulation in Section 5.3.2, specifically with Eq. (5.7) therein. The proxy is $\hat{\varphi}(t,s) = \varphi(t,s;\mathbb{E}[Z(t)])$, which is used to generate actions $\hat{U}^t$ through the LP (5.9) that defines it.

Specifically, we use maximizers X^t of $(P[B^t, \mu(t)])$ as estimators for solutions of $(P[B^t, Z(t)])$, where $\mu(t) \coloneqq \mathbb{E}[Z(t)]$. If the type of the arrival at time t is j, then $X^t(j,\mathtt{a}), X^t(j,\mathtt{r}), X^t(j,\mathtt{p})$ indicate how much of type j we want to accept, reject and probe, respectively. We choose the action with largest variable value. In case we decide to probe, we repeat the logic with variables $X^t(j,k,\mathtt{a}), X^t(j,k,\mathtt{r})$.

Proof of Theorem 5.2.2. By Theorem 5.3.10, $\textsc{Reg} \le r_\varphi \sum_t (\mathbb{1}_{\mathcal{B}(t,S^t)} + \mathbb{1}_{\mathcal{Q}(t,S^t)})$. To bound this expression, we proceed in two steps: bounding the measure of the exclusion sets $\mathcal{B}$ and the "disagreement" sets $\mathcal{Q}$. We conclude using the fact that $r_\varphi \le \max_{j,k} r_{jk}$.

To bound the measure of the exclusion sets $\mathcal{B}$, let $\bar{X}$ be the solution to $(P[\mathbf{b}, Z(t)])$ and recall that Lemma 5.4.3 guarantees that under any of the following conditions there is zero Bellman Loss: (1) $\max\{\bar{X}_{j\mathtt{a}}, \bar{X}_{j\mathtt{r}}\} \ge 1$ or (2) $\bar{X}_{j\mathtt{p}} \ge 1$ and $\max\{\bar{X}_{jk\mathtt{a}}, \bar{X}_{jk\mathtt{r}}\} \ge 1$. We therefore bound the event when both (1) and (2) fail, which corresponds to the exclusion set.

We exploit the following two restrictions of $(P[\mathbf{b}, Z(t)])$: $x_{j\mathtt{a}} + x_{j\mathtt{p}} + x_{j\mathtt{r}} = Z_j(t)\ \forall j$ and $x_{jk\mathtt{a}} + x_{jk\mathtt{r}} = q_{jk} x_{j\mathtt{p}}\ \forall j,k$. If $Z_j(t) \ge 3$, then one of the variables $x_{j\mathtt{a}}, x_{j\mathtt{p}}, x_{j\mathtt{r}}$ must be at least 1. On the other hand, we need $q_{jk} x_{j\mathtt{p}} \ge 2$ to guarantee that one of $x_{jk\mathtt{a}}, x_{jk\mathtt{r}}$ is at least 1.

Putting this argument together, the event where (1) and (2) fail is bounded by

$$\mathbb{P}[\mathcal{B}(t,b)|\xi^{t-1/2}=(j,k)] \leq \mathbb{P}\left[Z_j(t) < \frac{6}{q_{jk}}\right] = \mathbb{P}\left[Z_j(t)-\mu_j(t) < -\mu_j(t)\left(1-\frac{6}{\mu_j(t)q_{jk}}\right)\right]. \tag{5.12}$$

A standard Chernoff bound (see [27]) implies

$$\mathbb{P}[\mathcal{B}(t,b)|\xi^{t-1/2}=(j,k)] \leq e^{-2(p_j/2)t} + \mathbb{1}_{\{t\leq 12/(p_jq_{jk})\}}.$$

Here, we applied the bound to the restricted range $\mu_j(t) \geq 12/q_{jk}$, which guarantees, in particular, that the the right-hand side of Eq. (5.12) is positive.

Finally,

$$\sum_t \mathbb{P}[\mathcal{B}(t,B^t)] \leq \sum_t \sum_j p_j e^{-2(p_j/2)t} + \sum_t \sum_{j,k} p_j q_{jk} \mathbb{1}_{\{t\leq 12/(p_jq_{jk})\}} \leq \sum_j \frac{2}{p_j} + 12.$$

This finishes the first part of the proof, i.e., bounding the exclusion sets.

We turn to the second part of the proof, which is to bound the Information Loss $\sum_t \mathbb{P}[\mathcal{Q}(t,S^t)]$. Recall that $\mathcal{Q}(t,S^t) \subseteq \Omega$ is the event where $\hat{U}^t$ is not satisfying. Let $\bar{X}$ be a solution to $(P[\mathbf{b},Z(t)])$, t a first stage, and let $j=\xi^t$. We divide the analysis in two cases: if $\hat{U}^t \in \{\mathtt{a},\mathtt{r}\}$ or if $\hat{U}^t = \mathtt{p}$.

Assume $\hat{U}^t \in \{\mathtt{a},\mathtt{r}\}$. According to Lemma 5.4.3, accepting or rejecting is satisfying whenever $\max\{\bar{X}_{j\mathtt{a}}, \bar{X}_{j\mathtt{r}}\} \geq 1$. Since $X^t(\xi^t,\hat{U}^t) = \max\{X^t(\xi^t,u) : u = \mathtt{a},\mathtt{p},\mathtt{r}\}$ and we have the constraint $X^t_{j\mathtt{a}} + X^t_{j\mathtt{p}} + X^t_{j\mathtt{r}} = \mu_j(t)$, the error is given by

$$\mathbb{P}[\bar{X}(j,\hat{U}^t) < 1 | X^t(j,\hat{U}^t) \geq \mu_j(t)/3] \leq \mathbb{P}[||\bar{X}-X^t||_\infty \geq \mu_j(t)/3].$$

On the other hand, if $\hat{U}^t = \mathtt{p}$, the error is bounded by

$$\mathbb{P}\left[\bar{X}_{j\mathtt{p}} < 1 \text{ or } \bar{X}_{\xi^{t-1/2},u} < 1 \middle| X^t_{j\mathtt{p}} \geq \frac{\mu_j(t)}{3}, X^t_{\xi^{t-1/2},u} \geq \frac{q_{\xi^{t-1/2}}\mu_j(t)}{6}\right] \leq \mathbb{P}\left[||\bar{X}-X^t||_\infty \geq \frac{q_{\xi^{t-1/2}}\mu_j(t)}{6}\right],$$

where u is the action with largest value between the variables $X^t(\xi^{t-1/2},\mathtt{a}), X^t(\xi^{t-1/2},\mathtt{r})$.

We conclude that, regardless of the action $\hat{U}^t$, the probability of choosing a non-satisfying action is bounded by $\mathbb{P}[||\bar{X} - X^t||_\infty \geq \min_k q_{jk} \cdot \mu_j(t)/6]$. From the LP sensitivity result [84, Theorem 2.4], we deduce the existence of κ that depends on $\mathbf{q}, n, m$ only s.t. $||\bar{X} - X^t||_\infty \leq \kappa||Z(t) - \mu(t)||_1$. Finally, the number of times that ONLINE chooses a non-satisfying action (measure of all sets $\mathcal{Q}$) is bounded by

$$\sum_t \mathbb{P}[\mathcal{Q}(t, S^t)] \leq \sum_t \mathbb{P}[||Z(t) - \mu(t)||_1 \geq \min_k q_{jk} \cdot \mu_j(t)/6\kappa] < \infty. \tag{5.13}$$

The summability follows from standard concentration bounds (see Chapter 3) and Theorem 5.2.2 now follows from Theorem 5.3.10. □

Remark 5.4.4 (non-i.i.d arrival processes). The key property of the arrival process that we used is a tail bound of the form $\mathbb{P}[||Z(t) - \mathbb{E}[Z(t)]|| \geq c\mathbb{E}[Z(t)]] \leq g(t)$, see Eq. (5.13). The result thus holds for any arrival process that satisfies such deviation bounds.

Remark 5.4.5 (Probing cost). Suppose that the controller has infinite probing budget ($B_p = \infty$), but incurs a penalty c_j when probing a type-j arrival. All else remains unchanged. The only change to results and proofs is the definition of $P[\mathbf{b}, Z]$ to one where the probing budget is dropped and a probing cost is added to the objective function:

$$\begin{aligned}
(P[\mathbf{b}, \mathbf{z}]) \qquad \max \quad & \textstyle\sum_{j,k} r_{jk} x_{jk\mathtt{a}} + \sum_j \bar{r}_j x_{j\mathtt{a}} - \sum_j c_j x_{j\mathtt{p}} & & \\
\text{s.t.} \quad & \textstyle\sum_{j,k} x_{jk\mathtt{a}} + \sum_j x_{j\mathtt{a}} \leq b_h & & \\
& x_{j\mathtt{a}} + x_{j\mathtt{p}} + x_{j\mathtt{r}} = z_j & & \forall j \in [n] \\
& x_{jk\mathtt{a}} + x_{jk\mathtt{r}} = q_{jk} x_{j\mathtt{p}} & & \forall j \in [n], k \in [m] \\
& x_{j\mathtt{p}}, x_{j\mathtt{r}}, x_{jk\mathtt{a}}, x_{jk\mathtt{r}} \geq 0 & & \forall j \in [n], k \in [m]
\end{aligned}$$

5.5 Pricing

There is a set of d resources and n customer types. Each customer type requests a set of resources and has a private reward if the resources are allocated to him. A purchase occurs

iff there is availability of resources and the controller posts a price below the private reward. The resource consumption is encoded in a matrix $A \in \{0,1\}^{d \times n}$.

At each time t, the controller first observes the type $j \in [n]$, which arrives with probability p_j, then posts one price f_{jl} among $\{f_{j1}, \ldots, f_{jm}\}$. Subsequently, the customer draws a private reward $R^t \sim F_j$ (the customer's valuation) and a purchase occurs iff $R^t > f_{jl}$. If the customer buys, f_{jl} is collected and the inventory decreases according to A_j. On the other hand, if the customer does not buy, the controller collects zero and the inventory remains unchanged.

5.5.1 Pricing RABBI

The Pricing RABBI algorithm uses as proxy $\hat{\varphi}$ the following LP parameterized by $(\mathbf{b}, \mathbf{q}, \mathbf{z})$

$$
\begin{array}{lllll}
(P[\mathbf{b}, \mathbf{q}, \mathbf{z}]) & \max & \sum_{j,l} f_{jl} q_{jl} x_{jl} & & \\
& \text{s.t.} & \sum_{j,l} a_{ij} q_{jl} x_{jl} & \leq b_i & \forall i \in [d] \\
& & \sum_{l} x_{jl} + x_{j\mathtt{r}} & = z_j & \forall j \in [n] \\
& & \mathbf{x} & \geq 0. &
\end{array}
\tag{5.14}
$$

If we take $q_{jl} = \bar{F}_j(f_{jl})$, i.e., the probability that price f_{jl} is accepted by a type-j customer and z_j is the number of type-j arrivals, then the problem $(P[\mathbf{b}, \mathbf{q}, \mathbf{z}])$ can be interpreted as follows: the variable x_{jl} represents the number of times that price f_{jl} is offered, with $\sum_{j,l} f_{jl} q_{jl} x_{jl}$ the expected reward from the corresponding arrivals. Each time price f_{jl} is offered, $a_{ij} q_{jl}$ units of resource i are consumed in expectation, and hence $\sum_{j,l} a_{ij} q_{jl} x_{jl}$ is the total expected consumption of resource i. Finally, at most one price is offered per arrival, which is captured by $\sum_l x_{jl} + x_{j\mathtt{r}} = z_j$, where $x_{j\mathtt{r}}$ is the number of rejected type-j customers. The resulting online policy is presented in Algorithm 8. The relaxation φ is defined in the next subsection.

Algorithm 8 Pricing RABBI

Input: Access to solutions of $(P[\mathbf{b}, \mathbf{q}, \mathbf{z}])$
Output: Sequence of decisions for ONLINE.

Set $B^T \leftarrow B$ as the given initial budget and $q_{jl} \leftarrow \bar{F}_j(f_{jl})$
for $t = T, ..., 1$ **do**
    If the arrival is type j and $A_j \not\leq B^t$: not enough resources, reject and go to $t-1$
    Compute X^t, an optimal solution to $(P[B^t, \mathbf{q}, t\mathbf{p}])$
    Let $l \in \text{argmax}\{X^t_{j,l} : l = 1, \ldots, m, \mathtt{r}\}$. If $l = \mathtt{r}$, reject and go to $t-1$. Else post price f_{jl}
    If $R^t > f_{jl}$, collect f_{jl} and $B^{t-1} \leftarrow B^t - A_j$; else $B^{t-1} \leftarrow B^t$

Intuition behind Offline. For each type j, there are $Z_j(T)$ arrivals and hence $Z_j(T)$ draws from the distribution F_j. We reveal to OFFLINE the histogram of these draws: for each j, OFFLINE knows the empirical distribution of the $Z_j(T)$ rewards. We highlight that OFFLINE does not know the identity of the private rewards, therefore it is not a full information filtration. For example, say $Z_1(T) = 15$ and we reveal that 10 arrivals type-1 have private reward \$1 and 5 arrivals type-1 have private reward \$2, then, upon observing a type-1 arrival, OFFLINE concludes that the reward is \$2 with probability $\frac{5}{15}$. Crucially, after OFFLINE posts a price, he observes a realization, say it was \$1, then, the next time he observes a type-1, he will use that the reward is \$2 with probability $\frac{5}{14}$.

5.5.2 Offline Information and Relaxation

Assume, w.l.o.g., that, for each j, the prices are ordered $f_{j1} > f_{j2} > \ldots > f_{jm}$. Let us denote $\xi^t \in [n]$ the type of the arrival at time t. To formulate a suitable OFFLINE, we introduce a sequence of independent random vectors $\{Y^t : t = T, T-1, \ldots, 1\}$ defined by the relation $Y^t_{jl} := \mathbb{1}_{\{\xi^t = j, R^t > f_{jl}\}}$. In other words, Y^t_{jl} counts if a price price f_{jl} or lower would be accepted by the type-j at time t.

Define $Q_{jl}(t) := \frac{1}{Z_j(t)} \sum_{\tau=1}^{t} Y^\tau_{jl}$ as the empirical average of the last t periods; $Q_{jl}(t)$ is the

fraction of type-j customers who would accept price f_{jl}. Observe that $Q_{jl}(t)$ is a martingale with $\mathbb{E}[Q_{jl}(t)] = \bar{F}_j(f_{jl})$ and $Q_{jl}(t) = \frac{Z_j(t+1)}{Z_j(t)} Q_{jl}(t+1) - \frac{1}{Z_j(t)} Y_{jl}^{t+1}$.

Offline's information is given by the filtration $\mathcal{G}_t = \sigma(\{Q(\tau), Z(\tau) : \tau \geq t\})$, i.e., at every time t, Offline knows the total demand $Z_j(t)$ and the empirical averages $Q_{jl}(t)$, but does not know at what time which private reward arrived. This coincides with the canonical filtration (Definition 5.3.1) with variables $(Q_{jl}(T), Z_j(T) : j \in [n], l \in [m])$. The filtration $\mathcal{G}$ is strictly coarser than the full information filtration, which would correspond to revealing all the variables $Y^T, Y^{T-1}, \ldots, Y^1$ instead of their averages. We take the relaxation $\varphi(t, \mathbf{b}|\mathcal{G}_t) \coloneqq P[\mathbf{b}, Q(t), Z(t)]$, i.e.,

$$\begin{aligned} \varphi(t, \mathbf{b}|\mathcal{G}_t) \quad = \quad & \max \quad && \textstyle\sum_{j,l} f_{jl} Q_{jl}(t) x_{jl} && \\ & \text{s.t.} && \textstyle\sum_{j,l} a_{ij} Q_{jl}(t) x_{jl} \;\; \leq b_i && \forall i \in [d] \\ & && \textstyle\sum_{l} x_{jl} + x_{j\mathtt{r}} \;\; = Z_j(t) && \forall j \in [n] \\ & && \mathbf{x} \;\; \geq 0. && \end{aligned}$$

5.5.3 Bellman Inequalities and Loss

Proposition 5.5.1 shows that our choice of φ satisfies the Bellman Inequalities. The proof of the initial ordering is similar to that of Lemma 5.4.1, hence we omit it. We present the main ingredients to obtain the second part of Proposition 5.5.1 (monotonicity). For ease of exposition, when the controller rejects, he can equivalently post $f_{j\mathtt{r}} = \infty$ such that $\bar{F}_j(f_{j\mathtt{r}}) = 0$ with the convention $0 \times \infty = 0$.

Proposition 5.5.1. *Let $\varphi(t, \mathbf{b}|\mathcal{G}_t) = P[\mathbf{b}, Q(t), Z(t)]$ with optimal solution X.*

1. *Let $V(T, B|\mathcal{G}_T)$ be the value of* Offline*'s optimal policy, then $\mathbb{E}[V(T, B|\mathcal{G}_T)] \leq \mathbb{E}[\varphi(T, B|\mathcal{G}_T)]$, hence φ satisfies the initial ordering condition.*
2. *If the current type is j and $\max_l\{X_{jl}\} \geq 1$, then $\mathbb{E}[L_B(t, \mathbf{b})] \leq 0$.*

3. If the current type is j and $X_{jl} \geq 1$, then posting f_{jl} is a satisfying action.

We start by recalling the monotonicity condition (Definition 5.3.3). Denote $\mathbb{E}_t[\cdot] = \mathbb{E}[\cdot|\mathcal{G}_t]$. If the inventory is $\mathbf{b} \geq A_j$, the random reward of posting price f_{jl} at t is $f_{jl}Y_{jl}^t$ and the random new inventory is $\mathbf{b} - A_jY_{jl}^t$, thus monotonicity corresponds to:

$$\varphi(t+1, \mathbf{b}) \leq \max_{l \in [m] \cup \{\mathbf{r}\}} \{\mathbb{E}_{t+1}[f_{jl}Y_{jl}^{t+1} + \varphi(t, \mathbf{b} - A_jY_{jl}^{t+1})]\} + \mathbb{E}_{t+1}[L_B(t+1, \mathbf{b})].$$

Because Q is a martingale, we have $\mathbb{E}_t[Y^t] = Q(t)$ and we can further simplify the condition to

$$\varphi(t+1, \mathbf{b}) \leq \max_{l \in [m] \cup \{\mathbf{r}\}} \{f_{jl}Q_{jl}(t+1) + \mathbb{E}_{t+1}[\varphi(t, \mathbf{b} - A_jY_{jl}^{t+1})]\} + \mathbb{E}_{t+1}[L_B(t+1, b)]. \quad (5.15)$$

Let us define $L_B(t+1, b, j, l) \coloneqq \varphi(t+1, \mathbf{b}) - f_{jl}Q_{jl}(t+1) - \mathbb{E}_{t+1}[\varphi(t, \mathbf{b} - A_jY_{jl}^{t+1})]$, which corresponds to the loss in Eq. (5.15) when we assume a specific price f_{jl} is posted.

Lemma 5.5.2. *If the arrival is of type j, X solves $P[\mathbf{b}, Q(t+1), Z(t+1)]$ and $X_{jl} \geq 1$, then denoting $Y = Y^{t+1}$, we can write the loss in the Bellman inequality as*

$$L_B(t+1, \mathbf{b}, j, l) = P[\mathbf{b} - Q_{jl}(t+1), Q(t+1), Z(t)] - \mathbb{E}_{t+1}[P[\mathbf{b} - A_jY_{jl}, Q(t), Z(t)]] \quad (5.16)$$

Proof. By our definition,

$$\begin{aligned} L_B(t+1, b, j, l) &= \varphi(t+1, \mathbf{b}) - f_{jl}Q_{jl}(t+1) - \mathbb{E}_{t+1}[\varphi(t, \mathbf{b} - Y_{jl})] \\ &= P[\mathbf{b}, Q(t+1), Z(t+1)] - f_{jl}Q_{jl}(t+1) - \mathbb{E}_{t+1}[P[\mathbf{b} - A_jY_{jl}, Q(t), Z(t)]]. \end{aligned}$$

Using Lemma 5.3.5, we have $P[\mathbf{b}, Q(t+1), Z(t+1)] = f_{jl}Q_{jl}(t+1) + P[\mathbf{b} - A_jQ_{jl}(t+1), Q(t+1), Z(t)]$, which finishes the proof. □

Observe that $L_B(t, \mathbf{b}, j, l)$ is characterized by a random LP that depends on Y^{t+1} (which is unknown at time $t+1$), see Eq. (5.16). To complete item (2) of Proposition 5.5.1, it

remains to prove that $L_B(t, \mathbf{b}, j, l)$ characterized in (5.16) satisfies $\mathbb{E}_t[L_B(t, \mathbf{b}, j, l)] \leq 0$. This is proved in Appendix D.2 by arguing that the term in (5.16) is upper bounded by a zero-mean random variable.

We can then conclude that, for each l with $X_{jl} \geq 1$, $\mathbb{E}_{t+1}[L_B(t+1, b, j, l)] \leq 0$ so that $\varphi(t+1, \mathbf{b}) \leq \mathbb{E}_{t+1}[f_{jl} Q_{jl}(t+1) + \varphi(t, \mathbf{b} - A_j Y_{jl}^{t+1})]$, implying that posting price f_{jl} is a satisfying action, which is item (3) of Proposition 5.5.1.

5.5.4 Information Loss and Overall Performance Guarantee

We have already established in Proposition 5.5.1 that, if $X_{jl} \geq 1$, then posting f_{jl} is a satisfying action. We will now study the measure of the disagreement sets $\mathcal{Q}(t, B^t)$ and specifically bound the information loss $\sum_t \mathbb{P}[\mathcal{Q}(t, B^t)]$.

Proposition 5.5.3. *Let X be a solution of $(P[\mathbf{b}, Q(t), Z(t)])$. If $X_{jl} \geq 1$, then posting f_{jl} is a satisfying action. Furthermore, the information loss is bounded by $\mathbb{P}[\mathcal{Q}(t, B^t)] \leq 1/t^2$ for all $t \geq c$, where c depends only on $(\mathbf{f}, \mathbf{p}, A, F_1, \ldots, F_n)$.*

In the remainder of this subsection we first give intuition for Proposition 5.5.3 and then elicit the main elements needed for the proof. Recall that RABBI chooses l as the maximum entry of the solution to $(P[\mathbf{b}, \mathbb{E}[Q(t)], \mathbb{E}[Z(t)])$, which is a perturbed version of the object of interest, thus ONLINE needs to guess l such that $X_{jl} \geq 1$ without the knowledge of $Q(t)$ and $Z(t)$, creating an information loss.

To build intuition, we now give a characterisation of the solution of the one-dimensional problem, i.e., $d = 1$ and $n = 1$, which correspond to selling multiple copies of an item to homogeneous customers; since there is only one type, we drop the index j. See Fig. 5.4 for an illustration.

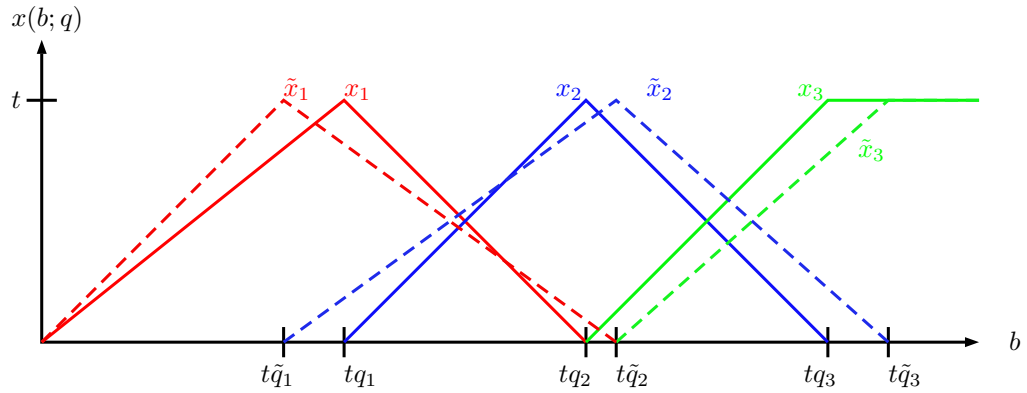


Figure 5.4: Solution to the pricing LP in Eq. (5.14) for the case $d = 1$ and $n = 1$, which correspond to selling multiple copies of an item to homogeneous customers. If $b/t \in (q_l, q_{l+1}]$, the prices used by the LP are f_l, f_{l+1} and the amount of time we offer each is piece-wise linear in the budget. For a perturbation $\tilde{\mathbf{q}}$ of $\mathbf{q}$, we superpose the solutions with these different parameters. We would guess incorrectly when $\tilde{x}_l >> 1$ and $x_l < 1$, which necessitates a substantial perturbation of $\mathbf{q}$.

Lemma 5.5.4. *Consider the case $d = 1$ and $n = 1$. For given $(t, b, \mathbf{q})$, with $f_1 > \ldots > f_m$ and $q_1 < \ldots < q_m$, the solution of $P[b, \mathbf{q}, t]$ is as follows. (i) If $b \leq tq_1$, then $x = (b/q_1, 0, \ldots, 0)$. (ii) If $b > tq_m$, then $x = (0, \ldots, 0, t)$*

The proof of Lemma 5.5.4 is straightforward. If b is neither too big nor too small, say $b \in (tq_l, tq_{l+1}]$, then $x_{l'} = 0$ for $l' \neq l, l+1$, and

$$x_l = \frac{tq_{l+1} - b}{q_{l+1} - q_l} \quad \text{and} \quad x_{l+1} = \frac{b - tq_l}{q_{l+1} - q_l}.$$

As illustrated in Fig. 5.4, the intuition is that $Q(t)$ and $\mathbb{E}[Q(t)]$ must deviate considerably for RABBI's guess to be incorrect. This intuition carries over to multiple dimensions.

The next concentration result is a direct application of the DKW inequality [87] and characterises the deviations of Q w.r.t. its mean.

Lemma 5.5.5. *For any $j \in [n]$, there is a constant c_j depending on p_j only such that, for any time t, $\mathbb{P}\left[\max_l |Q_{jl}(t) - \mathbb{E}[Q_{jl}(t)]| > \sqrt{\frac{\log(t)}{t}}\right] \leq \frac{c_j}{t^2}$.*

PROOF. Since $Q_{jl}(t)$ is the empirical distribution of $Z_j(t)$ draws from F_j, the DKW inequality [87] states that

$$\mathbb{P}\left[\sup_l |Q_{jl}(t) - \bar{F}(f_{jl})| > \lambda \middle| Z(t)\right] \leq 2e^{-2\lambda^2 Z_j(t)}.$$

This corresponds to the moment generating function of $Z_j(t) \sim \text{Bin}(t, p_j)$. Setting $\lambda = \sqrt{\log(t)/t}$ and using the formula $\mathbb{E}[e^{-\theta \,\text{Bin}(t,p)}] = (1 - p + pe^{-\theta})^t$ we obtain

$$\mathbb{P}\left[\sup_l |Q_{jl}(t) - \bar{F}(f_{jl})| > \sqrt{\frac{\log(t)}{t}}\right] \leq 2(1 - p_j + p_j e^{-\theta})^t \quad \text{where } \theta = 2\log(t)/t.$$

Using the inequality $e^{-\theta} \leq 1 - \theta + \theta^2/2$, an algebraic check confirms the desired inequality. □

Stability of Left-Hand Side Perturbations. As stated in Algorithm 8, ONLINE takes actions based on $P[\mathbf{b}, \mathbb{E}[Q(t)], \mathbb{E}[Z(t)]]$, while OFFLINE uses $P[\mathbf{b}, Q(t), Z(t)]$. Therefore, for fixed $(t, \mathbf{b})$, we need to compare solutions of $P[\mathbf{b}, \mathbf{q}, \mathbf{z}]$ to those of $P[\mathbf{b}, \mathbf{q} + \Delta\mathbf{q}, \mathbf{z} + \Delta\mathbf{z}]$, where Δ is the perturbation. For the remainder of this subsection, we set $\mathbf{q} = \mathbb{E}[Q(t)]$, $\mathbf{z} = \mathbb{E}[Z(t)]$, $\Delta\mathbf{q} = Q(t) - \mathbb{E}[Q(t)]$, and $\Delta\mathbf{z} = Z(t) - \mathbb{E}[Z(t)]$.

Lemma 5.5.6 (Selection Program). *Let $V_t = P[\mathbf{b}, \mathbf{q} + \Delta\mathbf{q}, \mathbf{z} + \Delta\mathbf{z}]$ and fix a component (j', l'). Then posting price $f_{j'l'}$ is satisfying if $P_S[V_t, \mathbf{q} + \Delta\mathbf{q}, \mathbf{z} + \Delta\mathbf{z}] \geq 1$, where*

$$(P_S[V_t, \mathbf{q} + \Delta\mathbf{q}, \mathbf{z} + \Delta\mathbf{z}]) \ \max\left\{x_{j'l'} : \sum_{j,l} f_{jl}(q_{jl} + \Delta q_{jl}) x_{jl} \geq V_t, \mathbf{x} \text{ feasible for } P[\mathbf{b}, \mathbf{q} + \Delta\mathbf{q}, \mathbf{z} + \Delta\mathbf{z}]\right\}.$$

In other words, $\mathcal{Q}(t, b, l) = \{\omega \in \Omega : P_S[V_t[\omega], Q(t), Z(t)] < 1\}$.

PROOF. This problem selects, among all the solutions of $P[\mathbf{b}, \mathbf{q} + \Delta\mathbf{q}, \mathbf{z} + \Delta\mathbf{z}]$, one with the largest component $X_{j'l'}$. From Proposition 5.5.1 we know that, if $X_{j'l'} \geq 1$, then posting $f_{j'l'}$ is satisfying. □

We turned the condition "there exists X solving $P[v, \mathbf{q} + \Delta\mathbf{q}, \mathbf{z} + \Delta\mathbf{z}]$ with $X_{j'l'} \geq 1$" to an optimization program. Let $\bar{\mathbf{x}}$ be the solution to the proxy $P[\mathbf{b}, \mathbf{q}, \mathbf{z}]$ and let v_t be the objective

value (recall that V_t is the value of $P[\mathbf{b}, \mathbf{q}+\Delta\mathbf{q}, \mathbf{z}+\Delta\mathbf{z}]$). Since the algorithm picks the price with the largest component, assume $\bar{x}_{j'l'} = \max_l \bar{x}_{j'l} >> 1$. In particular, $P_S[v_t, \mathbf{q}, \mathbf{z}] >> 1$ for this fixed (j', l'). We want to show that $P_S[V_t, \mathbf{q}+\Delta\mathbf{q}, \mathbf{z}+\Delta\mathbf{z}] \geq 1$ for that particular (j', l'). To that end, we need to bound the difference between $P_S[V_t, \mathbf{q}+\Delta\mathbf{q}, \mathbf{z}+\Delta\mathbf{z}]$ and $P_S[v_t, \mathbf{q}, \mathbf{z}]$. This difference depends on (i) $v_t - V_t$, (ii) Δ, and (iii) the dual variables of $(P_S[V_t, \mathbf{q}+\Delta\mathbf{q}, \mathbf{z}+\Delta\mathbf{z}])$. Observe that the quantities (i)-(iii) are random.

In Lemma 5.5.7 below we state the precise stability result, we relegate the proof to Appendix D.2.

Lemma 5.5.7. *There is a constant c that depends only on $(\mathbf{f}, \mathbf{p}, A, F_1, \ldots, F_n)$ such that, for all $t \geq c$, with probability $1 - c/t^2$, $P_S[V_t, Q(t), Z(t)] - P_S[v_t, \mathbb{E}[Q(t)], \mathbb{E}[Z(t)]] \geq -c\sqrt{t\log(t)}$*

.

Lemma 5.5.7 readily proves our bound for the information loss in Proposition 5.5.3. Indeed, since the LP in Eq. (5.14) has the constraint $\sum_{l\in[m]\cup\{\mathtt{r}\}} \bar{x}_{jl} = tp_j$, the maximum entry is guaranteed to have a value of at least $tp_j/(m+1)$. Therefore, by definition of the selection program, $P_S[v_t, \mathbb{E}[Q(t)], \mathbb{E}[Z(t)]] \geq tp_j/(m+1)$. We know that posting $f_{jl'}$ is satisfying whenever $P_S[V_t, Q(t), Z(t)] \geq 1$ (see Lemma 5.5.6), hence posting the maximum entry is satisfying provided that $tp_j/(m+1) - c\sqrt{t\log(t)} \geq 1$, which holds for all t large enough.

5.5.5 Numerical Demonstration

We test our algorithm on two systems, henceforth the "small system" and the "large system". For each system, we consider a sequence of instances with increasing horizons and initial inventories.

The small system corresponds to the one-dimensional problem ($n = 1$ and $d = 1$); in this case we can solve the DP for small enough horizons and directly compute the optimality gap. The large system corresponds to a multi-dimensional problem with $n = 20$, $d = 25$ and $m = 3$. The DP solution is intractable for the large system, yet we can compute the offline benchmark and compare our algorithm against it. The optimality gap, recall, is bounded by the offline vs. RABBI gap.

For the small system, the k-th instance has budget $B = 6k$ and horizon $T = 20k$. For each scaling k, we run $100,000$ simulations. We consider the following primitives: prices are $(1, 2, 3)$ and the private reward R^t has an atomic distribution on $(1, 2, 3)$ with probabilities $(0.3, 0.4, 0.3)$. The instance is chosen such that it is dual degenerate for (5.14) which is supposedly the more difficult case [77]. For large system, the parameters were generated randomly, the k-th instance has horizon $T = 100k$ and budgets $B_i = 10k$ for all $i \in [25]$.

For the small system we consider k small enough (short horizon) so that we can compute the optimal policy; this computation becomes intractable already for moderate values of k (RABBI however scales gracefully with k as it only requires re-solving an LP in each period). In Fig. 5.5 (LEFT) we display the gap between the optimal solution and both the RABBI and OFFLINE's value. We make two observations: (i) the OFFLINE benchmark outperforms the optimal (as it should), but by a rather small margin, and (ii) RABBI has a constant regret (i.e., independent of k) relative to OFFLINE, and hence constant optimality gap. In contrast, a full information benchmark would outperform the optimal by too much to be useful.

In Fig. 5.5 (RIGHT), we compare RABBI to the optimal static pricing policy which has regret $\Omega(\sqrt{k})$ [66]. In particular, if $D(f)$ denotes the demand at fare f, we choose the static price to be the one that maximizes the revenue function $f \cdot D(f) = f \cdot T \cdot \bar{F}(f)$ subject to the constraint $D(f) \leq B$. The solution is the better of two prices: (i) the market clearing price, i.e., that satisfies $D(f) = B$ or (ii) the monopoly price which maximizes $fD(f)$. We

note though that when a continuum of prices are allowed, [77] propose an algorithm (that, like RABBI, is based on resolving an optimization problem in each period) which achieves a regret that is logarithmic in k under certain non-degeneracy assumptions on the optimization problem and differentiability assumptions on the valuation distribution. In contrast, our constant regret guarantees hold under a finite price menu.

In Figure 5.6 we display the results for the large system. Here, recall, since the DP is intractable, we use the offline benchmark. Evidently, the regret is constant and bounded. This regret is negligible relative to the total value as captured by the approximation-factor on the right-hand side of the figure. We also present the competitive ratio of OFFLINE against the full-information benchmark (this upper bounds the competitive ratio of *any non-anticipatory policy*) and observe that is bounded away from 1, hence showing that the full-information benchmark is $\Omega(T)$ away from the DP in our randomly generated instance, which confirms the need for our refined benchmark.

5.6 Online Knapsack With Distribution Learning

We study first the full feedback setting and in Section 5.6.5 extend to censored feedback. As in the baseline `OnlineKnapsack`, at each time t, the arrival is of type $j \in [n]$ with known probability p_j. Type j has a random reward R_j, drawn from a distribution F_j that is *unknown* to ONLINE, and a known weight w_j. The reward R_j is revealed only after the decision of accept/reject has been made. At the end of each period, we observe the realization of both accepted and rejected items. In contrast, OFFLINE has access to the distribution F_j, *but not to the realizations*. Let $r_j := \mathbb{E}[R_j]$ and R_j^t be the empirical average of the type-j rewards observed with t periods to go. We assume that, before the process starts, we are given one sample of each type. *There is no probing or pricing in this setting.* The variation relative to

the baseline `OnlineKnapsack` is in terms of what is unknown to the decision maker.

5.6.1 Learning RABBI

The main object for the algorithm is the following LP parametrized by $(b, \mathbf{y}, \mathbf{z}) \in \mathbb{R}_{\geq 0} \times \mathbb{R}^n \times \mathbb{R}^n_{\geq 0}$

$$\begin{array}{lllll} (P[b,\mathbf{y},\mathbf{z}]) & \max & \sum_j y_j x_j & & \\ & \text{s.t.} & \sum_j w_j x_{j\mathtt{a}} & \leq b & \\ & & x_{j\mathtt{a}} + x_{j\mathtt{r}} & = z_j & j \in [n] \\ & & x_{j\mathtt{a}}, x_{j\mathtt{r}} & \geq 0 & j \in [n]. \end{array} \tag{5.17}$$

If we knew the average rewards $\mathbf{r}$, then setting $\mathbf{y} = \mathbf{r}$ we obtain the same LP we used for the baseline case. Hence, we interpret $\mathbf{y}$ as our best guess for the unknown rewards $\mathbf{r}$. Otherwise, the interpretation is the same as in Section 5.2.2. Recall that R^t are the empirical averages, hence the instantiation of RABBI in Algorithm 10 uses $\mathbf{y} = R^t$ as the natural estimator.

Algorithm 9 Learning RABBI

Input: Access to solutions of $(P[b, \mathbf{y}, \mathbf{z}])$
Output: Sequence of decisions for ONLINE.

1: Set $B^T \leftarrow B$ as the given initial state and R^T as the single sample of each j.
2: **for** $t \in \{T, T-1, \ldots, 1\}$ **do**
3: Compute X^t, an optimal solution to $(P[B^t, R^t, \mathbb{E}[Z(t)]])$.
4: Observe the arrival type (context), say $\xi^t = j$, and take any action $\hat{U}^t \in \operatorname{argmax}_{u=\mathtt{a},\mathtt{r}}\{X^t_{ju}\}$
5: If $\hat{U}^t = \mathtt{a}$, collect random reward R_j and reduce the budget $B^{t-1} \leftarrow B^t - w_j$. Else, $B^{t-1} \leftarrow B^t$.
6: Update empirical averages R^{t-1} based on R^t and the observation R_j.

Intuition behind Offline. The LP in Eq. (5.17) is a knapsack problem. Given the true average rewards $\mathbf{r}$, it is known that the solution is to sort items by "bang for the buck" ratios r_j/w_j and take them greedily. In other words, the solution can be computed if we knew the ranking induced by $\mathbf{r}$. OFFLINE is the controller that, besides the number of arrivals $Z_j(T)$

for each j, knows the ranking. On the other hand, ONLINE needs to learn the ranking from the samples.

5.6.2 Dynamic Programming Formulation

As in probing, it is convenient to divide each period $t \in \{T, T-1, \dots, 1\}$ into two stages, t and $t-1/2$. In the first stage (i.e., period t) the input reveals the type $j \in [n]$, and in second stages (i.e., period $t-1/2$) the reward is revealed. The random inputs are given by $\xi^t \in [n]$ and $\xi^{t-1/2} \in \mathbb{R}$ The state space is $\mathcal{S} = \mathbb{R}_{\geq 0} \times \{\varnothing, \mathtt{a}, \mathtt{r}\}$, where the first component is the remaining knapsack capacity (or hiring budget). At a first stage, given a state of the form $s = (b, \varnothing)$, we choose an action $u \in \{\mathtt{a}, \mathtt{r}\}$, discount the capacity if $u = \mathtt{a}$, and transition to a second-stage state with $\diamond = u$. At the second stage, the state is of the form $s = (b, \diamond)$, and we collect the reward only if $\diamond = \mathtt{a}$. The only feasible action at a second stage is $u = \varnothing$, which transitions to a state with the same budget. Formally, the rewards are $\mathcal{R}((b, \mathtt{a}), \xi^{t-1/2}, \varnothing) = \xi^{t-1/2}$ and $\mathcal{R}((b, \mathtt{r}), \xi^{t-1/2}, \varnothing) = 0$.

5.6.3 Offline Information, Relaxation, and Bellman Loss

We define OFFLINE through the filtration $\mathcal{G}_t = \sigma(\{\xi^t : t \in [T]\} \cup \{\xi^\tau : \tau \geq t\})$. This is a canonical filtration (see Definition 5.3.1) with variables $(G_\theta : \theta \in \Theta) = (\xi^t : t \in [T])$. Observe that the future rewards, corresponding to times $t-1/2$, are not revealed. The relaxation builds on the LP Eq. (5.17) and is defined as $\varphi(t, s|\mathcal{G}_t) = P[b, \mathbf{r}, Z(t)]$ for first stages and

$$\varphi(t-1/2, s|\mathcal{G}_t) = \begin{cases} P[b, \mathbf{r}, Z(t-1)] & \diamond = \mathtt{r} \\ \xi^{t-1/2} + P[b, \mathbf{r}, Z(t-1)] & \diamond = \mathtt{a}. \end{cases} \tag{5.18}$$

Lemma 5.6.1. *The relaxation φ defined in (5.18) satisfies the Bellman Inequalities with exclusion sets*

$$\mathcal{B}(t,b) = \{\omega \in \Omega : \nexists X \text{ solving } (P[b, \mathbf{r}, Z(t)]) \text{ s.t. } X_{\xi^t,a} \geq 1 \text{ or } X_{\xi^t,r} \geq 1\}.$$

Proof. The initial ordering in Definition 5.3.3 follows from an argument identical to that of Lemma 5.4.1. The monotonicity property follows from Proposition D.1.1. □

5.6.4 Information Loss and Overall Performance Guarantee

To complete the proof of Theorem 5.2.4, we first give an overview of the estimation process behind Algorithm 9. The relaxation relies on the knowledge of $\mathbf{r}$ (the true expectation) and $Z(t)$. The natural estimators are the empirical averages R^t and expectation $\mu(t) = \mathbb{E}[Z(t)]$, respectively. Specifically, we use maximizers X^t of $(P[b, R^t, \mu(t)])$ to "guess" those of $(P[b, \mathbf{r}, Z(t)])$.

The overall regret bound is $r_\varphi(\text{Reg}_1 + \text{Reg}_2)$, where Reg_1 and Reg_2 are two specific sources of error. When the estimators R^t of $\mathbf{r}$ are accurate enough, the error is Reg_1 and is attributed to the incorrect "guess" of a satisfying action, i.e., Reg_1 is an algorithmic regret. The second term, Reg_2, is the error that arises from insufficient accuracy of R^t, i.e., Reg_2 is the learning regret. The maximum loss satisfies $r_\varphi \leq \max_{j,i}\{w_i r_j / w_j - r_i\}$ and we can show that

$$\text{Reg}_1 \leq 2\sum_j \frac{(w_{\max}/w_j)^2}{p_j} \quad \text{and} \quad \text{Reg}_2 \leq 16\sum_j \frac{1}{p_j(w_j\delta)^2}.$$

In sum, the regret is bounded by $(\max_{j,i}\{w_i r_j/w_j - r_i\}) \cdot (2\sum_j \frac{(w_{\max}/w_j)^2}{p_j} + 16\sum_j \frac{1}{p_j(w_j\delta)^2})$.

Proof of Theorem 5.2.4. To apply Theorem 5.3.10, we first bound the measure of the exclusion sets $\mathcal{B}$ and the "disagreement" sets $\mathcal{Q}$. Recall that $\mathcal{B}(t,b)$ is given in Lemma 5.6.1

and $\mathcal{Q}(t,b)$ is the event where $\hat{U}^t$ is not a satisfying action.

Let $\sigma : [n] \to [n]$ be an ordering of $[n]$ w.r.t. the ratios $\bar{r}_j := \frac{r_j}{w_j}$ such that $\sigma_j = 1$ if j has the highest ratio. Similarly, let $\hat{\sigma}^t : [n] \to [n]$ be the ordering w.r.t. ratios $\bar{R}^t_j \coloneqq R^t_j / w_j$.

Call E^t the event $\mathcal{B}(t, B^t) \cup \mathcal{Q}(t, B^t)$, then

$$\mathbb{P}[E^t] = \mathbb{P}[E^t, \sigma = \hat{\sigma}^t] + \mathbb{P}[E^t, \sigma \neq \hat{\sigma}^t] \leq \mathbb{P}[E^t, \sigma = \hat{\sigma}^t] + \mathbb{P}[\sigma \neq \hat{\sigma}^t].$$

Let N^t_j be the number of type-j samples observed by the beginning of period t. By definition, since we are given a sample of each type before the process starts, we have $N^t_j = Z_j(T) - Z_j(t) + 1$. Since the reward distribution is sub-Gaussian, it satisfies the Chernoff bound [27]

$$\mathbb{P}[R^t_j - r_j \geq x | N^t_j], \mathbb{P}[R^t_j - r_j \leq x | N^t_j] \leq e^{-N^t_j x^2/2} \qquad \forall x \in \mathbb{R}, \tag{5.19}$$

A union bound relying on Eq. (5.19) gives that

$$\mathbb{P}[\sigma \neq \hat{\sigma}^t | \mathcal{F}_t] \leq \mathbb{P}[\exists j \text{ s.t. } |\bar{r}_j - \bar{R}^t_j| \geq \delta/2 | \mathcal{F}_t] \leq 2 \sum_j e^{-N^t_j (w_j \delta)^2/8}.$$

The variable N^t_j, recall, is the number of type-j samples observed by the beginning of period t, hence $N^t_j - 1$ is a $\mathrm{Bin}(T - t, p_j)$ random variable. It a known fact that, given $\theta > 0$, $\mathbb{E}[e^{-\theta \,\mathrm{Bin}(p,m)}] = (1 - p + pe^{-\theta})^m$, thus

$$\mathbb{P}[\sigma \neq \hat{\sigma}^t] = \mathbb{E}[\mathbb{P}[\sigma \neq \hat{\sigma}^t | \mathcal{F}_t]] \leq 2 \sum_j e^{-(w_j \delta)^2/8} (1 - p_j + p_j e^{-(w_j \delta)^2/8})^{T-t}.$$

Upper bounding by a geometric sum yields

$$\textsc{Reg}_2 \coloneqq \sum_t \mathbb{P}[\sigma \neq \hat{\sigma}^t] \leq 2 \sum_j \frac{1}{p_j (e^{(w_j \delta)^2/8} - 1)} \leq 2 \sum_j \frac{8}{p_j (w_j \delta)^2}. \tag{5.20}$$

We are left to bound $\mathbb{P}[E^t, \sigma = \hat{\sigma}^t]$. Let us assume w.l.o.g. that the indexes are ordered so that $\bar{r}_1 \geq \bar{r}_2 \geq \ldots \geq \bar{r}_n$. The optimal solution of $(P[B^t, \mathbf{r}, Z(t)])$, i.e., Offline's problem,

is to sort the items and accept starting from $j = 1$, without exceeding the capacity B^t or the number of arrivals $Z_j(t)$. Mathematically, the optimal solution $X^{\star t}$ to $(P[B^t, \mathbf{r}, Z(t)])$ is

$$X^{\star t}_{1\mathtt{a}} = \min\left\{Z_1(t), \frac{B^t}{w_1}\right\}, \quad X^{\star t}_{j\mathtt{a}} = \min\left\{Z_j(t), \frac{B^t - \sum_{i<j} w_i X^{\star t}_{i\mathtt{a}}}{w_j}\right\} \quad j = 2, \dots, n.$$

For the proxy $(P[B^t, R^t, \mu(t)])$, the optimal solution has the same structure with $Z_j(t)$ replaced everywhere by $\mu_j(t)$.

Let $\xi^t = j$ and U be any action in $\operatorname{argmax}\{X^t_{j,u} : u = \mathtt{a}, \mathtt{r}\}$. We study first the case $U = \mathtt{a}$. If $X^{\star t}_{j,\mathtt{a}} \geq 1$ then $U = \mathtt{a}$ would be, by Lemma 5.6.1, a satisfying action. If it is not a satisfying action it must then be that $X^{\star t}_{j,\mathtt{a}} < 1$ and since the algorithm chooses to accept it must be also that $X^t_{j,\mathtt{a}} \geq \mu_j(t)/2$. Thus we obtain the following two conditions

$$X^{\star t}_{j,\mathtt{a}} < 1 \Rightarrow \sum_{i<j} w_i Z_i(t) \geq b \quad \text{and} \quad X^t_{j,\mathtt{a}} \geq \mu_j(t)/2 \Rightarrow \sum_{i<j} w_i \mu_i(t) + w_j \mu_j(t)/2 \leq b.$$

In the case $U = \mathtt{r}$, $X^{\star t}_{j,\mathtt{r}} < 1$ and $X^t_{j,\mathtt{r}} \geq \mu_j(t)/2$ imply

$$\sum_{i\leq j} w_i Z_i(t) \leq b \quad \text{and} \quad \sum_{i<j} w_i \mu_i(t) + w_j \mu_j(t)/2 \geq b.$$

In conclusion,

$$\mathbb{P}[E^t, \sigma = \hat{\sigma}^t] \leq \max\left\{\mathbb{P}\left[\sum_{i\leq j} w_i(Z_i(t) - \mu_i(t)) \geq \frac{w_j \mu_j(t)}{2}\right], \mathbb{P}\left[\sum_{i\leq j} w_i(Z_i(t) - \mu_i(t)) \leq -\frac{w_j \mu_j(t)}{2}\right]\right\}.$$

These probabilities are bounded symmetrically using the method of averaged bounded differences [54, Theorem 5.3]. Indeed, using the natural linear function $f(\xi^1, \dots, \xi^t) = \sum_i w_i \sum_{l=1}^t \mathbb{1}_{\{\xi^l = i\}}$, the differences are bounded by $|\mathbb{E}[f|\mathcal{F}_l] - \mathbb{E}[f|\mathcal{F}_{l-1}]| \leq w_{\max}$, hence

$$\textsc{Reg}_1 := \sum_t \mathbb{P}[E^t, \sigma = \hat{\sigma}^t] \leq \sum_t \sum_j p_j \exp\left(-\frac{2(w_j \mu_j(t)/2)^2}{t w_{\max}^2}\right) \leq 2 \sum_j \frac{(w_{\max}/w_j)^2}{p_j}.$$

Together with Eq. (5.20), we have the desired bound. □

Remark 5.6.2 (non-i.i.d arrival processes). We used the i.i.d. arrival structure to bound two quantities in the proof of Theorem 5.2.4: (1) $\mathbb{P}[||Z(t) - \mathbb{E}[Z(t)]|| \geq c\mathbb{E}[Z(t)]]$ and (2) $\mathbb{E}[e^{-cN_j^t}]$, where, recall, N_j^t is the number of type-j observations. The result holds for other arrival processes that admit these tail bounds.

5.6.5 Censored Feedback

We consider now the case where only accepted arrivals reveal their reward. We retain the assumption of Theorem 5.2.4 that there is a separation $\delta > 0$: $|\bar{r}_j - \bar{r}_{j'}| \geq \delta$ for all $j \neq j'$, where $\bar{r}_j = \mathbb{E}[R_j]/w_j$.

In the absence of full feedback, we will introduce a unified approach to obtaining the optimal regret (up to constant factors), that takes the learning method is a plug-in. The learning algorithm will decide between explore or exploit actions. Examples of learning algorithms, that also give bounds that are explicit in t, include modifications of UCB [109], ε-Greedy or simply to set apart some time for exploration (see Corollary 5.6.4 below).

Recall that $\sigma : [n] \to [n]$ is the ordering of $[n]$ w.r.t. the ratios $\bar{r}_j = r_j/w_j$ and $\hat{\sigma}^t : [n] \to [n]$ is the ordering w.r.t. ratios $\bar{R}_j^t = R_j^t/w_j$. The discrepancy $\mathbb{P}[\sigma \neq \hat{\sigma}^t]$ depends on the plug-in learning algorithm (henceforth BANDITS). BANDITS receives as inputs the current state S^t (remaining capacity), time, and the natural filtration $\mathcal{F}_t$. The output of BANDITS is an action in $\{\texttt{explore}, \texttt{exploit}\}$. If the action is `explore`, we accept the current arrival in order to gather information, otherwise we call our algorithm to decide, as summarized in Algorithm 10. Note that $\mathcal{F}_t$ has information only on the observed rewards, i.e., accepted items.

Theorem 5.6.3. *Let* REG_1 *be the regret of Algorithm 9, as given in Theorem 5.2.4. Define the indicators* $\texttt{explore}_t, \texttt{exploit}_t$ *which denote the output of* BANDITS *at time* t*. The regret*

Algorithm 10 Bandits RABBI

Input: Access to BANDITS and Algorithm 9.
Output: Sequence of decisions for ONLINE.

Set S^T as the given initial state
for $t = T, \dots, 1$ **do**
    Observe input ξ^t and let $U \leftarrow \text{BANDITS}(T, t, S^t, \mathcal{F}_t)$.
    If $U =$ `explore`, accept the arrival
    If $U =$ `exploit`, take the action given by Algorithm 9
    Update state $S^{t-1} \leftarrow S^t - w_{\xi^t}$ if accept or $S^{t-1} \leftarrow S^t$ if reject.

of Algorithm 10 is at most $r_\varphi M$, where

$$M = \text{REG}_1 + \mathbb{E}\left[\sum_t \texttt{explore}_t\right] + \mathbb{E}\left[\sum_t \mathbb{P}[\sigma \neq \hat{\sigma}^t]\,\texttt{exploit}_t\right].$$

The expected regret of Algorithm 10 is thus bounded by the regret of Algorithm 9 in the full feedback setting, plus a quantity controlled by BANDITS. In the periods where BANDITS says `explore` (which, in particular, implies accepting the item), the decision might be the wrong one (i.e., different than OFFLINE's). We upper bound this by the number of exploration periods. This is the second term in M. The decision might also be wrong if BANDITS says `exploit` (in which case we call Algorithm 9), but the (learned) ranking at time t, $\hat{\sigma}^t$, is different than σ^t. This is the last term in M. Finally, even if the learned ranking is correct, `exploit` can lead to the wrong "guess" by Algorithm 9 because the arrival process is uncertain. This is the first term in M.

Corollary 5.6.4 uses a naive BANDITS which explores until obtaining $\Omega(\log T)$ samples and achieves the optimal (i.e., logarithmic) regret scaling. The constants may be better depending on the BANDITS module we plug into our general algorithm. Any such algorithm has the guarantee given by Theorem 5.6.3. With the naive BANDITS, the bound easily follows from a generalization of the coupon collector problem [99].

Corollary 5.6.4. *If we first obtain $\frac{8}{(w_j \delta)^2} \log T$ samples of every type j, then we can obtain $O(\log T)$ regret, which is optimal up to constant factors.*

5.7 Discussion

We developed a framework that provides rigorous support to the use of simple optimization problems as a basis for online re-solving algorithms. The framework is based on comparing ONLINE to a carefully chosen offline benchmark. The (often intuitive) optimization problem that guides the online algorithm is an outcome of two approximation steps: (i) an approximation for OFFLINE's value function and (ii) a projection thereof to ONLINE's smaller information set that produces the optimization problem guiding the online algorithm. The *translation* (or interpretation) of the solution to this optimization problem into actions is strongly grounded in OFFLINE's approximate value function.

The regret bounds follow from our use of Bellman Inequalities and a useful distinction between Bellman Loss and Information Loss. As is often the case in approximate dynamic programming, the identification of a function φ satisfying the Bellman Inequalities requires some ad-hoc creativity but, as our example illustrate, is often rather intuitive. In Appendix D.1 we provide sufficient conditions, applicable to cases where φ has a natural linear representation, to verify the Bellman inequalities. These conditions are intuitive and likely to hold for a variety of resource allocation problems. Importantly, once such a function is identified, RABBI provides a way of obtaining online policies from φ and mathematical results that produce upper bound on (i) the offline approximation gap and (ii) the online optimality gap.

The `OnlineKnapsack` with probing is an instance of the larger family of two-stage decision problems wherein there is an inherent trade-off between refined information and the cost of obtaining it. `OnlineKnapsack` with pricing is a well-studied problem and it is representative of settings where rewards and transitions are random. Our solution to the `OnlineKnapsack` with learning showcases a separation of the underlying combinatorial problem from the parameter estimation problem.

It is our hope that this structured framework will be useful in developing online algorithms for other problems, whether these are extensions of those we studied here or completely different.

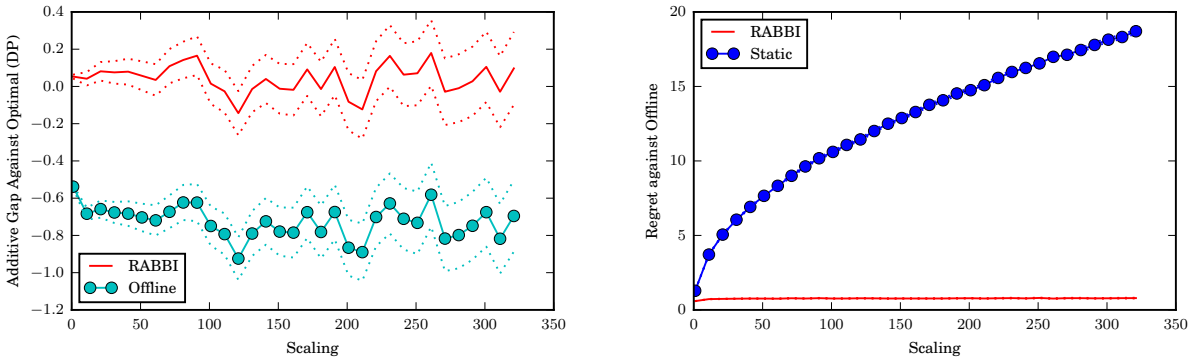


Figure 5.5: Regret of the small system ($n = 1$ and $d = 1$) for different scaling k, i.e., the horizon is $T = 20k$ and initial budget $B = 6k$, where $k = 1, 10, 20, \ldots, 340$. Dotted lines represent 90% confidence intervals. (LEFT) Gap against the optimal policy, i.e., $V^{DP} - V^{\text{RABBI}}$ and $V^{DP} - V^{\text{OFFLINE}}$ (RIGHT) Regret of two pricing policies against OFFLINE.

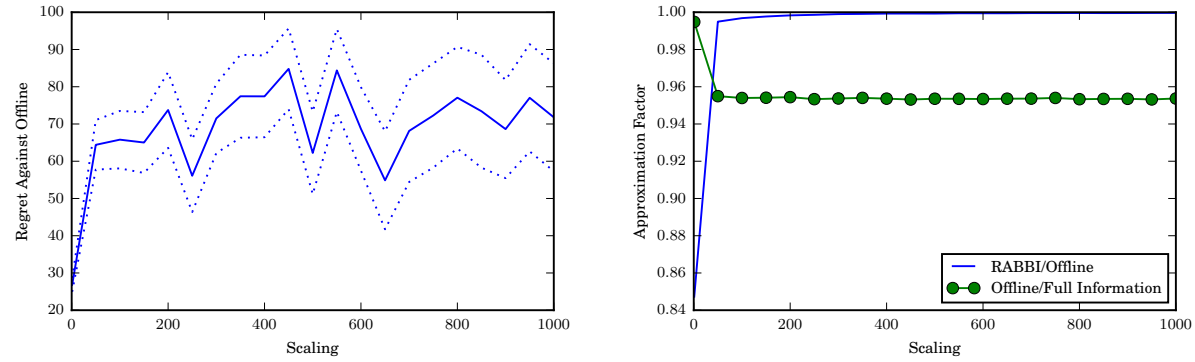


Figure 5.6: Performance in the large system ($n = 20$ and $d = 25$) for different scaling k, i.e., the horizon is $T = 100k$ and initial budget $B_i = 10k$ for $i \in [25]$. (LEFT) Regret against OFFLINE, i.e., $V^{\text{OFFLINE}} - V^{\text{RABBI}}$. Dotted lines represent 90% confidence intervals. (RIGHT) Approximation factor of RABBI against the DP captured by $V^{\text{RABBI}}/V^{\text{OFFLINE}}$ and approximation factor of OFFLINE against the full-information benchmark $V^{\text{OFFLINE}}/V^{\text{Full-Info}}$, which upper bounds the competitive ratio of *any non-anticipatory policy*, hence showing that the full-information benchmark is indeed too loose, as it is $\Omega(T)$ away from the DP.

CHAPTER 6

USING HISTORICAL DATA

6.1 Introduction

Online decision-making problems arise in many domains, and it is fundamental to understand the role of historical data in such problems, and how to use it for designing algorithms. In this work, we study this question in the context of online resource allocation problems. We focus on a general setting where a controller faces a stream of requests of various types over a finite horizon T, and must decide dynamically how to allocate resources, while collecting a reward associated with the request's type. There is a finite initial stock (inventory) of resources, which is depleted as requests are accepted. Many well-studied problems fit in this description (see Section 6.2.2 for details).

The main source of uncertainty in online resource-allocation is in the request-arrival process, and different approaches to online decision-making make different assumptions about how this uncertainty is realized. Moreover, these problems are typically not just run once, but repeatedly over many length-T *episodes*, and at any time, a controller typically has data of arrivals from past episodes, as well as of past arrivals in the current episode. To distinguish between these, we henceforth refer to data from previous episodes as *historical traces*, and data from the current episode as the *online trace*.

In practical settings, arrival processes may vary across different episodes (due to weekly/seasonal variations, etc.). What is often true, however, is that the controller has *some historical traces which are representative of the current episode*. For example, a network router can use request traces over the last hour; smart-grid operators can use electricity demand from the same hour yesterday; a rideshare platform can use ride requests from the

same day in the previous week; hotels and airlines can use reservations from last year's holiday period to model this year's holidays. One can model this by assuming arrivals in an episode are drawn from some unknown underlying process, and *the controller is given a single (or few) historical trace drawn from the same process.* How can we use this to design good policies?

Existing methods for online decision-making take two broad approaches for dealing with uncertainty: (*i*) Bayesian approaches posit an underlying stochastic model for the uncertainty, and (*ii*) worst-case approaches view decision-making as a game against an adversary generating the uncertainty. In terms of data dependence, these approaches lie at two ends of a spectrum. Bayesian approaches like approximate dynamic programming (ADP) and reinforcement learning (RL) require extensive historical traces to learn good policies; worst-case approaches like online learning and competitive analysis only use the online trace to guide decisions. A natural question is if there are variants of these approaches that can leverage the problem structure to find good policies using only a few historical traces.

In this paper, we propose a novel policy that is able to leverage few traces to get strong performance guarantees in theory and practice. Our algorithm builds on the framework introduced in Chapter 5, which uses a sequence of offline optimization programs to get constant regret algorithms for online resource allocation. Surprisingly, we show that we only need *a single trace of historical data to obtain constant regret* (i.e., independent of the size of the state-space and length of horizon T) in many problem settings, and under a wide class of processes including non-stationary and correlated processes.

In adapting the algorithm from Chapter 5 to work with traces, we bring out a natural robustness property of this paradigm relative to other ADP approaches. Informally, the core idea behind our proposed algorithm is that, by re-solving offline relaxations, a controller can aggregate uncertainty in a way that reduces sensitivity to estimation errors. To highlight this,

we show that we outperform common ADP approaches. In particular, standard approaches to data-driven ADP are based on first estimating the parameters of the arrival model (or more directly, the Q-functions for each state [85]) and then running an ADP algorithm with the estimated model as input. A hurdle in this approach is how to handle ties, i.e., when multiple actions look equally good relative to the (estimated) value function. We show that the tie-breaking rule is crucial for good performance, and standard approaches, in particular randomized rules, lead to sub-optimal performance (cf. Fig. 6.1). Our algorithm provides one such family of data-driven tie-breaking rules that ensures good performance.

Finally, the intuition behind our theoretical bounds also informs the design of algorithms for problems that lie beyond the theory. We illustrate this in the context of reusable resources (e.g. inventory control, allocation of cloud resources and hotel rooms, etc.) where it is known that no online algorithm can have constant regret. Our framework generates a straightforward algorithm that is computationally efficient and showcases strong performance, beating other state-of-the-art specialized algorithms.

6.2 Setting and Overview of Results

We consider the following general online resource-allocation problem, which takes place over a finite horizon of T periods. We use T to denote the first period, and index time in terms of *periods to go*, i.e., $t \in \{T, T-1, \ldots, 1\}$.

We are given a set of d resource types, and an initial state $S^T \in \mathbb{N}^d$ comprising of an initial stock of S_i^T for each resource type $i \in [d]$. In each period, an incoming request $\xi^t \in [n]$ is generated from a set of n request types via an *exogenous stochastic process*. Faced with a request of type j, the controller can choose one from a set $\mathcal{U}$ of controls, with associated rewards r_{uj} for $u \in \mathcal{U}$ and $j \in [n]$. A control $u \in \mathcal{U}$ applied to a request type j depletes

resources $A_{uj} \in \mathbb{Z}^d$, i.e., if the current stock is S, then after applying u the levels change to $S - A_{uj}$.

The above problem setting encapsulates many important Markov decision processes (MDPs) (cf. Section 6.2.2). A critical aspect of all settings we study is that the randomness arises via an exogenous input process $(\xi^t : t \in [T])$, which is independent of the state and actions of the controller. Note that If the controller has full knowledge of the underlying generative process, then it can use this to find optimal control policies via ADP techniques.

Our work instead assumes that the controller's information about the process is only *a limited number of historical traces*. Formally, we assume the controller is provided with K historical traces $\mathcal{H} = \{(\widehat{\xi}^{t,k} : t \in [T]) : k \in [K]\}$ of the exogenous request process, which are sampled from the underlying generative model. We focus on the case of $K = 1$, i.e., where the controller has only a single historical trace. Using this, the controller makes real-time decisions on new incoming requests. Henceforth we consistently refer to $(\xi^t : t \in [T])$ as the *online trace* and $(\widehat{\xi}^{t,k} : t \in [T])$ as *historical traces*.

Unlike in MDPs, in the context of online optimization with historical traces there is no clear notion of an optimal policy. Nevertheless, a natural benchmark for any policy is the so-called *prophet* (or offline/hindsight) benchmark – the performance of a controller that has full information of the online trace. Obviously, the offline controller has no use for the historical traces as it sees the full online trace, and uses it to solve the following problem:

$$\begin{aligned} &\max_{\mathbf{x} \in \mathbb{R}^{[n]\times[T]\times\mathcal{U}}_{\geq 0}} && h(\mathbf{x}; \xi^T, \ldots, \xi^1) \\ &\text{s.t.} && g(\mathbf{x}; S^T, \xi^T, \ldots, \xi^1) \geq \mathbf{0}. \end{aligned} \tag{6.1}$$

In most problem of interest, the decision variable $x^t_{u,\xi}$ is binary and represents if the control u is used at time t when an input of type ξ is presented. We assume that the functions h and g, encoding the structure of the problem, are known. The following example,

also referred to as unit-weight knapsack and online k-uniform matroid, suffices to understand our main results (see Section 6.2.2 for additional problems).

Example 6.2.1 (Multi-Secretary). Job candidates arrive sequentially for the selection to one of $S^T \in \mathbb{N}$ vacant positions. The candidate arriving at time t has ability (reward) r_j with probability p_j^t, independent of other candidates. The controller must irrevocably accept ($\mathtt{a}$) or reject ($\mathtt{r}$) the candidate. The goal is to select S^T candidates so as to maximize the total reward.

In this setting, the offline problem can be described by the total number of accepted and rejected type-j arrivals, $y_{\mathtt{a},j}$ and $y_{\mathtt{r},j}$ respectively. Let Z_j^t denote the total number of type-j arrivals over the last t periods, and S^t denote the number of vacant slots at the start of period t. Now Eq. (6.1) (computed at time-to-go t) takes the following form:

$$\max_{\mathbf{y}\geq 0}\left\{\sum_{j\in[n]} r_j y_{\mathtt{a},j} : \sum_{j\in[n]} y_{\mathtt{a},j} \leq S^t, y_{\mathtt{a},j} + y_{\mathtt{r},j} = Z_j^t \,\forall\, j\right\}. \tag{6.2}$$

Our approach is to solve Eq. (6.2) with $\widehat{\xi}^{t:1}$ as input, i.e., obtain cumulative arrivals $\widehat{Z}^t$ from the historical traces, and use the solution $\{\widehat{y}_{a,j}, \widehat{y}_{r,j}\}$ as *scores* given to the actions accept and reject respectively. We take the action with the highest score in each period. This idea, which forms the basis of our proposed RABBI algorithm, leads to significant improvements in the regret, as we quantify both in terms of theoretical guarantees and simulations.

6.2.1 Overview of our Results

We present the performance guarantees of our algorithm, formally defined in Section 6.3, which is a data-integrated adaption of RABBI introduced in Chapter 5. We provide here high-level statements of the main results, their formal statements and analysis is in Section 6.4.

For a given online trace $\xi^{T:1}$, let $V^{\text{off}}(S^T, \xi^{T:1})$ denote the reward collected by the offline controller, i.e., the objective value of the optimization problem in Eq. (6.1). An algorithm ALG that uses the historical traces in $\mathcal{H} = (\widehat{\xi}^{T:1,k} : k \in [K])$, collects total reward $V^{\text{ALG}}(T, S^T, \xi^{T:1}, \mathcal{H})$, and incurs a *regret* defined as

$$\text{REG}^{\text{ALG}}(T, S^T) = \mathbb{E}\left[V^{\text{off}}(S^T, \xi^{T:1}) - V^{\text{ALG}}(T, S^T, \xi^{T:1}, \mathcal{H})\right],$$

where the expectation is taken w.r.t. the historical and online traces as well as ALG's internal randomness (if any).

Our main algorithmic contribution is a general data-driven policy for online resource allocation problems, which we refer to as the RABBI algorithm (presented in Section 6.3). The basic idea behind RABBI is to act greedily based on action-scores computed via Eq. (6.1) using the historical trace (see discussion in previous section on the multi-secretary problem). The following theorem encapsulates our main performance guarantee for this algorithm.

Theorem 6.2.2. *For a large class of online allocation problems (cf. Section 6.2.2), and stochastic arrival processes (cf. Section 6.2.3),* RABBI *achieves constant regret with* $K = 1$ *historical trace. In other words,* $\text{REG}^{\text{RABBI}}(T, S^T) \leq \rho$ *for some constant* ρ *independent of* T *and* S^T*.*

Notice that constant regret implies, in particular, an approximation ratio of $1 - \mathcal{O}(\frac{1}{V^{\text{off}}})$, where $V^{\text{off}} = \mathbb{E}[V^{\text{off}}(S^T, \xi^{T:1})]$; see for example Figure 6.1 in Section 6.5.

The importance of score-based decisions is highlighted when we consider the natural way to incorporate historical data into a Dynamic Program (DP). A standard approach is to estimate the parameters and subsequently run the DP. Formally, consider an independent time-varying arrival process: $\mathbb{P}[\xi^t = j] = p_j^t$, with $\mathbf{p} \in \mathbb{R}_{\geq 0}^{n \times T}$ unknown parameters. Let $V^t(S, j; \mathbf{p})$ represent the total value if there are t periods to go, the current input is $\xi^t = j$, the

stock levels are S, and the probabilities are $\mathbf{p}$, then the value function is updated according to:

$$V^t(S, j; \mathbf{p}) = \max_{u \in \mathcal{U}} \left\{ r_{uj} + \sum_{j' \in [n]} p_{j'}^{t-1} V^{t-1}(S - A_{uj}, j'; \mathbf{p}) \right\}.$$

With historical traces, the parameters $\mathbf{p}$ are estimated from historical data, hence with K traces DP uses the empirical averages $\widehat{p}_j^t = \frac{|\{k \in K : \widehat{\xi}^{t,k} = j\}|}{K}$.

When executing the policy that arises from the DP, there is a degree of freedom in determining how to break ties — i.e. how to choose between controls that are equally optimal relative to the *estimated* value function. That is, if there is a unique control that is optimal relative to $V^t(S, j; \widehat{\mathbf{p}})$, then that control must be taken. Conversely, if there are multiple such controls, one must choose between them.

Viewed in the above framework, RABBI can be used as follows: when DP encounters a tie, it calls a single re-solve of RABBI and uses its scores to choose among actions. We prove that DP, equipped with this tie-breaking rule, achieves constant regret for all the problems for which RABBI achieves constant regret. In contrast, other natural tie-breaking rules, in particular randomized tie-breaking, may have $\omega(1)$ regret, see Fig. 6.1.

Theorem 6.2.3. *For a large family of problems and independent time-varying processes,* DP *with the tie-breaking rule from* RABBI *achieves constant regret with $K = 1$ historical trace. In other words, there is a constant ρ independent of T and S^T such that $\mathrm{REG}^{\mathrm{DP}}(T, S^T) \leq \rho$.*

6.2.2 Application to Specific Problems

We now discuss several problems of practical interest that can be encoded via our general setting. The controls $\mathcal{U}$ correspond to allocation of resources and the specific dynamics give rise to different problems, whereas the following description is common to all problems.

Let us denote Z_j^t as the cumulative number of type-j requests in the last t periods, i.e., $Z_j^t \coloneqq \sum_{\tau=1}^{t} \mathbb{1}_{\{\xi^\tau = j\}}$. Additionally, if x_{uj}^t represents that the control u was used at t when an input type j was presented, then all the problems we consider satisfy the following natural conditions, which are interpreted as "some control is used at every time period":

$$\sum_{u \in \mathcal{U}} \sum_{\tau=1}^{t} x_{uj}^\tau = Z_j^t \quad \forall j \in [n], t \in [T]. \tag{6.3}$$

Settings Admitting Uniform Regret Guarantees The following represent important special cases of settings in which our main theorem, Theorem 6.2.2, guarantees that RABBI incurs $O(1)$ regret (in particular, they satisfy the requirements R1-R4 given in in Section 6.4).

1. Online Packing (Network Revenue Management): Each type $j \in [n]$ is associated with a vector $A_j \in \{0,1\}^d$ of resources and a reward $r_j \geq 0$. At each time, we must decide if we allocate all the resources requested (encoded in A_j) or if we reject the request. Hence, the control set has two actions, $\mathcal{U} = \{\mathtt{a}, \mathtt{r}\}$. The function g encodes the following constraint for all resources $i \in [d]$: $\sum_j A_{ij} \sum_{t=1}^{T} x_j^t \leq S_i^T$, meaning that the amount of allocated resources do not exceed the initial stock.
2. Online Matching: Each type $j \in [n]$ has associated a reward vector $r_j \in \mathbb{R}^d_{\geq 0}$. At each time, we must decide to which resource i we match the current arrival j, which would generate reward r_{ij}. The control set is $\mathcal{U} = [d]$. If $r_{ij} = 0$, we interpret that j cannot be assigned a resource type i. The function g encodes $\sum_j \sum_{t=1}^{T} x_{ij}^t \leq S_i^T$.
3. Generalized Assignment: each $i \in [d]$ represents a bin and, if we place a type j in bin i, it uses $a_{ij} \geq 0$ space while generating a reward of r_{ij}. The control set is again $\mathcal{U} = [d]$ and g encodes $\sum_j \sum_{t=1}^{T} a_{ij} x_{ij}^t \leq S_i^T$.
4. Online Probing: Packing problems with unknown random rewards, where the controller can probe up to S_0^T requests to reveal their true reward before deciding on a control. The available controls are 'accept', 'probe', and 'reject'. The functions h and g are again linear objectives and constraints (cf. Chapter 5 for details).

Settings we Study Numerically A common feature of the above settings is that actions are *exchangeable* across arrivals of the same type – the objective and constraints only depend on the number of times an action u is taken for arrival type j, and not when they were taken (this is encoded in requirement R3 in Section 6.4). In settings where the timing of actions matters, uniform regret guarantees may not be possible. Nevertheless, these settings are still captured by our formulation, and we show via simulations that RABBI has a superior performance even compared to state-of-the-art algorithms crafted specially for these problems. In particular, we consider the following *reusable resource* problems, where allocated resources are returned to the platform after some time.

1. Reusable Matching: As in online matching, type-j arrivals can be matched to at most one resource i for reward r_{ij}, but now the allocated resource is returned after l_j periods. For known distributions, recent work [97] presents a 0.5-approximation for this setting using ADP with linear basis functions.
2. Reusable Packing: Multidimensional packing problems, where resources are released after some time. To the best of our knowledge, there are no algorithms with provable guarantees for this setting, and it is unclear how to extend other algorithms for this case.

6.2.3 Stochastic Generative Models for Arrivals

We consider Markov modulated arrival processes, which capture independent time-varying processes as well as other correlated models. Let (M^t) be a Markov chain with state space V and initial distribution $\nu : V \to [0, 1]$. The arrival process is modulated by (M^t) as follows: at time t there is an arrival of type j with probability $p_j(M^t)$, where $p_j : V \to [0, 1]$, i.e., $\mathbb{P}[\xi^t = j|\mathcal{F}_t] = p_j(M^t)$. In summary, to completely specify the arrival process, we need the law of (M^t) as well as the functions $p_j(\cdot)$, both of which are assumed to be unknown. We also consider case where the Markov chain is hidden.

This covers the case of i.i.d arrivals; the modulation process has just one state ($|V = 1|$), i.e., $p_j(M^t) = p_j$. Additionally, it covers the case of time-varying independent arrivals if we define the chain's state-space be $V = [T]$ and endow it with deterministic transitions $\mathbb{P}[M^{t-1} = t - 1|M^t] = 1$.

6.2.4 Related Work

As we mention above, our work is closely related to ADP approaches for online allocation problems [65, 78, 10, 37], bandit approaches with iid arrivals from an unknown distribution [15], and worst-case models for online packing [36, 51], and more generally, online convex optimization [6, 8]. These methods do not naturally use traces, and hence guarantees are difficult to compare; nevertheless we numerically demonstrate that our algorithm performs much better than (i) state-of-the-art parametric ADP approaches with full model information, and (ii) worst-case models even under complex input processes.

Our work is closely related to the literature on *prophet inequalities* [80] and in particular, sample-based variants [12, 96]. These works however focus on worst-case distributions, and have poor performance for typical distributions. Our work builds on a more recent approach towards getting distribution-dependent prophet inequalities, and in particular, on the work of [10] for multi-secretary problems. The general algorithm we introduce here adapts the RABBI algorithm in Chapter 5 to incorporate historical traces.

6.3 The Data-Driven rabbi Algorithm

Recall the offline benchmark problem in Eq. (6.1):

$$\begin{aligned} &\max_{\mathbf{x}\in\mathbb{R}^{[n]\times[T]\times\mathcal{U}}_{\geq 0}} && h(\mathbf{x};\xi^T,\ldots,\xi^1) \\ &\text{s.t.} && g(\mathbf{x};S^T,\xi^T,\ldots,\xi^1)\geq \mathbf{0}. \end{aligned}$$

RABBI stipulates resolving, at each time t, a problem with estimated functions $\bar{h}^t \approx h, \bar{g}^t \approx g$ and updated state S^t. Formally, define $\hat{\varphi}(t,s)$ as the following program:

$$\begin{aligned} \hat{\varphi}(t,s) = \max_{\mathbf{x}\in\mathbb{R}^{[n]\times[T]\times\mathcal{U}}} \quad & \bar{h}^t(\mathbf{x};\xi^T,\ldots,\xi^1) \\ \text{s.t.} \quad & \bar{g}^t(\mathbf{x};s,\xi^T,\ldots,\xi^1)\geq \mathbf{0}, \end{aligned} \tag{6.4}$$

where $\bar{h}^t, \bar{g}^t$ are the empirical estimates given by $\bar{h}^t = \frac{\sum_{k\in[K]} h(\mathbf{x};\widehat{\xi}^{t:1,k})}{K}$ and $\bar{g}^t = \frac{\sum_{k\in[K]} g(\mathbf{x};S^t,\widehat{\xi}^{t:1,k})}{K}$.

Let $\widehat{\mathbf{x}}$ denote the optimal solution to Eq. (6.4), and define *score* $\widehat{y}^t_{u,j} = \sum_{\tau=1}^{t} \widehat{x}^{\tau}_{u,j}$ for each action u when faced with a type-j request. The main idea in RABBI is to re-compute these scores in each period, and act greedily with respect to them. The formal algorithm is as follows:

RABBI (Re-solve and Act Based on Bellman Inequalities)

Input: Access to historical traces $(\widehat{\xi}^{t,k} : t\in[T])$, $k\in[K]$.
Output: Sequence of decisions $\hat{U}^t$ for each $t\in[T]$.

1: Set S^T as the given initial state
2: **for** $t = T,\ldots,1$ **do**
3: Re-solve $\hat{\varphi}(t,S^t)$ as in Eq. (6.4)
4: Compute scores $\widehat{\mathbf{y}} = \{\widehat{y}^t_{u,j}\}_{u\in\mathcal{U},j\in[n]}$
5: Given ξ^t, choose action $\hat{U}^t$ with highest score $\widehat{y}^t_{u,\xi^t}$
6: Collect reward $r_{\hat{U}^t,\xi^t}$. Update $S^{t-1}\leftarrow S^t - A_{\hat{U}^t,\xi^t}$

6.4 Analysis

We now present formal regret guarantees for RABBI based on a single trace. Our results hold for any problem satisfying three properties that encapsulate (i) the structure of the optimization problem, (ii) the predictability of the stochastic process, and (iii) the interaction between the optimization problem and the stochastic process.

Our first requirement is that the offline problem is linear and 'well-posed', in that it captures underlying transitions.

R1 *Well-Posed Problem*: The reward h has the form $h(\mathbf{x}; \xi^{t:1}) = \sum_{j,u} x_{ju} r_{ju}$. The function g is linear, and includes the constraints $\sum_u x_{ju} = Z_j^t \, \forall \, j \in [n]$, i.e, an action must be selected for every arrival. For all $u \in \mathcal{U}, j \in [n]$, if control u is applied to type j, then the constraint g captures the resulting depletion of resources; formally, $g(\mathbf{e}_{u,j}^t; s, \xi^{t:1}) \leq g(\mathbf{0}; s - A_{uj}, \xi^{t:1})$.

Next, we require that the stochastic arrival process satisfies a concentration bound, which is a variant of standard bounds based on 'self-normalized exchangeable pairs'.

R2 *All Time Concentration of Arrivals*: $\exists \, \theta_1, \theta_2 > 0$, s.t.

$$\mathbb{P}[||Z^t - \widehat{Z}^t||_\infty \geq \widehat{Z}_k^t / c] \leq \theta_1 e^{-\frac{\theta_2 t}{c^2}}, \quad \forall k \in [n]. \tag{6.5}$$

Finally our third condition requires the constraints can be expressed as an additive function of the actions and arrivals.

R3 *Noise Interaction*: The constraint function satisfies $g(\mathbf{x}; s, \xi^{t:1}) = g_1(\mathbf{x}; s) + g_2(s; \xi^{t:1})$ and $||g_2(s; \xi^{t:1}) - g_2(s; \widehat{\xi}^{t:1})||_\infty \leq ||Z^t - \widehat{Z}^t||_\infty$. In other words, the noise interacts additively with the constraints and it is dominated by the demand.

Example 6.4.1. We show that the multi-secretary problem (Example 6.2.1) satisfies requirements R1 and R3. Recall that the problem has two sets of constraints, $\sum_j x_{\mathtt{a},j} \leq S^t$

(acceptances do not exceed the available positions) and $x_{\mathtt{a},j} \leq Z_j^t$ (acceptance of j does not exceed arrivals of j). R1 follows by inspection. R2 follows because the constraint that involves noise is $\mathbf{x}_\mathtt{a} \leq Z^t$, which is additive in the noise with $g_2(s; \xi^{t:1}) = Z^t$.

In what follows we use the constant κ that identifies the Lipshitz continuity of the underlying LP. Let the matrix $\bar{A}$ be the constraint matrix encoded by g. Then, we write $\kappa = \kappa(\bar{A})$ to be the constant such that for any two right-hand side values $\mathbf{b}^1, \mathbf{b}^2$, the set of optimal solutions $\mathcal{X}(v) := \text{argmax}_\mathbf{x}\{\mathbf{r}'\mathbf{x} : \bar{A}\mathbf{x} \leq \mathbf{b}\}$ satisfies $||\mathcal{X}(\mathbf{b}^1) - \mathcal{X}(\mathbf{b}^2)||_\infty \leq \kappa ||\mathbf{b}^1 - \mathbf{b}^2||_\infty$. The existence of this constant κ follows, e.g., from [84].

Theorem 6.4.2. *Suppose that the optimization problem (6.1) satisfies the three requirements listed above. Then, the regret of* RABBI *with a single trace is at most $dr_\varphi |\mathcal{U}|^2 \theta_1 \kappa^2 / \theta_2$, where $r_\varphi = \max_{j \in [n], u \in \mathcal{U}} r_{uj}$ is the maximum reward and d is the number of resource types.*

We complement our result by showing that a large family of stochastic processes satisfy Eq. (6.5).

Proposition 6.4.3. *For time-varying independent processes with $p_{jt} \geq \beta$ for all j, t, Eq. (6.5) is satisfied with $\theta_1 = 4n$ and $\theta_2 = \beta^2/12$, hence the regret is at most $\mathcal{O}(dr_\varphi n \kappa^2 / \beta^2)$.*

The next example emphasises that, if the arrival process is not independent, in general a single trace is insufficient. Motivated by this, we study in Propositions 6.4.5 and 6.4.7 particular forms of correlation.

Example 6.4.4. Consider the arrival process where, at time $t = T$, i.e., at the beginning of the horizon, a fair coin is flipped. With probability 1/2, we take ξ^t to be independent random variables with $\mathbb{P}[\xi^t = j] = p^t_{j,\text{head}}, j \in [n], t \in [T]$. With the remaining probability 1/2, $\mathbb{P}[\xi^t = j] = p^t_{j,\text{tail}}$. In this case, with probability 1/2 the online trace comes from a different distribution that the single historical trace, hence the latter contains no relevant information.

In the following two results, we assume that the underlying Markov chain has a stationary distribution $\pi : V \to [0,1]$. We denote M^t the online chain and $\widehat{M}^t$ the historical chain, modulating the online and historical trace, respectively. We give *parametric results* in terms of (i) the absolute spectral gap and (ii) mixing time of M^t.

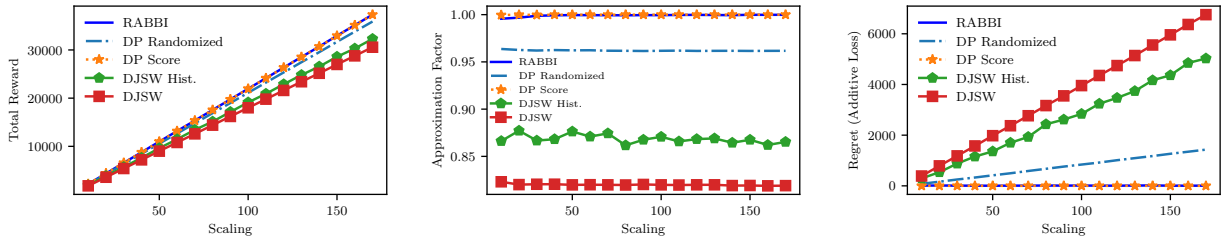
For a function $h : V \to \mathbb{R}$ we define $||h||_\pi^2 = \int h(v)^2 \pi(dv)$ Given a Markov operator P, the absolute spectral gap is $\lambda(P)$, where $\lambda(P) = \sup\{||Ph||_\pi : ||h||_\pi = 1, \int h(v)\pi(dv) = 0\}$; see [58].

Proposition 6.4.5. *Suppose that the arrival process is Markov modulated with a chain M^t that has invariant measure π and absolute spectral gap λ. Then, if M^T and $\widehat{M}^T$ are initialized with π, Eq. (6.5) holds with $\theta_1 = 4n$ and $\theta_2 = 48\frac{\lambda}{1+\lambda}\beta^2$, hence the regret is at most $\mathcal{O}(\frac{dr_\varphi n\kappa^2}{(\lambda)\beta^2})$. Furthermore, the chain can be hidden from the controller.*

Example 6.4.6. The 2-state Markov chain with states $V = \{1,2\}$ and transition matrix $P = \left(\begin{smallmatrix} 1-\delta & \delta \\ \delta & 1-\delta \end{smallmatrix}\right)$ has spectral gap $\lambda = 3\delta(1-\delta)$. The modulated arrival process has parameters $p_j(1), p_j(2)$. With a single historical trace, if $\delta \approx 0$, it could be that the online trace evolves with parameters $p_j(1)$ for most periods, whereas the historical trace with $p_j(2)$ for most periods. This captures the intuitive notion that a slower-mixing chain releases less information in a single trace. In other words, if $\delta \approx 0$ a single trace is uninformative. This is reflected in our regret bound: it grows as $1/\delta$ as $\delta \downarrow 0$.

It is, in fact, not necessary for the chain to be initialized with its stationary distribution. With other conditions, it suffices that M^T and $\widehat{M}^T$ are initilaized in the same state. To state this, the mixing time of a Markov chain is given by $t_{\text{mix}} := \min\{t : \sup_{v\in V} d_{TV}(P^t(v,\cdot),\pi) \leq 1/4\}$, where d_{TV} is the total variation distance; see e.g. [91].

Proposition 6.4.7. *Suppose that the arrival process is Markov modulated with a chain M^t that has invariant measure π and mixing time t_{mix}. Then, if both M^T and $\widehat{M}^T$ start at the*

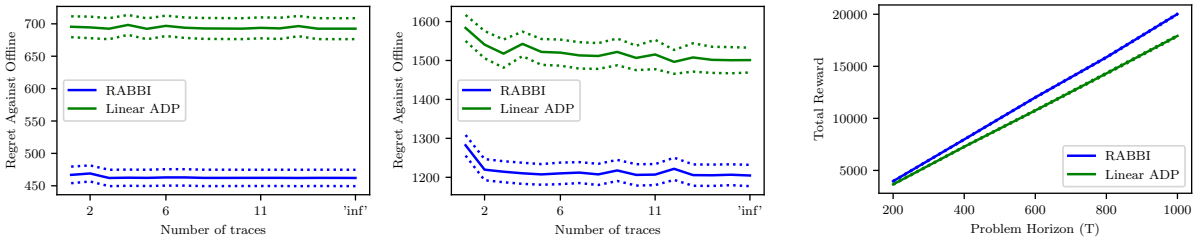


(a) Total reward vs. problem size

(b) Approximation factor vs. problem size

(c) Regret vs. problem size scaling

Figure 6.1: Performance of different policies for the multi-secretary problem: The plots show the total reward and approximation factor for different algorithms as the size of the problem instance scales. We compare RABBI against (*i*) the DP with empirical value-functions and randomized tie-breaking, (*ii*) the DP with empirical value-functions with RABBI's score-based tie breaking, and (*iii*) two variants of the online learning policy of [51], which we refer to here as DJSW. All algorithms are given a single historical trace.



(a) RABBI vs ADP for different matching instances and increasing traces:
Stationary arrivals (left) and time-varying arrivals (right).

(b) Performance (total reward) vs. increasing horizon for RABBI and ADP

Figure 6.2: Performance of RABBI in the reusable matching problem. We compare RABBI against a specialised algorithm based on Approximate DP with Linear Basis Functions. We give both algorithms an increasing number of historical traces as well as the full arrival model (denoted as 'inf' in plots (a) and (b)). Dotted lines represent 90% confidence intervals.

same state, Eq. (6.5) is satisfied with $\theta_1 = 4n$ *and* $\theta_2 = 48\beta^2/t_{mix}$*, hence the regret is at most* $\mathcal{O}(dr_\varphi n\kappa^2 t_{mix}/\beta^2)$*. Furthermore, the chain can be hidden from the controller.*

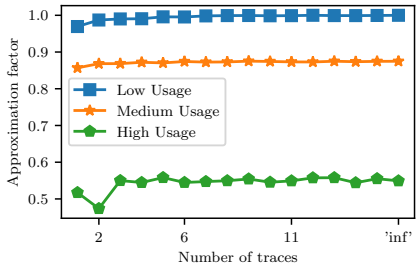


Figure 6.3: Performance of RABBI in different instances of reusable packing with increasing lengths l_j (number of periods that requests use the resources). In the x-axis we denote as 'inf' the case where we give RABBI the full distribution.

6.5 Numerical Experiments

We now present numerical experiments to emphasize the following: (i) a single trace suffices for bounded regret, (ii) the importance and usefulness—beyond our algorithm—of using *action scores* as a tie-breaking rule for estimated value functions, and (iii) the effectiveness of our approach in settings beyond those covered by Theorem 6.2.2. We use the multi-secretary problem (Example 6.2.1) for the first two objectives, and online packing and matching with reusable resources (Section 6.2.2) for the third.

6.5.1 Multi-Secretary with Single Trace

We simulate an instance with $n = 4$ types, and non-stationary arrival process (in particular, we choose p_j^t to be sinusoidal with type-dependent shift, frequency, and translation). The base instance has horizon of length $T = 100$ and initial resource stock of $S^T = 60$; we then scale this to get a family of instances, where the k-th instance has horizon kT and initial resource stock kS^T. Fig. 6.1 compares different algorithms—RABBI, empirical value-function approximation (DP) with randomized tie-breaking and score-based tie-breaking, and the minimax optimal policy (DJSW) [51]—for this setting.

Fig. 6.1b shows that RABBI achieves almost 100% of the offline reward. Moreover, we also see that DP with a random tie-breaking rule has regret linear in T, but if we instead use the score-based tie-breaking rule, then it has the same performance as RABBI (thereby confirming Theorem 6.2.3). We note though that it is not feasible to run DP in higher dimensions, while RABBI is always efficient.

We also benchmark RABBI against the minimax optimal algorithm for this setting [51], which we indicate as DJSW. This policy has a tuning parameter ε, and we search over this parameter and report the best performance. While DJSW does not naturally incorporate historical traces, we also test a modified version where we first feed DJSW the historical trace for T periods (but ignore its actions), and then the online trace for the next T periods. Both algorithms perform much worse, though we note that this is expected since they are designed for maximizing worst-case performance.

6.5.2 Online Matching with Reusable Resources

Next we consider online matching with reusable resources (cf. Section 6.2.2): A type-j customer generates reward $r_{ij} \geq 0$ if assigned a resource $i \in [d]$, and utilises i for $l_j \geq 1$ periods, after which it is returned to the platform. The offline problem at time t is thus $\max_{\mathbf{y} \in \mathbb{R}_{\geq 0}^{n \times [t]}} \sum_{i,j} r_{ij} y_{ij}^t$ subject to the following constraints

$$
\begin{aligned}
S_i^\tau &= S_i^T - \sum_{j \in [n]: l_j > 1} A_{ij}(y_{ij}^{\tau + l_j - 1} - y_{ij}^\tau), \quad \tau \in [t] \\
y_{ij}^{\tau - 1} &\leq y_{ij}^\tau, \quad \tau = 2, \ldots, t \\
\sum_i y_{ij}^\tau &\leq Z_j^\tau, \quad \tau \in [t] \\
y_{ij}^\tau &= \bar{y}_{ij}^\tau, \quad j \in [n], \tau = t+1, \ldots, t + l_j.
\end{aligned}
$$

The parameters $\bar{y}_{ij}^{\tau}$ correspond to the controls over the interval $[t + l_j, t + 1]$, i.e., past controls that impact the system at t. As time progresses, we simply set the values of $\bar{y}_{ij}^{\tau}$ to the actions performed in the past.

In Fig. 6.2, we compare the performance of RABBI to the state-of-the-art approach (ADP with tuned linear basis functions) for this setting [97]. For fixed T, we study two instances, with stationary (Fig. 6.2a) and with time-varying (Fig. 6.2b) arrival processes, and report the regret as a function of number of traces (with 'inf' denoting full model specification). Additionally, for the latter instance, we also demonstrate the performance (reward) as we scale the horizon T (Fig. 6.2b). Our experiments show that RABBI with a single trace, beats the specialized algorithm *even if it is provided the full distribution.*

6.5.3 Online Packing with Reusable Resources

Finally we study online packing with reusable resources (Section 6.2.2): A type-j customer offers reward r_j for using resources $\{i \in [d] : A_{ij} = 1\}$ for $l_j \geq 1$ periods. The period-t offline problem encodes the natural packing constraints. For this setting we are unaware of any specialized algorithm (with the exception of the single-resource, infinite horizon case [82]), and so we compare only to the offline benchmark.

We consider instances with horizon $T = 200$, and test RABBI in 3 different instances: a low-usage instance with $d = 2$ resources, $n = 6$ request types, and random holding times ℓ_j with average 3; a medium-usage instance with $d = 4$, $n = 7$, and average holding-time 6; and a high-usage instance with $d = 7$, $n = 5$, and average holding-time 10. We report results in Fig. 6.3. Models with larger holding-times naturally lead to harder problems, as supported by our experiments. Moreover, additional traces leads to small performance improvements, as is observed in the reusable matching and multi-secretary settings.

6.6 Proofs of Main Results

PROOF OF THEOREM 6.4.2. We use the framework introduced in Chapter 5. In particular, we can bound the regret of RABBI in terms of the information loss and Bellman loss, as $\sum_{t=1}^{T}(dr_{\varphi}\mathbb{P}[\mathcal{Q}(t,S^t)] + \mathbb{E}[L_B(t,S^t)])$, where S^t denotes the state of RABBI, i.e., the process that evolves according to using controls given by RABBI; $\mathcal{Q}(t,s)$ is the disagreement set, which corresponds to the set of online traces where the control chosen by RABBI is not optimal for offline given the same state; and $L_B(t,s)$ is the Bellman loss, which corresponds to the violation of $\hat{\varphi}$ in Eq. (6.4) w.r.t. the Bellman equations.

Using assumptions 1 and 4, we conclude from Proposition D.1.1 that the Bellman loss is bounded by dr_{φ} since the functions h and g satisfy both assumptions required in that result. Moreover, we claim that $\mathbb{P}[\mathcal{Q}(t,S^t)] \leq \mathbb{P}[||Z^t - \widehat{Z}^t|| \geq \widehat{Z}^t_j/|\mathcal{U}|\kappa]$, where $j = \xi^t$ is the current arrival. Assuming this claim and using Eq. (6.5), we obtain $\sum_{t=1}^{T}\mathbb{P}[\mathcal{Q}(t,S^t)] \leq \sum_{t=1}^{T}\theta_1 e^{-\frac{\theta_2 t}{(|\mathcal{U}|\kappa)^2}}$, proving the final regret bound.

To prove the claim we use the Lipschitz continuity of LPs as discussed in Section 6.4. Fix a time t and let $\widehat{X}$ be the solution to the optimization problem in Eq. (6.4). There exists a solution X to the same optimization problem but with Z^t in lieu of $\hat{Z}^t$ such that $||X - \widehat{X}||_{\infty} \leq \kappa||Z^t - \widehat{Z}^t||_{\infty}$. We remark that X corresponds to the problem solved by the offline controller. Let u be the control chosen by RABBI. This control is not optimal for offline only if $X_{uj} < 1$, which formally is $\mathcal{Q}(t,S^t) = \{\omega \in \Omega : X_{uj} < 1\}$. Intuitively, the control u is not optimal only if the offline problem never uses the control u for type j ($X_{uj} < 1$).

Since RABBI chooses the control u with maximum entry and we have the constraint $\sum_{u' \in \mathcal{U}} \widehat{X}_{u'j} = Z^t_j$, necessarily $\widehat{X}_{uj} \geq \frac{Z^t_j}{|\mathcal{U}|}$. In conclusion $X_{uj} < 1$ necessitates $||X - \widehat{X}||_{\infty} \geq \frac{Z^t_j}{|\mathcal{U}|}$. Applying the Lipshichtz property, we have $\mathbb{P}[||X - \widehat{X}||_{\infty} \geq \frac{Z^t_j}{|\mathcal{U}|}] \leq \mathbb{P}[||Z^t - \widehat{Z}^t||_{\infty} \geq \frac{Z^t_j}{\kappa|\mathcal{U}|}]$

concluding the proof of the claim. □

PROOF OF THEOREM 6.2.3. With a single trace, DP has the following description. At time t, DP solves an *integer program* that corresponds to Eq. (6.4) with integrality constraints. Let X be the solution of said IP. Then, if $\xi^t = j$, the control u is a maximizer of the Bellman equation if $X_{uj} \geq 1$. Observe that *there could be multiple such controls.* If in case of ties DP uses RABBI's decision rule, i.e., play u with maximum X_{uj}, then the analysis is exactly the same as in the proof of Theorem 6.2.2 except that we use the Lipschitz property of integer programs [46]. □

PROOF OF PROPOSITION 6.4.3. For brevity, all the norms $||\cdot||$ denote the infinity norm. We prove that $\mathbb{P}[||Z^t - \widehat{Z}^t|| \geq \widehat{Z}^t_k/c] \leq 4ne^{-\frac{\beta^2 t}{3(2c+1)^2}}$. Standard concentration results imply that, for $x < \min_j \mu^t_j$,

$$\mathbb{P}[||Z^t - \mu^t|| \geq x] \leq 2\sum_j e^{-\frac{x^2}{3\mu^t_j}}. \tag{6.6}$$

Call E_x the event where $||Z^t - \mu^t|| < x$ and $||\widehat{Z}^t - \mu^t|| < x$. We use total probabilities to compute:

$$\begin{aligned}\mathbb{P}\left[||Z^t - \widehat{Z}^t|| \geq \frac{\widehat{Z}^t_k}{c}\right] &\leq \mathbb{P}\left[||Z^t - \widehat{Z}^t|| \geq \frac{\widehat{Z}^t_k}{c}|E_x\right] + \mathbb{P}[E_x^c] \\ &\leq \mathbb{1}_{\left\{x+x>(\mu^t_k - x)/c\right\}} + \mathbb{P}[E_x^c].\end{aligned}$$

The last inequality is because, in E_x, $\widehat{Z}^t_k > \mu^t_k - x$. Set $x = \frac{\min_j \mu^t_j}{2c+1}$ and observe that, with this choice, the indicator is zero and we satisfy the requirements of Eq. (6.6). Further, by our condition we have $x \geq \frac{\beta t}{2c+1}$ and it always holds that $\mu^t_j \leq t$, hence

$$\mathbb{P}[||Z^t - \widehat{Z}^t||_\infty \geq \widehat{Z}^t_k/c] \leq \mathbb{P}[E_x^c] \leq 4\sum_j e^{-\frac{\beta^2 t}{3(2c+1)^2}}.$$

This concludes the proof. □

Proof of Proposition 6.4.5. Observe that the crucial fact in the proof of Proposition 6.4.3 is that both Z^t and $\widehat{Z}^t$ concentrate around μ^t, see Eq. (6.6), where μ^t does not depend on the realization, i.e., it is fixed for all t. We prove a similar fact, but with another quantity that is also common to both traces. We claim that

$$\mathbb{P}[||Z^t - t\pi(p)||_\infty \geq x] \leq 4ne^{-\frac{\lambda}{12t}x^2}, \tag{6.7}$$

where $\pi(p)_j \coloneqq \sum_{v\in V} \pi(v)p_j(v)$. Assuming Eq. (6.7), then the proof proceeds exactly as in Proposition 6.4.3.

What remains is to prove Eq. (6.7). Conditioned on $M^t = v^t, \ldots, M^1 = v^1$, the arrivals of type j are independent with probabilities $p_j(v^\tau)$ for $\tau = t, \ldots, 1$. Denoting $p_j(v^{t:1}) = p_j(v^t) + \ldots + p_j(v^1)$ and $E(v^{t:1})$ the event $M^t = v^t \ldots, M^1 = v^1$, let us define the following r.v. conditioned on M^t:

$$\eta_j^t = \sum_{v^t,\ldots,v^1} \mathbb{1}_{E(v^{t:1})}(p_j(M^t) + p_j(v^{t-1:1})). \tag{6.8}$$

Then, $\mathbb{P}[|Z_j^t - \eta_j^t| \geq x|M^t]$ is upper bounded by

$$\begin{aligned}
&\sum_{v^t,\ldots,v^1} \mathbb{P}[E(v^{t:1})|M^t]\mathbb{P}\left[\left|Z_j^t - \frac{p_j(l^{t:1})}{t}\right| \geq x|E(v^{t:1})\right] \\
&\leq \sum_{v^t,\ldots,v^1} \mathbb{P}[E(v^{t:1})|M^t]2e^{-\frac{x^2}{3t}} = 2e^{-\frac{x^2}{3t}}.
\end{aligned} \tag{6.9}$$

The inequality follows from Eq. (6.6).

We know that Z_j^t concentrates around η_j^t and $\widehat{Z}_j^t$ concentrates around $\widehat{\eta}_j^t$ (as defined in Eq. (6.8), but with $\widehat{M}^t$ replacing M^t). Now we will show that both η_j^t and $\widehat{\eta}_j^t$ concentrate around $t\pi(p)_j$. Indeed, applying [58, Theorem 2.1], provided that $M^T \sim \pi$, we obtain the following

$$\mathbb{P}\left[\left|\eta_j^t - t\pi(p_j)\right| > x\right] \leq 2e^{-2\frac{\lambda}{1+\lambda}\frac{x^2}{t}}. \tag{6.10}$$

We use $|Z_j^t - t\pi(p)_j| \leq |Z_j^t - \eta_j^t| + |\eta_j^t - t\pi(p)_j|$ and evaluate Eqs. (6.9) and (6.10) with $x/2$ to obtain

$$\mathbb{P}[|Z_j^t - t\pi(p)_j| \geq x] \leq 2e^{-\frac{x^2}{12t}} + 2e^{-\frac{\lambda}{1+\lambda}\frac{x^2}{2t}}.$$

A union bound finishes the proof of Eq. (6.7). □

Proof of Proposition 6.4.7. We reason as in the proof of Proposition 6.4.7, but in this case we prove that Z^t and $\widehat{Z}^t$ concentrate around $\mathbb{E}[\eta^t]$ which is also invariant of the trace (since both chains start at the same state). Formally, we claim $\mathbb{P}[||Z^t - \mathbb{E}[\eta^t]||_\infty \geq x] \leq 4ne^{-\frac{\lambda}{12t}x^2}$.

To prove the claim, we use $|Z_j^t - \mathbb{E}[\eta_j^t]| \leq |Z_j^t - \eta_j^t| + |\eta_j^t - \mathbb{E}[\eta_j^t]|$ so that

$$\mathbb{P}[|Z_j^t - \mathbb{E}[\eta_j^t]| \geq x] \leq \mathbb{P}\Big[|Z_j^t - \eta_j^t| \geq \frac{x}{2} \text{ or } |\eta_j^t - \mathbb{E}[\eta_j^t]| \geq \frac{x}{2}\Big].$$

The first event is bounded by Eq. (6.9) whereas for the second we apply [91, Corollary 2.10] to obtain $\mathbb{P}[|\eta_j^t - \mathbb{E}[\eta_j^t]| \geq x] \leq 2e^{-2\frac{x^2}{tt_{\text{mix}}}}$. Finally, a union bound finishes the proof of the claim. □

6.7 Discussion

In our numerical experiments it was not necessary to tune the algorithm to obtain good performance. Nevertheless, we recommend two different ways of tuning RABBI to boost the performance in practice. First, in Step 4 of RABBI, instead of choosing the action with maximum score $y_{u,\xi}$, we can chooss the action with maximum weighted scored $w_{u,\xi}y_{u,\xi}$, where $\mathbf{w} \in \mathbb{R}^{\mathcal{U}\times[n]}$ is now a parameter that can be tuned. Second, we can influence the aggressiveness of the algorithm by modifying the initial state S^T; in the case where S^T represents inventory levels (capacities) of d resources, we can change S^T to be $S^T + \mathbf{w}$, where now $\mathbf{w} \in \mathbb{R}^d$ is a tuning parameter.

APPENDIX A

APPENDIX FOR CHAPTER 2

A.1 Different notions of HD

We give below the main notions of HD. We note that they are all similar and they are interlinked in terms of their properties, see [61, Section 9] for several results on this. We limit the discussion here to highlight the main differences.

- [2]: this is the closest to our definition, but we weaken it (make it easier to satisfy). The difference is that they make the notion of r-significant stronger by also calling r-significant paths P whose length is $\ell(P) < r$, but can be extended by adding at most one node at each end of P to obtain a path P' with $\ell(P')$.
- [1]: it is also very similar, but technically different in that we define path neighbourhoods $S_r(v)$ as sets of paths (while they use the union of nodes belonging to those paths), hence in our definition there are fewer elements to hit. On the other hand, they consider paths of lengths between r and $2r$, while we consider paths longer than r. We could also add the restriction of length at most $2r$ without changing any of the results, but at the cost of making the definition even more involved.
- [4]: it is slightly weaker, in that they look at paths are r-significant and contained in a larger ball (of radii $4r$) vs the notion of r-significant and within distance $2r$ of a node. Specifically, for each v, they ask for hitting sets of paths $P \subseteq B_{4r}(v)$ such that $\ell(P) > r$.

For $\mathcal{P}^*$, a consequence of considering less-restrictive path-neighborhoods is that the highway dimension returned by our definition is smaller than that of [2]. In particular, unlike

[2], the HD of G as per our definition is not an upper bound to the maximum degree Δ or the doubling constant α. The notion of [4] does not bound the doubling constant.

With respect to our average HD in Section 2.3.4, we note the following.

Remark A.1.1. The algorithm in Theorem 2.3.18 makes one call to the VC-dimension solver for each C_i. On the other hand, the algorithm in [2] calls up to n times the solver for each C_i. Finally, there is an extra $\log n$ factor in the approximation guarantee, but now the value of h can be much smaller.

We now discuss how our results extend to the definition in [2], which we refer to as strong-HD. The strong-HD defines a path P to be r-significant if, by adding at most one hop at each end, we get a shortest path P' longer than r. The path P' is called an r-witness for P. Intuitively, a path is significant if it represents a long path. Observe that, if $P \in \mathcal{P}^*$ is such that $\ell(P) > r$, then P is r-significant by definition. We remark also that a path can have many r-witnesses.

Finally, the path neighborhood must also be strengthened. The path $P \in \mathcal{P}^*$ belongs to $S_r^+(v)$ if, P has some r-witness P' such that $\text{dist}(v, P') \leq 2r$. The reverse neighborhood $S_r^-(v)$ is defined analogously. With this modified versions of r-significant and neighborhood, the notions of LSHS and HD are the same as our previous definitions.

Under the strong-HD, we have $\Delta \leq h$ and $\alpha \leq h + 1$. Additionally, this definition allows proving results for CH. Finally, we show that even for the strong-HD, CHD and HD can still be off by a factor of n.

Proposition A.1.2. *For any h, we can construct a family of networks such that the sparsity of LSHS is h and that of EPHS is arbitrarily worse than h.*

Proof. First, we construct an example where the sparsity grows from h to h^2. Consider an

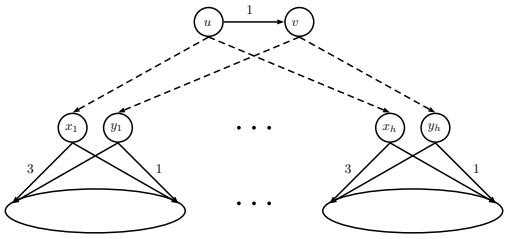


Figure A.1: Example where the EPHS is much larger than the LSHS.

h-ary tree rooted at u with three levels, i.e., with $1 + h + h^2$ nodes. Now add a node v with h children as in Figure A.1. The grandchildren of v are the same as the grandchildren of u.

All the edges are bidirectional and have unit cost. The lengths are as follows: ux_i and vy_i (dashed in Figure A.1) are zero; uv and from y_i to the leafs is one; from x_i to the leafs is three. It is easy to see that the sparsity of a LSHS is $h + 1$.

On the other hand, every leaf w is a 2-efficient path. Indeed, it can be extended to $x_i w$ that is the shortest path from x_i to w with constraint 1. All the leafs are in the ball $B_4(u)$, so the sparsity is at least h^2.

The general case works in the same fashion. We make the sparsity grow to h^k by creating two complete, k-level, h-ary trees T and T'. Connect the root of T to the root of T' and the leafs of both trees are shared. Observe that the number of nodes is

$$\begin{aligned} n &= [k\text{-level } h\text{-ary tree}] + [(k-1)\text{-level } h\text{-ary tree}] \\ &= (h^{k+1} - 1)/(h-1) + (h^k - 1)/(h-1), \end{aligned}$$

therefore the sparsity is $\Theta(n)$, the worst possible. □

A.2 Contraction Hierarchies

We present here how to extend the concept of HD in order to prove the efficiency of CH in directed graphs. Given a rank in the nodes, the shortcut process works as in the non-directed case:

1. Let G' be a temporary copy of G.
2. Remove nodes of G' and its edges in increasing rank.
3. When removing v, if some unique shortest path in G uses uvw, add (u, w) to G' with length $\ell(u, v) + \ell(v, w)$.

Call E^+ the set of edges created in the shortcut process. A source-destination query runs bidirectional Dijkstra, but each search only considers paths of increasing ranks.

As in the non-directed case, let $Q_i = C_i \setminus \cup_{j>i} C_j$ be the partition of V. All the ranks in Q_i are smaller than those in Q_{i+1}, within each Q_i the rank is arbitrary.

Lemma A.2.1. *Let P be a shortest path in the original graph. If P has at least three vertices and $\ell(P) > 2^\gamma$, then some internal vertex of P belongs to a level Q_x, $x > \gamma$.*

Proof. The path P' obtained by removing the endpoints of P is $\ell(P)$-significant. By definition of the C_i's, $C_{\gamma+1}$ hits P' at some node u. By construction of the partition, $u \in Q_x$ with some $x > \gamma$. □

Now we show that each node adds at most h to its out-degree for each Q_i, so the process adds at most $h \log D$ to the out-degree of each node.

Lemma A.2.2. *Assume the network admits the C_i's. For any v and fixed j, the number of shortcuts (v, w) with $w \in Q_j$ is at most h.*

PROOF. Let i be the level such that $v \in Q_i$ and define $\gamma := \min(i, j)$. We claim that $w \in B^+_{2^\gamma}(v)$. Assume the claim, then the number of shortcuts is at most $|Q_j \cap B^+_{2^\gamma}(v)|$, but using local sparsity and set inclusion:

$$|Q_j \cap B^+_{2^\gamma}(v)| \leq |C_j \cap B^+_{2 \cdot 2^{j-1}}(v)| \leq h.$$

All that remains is to prove the claim. The shortcut (v, w) was created when the process removed the last internal vertex of the shortest path $P(v, w)$ in G. Necessarily all the internal vertices are in levels at most γ, because they were removed before v and w, hence they have lower rank. Finally, apply Lemma A.2.1 to conclude that $\ell(P(v, w)) \leq 2^\gamma$. □

We need to bound the in-degree, because it could be that some node v is receiving many edges. The proof is basically the same.

Lemma A.2.3. *Assume the network admits the C_i's. For any v and fixed j, the number of shortcuts (w, v) with $w \in Q_j$ is at most h.*

PROOF. Same as in the previous lemma, but now $w \in B^-_{2^\gamma}(v)$. □

We can conclude now that the number of shortcuts, i.e. $|E^+|$, is at most $2nh \log D$.

As we mentioned before, the query performs Dijkstra from the source and target, but always constructing paths of increasing rank. When scanning a vertex v, the forward search has a label $\text{dist}(s, v)'$. The labels always satisfy $\text{dist}(s, v)' \geq \text{dist}(s, v)$, but, since the algorithm only goes to higher ranks, equality is not guaranteed.

We add a pruning rule analogous to the non-directed case: when the forward search scans a node v, if $(v, w) \in E \cup E^+$ and $w \in Q_i$, then w is added to the priority queue only if $\text{rank}(w) > \text{rank}(v)$ and $\text{dist}(s, v)' + \ell(v, w) \leq 2^i$. For the reverse search, the condition is the analogous $\text{dist}(v, t)' + \ell(w, v) \leq 2^i$ when $(w, v) \in E \cup E^+$.

Proposition A.2.4. *The query with additional pruning returns the correct distance. Additionally, each Dijkstra scans at most h nodes in each level.*

PROOF. Let us analyse the forward search. Say the node v is being scanned, $w \in Q_i$ is a candidate and $\text{dist}(s,v)' + \ell(v,w) > 2^i$. If the current path P' to w is optimal, then $P(s,w)$ is 2^i-significant and it is hit by C_{i+1}. As a consequence, $P(s,w)$ contains an internal vertex with higher rank than w. This vertex cannot be in P' nor a shortcut containing it, thus contradicting the optimality of P'. We conclude that P' is not optimal and w can be ignored.

Bounding the number of scanned nodes is easy; every $w \in Q_i$ added to the queue satisfies $w \in B^+_{2^i}(s)$, so applying local sparsity we finish the proof. □

As a result, the forward search adds at most $h \log D$ nodes to the queue; each of node amounts to $O(\text{outdeg}(G^+))$ operations, i.e., $O(\text{outdeg}(G) + h \log D)$ operations.

A.3 CHD vs. HD: Extensions

So far we have only used the structure of shortest paths in G, which, naturally, does not capture all the information in the network. It is natural to think that there is structure if we look at, for example, only zero cost edges.

Let G_0 be obtained from G by removing all the edges with cost. The networks G and G_0 define two hierarchies of roads; shortest paths in G_0 are free, but not as fast as the ones in G. Our main hypothesis now is that an efficient path P does not alternate between these two hierarchies. For example, a path that enters and exits multiple times a highway is not desirable because of turning costs.

Proposition A.3.1. *Let $\mathcal{Q}, \mathcal{Q}'$ be two path systems with HD h and h' respectively. The HD of the system $\mathcal{Q} \cup \mathcal{Q}'$ is at most $h + h'$.*

PROOF. Given $v \in V$, the union of $H_{v,r}$ and $H'_{v,r}$ hits all the paths in $S_r(v, \mathcal{Q}) \cup S_r(v, \mathcal{Q}')$. □

We now relax the assumption that a system witnesses another. It could be that the efficient paths are sometimes witnessed by free paths and sometimes by shortest paths.

Theorem A.3.2. *Assume that G has doubling constant α and $\mathcal{Q}, \mathcal{Q}'$ are systems with HD h, h' respectively. Moreover, suppose $\mathcal{P}^E$ does not alternate between $\mathcal{Q}$ and $\mathcal{Q}'$, that is, for some $\beta, \beta' > 0$, each path $P \in \mathcal{P}^E$ is either β-witnessed by some $Q \in \mathcal{Q}$ or β'-witnessed by $Q' \in \mathcal{Q}'$. Then G admits $(\alpha^\beta h + \alpha^{\beta'} h', r)$-EPHS.*

A.3.1 Correlated Costs

We have studied so far the case where $c(P)$ is just the sum of individual edge costs. In practice it could be that the cost depends on combinations of arcs. Think of a turn in a road network; we can turn right quickly, but turning left means waiting for a green arrow in most cases. Another example is minimizing expectation subject to bounded variance. If there is no independence, the variance of a path is not the sum of individual variances.

We explain now how to deal with more general cases using the same framework. Assume the cost function $c_2 : E \times E \to \mathbb{N} \cup \{0\}$ depends on pairs of edges, so if a path is $P = e_0 e_1 \dots e_k$, then the cost would be $c_2(P) = \sum_{i=1}^{k} c_2(e_{i-1}, e_i)$. The nodes in the augmented graph will be triplets $\langle u, v, b \rangle$, where v is the current state, u is the previous state and b is the available budget. The arcs are given by

$$(\langle u, v, b \rangle, \langle v, w, b' \rangle), \quad uv, vw \in E, b' = b - c_2(u, v, 2).$$

Define analogously the concept of efficient paths. It is easy to see that, as in the previous case, shortest paths in the augmented graph are efficient paths. The system $\tilde{\mathcal{P}}^E$ of such paths may also allow for a β-witness. With the previous properties we can construct the hub labels in the same fashion to prove the following result.

Theorem A.3.3. *Assume the system $\tilde{\mathcal{P}}^E$ has HD $\tilde{h}$. Then, there exists HL such that queries s, t, b can be answered in time $O(b\Delta\tilde{h}\log D)$ and the space requirement is $O(Bn\cdot\Delta B\tilde{h}\log D)$. In particular, if $\mathcal{P}^*$ is a β-witness for $\tilde{\mathcal{P}}^E$, then $\tilde{h} \leq h\alpha^\beta$.*

A.4 Additional Proofs

Proof of Theorem 2.3.9. To get this stronger bound, we need to modify the HL construction. The algorithm for forward hub construction is given in Algorithm 12, and for reverse hubs in Algorithm 13. Note that the two must be run sequentially, as the latter uses the nodes marked in the former. We make the forward hubs $L^+(\langle v, b\rangle)$ slightly bigger by storing, for each node the distance from $\langle v, b\rangle$ and also the *budget surplus*. Let C_i be the $(h_c, 2^{i-1})$-EPHS and $\mathcal{P}^E_{s,t}$ the efficient paths from s to t.

Observe that, whenever a node $v \in C_i$ is added, $v \in B^+_{2^i}(s)$ guarantees that at most h_c such points are needed for the whole process. Additionally, every such v is added at most $g(b)$ times in the hub of $\langle s, b\rangle$. The data requirement guarantee follows.

The bound for data requirements is $g(B)h_c \log D$, the argument is analogous to the forward case. Finally, we need to prove the cover property. Take any query $SP(\langle s, b\rangle, t^-)$ and let P be the solution. In $Lf(\langle s, b\rangle)$ there is a node v_P added by Algorithm 12. By construction, the same node v_P was added to $L^-(\langle d, 0\rangle)$. The result follows. □

Proof of Proposition 2.3.8. We extend some arguments from [2, Theorem 8.2]. Denote

Algorithm 12 Construction of forward hub

Input: Node $s \in V$, efficient paths $\mathcal{P}^E_{s,t} \forall\, t$, EPHS $\{C_i\}$.
Output: Forward hubs $L^+(\langle s, b\rangle)$ for $b = 0, \ldots, B$ and a marked node v_P for every path.

Order each $\mathcal{P}^E_{s,t}$ by increasing cost and remove paths consuming more than B.
for $t \in V \setminus s$ **do**
    for $P \in \mathcal{P}^E_{s,t}$ **do**
        $b \leftarrow c(P)$, $b' \leftarrow c(P')$, where P' is the next path in the list ($b' = B$ if no such path).
        Find the largest i such that P is 2^{i-1}-efficient.
        Find $v \in C_i$ hitting P and mark v as v_P.
        Add $\langle v, c(P[v,t])\rangle$ to $L(\langle s, b\rangle)^+$ with distance $\ell(P[s,v])$ and surplus zero.
        for x between b and b' **do**
            Add $\langle v, c(P[v,t])\rangle$ to $L(\langle s, x\rangle)^+$ with distance $\ell(P[s,v])$ and surplus $x - b$.

Algorithm 13 Construction of reverse hub

Input: Node $t \in V$, efficient paths $\mathcal{P}^E_{s,t} \forall\, s$, marked nodes and EPHS C_i.
Output: Backward hub $L^-(\langle t, 0\rangle)$.

Order each $\mathcal{P}^E_{s,t}$ by increasing cost and remove paths consuming more than B.
$L^-(\langle t, 0\rangle) \leftarrow \varnothing$
for $s \in V \setminus t$ **do**
    for $P \in \mathcal{P}^E_{s,t}$ **do**
        Find the largest i such that P is 2^{i-1}-efficient.
        Take v as the marked node v_P.
        Add $\langle v, c(P[v,t])\rangle$ to $L^-(\langle t, 0\rangle)$ with distance $\ell(P[v,t])$.

$S_r(v) := S_r^+(v, \mathcal{Q}) \cup S_r^-(v, \mathcal{Q})$. Observe that, for fixed $v \in V$, the set system $(E, \{\pi(Q) : Q \in S_r(v)\})$ admits a hitting set of size $h\Delta$. Indeed, we know that exists $H_{v,r} \subseteq V$, $|H_{v,r}| \leq h$, hitting every path in $S_r^+(v, \mathcal{Q})$ and in $S_r^-(v, \mathcal{Q})$. The desired hitting set consists of all the edges adjacent to a node in $H_{v,r}$.

If the minimum size of a set system is s and the VC-dimension is d, then the algorithm in [57] obtains, in polynomial time, a hitting set of size at most $O(sd \log(sd))$. In particular, we can use the algorithm to obtain a set $\tilde{F}_{v,r} \subseteq E$, of size at most $h' = O(h\Delta \log(h\Delta))$, hitting the set system $(E, \{\pi(Q) : Q \in S_r(v)\})$.

Consider the set $F_{v,r} \subseteq V$ that contains all the endpoints of edges in $\tilde{F}_{v,r}$. It follows that $F_{v,r} \subseteq V$ can be obtained in polynomial time and is a hitting set for $S_r(v)$ of size $|F_{v,r}| \leq 2h'$. Assume for now that we know the value of h. Note that the value h' can be

computed from h and the guarantee given by the oracle, i.e., the constant inside the big-O. We construct the $(2h', r)$-LSHS iteratively. At each iteration i we maintain the following invariant: C_i hits every path in $\mathcal{Q}_r$. In an iteration we check if C_i is locally sparse, if not, we strictly reduce the cardinality of C_i while maintaining the invariant. Start with $C_0 = V$. Let $B_{2r}(v) := B^+_{2r}(v) \cup B^-_{2r}(v)$. Assume $v \in V$ is such that $|B_{2r}(v) \cap C_i| > 2h'$ and let $C_{i+1} := (C_i \setminus B_{2r}(v)) \cup F_{v,r}$. The cardinality strictly decreases and we only need to check the invariant. Consider the paths hit by nodes removed in C_i, this set is

$$\{Q \in \mathcal{Q}_r : Q \cap C_i \cap B_{2r}(v) \neq \varnothing\} \subseteq \{Q \in \mathcal{Q}_r : Q \cap B_{2r}(v) \neq \varnothing\} \subseteq S_r(v).$$

Since $F_{v,r}$ hits $S_r(v)$, the proof is completed.

If we do not know the value of h, we can do a doubling search for h'. Indeed, if the guess of h' is low, then at some point it could be that $|F_{v,r}| > 2h'$, then we double h' and restart the process. □

APPENDIX B

APPENDIX FOR CHAPTER 3

B.1 The Fluid Benchmark

PROOF OF PROPOSITION 3.2.1. To build intuition, we start with a description of dual degeneracy for the online knapsack problem with budget $B \leq T$. We assume w.l.o.g. $r_1 \geq r_2 \geq \ldots \geq r_n$ and denote $Z = Z(T)$. The primal and dual are given by

$$\begin{aligned} (P[Z])\ \max \quad & r'x \\ \text{s.t.} \quad & \textstyle\sum_{j\in[n]} x_j \leq B \\ & x \leq Z \\ & x \geq 0, \end{aligned} \qquad \begin{aligned} (D[Z])\ \min \quad & \alpha B + \beta' Z \\ \text{s.t.} \quad & \alpha + \beta_j \geq r_j \,\forall j \\ & \alpha \geq 0 \\ & \beta \geq 0. \end{aligned}$$

Let us denote $\mu \coloneqq \mathbb{E}[Z]$. If the fluid $(P[\mu])$ is degenerate, then we have $n+1$ active constraints. It is straightforward to conclude that there must be an index $j^\star$ such that $\sum_{j\leq j^\star} \mathbb{E}[Z_j] = B$. The fluid solution is thus $x_j = \mathbb{E}[Z_j]$ for $j \leq j^\star$ and $x_j = 0$ for $j > j^\star$. We can construct two dual solutions as follows. Let $\alpha^1 = r^\star_j$ and $\alpha^2 = r_{j^\star+1}$, these correspond to the shadow prices for alternative budgets $B-\varepsilon$ and $B+\varepsilon$ respectively. The corresponding variables β^1, β^2 are given by $\beta^k_j = (r_j - \alpha^k)_+$ for $k = 1, 2$. Intuitively, the fluid is indifferent between these two dual bases, but, given a realization of Z, OFFLINE will prefer one over the other; this causes a discrepancy between the expectations.

Now we turn to the case of any packing problem, the assumption is that we are given two optimal dual solutions (α^k, β^k), with $\beta^1 \neq \beta^2$. The dual is a minimization problem and (α^k, β^k) are always dual feasible, thus defining $\beta \coloneqq \beta^1 - \beta^2$ and $\alpha \coloneqq \alpha^1 - \alpha^2$,

$$v(D[Z]) \leq \min_{k=1,2}\{B'\alpha^k + Z'\beta^k\} = (B'\alpha^1 + Z'\beta^1)\mathbb{1}_{\{B'\alpha+Z'\beta<0\}} + (B'\alpha^2 + Z'\beta^2)\mathbb{1}_{\{B'\alpha+Z'\beta\geq 0\}}.$$

The rest of the proof is reasoning that interchanging expectations $\mathbb{E}[\min_{k=1,2}\{B'\alpha^k + Z'\beta^k\}]$ for $\min_{k=1,2}\{B'\alpha^k + \mathbb{E}[Z]'\beta^k\}$ induces a $\Omega(\sqrt{T})$ error.

Since the two dual solutions have the same dual value, $B'\alpha^1 + \mu'\beta^1 = B'\alpha^2 + \mu'\beta^2$, we conclude $B'\alpha = -\mu'\beta$. We can use this condition to rewrite our bound as

$$v(P[Z]) \leq v(D[Z]) \leq (B'\alpha^1 + Z'\beta^1)\mathbb{1}_{\{(\mu-Z)'\beta>0\}} + (B'\alpha^2 + Z'\beta^2)\mathbb{1}_{\{(\mu-Z)'\beta\leq 0\}}.$$

Since $v(P[\mu]) = B'\alpha^k + \mu'\beta^k$ for $k = 1, 2$, we take a random convex combination to obtain

$$v(P[\mu]) = (B'\alpha^1 + \mu'\beta^1)\mathbb{1}_{\{(\mu-Z)'\beta>0\}} + (B'\alpha^2 + \mu'\beta^2)\mathbb{1}_{\{(\mu-Z)'\beta\leq 0\}}.$$

Now combine the last with our upper bound for $v(P[Z])$ and take expectations to obtain

$$\begin{aligned} v(P[\mu]) - \mathbb{E}[v(P[Z])] &\geq \mathbb{E}[(\mu - Z)'\beta^1\mathbb{1}_{\{(\mu-Z)'\beta>0\}}] + \mathbb{E}[(\mu - Z)'\beta^2\mathbb{1}_{\{(\mu-Z)'\beta\leq 0\}}] \\ &= \mathbb{E}[(\mu - Z)'\beta^1\mathbb{1}_{\{(\mu-Z)'\beta>0\}}] + \mathbb{E}[(\mu - Z)'\beta^2(1 - \mathbb{1}_{\{(\mu-Z)'\beta>0\}})] \\ &= \mathbb{E}[(\mu - Z)'\beta\mathbb{1}_{\{(\mu-Z)'\beta>0\}}]. \end{aligned}$$

Let us define ξ as the normalized vector Z, i.e., $\xi \coloneqq \frac{1}{\sqrt{T}}(\mu - Z)$. We conclude that

$$v(P[\mu]) - \mathbb{E}[v(P[Z])] \geq \sqrt{T}\mathbb{E}[\xi'\beta\mathbb{1}_{\{\xi'\beta>0\}}].$$

Reducing by the standard deviation and applying the Central Limit Theorem, we arrive at a half-normal (also known as folded normal), which has constant expectation. This concludes the desired result. □

B.2 Additional Details and Proofs

B.2.1 Poisson Process in Discrete Periods

We explain how a continuous time Poisson process can be reduced to our setting. We are given a time horizon T, where time $t \in [0, T]$ still denotes time to go and, according to an

exponential clock, arrivals occur at some times $t_1 > t_2 > \ldots > t_N \in [0, T]$, where N is random and corresponds to the total number of arrivals, i.e., $N = \sum_{j \in [n]} Z_j(T)$.

Treating times t_k as periods, there is one arrival per period. Observe that OFFLINE knows N, therefore his Bellman Equation is well defined. ONLINE acts on these discrete periods, i.e., he is event-driven, thus making at most N decisions. Finally, we note that, at some time t_k, $\mathbb{E}[Z_j(t_k)] = \lambda_j t_k$ if the process is homogeneous or $\mathbb{E}[Z_j(t_k)] = \int_0^{t_k} \lambda_j(t)dt$ if the process is nonhomogeneous. In conclusion, ONLINE can compute all the required expectations without knowing N, but rather the knwoledge of t_k and $\lambda(\cdot)$ is enough.

B.2.2 Bayes Selector Based on Marginal Compensations

A somewhat more powerful oracle is one which, for every time t, state s and action a, returns estimates of the *marginal compensation* $\bar{R}(t, a, s) \cdot \mathbb{1}_{Q(t,a,s)}$. This suggests a stronger form of the Bayes selector based on marginal compensations, as summarized in Algorithm 14.

The following result follows directly from Lemma 3.3.9 and gives a performance guarantee for this algorithm.

Algorithm 14 Marginal-Compensation Bayes Selector

Input: Access to over-estimates $\hat{l}(t, a, s)$ of the expected compensation, i.e., $\hat{l}(t, a, s) \geq \mathbb{E}[\bar{R}(t, a, s)\mathbb{1}_{Q(t,a,s)}]$

Output: Sequence of decisions for ONLINE.

1: Set S^T as the given initial state
2: **for** $t = T, \ldots, 1$ **do**
3: Observe arrival ξ^t, and take any action that minimizes marginal compensation, i.e., $a \in \text{argmin}\{\hat{l}(t, a, S^t) : a \in \mathcal{U}\}$.
4: Update state $S^{t-1} \leftarrow \mathcal{T}(a, S^t, \xi^t)$.

Corollary B.2.1 (Regret Of Marginal-Compensation Bayes Selector). *Consider Algorithm 14 with overestimates $\hat{l}(t, a, s)$, i.e., $\hat{l}(t, a, s) \geq \mathbb{E}[\bar{R}(t, a, s)\mathbb{1}_{Q(t,a,s)}]$. If A^t denotes*

the policy's action at time t, then

$$\mathbb{E}[\textsc{Reg}] \leq \sum_t \mathbb{E}[\hat{l}(t, A^t, S^t)].$$

B.2.3 Multi-Secretary Problem

Proof of Theorem 3.4.2. Assume w.l.o.g. that $r_1 \geq r_2 \geq \ldots \geq r_n$. This one dimensional version can be written as follows.

$$\begin{aligned} (P_t^\star)\ \max \quad & r'x \\ \text{s.t.} \quad & \textstyle\sum_{j\in[n]} x_j \leq B^t \\ & x_j \leq Z_j(t)\ \forall j \\ & x \geq 0. \end{aligned} \qquad \begin{aligned} (P_t)\ \max \quad & r'x \\ \text{s.t.} \quad & \textstyle\sum_{j\in[n]} x_j \leq B^t \\ & x_j \leq tp_j\ \forall j \\ & x \geq 0. \end{aligned}$$

The optimal solution to $(P_t^\star)$ is to sort all the arrivals by reward and pick the top ones. The solution to (P_t) is similar, except that it can be fractional; we saturate the variable x_1 to tp_1, then x_2 to tp_2, and continue as long as $\sum_{i\leq j} tp_i \leq B^t$ for some j. Define the probability of 'arrival j or better' by $\bar{p}_j \coloneqq \sum_{i\leq j} p_i$. Observe that we can saturate all variables $1, \ldots, j$ iff $t\bar{p}_j \leq B^t$. The solution to (P_t) is therefore to pick the largest j such that $t\bar{p}_j \leq B^t$, then make $X_i^t = tp_i$ for $i \leq j$ and $X_{j+1}^t = B^t - t\bar{p}_j$. When we round this solution according to Algorithm 2, we arrive at the following policy: First, if $B^t = 0$, end the process. Second (assuming $B^t \geq 1$), always accept class $j = 1$. Third (assuming $B^t \geq 1$), if class $j > 1$ arrives, accept if $B^t/t \geq \bar{p}_j - p_j/2$ and reject if $B^t/t < \bar{p}_j - p_j/2$.

Recall that $q(t, b)$ is the probability that Offline is not satisfied with Online's action at time t if the budget is b. We denote $q_j(t, b)$ as the probability conditioned on $\xi^t = j$. Our aim in the rest of the section is to show that $q_j(t, b)$ is summable over t.

As we observed before: (1) Offline is not satisfied rejecting a class j iff he accepts all the future arrivals type j, i.e., $X_j^{\star t} > Z_j(t) - 1$. (2) Offline is not satisfied accepting class j

iff he rejects all future type j arrivals, i.e., $X_j^{\star t} < 1$. We use the following standard Chernoff bound in [54, Theorem 1.1]. For any $\alpha \in [0,1]$, if $X \sim \text{Bin}(t, \alpha)$:

$$\mathbb{P}[X - \mathbb{E}[X] \leq -t\varepsilon] \leq e^{-2\varepsilon^2 t}\ ,\ \mathbb{P}[X - \mathbb{E}[X] \geq t\varepsilon] \leq e^{-2\varepsilon^2 t}. \tag{B.1}$$

We now bound the disagreement probabilities $q_j(t, B^t)$. Take j rejected by Online, i.e., it must be that $j > 1$ and $B^t/t < \bar{p}_j - p_j/2$. Since we are rejecting, a compensation is paid only when condition (1) applies, thus $X_j^{\star t} = Z_j(t)$. By the structure of Offline's solution, all classes $j' \leq j$ are accepted in the last t rounds, i.e., it must be that $X_{j'}^{\star t} = Z(t)_{j'}$ for all $j' \leq j$. We must be in the event $\sum_{j' \leq j} Z(t)_{j'} \leq B^t$. We know that $\sum_{j' \leq j} Z(t)_{j'} \sim \text{Bin}(t, \bar{p}_j)$. Since $B^t/t < \bar{p}_j - p_j/2$, the probability of error is:

$$q_j(t, B^t) \leq \mathbb{P}\left[\sum_{j' \leq j} Z(t)_{j'} \leq B^t\right] = \mathbb{P}[\text{Bin}(t, \bar{p}_j) \leq B^t] \leq \mathbb{P}[\text{Bin}(t, \bar{p}_j) \leq t\bar{p}_j - tp_j/2].$$

Using Eq. (B.1), it follows that $q_j(t, B^t) \leq e^{-p_j^2 t/2}$.

Now let us consider when j is accepted by Online. A compensation is paid only when $j > 1$ and condition (2) applies, thus $X_j^{\star t} = 0$. Again, by the structure of $X^{\star t}$, necessarily $X_{j'}^{\star t} = 0$ for $j' \geq j$. Therefore we must be in the event $\sum_{j' < j} Z(t)_{j'} \geq B^t$. Recall that j is accepted iff $B^t/t \geq \bar{p}_j - p_j/2 = \bar{p}_{j-1} + p_j/2$, thus

$$q_j(t, B^t) \leq \mathbb{P}\left[\sum_{j' < j} Z(t)_{j'} \geq B^t\right] = \mathbb{P}[\text{Bin}(t, \bar{p}_{j-1}) \geq B^t] \leq \mathbb{P}[\text{Bin}(t, \bar{p}_{j-1}) \geq t\bar{p}_{j-1} + tp_j/2].$$

This event is also exponentially unlikely. Using Eq. (B.1), we conclude $q_j(t, B^t) \leq e^{-p_j^2 t/2}$. Overall we can bound the total compensation as:

$$\sum_{t \leq T} q(t, B^t) \leq \sum_{j>1} p_j \sum_{t \leq T} e^{-p_j^2 t/2} \leq \sum_{j>1} p_j \frac{2}{p_j^2}.$$

Using compensated coupling (Lemma 3.3.9), we get our result. □

Proof of Corollary 3.4.3. By Corollary 3.3.12, if A^t is the action using over-estimates $\hat{q}$, then $\mathbb{E}[\text{Reg}] \leq r_\varphi \sum_{t \in [T]} (\mathbb{E}[\hat{q}(t, A^t, B^t)] + \Delta^t)$. Recall that A^t is chosen to minimize

disagreement, hence, given the condition $|q(t,a,b) - \hat{q}(t,a,b)| \leq \Delta^t$, we have $\hat{q}(t, A^t, B^t) \leq \min_{a \in \mathcal{U}} q(t, a, B^t) + \Delta^t$. In conclusion,

$$\mathbb{E}[\textsc{Reg}] \leq r_\varphi \sum_{t \in [T]} \left(\mathbb{E}\left[\min_{a \in \mathcal{U}} q(t, a, B^t) \right] + 2\Delta^t \right).$$

We prove that $\min_{a \in \mathcal{U}} q_j(t, a, b) \leq e^{-p_j^2 t/2}$ for all $t \in [T], j \in [n], b \in \mathbb{N}$, hence the corollary follows by summing all the terms.

Let us denote $a = 1$ the action accept and $a = 0$ reject. In the proof of Theorem 3.4.2 we concluded that the following are over-estimates of the disagreement probabilities q:

$$\hat{q}_j(t,1,b) = \begin{cases} e^{-p_j^2 t/2} & \text{if } \frac{X_j^t}{t p_j} \geq 1/2 \\ 1 & \text{otherwise.} \end{cases} \quad \text{and} \quad \hat{q}_j(t,0,b) = \begin{cases} e^{-p_j^2 t/2} & \text{if } \frac{X_j^t}{t p_j} < 1/2 \\ 1 & \text{otherwise.} \end{cases} \tag{B.2}$$

Crucially, observe that the term $e^{-p_j^2 t/2}$ is *independent of the state* b. This proves that $\sup_{b \in \mathbb{N}} \min\{q_j(t,0,b), q_j(t,1,b)\} \leq e^{-p_j^2 t/2} \, \forall \, t \in [T], \, \forall \, j \in \mathcal{J}$. The proof is completed. □

B.2.4 Other Arrival Processes

Proof of Example 3.4.12. This follows from an application of [42, Theorem 3.1], which guarantees that, for some constants c', m that depend on P only,

$$\mathbb{P}[|Z_k(t) - \nu_k t| \geq \delta \nu_k t] \leq c' e^{-\frac{\delta^2 \nu_k t}{72m}}, \quad \forall t \in [T], \delta \in [0,1], k \in [n]. \tag{B.3}$$

To obtain Eq. (3.10), we fix $j \in [n]$ and use a union bound taking the worst case in Eq. (B.3); we let $\nu_{\min} \coloneqq \min_{k \in [n]} \nu_k$, $\nu_{\max} \coloneqq \max_{k \in [n]} \nu_k$ and set $\delta = \nu_j / 2\kappa_j \nu_{\max}$ in Eq. (B.3) to obtain the result. The constants are thus $c_j = (\nu_j / 2\kappa_j \nu_{\max})^2 \nu_{\min} / 72m$. Finally, we mention that the constants c' and m are related to the spectral gap and mixing time of P, for details see [42]. □

PROOF OF EXAMPLE 3.4.13. To prove the All Time Deviation, we use that, from the proof of [52, Lemma 3], $\mathbb{P}(|X - \mathbb{E}[X]| \geq \varepsilon \mathbb{E}[X]) \leq 2e^{-\mathbb{E}[X]\varepsilon^2/4}$ is valid for any Poisson r.v. X and any $\varepsilon > 0$. Now we proceed as in Example 3.4.12: taking $X_k = Z_k(t)$ and $\varepsilon = \frac{\mathbb{E}[Z_j(t)]}{2\kappa_j \mathbb{E}[Z_k(t)]}$ we obtain

$$\mathbb{P}\left[||Z(t) - \mathbb{E}[Z(t)]||_\infty \geq \frac{\mathbb{E}[Z_j(t)]}{2\kappa_j}\right] \leq 2 \sum_{k \in [n]} e^{-\frac{\mathbb{E}[Z_j(t)]^2}{8\kappa_j^2 \mathbb{E}[Z_k(t)]}}.$$

Finally, from Eq. (3.11) we have $\mathbb{E}[Z_k(t)] \leq g(t)\mathbb{E}[Z_j(t)]$ and from Eq. (3.12) we have $\mathbb{E}[Z_j(t)] \geq g(t)f(t)\log(t)$. From these bounds, we conclude

$$\mathbb{P}\left[||Z(t) - \mathbb{E}[Z(t)]||_\infty \geq \frac{\mathbb{E}[Z_j(t)]}{2\kappa_j}\right] \leq 2ne^{-f(t)\log(t)/8\kappa_j^2}$$

and the existence of constants τ_j, c_j satisfying the All Time Deviation follows.

□

B.2.5 Proof of Proposition 3.5.3

We denote $\mathbf{x} \in \mathbb{R}^{nd}$ the vector of the form $\mathbf{x} = (x_{11}, x_{21} \ldots, x_{d1}, x_{12}, \ldots)'$, i.e., we concatenate the components x_{ij} by j first. We can write the feasible region of $P[z, b]$ as $\{\mathbf{x} : C\mathbf{x} \leq b, D\mathbf{x} \leq z, \mathbf{x} \geq 0\}$, where $C \in \mathbb{R}^{d \times nd}$ and $D \in \mathbb{R}^{n \times nd}$. It follows from a slight strengthening of [84, Theorem 2.4] that $||\mathbf{x}^1 - \mathbf{x}^2||_\infty \leq \kappa ||z^1 - z^2||_1$, where

$$\kappa = \sup\left\{ ||v||_\infty : ||C'u + D'v||_1 = 1, \text{ support} \begin{pmatrix} u \\ v \end{pmatrix} \text{corresponds to l.i. rows of} \begin{pmatrix} C \\ D \end{pmatrix} \right\}$$

If we study Eq. (3.13), denoting I_d the d-dimensional identity and $1_d, 0_d$ d-dimensional row vectors of ones and zeros, we can write the matrices C, D as follows. We sketched the

multipliers u_i, v_j next to the rows,

$$C = [I_d|I_d|\dots|I_d] = \begin{pmatrix} 1 & 0 & \cdots & 0 & 1 & \cdots \\ 0 & 1 & \cdots & 0 & 0 & \cdots \\ \vdots & \vdots & \ddots & \vdots & \vdots & \cdots \\ 0 & 0 & \cdots & 1 & 0 & \cdots \end{pmatrix} \begin{matrix} \leftarrow u_1 \\ \leftarrow u_2 \\ \vdots \\ \leftarrow u_d \end{matrix}$$

and similarly

$$D = \begin{pmatrix} 1_d & 0_d & \cdots & 0_d \\ 0_d & 1_d & \cdots & 0_d \\ \vdots & \vdots & \ddots & \vdots \\ 0_d & 0_d & \cdots & 1_d \end{pmatrix} \begin{matrix} \leftarrow v_1 \\ \leftarrow v_2 \\ \vdots \\ \leftarrow v_n \end{matrix}$$

We have two cases: either $u_i = 0$ for some $i \in [d]$ or $u_i \neq 0$ for all $i \in [d]$. On the first case, say w.l.o.g. $u_1 = 0$ and take any $j \in [n]$. Observe that the constraint $||C'u + D'v||_1 = 1$ implies (studying all the components involving j) $\sum_{i\in[d]}|u_i + v_j| \leq 1$. Since $u_1 = 0$, this reads as $|v_j| + \sum_{i>1}|u_i + v_j| \leq 1$, thus $|v_j| \leq 1$ as desired.

For the other case we assume $u_i \neq 0$ for all i, hence $v_j = 0$ for some j, since otherwise we would violate the l.i. restriction on the support. Assume w.l.o.g. $v_1 = 0$ and let us study some v_j. The constraint $||C'u + D'v||_1 = 1$ implies (looking at the first n components and the components involving j) $\sum_{i\in[d]}|u_i| + \sum_{i\in[d]}|v_j + u_i| \leq 1$. By triangle inequality,

$$d|v_j| = \left|\sum_{i\in[d]}(u_i + v_j) - \sum_{i\in[d]} u_i\right| \leq \sum_{i\in[d]}|v_j + u_i| + \sum_{i\in[d]}|u_i| \leq 1.$$

This shows $|v_j| \leq 1$ and the proof is complete. □

B.2.6 Additional Details for Online Stochastic Matching

The stochastic bipartite matching is defined by a set of static nodes U, $|U| = d$, and a random set of nodes arriving sequentially. At each time a node ξ^t is chosen from a set

V, $|V| = n$, and we are given its set of neighbours in U. We identify the online bipartite matching problem in our framework as follows. The state S^t encodes the available nodes from U, an action corresponds to matching the arrival $\xi^t \in V$ to a neighbour $u \in U$ of ξ^t or to discard the arrival. In the latter case we say that it is matched to $u = \varnothing$.

For a graph G, we denote the size of its maximum matching as $M(G) \in \mathbb{N} \cup \{0\}$ and $G - (u, v)$ as the usual removal of nodes; in the case $u = \varnothing$, $G - (u, v) = G - v$. Recall that $Q(t, a, s)$ is the event when OFFLINE is not satisfied with action a and $q(t, a, s) = \mathbb{P}[Q(t, a, s)]$. Let us fix an ONLINE policy and define $G_t = (L, R)$ as the bipartite graph with nodes $L = S^t$ and $R = Z(t)$, i.e., the realization of future arrivals and current state. With the convention $\mathbb{1}_\varnothing = 0$ and $\mathbb{1}_u = 1$ for $u \in U$,

$$\bar{Q}(t, u, s) = \{\omega \in \Omega : M(G_t) = \mathbb{1}_u + M(G_t - (u, \xi^t))\}.$$

In words, OFFLINE is satisfied matching ξ^t to u if the size of the maximum matching with and without that edge differs by exactly 1. With this observation, a straightforward application of the compensated coupling Lemma 3.3.9 yields Lemma 3.5.4.

Finally, we provide an example for a negative result. Despite the fact that the regret is exactly the number of disagreements and the Bayes Selector minimizes each term, it is not an optimal policy.

Proposition B.2.2. *The Bayes Selector is sub-optimal for stochastic online bipartite matching.*

PROOF. Consider an instance with static nodes $U = \{a, b, c\}$ and four types of online nodes $V = [4]$. Type 1 matches to a only, 2 to a and b, 3 to c only, and 4 to b and c. Observe that the only types inducing error are 2 and 4.

Assume the arrival at $t = 3$ is $\xi^3 = 2$. Matching it to a is an error if arrivals are $\{1, 1\}, \{1, 3\}, \{1, 4\}$, so the disagreement is $p_1^2 + 2p_1p_3 + 2p_1p_4$. Matching it to b is an error

if arrivals are $\{4,4\},\{3,4\}$ with disagreement $p_4^2 + 2p_4p_3$. Now assume $p_4^2 + 2p_4p_3 = p_1^2 + 2p_1p_3 + 2p_1p_4$, so the bayes selector is indifferent and thus say it matches to a.

At $t = 2$, there is only an error if $\xi^2 = 4$, in which case matching it to b has disagreement p_2 and matching it to c disagreement p_3. In conclusion to the bayes selector pays $p_1^2 + 2p_1p_3 + 2p_1p_4$ in the first stage plus $\min\{p_2, p_3\}$ in the second with probability p_4.

The strategy that matches at $t = 3$ type 2 to b has disagreement $p_4^2 + 2p_4p_3 = p_1^2 + 2p_1p_3 + 2p_1p_4$, thus lower than the bayes selector. To see this, note that if we match to b there is no error at $t = 2$.

Finally, the equation $p_4^2 + 2p_4p_3 = p_1^2 + 2p_1p_3 + 2p_1p_4$ is satisfied, e.g., with $p_1 = p_4/2$ and $p_3 = p_4/4$. □

B.3 Additional Details from Numerical Experiments

Competitive is described as follows. For a given horizon T, let $K_j := \lceil p_j T \rceil$. We create a bipartite graph $G = (U, V, E)$, where U is the static side and V the online side. The static side contains B_i copies of each resource i, hence $|U| = \sum_{i\in[d]} B_i$. The online side contains K_j copies of each type j, hence $|V| = \sum_{j\in[n]} K_j$. The edge set E is the natural construction where each copy of $j \in [n]$ has edges to all copies of $i \in [d]$ according to the adjacency matrix A. The weight w_e on edge $e = (u, v)$ is r_{ij} if u is a copy of i and v is a copy of j. Finally, define the following matching LP on the graph G, where λ_{iljk} stands for the l-th copy of i

and k-th copy of j

$$\begin{aligned}(P)\ \max \quad & \textstyle\sum_{i\in[d]}\sum_{l\in[B_i]}\sum_{j\in[n]:A_{ij}=1}\sum_{k\in[K_j]} r_{ij}\lambda_{iljk} \\ \text{s.t.} \quad & \textstyle\sum_{j\in[n]}\sum_{k\in[K_j]}\lambda_{iljk} \le 1 \quad \forall i\in[d], l\in[B_i] \\ & \textstyle\sum_{i\in[d]}\sum_{l\in[B_i]}\lambda_{iljk} \le 1 \quad \forall j\in[n], k\in[K_j] \\ & \lambda \ge 0,\end{aligned}$$

Let $\lambda^\star$ be a solution to this LP. Whenever a type j arrives, Competitive draws $k \in [K_j]$ uniformly at random, then takes a vertex $u = il$ incident to $v = jk$ with probability $\lambda^\star_{iljk}$ and if $u = il$ is not taken, it matches v to u. We note that the process of copying nodes is not superfluous since the analysis of Competitive heavily relies on the fact that the LP is in this form.

Marginal Allocation is described as follows. Let x be a solution of (P_T) in Eq. (3.15), i.e., of the fluid LP, and let $f_i : [T] \times \{0, \dots, B_i\} \to \mathbb{R}_{\ge 0}$ be some functions specified later. When a type j arrives at t and the current budgets are $B^t \in \mathbb{N}^d$, Marginal Allocation uses $f_i(t, B_i^t) - f_i(t, B_i^t - 1)$ as the bid-price for each resource $i \in [d]$: the type is rejected if $r_{ij} < f_i(t, B_i^t) - f_i(t, B_i^t - 1)$ for all $i \in [d]$ such that $B_i^t > 0$ and otherwise it is matched to $\operatorname{argmax}\{r_{ij} - f_i(t, B_i^t) + f_i(t, B_i^t - 1) : i \in [d], B_i^t > 0\}$. Finally, the functions f are obtained with the following recursion

$$f_i(t+1, b) = f_i(t, b) + \frac{1}{T}\sum_{j\in[n]} x_{ij}(r_{ij} - f_i(t, b) + f_i(t, b-1))^+, \qquad f_i(1, \cdot) = 0, f_i(\cdot, 0) = 0.$$

		Type j									
		1	2	3	4	5	6	7	8	9	10
Resource i	1	10	6	0	0	9	8	2	0	0	1
	2	1	0	0	0	0	0	2	0	0	8
	3	0	0	0	0	0	0	2	0	0	6
	4	0	26	0	0	1	0	3	0	0	11
	5	1	4	0	0	0	0	0	0	0	13
	6	7	4	12	11	10	12	18	2	0	0
	p_j	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1	0.1

Table B.1: Parameters used for the second online matching instance. Coordinates (i, j) represent the reward r_{ij} and $r_{ij} = 0$ implies that it is not possible to match i to j.

APPENDIX C

APPENDIX FOR CHAPTER 4

C.1 Fast Restock

In this section we prove Lemma 4.2.2, i.e., that when Assumption 4.2.1 (slow restock) is violated, it is not possible to obtain constant regret against the offline benchmark. We consider as a counter example a network with one request type and one resource, both arriving with probability p. Figure C.1 is the numerical illustration of the $\sqrt{T}$ regret.

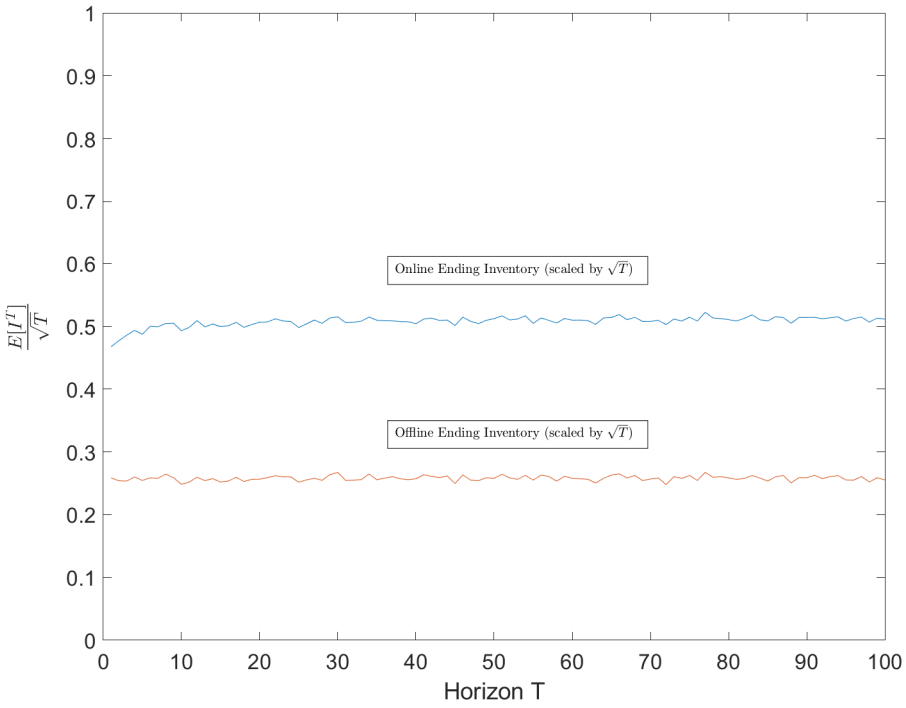


Figure C.1: Singe RAN with impatient requests and restock. The first line depicts the expected remaining inventory of the offline. The second line depicts the expected remaining inventory of the optimal online policy. The difference is the regret. The expectation is computed as an average over 10000 replications. The y-axis is the (expected) ending inventory divided by $\sqrt{T}$. The horizon runs between T=100 to T=10000.

We label the request by 1, the resource by a, and set $v_1 = 1$, $p_1 = p_a = p$. Let Z_1^t

be the number of request arrivals by time t and let Z_a^t be the restock by time t. Let I_{off}^T (respectively I_{on}^T) be the end-of-horizon residual inventory under offline (respectively online) policies. Then, $V_{off}^*(T) = \mathbb{E}[\min\{Z_a^t, Z_1^t\}] = \mathbb{E}[Z_a^T - I_{off}^T]$.

The optimal online policy is to always serve if possible. Let Y^t be the number of requests accepted by the online policy by time t. Then, $V_{on}^* = \mathbb{E}[Y^T] = \mathbb{E}[Z_a^T - I_{on}^T]$. Thus, $V_{off}^*(T) - V_{on}^*(T) = \mathbb{E}[I_{on}^T] - \mathbb{E}[I_{off}^T]$. We analyze now the two (end-of-horizon) inventory levels, starting with offline, which satisfies:

$$I_{off}^T = (Z_a^T - Z_1^T)^+.$$

The process $Z_a^t - Z_1^t$ is a random walk starting at 0 and with i.i.d zero-mean increments $X_1, \ldots, X_T$ taking values $\{-1, 0, 1\}$ with probabilities $p(1-p), 1-2p(1-p), p(1-p)$; X_t is the difference between the restock at t (0 or 1) and the request arrival (0 or 1). Write $S^T = \sum_{t=1}^T X_t$. By the central limit theorem

$$\frac{1}{\sqrt{T}} \sum_{i=1}^{T} X_i \Rightarrow \mathcal{N}(0, \sigma^2),$$

where $\sigma^2 \coloneqq 2p(1-p)$. Since $I_{off}^T = (S^T)^+$, by the continuous mapping theorem we have $\frac{1}{\sqrt{T}}\mathbb{E}[I_{off}^T] \to \mathbb{E}[(\mathcal{N}(0, \sigma^2))^+]$. On the other hand, the online inventory satisfies the queueing recursion $I_{on}^{t+1} = [I_{on}^t + X_t]^+$ so that

$$I_{on}^T = \sup_{t \le T}(S^T - S^t).$$

It is well-known (reflection principle) that the following equivalence in law holds

$$\sup_{t \le T}(S^T - S^t) \stackrel{\mathcal{L}}{=} \sup_{t \le T} S^t.$$

It is also well-known that

$$\frac{1}{\sqrt{T}} \sup_{t \le T} S^t \Rightarrow \mathcal{Z},$$

where $\mathcal{Z}$ has the distribution of the supremum of a Brownian motion. The convergence holds again also in expectation. The reflection principle for Brownian motion then guarantees that

$\mathbb{P}\{\mathcal{Z}' > a\} = 2\mathbb{P}\{\mathcal{N}(0,\sigma^2) > a\}$ and this allows us to conclude that

$$\mathbb{E}[I_{on}^T]/\mathbb{E}[I_{off}^T] \to 2 \quad \text{and} \quad \frac{1}{\sqrt{T}}(\mathbb{E}[I_{on}^T] - \mathbb{E}[I_{off}^T]) \to \mathbb{E}[\mathcal{N}(0,\sigma^2)^+] > 0, \quad \text{as } T \to \infty.$$

C.2 Proofs of Auxiliary Lemmas

Restatement of Lemma 4.4.10. *Fix a centroid $\mathcal{K}$ with associated basis $\mathcal{B}$. Let (K^+, K^-, K^0) be the zero-valued basic variables (Definition 4.4.7). Then,*

1. *The basis $\mathcal{B}$ is optimal for any right-hand side (R, D) of the form*

$$R = r_{\mathcal{K}}(D) + \alpha(A_{\kappa^+}D_{\kappa^+} - A_{\kappa^-}D_{\kappa^-}) + b,$$

   where $\kappa^+ \subseteq K^+, \kappa^- \subseteq K^-, \alpha \in [0,1]$, and $b \in \mathbb{R}^d_{\geq 0}$ is zero for components not in K^0, i.e., $b_i = 0$ for $i \notin K^0$. In particular, the set $\mathcal{K} \cup \kappa^+ \backslash \kappa^-$ is a centroid it is a neighbor of $\mathcal{K}$.

2. *The basis $\mathcal{B}$ is optimal for (R, D) if and only if R is of the form*

$$R = r_{\mathcal{K}}(D) + \sum_{\kappa^+ \subseteq K^+, \kappa^- \subseteq K^-} \alpha_{(\kappa^+,\kappa^-)}(A_{\kappa^+}D_{\kappa^+} - A_{\kappa^-}D_{\kappa^-}) + b,$$

   where b is as before, $\alpha \geq 0$, and $\sum_{\kappa^+ \subseteq K^+, \kappa^- \subseteq K^-} \alpha_{(\kappa^+,\kappa^-)} = 1$.

3. *$R \in \mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$ if and only if*

$$R - r_{\mathcal{K}}(D) = A_{K^+}x_{K^+} - A_{K^-}x_{K^-} + b,$$

   where $x_j \in [0, D_j/2]$ for $j \in K^+ \cup K^-$ and b is as before.

Proof of Lemma 4.4.10. **Item 1.** In virtue of Lemma 4.4.1, it suffices to show that

$\mathcal{B}^{-1}\begin{pmatrix} R \\ D \end{pmatrix} \geq 0$ to conclude the optimality of $\mathcal{B}$. Recall that the matrix $\bar{A}$ is given by

$$\bar{A} = \begin{bmatrix} A & 0 & I^d \\ I^n & I^n & 0 \end{bmatrix},$$

where the columns are associated, respectively to request $y \in \mathbb{R}^n$, unmet variables $u \in \mathbb{R}^n$ and surplus $s \in \mathbb{R}^d$. The sub-matrix $\mathcal{B}$ has a subset of these columns and can be written as

$$\mathcal{B} = \begin{bmatrix} A_{\mathcal{K}\cup K^+} & 0 & I^d_{K^0} \\ I^n_{\mathcal{K}\cup K^+} & I^n_{\mathcal{K}^c\cup K^-} & 0 \end{bmatrix}.$$

Recall that $\mathcal{B}$ is of dimension $(n+d)\times(n+d)$, and each column is associated to either a variable y_j, u_j, or s_i. We will write vectors of dimension $n+d$ in this same order, specifying the components associated to y, u, and s respectively, which is the same order in which we wrote $\bar{A}$ and $\mathcal{B}$ above.

By definition of centroid, all request variables $\mathcal{K}$ are saturated at $r_{\mathcal{K}}$, hence $K^+ \subseteq \mathcal{K}^c$; in other words, zero-valued requests cannot come from $\mathcal{K}$. Similarly, unfulfilled variables $\mathcal{K}^c$ are saturated at $r_{\mathcal{K}}$, therefore $K^- \subseteq \mathcal{K}$. We deduced the inclusions $\kappa^+ \subseteq \mathcal{K}^c \cap \mathcal{K}^+$ and $\kappa^- \subseteq \mathcal{K} \cap \mathcal{K}^-$. Using the previous fact, by inspection we have the following identities

$$\mathcal{B}\begin{pmatrix} D_{\kappa^-} \\ -D_{\kappa^-} \\ 0 \end{pmatrix} = \begin{pmatrix} A_{\kappa^-} D_{\kappa^-} \\ 0 \end{pmatrix}, \qquad \mathcal{B}\begin{pmatrix} D_{\kappa^+} \\ -D_{\kappa^+} \\ 0 \end{pmatrix} = \begin{pmatrix} A_{\kappa^+} D_{\kappa^+} \\ 0 \end{pmatrix}, \qquad \mathcal{B}\begin{pmatrix} 0 \\ 0 \\ b_{K^0} \end{pmatrix} = \begin{pmatrix} b_{K^0} \\ 0 \end{pmatrix}.$$

We are now ready to prove $\mathcal{B}^{-1}\begin{pmatrix} R \\ D \end{pmatrix} \geq 0$. Indeed, pre-multiplying the previous identities

by $\mathcal{B}^{-1}$,

$$\mathcal{B}^{-1}\begin{pmatrix} R \\ D \end{pmatrix} = \mathcal{B}^{-1}\left[\begin{pmatrix} A_{\mathcal{K}}D_{\mathcal{K}} \\ D \end{pmatrix} + \alpha\begin{pmatrix} A^{+}_{\kappa}D^{+}_{\kappa} \\ 0 \end{pmatrix} - \alpha\begin{pmatrix} A^{-}_{\kappa}D^{-}_{\kappa} \\ 0 \end{pmatrix} + \begin{pmatrix} b \\ 0 \end{pmatrix}\right]$$
$$= \begin{pmatrix} D_{\mathcal{K}} \\ D_{\mathcal{K}^c} \\ 0 \end{pmatrix} + \alpha\begin{pmatrix} D_{\kappa^+} \\ -D_{\kappa^+} \\ 0 \end{pmatrix} - \alpha\begin{pmatrix} D_{\kappa^-} \\ -D_{\kappa^-} \\ 0 \end{pmatrix} + \begin{pmatrix} 0 \\ 0 \\ b_{K^0} \end{pmatrix}.$$

Since we have the inclusions $\kappa^+ \subseteq \mathcal{K}^c \cap \mathcal{K}^+$ and $\kappa^- \subseteq \mathcal{K} \cap \mathcal{K}^-$, it can be verified that the last expression is non-negative, completing the proof of the first item.

Item 2. Assume R has the stated form and let us prove that $\mathcal{B}$ is optimal. If two right-hand sides have the same optimal candidate basis, then, in virtue of Lemma 4.4.1, any non-negative combination of the right-hand sides has the same optimal basis. Finally, by the first item of the lemma, we are taking non-negative combinations of right-hand sides which have $\mathcal{B}$ as optimal basis, so we conclude optimality.

We turn to prove that, if $\mathcal{B}$ is optimal, the R has the stated representation. Let $\begin{pmatrix} y \\ u \\ s \end{pmatrix} = \mathcal{B}^{-1}\begin{pmatrix} R \\ D \end{pmatrix}$, where y are the request variables, u are unmet variables s are surplus variables. Let K^+, K^- and K^0 as in Definition 4.4.7. By definition of centroid and the optimality of $\mathcal{B}$, we have the following:

$$y_j = D_j \text{ and } u_j = 0 \quad \forall j \in \mathcal{K}\backslash K^-,$$
$$y_j = 0 \text{ and } u_j = D_j \quad \forall j \in \mathcal{K}^c\backslash K^+.$$

For all other indices j, we know that $u_j = D_j - y_j$ and $y_j, u_j \geq 0$. Since $R \in \mathcal{N}_{\mathcal{K}}$ we also know that $y_j \geq D_j/2$ for all $j \in K^-$ and that $y_j < D_j/2$ for all $j \in K^+$. From the vector

(y, u, s) we subtract the vector $(\bar{y}, \bar{u}, \bar{s})$ given by

$$\bar{y}_j = \begin{cases} D_j & \text{for } j \in \mathcal{K} \backslash K^-, \\ 0 & \text{for } j \in \mathcal{K}^c \backslash K^+, \\ D_j/2 & \text{for } j \in K^- \\ 0 & \text{otherwise.} \end{cases}$$

and $\bar{u}_j = D_j/2$ for $j \in K^-$, $\bar{u}_j = D_j$ for $j \in K^+$. We also put $\bar{s} = s$ (subtracting fully the budget slack variables). Since $R \in \mathcal{N}_{\mathcal{K}}$, by definition that $y \geq \bar{y}$ and $u \leq \bar{u}$. Thus, $(y - \bar{y}, u - \bar{u}, s - \bar{s}) = (y - \bar{y}, u - \bar{u}, 0)$. We will study next the vector $(y - \bar{y}, u - \bar{u})$.

For convenience, let us re-label (and re-order) the indices so that indices in $K^+ \cup K^-$ are in the top of the vector (y, u, s). This portion of (y, u, s) then has the form

$$z = \begin{pmatrix} y_{K^+} \\ y_{K^-} - D_{K^-}/2 \\ u_{K^+} - D_{K^+} \\ u_{K^-} - D_{K^-}/2 \end{pmatrix} = \begin{pmatrix} y_{K^+} \\ y_{K^-} - D_{K^-}/2 \\ -y_{K^+} \\ D_{K^-}/2 - y_{K^-} \end{pmatrix}.$$

We will identify a representation for z. Since all other entries of (y, u, s) have fixed values, we will then append those to all vectors in the resulting combination.

Let us apply the following transformation

$$x = Pz \text{ where } P = 2\,\mathrm{diag}(1/D_{K^+}, 1/D_{K^-}, 1/D_{K^+}, 1/D_{K^-}).$$

By definition, all request elements of Pz are in $[0, 1]$ and unmet elements are in $[-1, 0]$. If x can be written as a convex combination of vectors $x_1, \ldots, x_m$ then $(y, u) - (\bar{y}, \bar{u})$ can be written as a convex combiantion of $P^{-1}x_1, \ldots, P^{-1}x_m$.

Vectors $x = Pz$ are elements in the polyhedron

$$\left\{ \begin{pmatrix} x_{K^+} \\ x_{K^-} \\ \mathfrak{s}_{K^+} \\ \mathfrak{s}_{K^-} \end{pmatrix} : x_j + \mathfrak{s}_j = 0, x_j \in [0,1], \mathfrak{s}_j \in [0,1] \right\}.$$

This polyhedron is integral. Indeed, this follows by noting that the constraint matrix is totally unimodular as it consists of only $\{0,1\}$ entries and has a single 1 per column. In turn, we can write each such vector as a convex combination of *binary* vectors of the form

$$\begin{pmatrix} x_{K^+} \\ x_{K^-} \\ -x_{K^+} \\ -x_{K^-} \end{pmatrix},$$

where $x_j \in \{0,1\}$. For such a vector x we have a set $\kappa^+ \subseteq K^+$ of entries such that $x_j = 1$ for $j \in \kappa^+$ and a set $\kappa^- \subseteq K^-$ with $x_j = 0$ for $j \in \kappa^-$. Thus, each of these binary vectors can be written as

$$\begin{pmatrix} e_{\kappa^+} \\ 0_{K^+ \setminus \kappa^+} \\ 0_{\kappa^-} \\ e_{K^- \setminus \kappa^-} \\ -e_{\kappa^+} \\ 0_{K^+ \setminus \kappa^-} \\ 0_{\kappa^-} \\ -e_{K^- \setminus \kappa^-} \end{pmatrix},$$

for some subsets $\kappa^+ \subseteq K^+$ and $\kappa^- \subseteq K^-$. Transforming back (multiplying by D^{-1} and adding $\bar{y}, \bar{u}$), we have that we can write (y, u, s) as a convex combination of vectors of the

form

$$\begin{pmatrix} D_{\kappa^+}/2 \\ 0_{K^+\setminus\kappa^+} \\ D_{\kappa^-}/2 \\ D_{\mathcal{K}\setminus\kappa^-} \\ D_{\kappa^+}/2 \\ D_{K^+\setminus\kappa^+} \\ D_{\kappa^-}/2 \\ 0_{\mathcal{K}\setminus\kappa^-} \\ s \end{pmatrix}.$$

Notice that multiplying this vector by $\mathcal{B}$ we get a vector of the form

$$r_{\kappa^+,\kappa^-,u} = r_{\mathcal{K}} + A_{\kappa^+}D_{\kappa^-}/2 - A_{\kappa^-}D_{\kappa^+}/2 + s$$

where we use the fact that $\mathcal{B}s$ (multiplying by vector of surplus) gives back the surplus. We conclude that we can write the top elements of y as a sum of a vector s and a convex combination of vectors (y, u) of the desired form.

Item 3. We just proved that $\mathcal{B}$ is optimal for (R, p) iff it can be written as

$$R = r_{\mathcal{K}}(D) + \sum_{\kappa^+\subseteq K^+,\kappa^-\subseteq K^-} \alpha_{(\kappa^+,\kappa^-)}(A_{\kappa^+}D_{\kappa^+} - A_{\kappa^-}D_{\kappa^-}) + b.$$

Observe that the sum ranges over subsets of K^+, K^-. Let us group it instead for each $j \in K^+ \cup K^-$. With this end, define

$$\alpha_j \coloneqq \sum_{(\kappa^+,\kappa^-):j\in\kappa^+} \alpha(\kappa^+,\kappa^-) \text{ for } j \in K^+ \quad \text{and} \quad \alpha_j \coloneqq \sum_{(\kappa^+,\kappa^-):j\in\kappa^-} \alpha(\kappa^+,\kappa^-) \text{ for } j \in K^-.$$

Now, if we put $x_j \coloneqq \alpha_j D_j$, we can write $R = r_{\mathcal{K}}(D) + A_{K^+}x_{K^+} - A_{K^-}x_{K^-}$. We claim that

the solution (y, u, s) associated to the right-hand side (R, D) is

$$\begin{pmatrix} y \\ u \\ s \end{pmatrix} = \begin{pmatrix} D_{\mathcal{K}} \\ D_{\mathcal{K}^c} \\ 0 \end{pmatrix} + \begin{pmatrix} x_{K^+} - x_{K^-} \\ -x_{K^+} + x_{K^-} \\ b \end{pmatrix}.$$

Assuming this claim, we can conclude since, by definition, $R \in \mathcal{N}_{\mathcal{K}}(D, \mathcal{B})$ if (1) the basis $\mathcal{B}$ is optimal and (2) we have $y_{\mathcal{K}} \geq \frac{1}{2} D_{\mathcal{K}}$ and $y_{\mathcal{K}^c} < \frac{1}{2} D_{\mathcal{K}^c}$. Indeed, condition (2) follows by recalling $K^+ \subseteq \mathcal{K}^c$, $K^- \subseteq \mathcal{K}$ and our definition $x_j = \alpha_j D_j$.

We are left to prove the claim. Using that the variables $\{y_j, u_j : j \in K^- \cup K^+\}$ and s_{K^0} are in the basis $\mathcal{B}$ we have

$$\mathcal{B} \begin{pmatrix} x_{K^+} - x_{K^-} \\ -x_{K^+} + x_{K^-} \\ b_{K^0} \end{pmatrix} = \begin{pmatrix} A_{K^+} x_{K^+} - A_{K^-} x_{K^-} + b \\ 0 \end{pmatrix}.$$

Finally, by definition of centroid

$$\mathcal{B} \begin{pmatrix} D_{\mathcal{K}} \\ D_{\mathcal{K}^c} \\ 0 \end{pmatrix} = \begin{pmatrix} r_{\mathcal{K}}(D) \\ D \end{pmatrix}.$$

The last two equations together prove the claim in virtue of Lemma 4.4.1. □

Proof of Lemma 4.4.13. The existence of separating vectors that have $\max_l \psi_l' x \leq 0$ for any $x \in \text{cone}(\mathcal{K}, \mathcal{B})$ follows from the standard results.

Per our construction of the set $\mathcal{N}_{\mathcal{K}}(\mathcal{B}, D)$ a vector x is in the cone if and only if x can be written as

$$x - r_{\mathcal{K}}(D) = \sum_{(\kappa^+, \kappa^-)} \alpha[\kappa^+, \kappa^-](A_{\kappa^+} z_{\kappa^+} - A_{\kappa^-} z_{\kappa^-}),$$

where $z \geq 0$. It follows immediately that in the convex combination it suffices to include κ^+ and κ^- that are minimal. That is such that $|\kappa^+| = 1$ or that $|\kappa^-| = 1$. We refer to these as extreme neighbors.

In turn, the separating vector must have $\psi_l = \psi[\kappa^+, \kappa^-]' A_{\kappa^+} = 0$ for those extreme neighbors. By the requirement that $\max_l \psi_l' x \leq 0$ it must also be that for other A_j that do not define this cone but do define the cone of $\mathcal{B}'$ (for some other basis $\mathcal{B}'$ that is optimal at $\mathcal{K}$) we have $\psi'[\kappa^+, \kappa^-]' A_j = 1$ if $j \in cup_{\mathcal{B}' \neq \mathcal{B}} K^+(\mathcal{B}')$ and $\psi'[\kappa^+, \kappa^-]' A_j = -1$ if $j \in \cup_{\mathcal{B}' \neq \mathcal{B}} K^-(\mathcal{B}')$. □

Proof of Lemma 4.4.15. Note that, if $\mathbb{E}[R^0_{\mathcal{K}}] = \frac{1}{T}(I^0 + p_{\mathcal{R}} T - A_{\mathcal{K}} Q^0_{\mathcal{K}}) \in \mathcal{N}_{\mathcal{K}}(p)$, then since $R^0 = \frac{1}{T}(I^0 + Z^T_{\mathcal{R}} - A_{\mathcal{K}} Q^0_{\mathcal{K}})$ we have for $\bar{Z}^T \in \mathcal{A}^\epsilon$, $d(R^0_{\mathcal{K}}, \mathcal{N}_{\mathcal{K}}(p)) \leq 2\epsilon$. Let $\widehat{\mathcal{N}}_{\mathcal{K}}$ be the $\mathcal{K}$ centroid neighborhood under the empirical distribution. Suppose that $R^0_{\mathcal{K}} \in \widehat{\mathcal{N}}^+_{\mathcal{K}}$ where the $+$ stands for the "bigger" neighborhood. (The one obtained by taking convex combination of the full lines rather than mid-points). If, in addition $R^0_{\mathcal{K}} - A_{\mathcal{K}} \bar{Z}^0_{\mathcal{K}} \in \text{cone}(\mathcal{N}_{\mathcal{K}}(\mathcal{B}))$. Then, it follows from our main geometric lemma that $\mathcal{B}$ is the optimal basis and from here that the solution to the LP has precisely the properties we need. To conclude the lemma notice that if $\mathbb{E}[R^0_{\mathcal{K}}] \in \mathcal{N}_{\mathcal{K}}$ then, on $\mathcal{A}^\epsilon$, $R^0_{\mathcal{K}} \in \widehat{\mathcal{N}}^+_{\mathcal{K}}$. □

Proof of Lemma 4.5.1. Let (y, u, s) be the solution to LP(R, D). Defining (y', u', s') to be the solution to LP$(R - s, D)$, we have that $y' = y$ and $u' = u$.

We will argue that we can choose ϵ' such that (i) $y_j \geq D_j/2 - \epsilon''$ for all $j \in \mathcal{K}$, (ii) $y_j \leq \epsilon''$ for all $j \notin \mathcal{K} \cup K^+$, and $u_j \leq \epsilon''$ for all $j \in \mathcal{K} \backslash K^-$. In which case we have that $R - s = \bar{R} \pm \|A\|_\infty \epsilon''$ where $\bar{R} = r_{\mathcal{K}}(D) + A_{K^+} x_{K^+} - A_{K^-} x_{K^-}$ for some x with $x_j \in [0, D_j/2]$ for all $j \in K^+ \cap K^-$.

Choosing ϵ' (and subsequently ϵ'') so that $\epsilon'' \leq \frac{\epsilon}{\sqrt{d}\|A\|_\infty}$, we conclude that $d(R - s, \mathcal{N}_{\mathcal{K}}(\mathcal{B}, D)) \leq \sqrt{d} \times d_\infty(R - s, \mathcal{N}_{\mathcal{K}}(\mathcal{B}, D)) \leq \epsilon$.

Item (i): Because, $d_\infty(R-s, \mathcal{N}_\mathcal{K}(D)) \le d(R-s, \mathcal{N}_\mathcal{K}(D)) \le \epsilon'$, the Lipschitz continuity of LP (see [46]) implies that $y_j \ge D_j/2 - M\epsilon'$ for all $j \in K^-$ and some constant M that depends on (v, A, p). Taking $\epsilon' = \epsilon''/M$ covers item (i) of the requirements.

Item (ii): Take $R \notin \mathcal{N}_\mathcal{K}(\mathcal{B}, D)$ that satisfies the assumptions and consider two cases:

First case ($\max_l \psi_l'(R - r_\mathcal{K}(D)) \le 0$): In this case, in particular, $R - r_\mathcal{K}(D) \in cone(\mathcal{K}, \mathcal{B})$. With ϵ' small, $d(R-s, \mathcal{N}_\mathcal{K}(D)) \le \epsilon'$ implies that $R - s \in \mathcal{N}_\mathcal{K}^+(D)$. Per Lemma 4.4.10, we then have that $y_j = 0$ for all $j \notin K \cup K^+$ and $u_j = 0$ for all $j \in \mathcal{K} \backslash K^-$ as required.

Second case ($\max_l \psi_l'(R - r_\mathcal{K}(D)) > 0$): Take l such that $0 < \psi_l'(R - r_\mathcal{K}(D)) \le \epsilon'$. Let κ^+, κ^- be such that $\psi_l = \psi[\kappa^+, \kappa^-]$. By Lemma 4.4.13, ψ is orthogonal to A_j for all $j \in \kappa^+ \cup \kappa^-$. Furthermore, defining $\delta_{lj} \coloneqq \psi_l' A_j$, we have $\delta_{lj} > 0$ for $j \notin \kappa^+$ but $j \in \cup_{\bar{\mathcal{B}} \neq \mathcal{B}} K^+(\bar{\mathcal{B}})$ and $\delta_{lj} < 0$ for each $j \notin \kappa^-$ but $j \in \cup_{\bar{\mathcal{B}} \neq \mathcal{B}} K^-(\bar{\mathcal{B}})$. We thus have that $\psi'(R - r_\mathcal{K}(D)) = \psi'(\sum_{j \notin \mathcal{K} \cup \kappa^+} A_j y_j + \sum_{j \in \mathcal{K} \backslash \kappa^-} A_j u_j) = \sum_{j \notin \mathcal{K} \cup \kappa^+} \delta_{lj} y_j + \sum_{j \in \mathcal{K} \backslash \kappa^-} |\delta_{lj}| u_j$. The right-hand side is, by assumption, bounded by ϵ' and we get the desired result by taking again $\epsilon' \le \epsilon''$ for a suitable constant M.

The proof of the lemma's second part is very similar and omitted. □

Proof of Lemma 4.5.4. First we argue $\theta = \theta_\mathcal{K}(y, D) \in \text{closure}(\mathcal{Y}(\mathcal{K}, D))$. Since y solves LP(R, D) and $R \in \mathcal{N}_{\mathcal{K}^0}(D)$:

$$y_j \ge \frac{D_j}{2}\ j \in \mathcal{K} \setminus \kappa^-, \qquad y_j \ge \frac{D_j}{2}\ j \in \kappa^+, \quad \text{and} \quad y_j < \frac{D_j}{2} j \in \kappa^-.$$

This implies $\theta_j \ge \frac{D_j}{2}$ for $j \in \mathcal{K}$ and $\theta_j \le \frac{D_j}{2}$ for $j \in \mathcal{K}^c$. Finally, we claim that θ is optimal for the ratio $R^\theta \coloneqq A\theta$. Indeed, if we take $\mathcal{B}$ as the basis that $\mathcal{K}$ and $\mathcal{K}^0$ share, then the

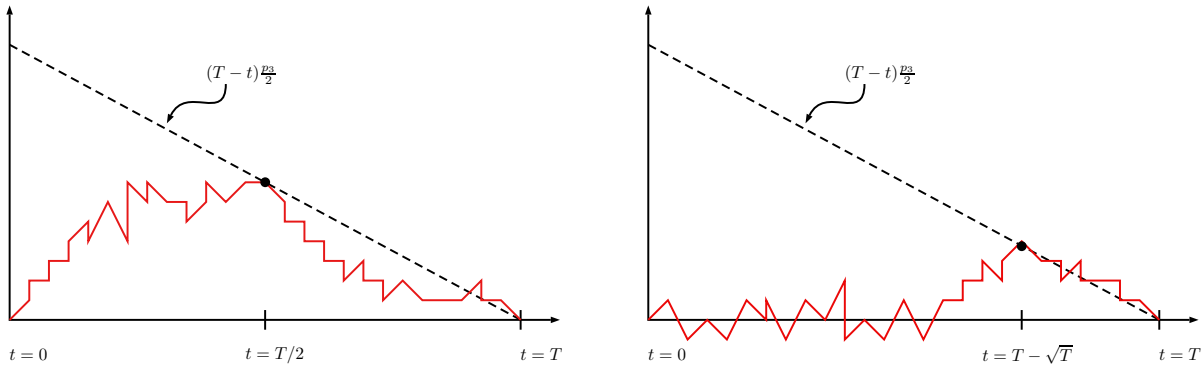


Figure C.2: Paths on $\mathcal{M} = \{R - e_{\mathcal{K}}\bar{Z}^T_{\mathcal{K}} \leq 0\}$. The two paths illustrate unlikely events. On the left: because the random walk has variation $\mathcal{O}(\sqrt{t})$ it cannot hit the target line by a time t such that $T - t = \Omega(n)$. On the right: the path could hit the target by a time of the form $t = T - \mathcal{O}(\sqrt{T})$. In this case, however, it does not have enough time to get back to 0 which it must on the event $\mathcal{M}$

support of θ is all basic variables and we have

$$\mathcal{B}\begin{pmatrix}\theta\\ D-\theta\\ 0\end{pmatrix}_{\mathcal{B}} = \begin{pmatrix}R^\theta\\ D\end{pmatrix},$$

which proves the optimality of θ at R^θ by Lemma 4.4.1 and concludes the first item.

For the second item, let u and s be the unmet and surplus variables for $\mathrm{LP}(R, D)$. Since both bases share the request variables $\kappa := \kappa^+ \cup \kappa^-$, we can write the following

$$\bar{\mathcal{B}}\begin{pmatrix}y_\kappa\\ 0\\ 0\end{pmatrix} = \mathcal{B}\begin{pmatrix}y_\kappa\\ 0\\ 0\end{pmatrix} \quad \text{and} \quad \bar{\mathcal{B}}\begin{pmatrix}y_\kappa\\ 0\\ 0\end{pmatrix} + \bar{\mathcal{B}}\begin{pmatrix}y_{\bar{\mathcal{B}}\setminus\kappa}\\ u_{\bar{\mathcal{B}}}\\ s_{\bar{\mathcal{B}}}\end{pmatrix} = \begin{pmatrix}R\\ D\end{pmatrix} \Longrightarrow \mathcal{B}^{-1}\begin{pmatrix}R\\ D\end{pmatrix} = \begin{pmatrix}y_\kappa\\ 0\\ 0\end{pmatrix} + \mathcal{B}^{-1}\bar{\mathcal{B}}\begin{pmatrix}y_{\bar{\mathcal{B}}\setminus\kappa}\\ u_{\bar{\mathcal{B}}}\\ s_{\bar{\mathcal{B}}}\end{pmatrix},$$

where the second equation is from the optimality of $\bar{\mathcal{B}}$. Finally, we claim that $\mathcal{B}^{-1}\bar{\mathcal{B}}$ has an identity in the columns corresponding to κ, which proves the result. To see this, note that by assumption $\mathcal{B}_\kappa = \bar{\mathcal{B}}_\kappa$ and we can separate by columns $\bar{\mathcal{B}} = [\mathcal{B}_\kappa|0_{\kappa^c}] + [0_\kappa|\bar{\mathcal{B}}_{\kappa^c}] = \mathcal{B} + [0_\kappa|\bar{\mathcal{B}}_{\kappa^c} - \mathcal{B}_{\kappa^c}]$. □

Lemma C.2.1. *Consider m random walks $S^1, \ldots, S^m$ and $S^i_t = S^i_0 + \sum_{t=1}^n X^i_t$ where, for each i, the increments $X^i_t, t \in [T]$ are zero-mean i.i.d and bounded ($\mathbb{E}[X^i_t] = 0$ and $\mathbb{P}\{|X^i_t| \leq b\} = 1$*

for some $b > 0$) and independent of S_0^i which satisfies $|S_0^i| \le \epsilon/2$. Let

$$\tau = \inf\{t \ge 0 : (T-t)\epsilon \le \max_i S_t^i\} \wedge T.$$

Then, for all $t \in [T]$

$$\mathbb{P}\{T - \tau > t, \max_i S_T^i \le 0\} \le \theta_1 e^{-\theta_2 t},$$

for constants $\theta_1, \theta_2 > 0$ that may depend on b, ϵ

PROOF. Fix M and notice that for $t \le n - M$

$$\begin{aligned}\mathbb{P}\{\tau \le t, S_n \le 0\} &\le \mathbb{P}\{\inf_{s \le t} S_n - S_s + \delta(T-s) \le 0\} \\ &= \mathbb{P}\{\inf_{u \ge n-t} S_u + \delta u \le 0\} \le Ce^{-(T-t)},\end{aligned}$$

where the last inequality follows from standard results; see e.g. [27]. We conclude that

$$\mathbb{E}[\tau \mathbb{1}\{S_n \le 0\}] = \sum_{t=0}^{n-1} \mathbb{P}\{\tau > t, S_n \le 0\} \ge n - M$$

for some constant M. □

Figure C.2 is a graphic illustration of the event whose probability is bounded in Lemma C.2.1, for the case of $m = 1$.

APPENDIX D

APPENDIX FOR CHAPTER 5

D.1 A Sufficient Condition for Bellman Inequalities

In this section, we construct φ based on a general optimization program and provide a sufficient condition to guarantee monotonicity. This serves to underscore some of the key elements in a problem's structure that allows one to construct low regret online policies. This guideline does not apply to all the examples we study here: in particular, it applies to the baseline and learning variants, but not to probing or pricing.

We study a particular case of canonical filtrations (see Definition 5.3.1), where the random variables G_θ that we reveal are the inputs ξ^θ for some fixed times Θ (see Fig. 5.2 for an illustration).

Recall that we reveal some inputs to OFFLINE, but not necessarily all of them; we call *concealed inputs* those not revealed to OFFLINE. Informally speaking, we will show that φ satisfies the Bellman Inequalities if (i) OFFLINE's relaxed value φ can be computed with a linear program and (ii) the concealed inputs are in the objective function only (not in the constraints). Requirements (i) and (ii) are appealing because they are verifiable directly from the problem structure without any computation.

Recall that, with t periods to go, OFFLINE knows the randomness $\{\xi^T, \ldots, \xi^t, \xi_\Theta\}$, where we denote $\xi_\Theta = (\xi^\theta : \theta \in \Theta)$. In other words, we reveal $\{\xi^T, \ldots, \xi^t, \xi_\Theta\}$, while the inputs $\{\xi^l : l < t, l \notin \Theta\}$ are concealed.

Suppose the relaxation is an LP with decision variables $\mathbf{x}$ (see Eq. (5.7)):

$$\varphi(t, s|\mathcal{G}_t) = \max_{\mathbf{x} \in \mathbb{R}^{\Xi \times [T] \times \mathcal{U}}} \{\mathbb{E}[h(\mathbf{x}; \xi^1, \ldots, \xi^T)|\mathcal{G}_t] : g(\mathbf{x}; s, \xi^T, \ldots, \xi^t, \xi_\Theta) \geq \mathbf{0}\}, \tag{D.1}$$

where $\mathcal{U}, \Xi$ are the control and input spaces. For input ξ, control u, and time t, we interpret $x_{\xi,t,u}$ as a variable indicating if OFFLINE uses u at time t when presented input ξ.

Proposition D.1.1. *Let h, g be linear functions and let φ be given by (D.1). Assume further that the following holds for all s, t, u*

(i) h captures rewards: $\mathbb{E}[h(\mathbf{e}_{\xi^t,t,u}; \xi^1, \ldots, \xi^T)|\mathcal{G}_t] \leq \mathcal{R}(s, \xi^t, u)$ for actions u that are feasible in state s.

(ii) g captures transitions: $g(\mathbf{e}_{\xi^t,t,u}; s, \xi^T, \ldots, \xi^t, \xi_\Theta) \leq g(\mathbf{0}; \mathcal{T}(s, \xi^t, u), \xi^T, \ldots, \xi^{t-1}, \xi_\Theta)$.

Then, φ satisfies monotonicity with exclusion sets

$$\mathcal{B}(t, s) = \{\omega \in \Omega : \nexists X[\omega] \text{ solving } \varphi(t, s|\mathcal{G}_t) \text{ s.t. } X_{\xi^t,t,u} \geq 1 \text{ for some } u \in \mathcal{U}\}.$$

It is natural to say that h captures the reward if the incremental effect of taking the action u given input ξ^t is equal to the immediate reward $\mathbb{E}[h(\mathbf{e}_{\xi^t,t,u}; \xi^1, \ldots, \xi^T)|\mathcal{G}_t] = \mathcal{R}(s, \xi^t, u)$. It is similarly natural to say that g captures transitions if it is *stable under the one-step transition*, namely, that $g(\mathbf{e}_{\xi^t,t,u}; s, \xi^T, \ldots, \xi^t, \xi_\Theta) = g(\mathbf{0}; \mathcal{T}(s, \xi^t, u), \xi^T, \ldots, \xi^{t-1}, \xi_\Theta)$; in other words, this means that taking the action u at time t, has the same effect as taking no action at the state $\mathcal{T}(s, \xi^t, u)$. This should hold in any reasonable resource consumption problem, e.g., consuming 1 with B units of budget remaining is the same as not consuming anything with $B-1$ units. In the result below we make the weaker assumption that these relationships hold as inequalities.

The baseline and learning variants are useful illustrations of Proposition D.1.1.

Example D.1.2 (Baseline). Let $\mathcal{G}$ be the full information filtration ($\Theta = [T]$). In Section 5.2.2 we introduced a linear relaxation for OFFLINE. We start by writing a relaxation in the form of Proposition D.1.1 and show how it subsequently simplifies to the final form in Section 5.2.2.

Recall that $\mathtt{a}, \mathtt{r}$ denote the actions accept and reject. A natural "expanded" linear program is

$$\max\left\{\sum_j \sum_{l=1}^{t} x_{j,l,\mathtt{a}} r_j : \sum_{j,l} w_j x_{j,l,\mathtt{a}} \le s, 0 \le x_{j,l,\mathtt{a}} \le \mathbb{1}_{\{\xi^l = j\}}\right\}.$$

Defining the auxiliary variables $x_j \coloneqq \sum_l x_{j,l,\mathtt{a}}$, this is equivalent to $\varphi(t, s|\mathcal{G}) = \max\{\mathbf{r}'\mathbf{x} : \mathbf{w}'\mathbf{x} \le s, 0 \le \mathbf{x} \le Z(t)\}$, where, recall $Z_j(t) = \sum_{l=1}^{t} \mathbb{1}_{\{\xi^l = j\}}$ counts the number of type-j arrivals in the last t periods.

This φ also has the form of Proposition D.1.1, with the functions h and g given by (note that the action $\mathtt{r}$ has zero objective coefficient)

$$h(\mathbf{x}; \xi^1, \dots, \xi^t) \coloneqq \sum_j x_{j\mathtt{a}} r_j \quad \text{and} \quad g(\mathbf{x}; s, \xi^T, \dots, \xi^1) \coloneqq \begin{pmatrix} s - \sum_j x_{j\mathtt{a}} \\ Z(t) - \mathbf{x} \end{pmatrix}.$$

Conditions (i) and (ii) can be easily verified now. The objective h is a linear function of the decision vector $\mathbf{x}$ and the constraint function g aggregates ξ into the sums $Z(t)$.

In the learning setting, Offline is presented with a public type j and must decide whether to accept or reject before seeing the private type, which is a reward R_j drawn from an unknown distribution.

Example D.1.3 (Learning). Let us model the problem with $2T$ time periods, where at even times the public type is revealed and at odd times the private (reward). In this model, the input ξ^t is an index $j \in [n]$ at even times and it is a reward $R \in \mathbb{R}$ at odd times. Also let us model the random rewards by drawing i.i.d. copies $\{R_{jt}\}_t$ of R_j.

Let us endow Offline with the information of all even times, i.e., Offline knows all the future arriving public types. Specifically, we set $\Theta = \{t \in [T] : t \text{ is even}\}$ (see Fig. 5.2 for a representation of $\mathcal{G}$). The realizations $\{R_{jt}\}_{j,t}$, drawn at times $t \notin \Theta$, are concealed. The expanded linear program is

$$\max\left\{\sum_j \sum_{l=1}^{t} x_{j,l,\mathtt{a}} \mathbb{E}[R_j] : \sum_{j,l} w_j x_{j,l,\mathtt{a}} \le s, 0 \le x_{j,l,\mathtt{a}} \le \mathbb{1}_{\{\xi^l = j\}}\right\}.$$

As before, we can simplify this LP by aggregating variables, see Section 5.6 for the details. Here we prefer to study the expanded LP because it exemplifies the conditions in Proposition D.1.1.

The objective function is $h(\mathbf{x}; \xi^1, \dots, \xi^t) = \sum_{j,l} x_{j,l,\mathtt{a}} R_{j,l}$, When we take expectations $\mathbb{E}[\cdot|\mathcal{G}_t]$ we arrive at the expression $\sum_{j,l} x_{j,l,\mathtt{a}} \mathbb{E}[R_j]$. The constraint function g is given by the feasibility region of the LP. Conditions (i) and (ii) of Proposition D.1.1 hold with equality.

Proof of Proposition D.1.1. Let $u \in \mathcal{U}$ be such that $X_{\xi^t,t,u} \geq 1$. Denote $\theta_t \coloneqq \{l \in [T] : l \geq t\} \cup \Theta$, so all the inputs $(\xi^l : l \in \Theta_t)$ are revealed at time t (the rest are concealed). By Lemma 5.3.5,

$$\varphi(t, s|\mathcal{G}_t) = \mathbb{E}[h(\mathbf{e}_{\xi^t,t,u}; \xi^1, \dots, \xi^T)|\mathcal{G}_t] + \max_{\mathbf{x}} \{\mathbb{E}[h(\mathbf{x}; \xi^1, \dots, \xi^T)|\mathcal{G}_t] : g(\mathbf{x} + \mathbf{e}_{\xi^t,t,u}; s, (\xi^l : l \in \Theta_t)) \geq \mathbf{0}\}.$$

Using (i) and (ii) yields

$$\varphi(t, s|\mathcal{G}_t) \leq \mathcal{R}(s, \xi^t, u) + \max_{\mathbf{x}} \{\mathbb{E}[h(\mathbf{x}; \xi^1, \dots, \xi^T)|\mathcal{G}_t] : g(\mathbf{x}; \mathcal{T}(s, \xi^t, u), (\xi^l : l \in \Theta_{t-1})) \geq \mathbf{0}\}. \tag{D.2}$$

Since $\mathcal{G}_t$ is coarser than $\mathcal{G}_{t-1}$, we know that $\mathbb{E}[\mathbb{E}[\cdot|\mathcal{G}_{t-1}]|\mathcal{G}_t] = \mathbb{E}[\cdot|\mathcal{G}_t]$. Using Eq. (D.2) and applying Jensen's Inequality (recall that the maximum of linear functions is a convex function) we obtain

$$\varphi(t, s|\mathcal{G}_t) \leq \mathcal{R}(s, \xi^t, u) + \mathbb{E}\Big[\max_{\mathbf{x}} \{\mathbb{E}[h(\mathbf{x}; \xi^1, \dots, \xi^T)|\mathcal{G}_{t-1}] : g(\mathbf{x}; \mathcal{T}(s, \xi^t, u), (\xi^l : l \in \Theta_{t-1})) \geq \mathbf{0}\}\Big|\mathcal{G}_t\Big].$$

This corresponds to the required inequality in Definition 5.3.3. □

The sufficient conditions in Proposition D.1.1 are not necessary; they are not satisfied in the probing setting (Section 5.4) or in the pricing setting (Section 5.5). Nevertheless, we are still able to show monotonicity and draw the desired regret bounds.

D.2 Proofs from Section 5.5

D.2.1 Proof of Proposition 5.5.1

Throughout this subsection, we fix some indexes j', l'. To complete the proof of the proposition, it remains to establish that, whenever $X_{j'l'} \geq 1$, then $\mathbb{E}[L_B(t+1, \mathbf{b}, j', l')|\mathcal{G}_t] \leq 0$, where

$$L_B(t+1, \mathbf{b}, j', l') = P[\mathbf{b} - Q_{j'l'}(t+1), Q(t+1), Z(t)] - \mathbb{E}_{t+1}[P[\mathbf{b} - A_{j'}Y_{j'l'}, Q(t), Z(t)]].$$

The Correction LP. Let us fix $(t, \mathbf{b}, \mathbf{q}, \mathbf{z})$ and denote $\bar{\mathbf{x}}$ the solution of $P[\mathbf{b} - A_{j'}q_{j'l'}, \mathbf{q}, \mathbf{z}]$. To bound the loss, we must bound the right-hand side of (5.16), which captures the perturbation of budgets from $\mathbf{b} - A_{j'}q_{j'l'}$ to $\mathbf{b} - A_{j'}Y_{j'l'}$ and the perturbation of fractions from $\mathbf{q}$ to $\mathbf{q} + \Delta$, where Δ is a zero-mean random vector.

Let us re-formulate $P[\mathbf{b} - A_{j'}Y_{j'l'}, \mathbf{q} + \Delta, \mathbf{z}]$ based on how much we need to correct $\bar{\mathbf{x}}$:

$$\begin{array}{lll} (P[\mathbf{b} - A_{j'}Y_{j'l'}, \mathbf{q} + \Delta, \mathbf{z}]) \quad \max_{\mathbf{y}} & \sum_{j,l} f_{jl}(q_{jl} + \Delta_{jl})(\bar{x}_{jl} - y_{jl}) & \\ \text{s.t.} & \sum_{j,l} a_{ij}(q_{jl} + \Delta_{jl})(\bar{x}_{jl} - y_{jl}) \leq b_i - a_{ij'}Y_{j'l'} & \forall i \\ & \sum_{l}(\bar{x}_{jl} - y_{jl}) \leq z_j & \forall j \\ & \bar{\mathbf{x}} - \mathbf{y} \geq 0. & \end{array}$$

The new formulation uses decision variables $\mathbf{y}$, which may be negative, and correspond to how much we movement there is from the initial solution $\bar{\mathbf{x}}$ to the new one.

Let us denote the resource-slack variables of $P[\mathbf{b} - A_{j'}q_{j'l'}, \mathbf{q}, \mathbf{z}]$ by $(s_i \geq 0 : i \in [d])$, i.e., $\sum_{j,l} a_{ij}q_{jl}\bar{x}_{jl} + s_i = b_i - a_{ij'}q_{j'l'}$. Similarly, let us denote the demand-slack variables by

$(u_j \geq 0 : j \in [n])$, i.e., $\sum_l \bar{x}_{jl} + u_j = z_j$. Using the slack variables, the problem simplifies to

$$\begin{aligned} P[\mathbf{b} - A_{j'}Y_{j'l'}, \mathbf{q} + \Delta, \mathbf{z}] = \textstyle\sum_{j,l} f_{jl}(q_{jl} + \Delta_{jl})\bar{x}_{jl} - \min_{\mathbf{y}} \quad & \textstyle\sum_{j,l} f_{jl}(q_{jl} + \Delta_{jl})y_{jl} & & \\ \text{s.t.} \quad & \textstyle\sum_{j,l} a_{ij}(q_{jl} + \Delta_{jl})y_{jl} & \geq \beta_i & \quad \forall i \\ & \textstyle\sum_l y_{jl} & \geq -u_j & \quad \forall j \\ & y & \leq \bar{\mathbf{x}}, & \end{aligned} \tag{D.3}$$

where we defined $\beta_i \coloneqq a_{ij'}(Y_{j'l'} - q_{j'l'}) - s_i + \sum_{j,l} a_{ij}\Delta_{jl}\bar{x}_{jl}$.

Observe that, since $\mathbb{E}[\Delta] = 0$, the first term outside the minimization, namely $\sum_{j,l} f_{jl}(q_{jl} + \Delta_{jl})\bar{x}_{jl}$, equals $\sum_{j,l} f_{jl}q_{jl}\bar{x}_{jl} = P[\mathbf{b} - A_{j'}q_{j'l'}, \mathbf{q}, \mathbf{z}]$ in expectation. The following result readily proves Proposition 5.5.1.

Lemma D.2.1 (Correction LP). *If we denote $\mathbf{q} = Q(t+1)$, then the Bellman Loss is bounded by $\mathbb{E}[L_B(t+1, \mathbf{b}, j', l')] \leq \mathbb{E}[P_C[Y_{j'l'}, \mathbf{q}, \Delta]]$, where $(P_C[Y_{j'l'}, \mathbf{q}, \Delta])$ is the minimization problem in Eq. (D.3). Furthermore, $\mathbb{E}[L_B(t+1, \mathbf{b}, j', l')] \leq 0$.*

Proof. Recall that $\beta_i = a_{ij'}(Y_{j'l'} - q_{j'l'}) - s_i + \sum_{j,l} a_{ij}\Delta_{jl}\bar{x}_{jl}$ and observe that $\mathbb{E}[\beta_i] \leq 0$ for all i. We will find some deterministic values c_i such that the objective value of $P_C[Y_{j'l'}, \mathbf{q}, \Delta]$ is upper bounded by $\sum_i c_i\beta_i$, which proves the result.

We argue the upper bound on $(P_C[Y_{j'l'}, \mathbf{q}, \Delta])$ by bounding the optimal dual solution. The dual of $P_C[Y_{j'l'}, \mathbf{q}, \Delta]$ is

$$\max_{\mu,\lambda,\theta \geq 0} \left\{ \beta'\mu - u'\lambda - \sum_{j,l} \bar{x}_{jl}\theta_{jl} : (q_{jl} + \Delta_{jl})A_j'\mu + \lambda_j - \theta_{jl} \leq f_{jl}(q_{jl} + \Delta_{jl}) \quad \forall j, l \right\}$$

This problem is the dual of a feasible and finite problem (see Eq. (D.3)), hence it has an optimal finite solution and we can bound $\mu_i \leq c_i$ for some deterministic values c_i. The objective value of this maximization problem is upper bounded by $\beta'c$, which proves the result. □

D.2.2 Proof of Lemma 5.5.7

Recall that we wish to establish the following: if $\hat{\varphi}$ (used by ONLINE) has a solution with $x_{jl'} = \max_l x_{jl} >> 1$, then posting price f_{jl} is a satisfying action. To establish this, it remains to bound the difference between the LP $P_S[v_t, \mathbf{q}, \mathbf{z}]$ and its "perturbed" version $P_S[V_t, \mathbf{q} + \Delta\mathbf{q}, \mathbf{z} + \Delta\mathbf{z}]$. To that end, we first establish a bound on $v_t - V_t$; see item (i) in the discussion following Lemma 5.5.6.

Lemma D.2.2. *For fixed* $\mathbf{b}$*, denote* $V_t = P[\mathbf{b}, Q(t), Z(t)]$ *and* $v_t = P[\mathbf{b}, \mathbb{E}[Q(t)], \mathbb{E}[Z(t)]]$*. If* $t \geq c$*, then, with probability at least* $1 - c/t^2$*, we have* $v_t - V_t \geq -c\sqrt{t \log(t)}$*. The constant* c *is independent of* $\mathbf{b}$ *and depends on* $(\mathbf{f}, F_1, \ldots, F_n)$ *only.*

PROOF. Set $\mathbf{q} = Q(t)$, $\mathbf{z} = Z(t)$, $\Delta\mathbf{q} = \mathbb{E}[Q(t)] - Q(t)$, and $\Delta\mathbf{z} = \mathbb{E}[Z(t)] - Z(t)$. Take $\bar{\mathbf{x}}$ to be a solution of V_t and use a correction program analogous to Eq. (D.3) to conclude

$$\begin{aligned} v_t = V_t + \textstyle\sum_{j,l} f_{jl} \Delta q_{jl} \bar{x}_{jl} - \min_{\mathbf{y}} \quad & \textstyle\sum_{j,l} f_{jl}(q_{jl} + \Delta q_{jl}) y_{jl} & & \\ \text{s.t.} \quad & \textstyle\sum_{j,l} a_{ij}(q_{jl} + \Delta q_{jl}) y_{jl} \geq \beta_i & & \forall i \\ & \textstyle\sum_l y_{jl} + t p_j \geq \sum_l \bar{x}_{jl} & & \forall j \\ & \mathbf{y} \leq \bar{\mathbf{x}}, & & \end{aligned} \tag{D.4}$$

where $\beta_i = -s_i + \sum_{j,l} a_{ij} \Delta q_{jl} \bar{x}_{jl}$. We will argue an upper bound on the minimisation problem by exhibiting a feasible solution.

Set $g(t) := \sqrt{log(t)/t}$ and consider the solution $y_{jl} = \bar{x}_{jl} \frac{g(t)}{q_{jl} + \Delta q_{jl}}$. First recall that, by Lemma 5.5.5, $|\Delta q_{jl}| \leq g(t)$ with high probability. The objective value of this solution is $\sum_{j,l} f_{jl} \bar{x}_{jl} g(t)$, hence from Eq. (D.4) we get $v_t \geq V_t - 2 \sum_{j,l} f_{jl} g(t) \bar{x}_{jl}$. From here, using the fact that $\bar{\mathbf{x}}$ solves an LP with the constraint $\sum_l x_{jl} \leq Z_j(t)$ forr all j and that $Z_j(t) \leq t$ a.s., we conclude the result by using that $\bar{x}_{jl} \leq t$.

We are left to check that our solution $\mathbf{y}$ is feasible for the LP in Eq. (D.4). The first set

of constraints is satisfied because $g(t) \geq \Delta q_{jl}$. The second set of constraints is satisfied since $\sum_l \bar{x}_{jl} \leq Z_j(t)$ and $Z_j(t) \leq tp_j + \sqrt{t \log(t)}$ w.h.p. Finally, the constraints $\mathbf{y} \leq \bar{\mathbf{x}}$ are satisfied since $g(t) \leq q_{jl} + \Delta q_{jl}$ for all t large enough. □

Proof of Lemma 5.5.7. Let us denote $\theta = (v, \mathbf{q}, \mathbf{z})$ and $\theta + \Delta\theta = (v + \Delta v, \mathbf{q} + \Delta\mathbf{q}, \mathbf{z} + \Delta\mathbf{z})$. Recall that $\mathbf{b}$ and t are fixed throughout. The selection program for a fixed component (j', l') is given by

$$\begin{aligned} (P_S[\theta]) \qquad \max \quad & x_{j'l'} & & \\ \text{s.t.} \quad & \textstyle\sum_{j,l} f_{jl} q_{jl} x_{jl} & \geq v & \\ & \textstyle\sum_{j,l} a_{ij} q_{jl} x_{jl} & \leq b_i & \qquad \forall i \\ & \textstyle\sum_l x_{jl} & \leq z_j & \qquad \forall j \\ & \mathbf{x} & \geq 0. & \end{aligned}$$

If $\bar{X}$ is the solution to $P[\mathbf{b}, Q(t), Z(t)]$ (used by Offline) and $\bar{\mathbf{x}}$ is the solution to $P[\mathbf{b}, \mathbb{E}[Q(t)], \mathbb{E}[Z(t)]]$ (used by Online), we want to prove $\bar{X}_{j'l'} \geq \bar{x}_{j'l'} - c\sqrt{t \log(t)}$. Equivalently, our aim is to prove the following:

$$P_S[\theta + \Delta\theta] \geq P_S[\theta] - c\sqrt{t \log(t)} \qquad \text{w.p. } 1 - c/t^2.$$

We argue via Lagrangian relaxation. The Lagrangian of the selection problem with parameters $\theta + \Delta$ is given by

$$\begin{aligned} L(\mathbf{x}, \lambda; \theta + \Delta\theta) &= x_{j'l'} + \lambda_0 \left(\sum_{j,l} f_{jl}(q_{jl} + \Delta q_{jl}) x_{jl} - v - \Delta v \right) \\ &\quad + \sum_i \lambda_i \left(b_i - \sum_{j,l} a_{ij}(q_{jl} + \Delta q_{jl}) x_{jl} \right) + \sum_j \lambda_j \left(z_j + \Delta z_j - \sum_l x_{jl} \right) \\ &= L(\mathbf{x}, \lambda; \theta) + \lambda_0 \left(\sum_{j,l} f_{jl} \Delta q_{jl} x_{jl} - \Delta v \right) - \sum_i \lambda_i \sum_{j,l} a_{ij} \Delta q_{jl} x_{jl} + \sum_j \lambda_j \Delta z_j \end{aligned}$$

Define $D := \{\mathbf{x} : \mathbf{x} \geq 0, ||\mathbf{x}||_\infty \leq t\}$. Observe that both $P_S[\theta]$ and $P_S[\theta + \Delta\theta]$ have solutions $\mathbf{x} \in D$. From Lemma 5.5.5 and Lemma D.2.2 we have the following with probability

$1 - c/t^2$:

$$|\Delta q_{jl} x_{jl}| \leq \sqrt{t \log(t)} \quad \forall \mathbf{x} \in D, \quad \Delta v \leq \sqrt{t \log(t)}, \quad \Delta z_j \geq \sqrt{t \log(t)}.$$

Let $\lambda^\star$ be the optimal dual variables of $P_S[\theta + \Delta\theta]$. We claim that there is a constant c such that $||\lambda^\star||_\infty \leq c$. Assuming this claim, from the previous equation we get

$$L(\mathbf{x}, \lambda; \theta + \Delta\theta) \geq L(\mathbf{x}, \lambda; \theta) - c\sqrt{t \log(t)} \quad \forall \mathbf{x} \in D.$$

Using Strong Duality for the problem $P_S[\theta + \Delta\theta]$ we have

$$\begin{aligned} P_S[\theta + \Delta\theta] &= \max_{\mathbf{x} \geq 0} L(\mathbf{x}, \lambda^\star; \theta + \Delta\theta) \\ &= \max_{\mathbf{x} \in D} L(\mathbf{x}, \lambda^\star; \theta + \Delta\theta) \\ &\geq \max_{\mathbf{x} \in D} L(\mathbf{x}, \lambda^\star; \theta) - c\sqrt{t \log(t)} \\ &\geq P_S[\theta] - c\sqrt{t \log(t)}. \end{aligned}$$

In the last step we used weak duality. Finally, to bound $||\lambda^\star||_\infty \leq c$ we observe that the dual feasible region is defined by $\lambda \geq 0$ and the following set of inequalities, where δ is the Kronecker delta:

$$-f_{jl} q_{jl} \lambda_0 + q_{jl} \sum_i a_{ij} \lambda_i + \lambda_j \geq \delta_{j'l'} \quad \forall j, l.$$

These inequalities are independent of $(t, \mathbf{b})$, hence we can bound uniformly the extreme points. □

D.3 Connections to Information Relaxations

Our work is related to the information-relaxation framework developed in [29, 18]. The information-relaxation framework is a fairly general way to endow OFFLINE with additional

information, but at the same time forcing him to pay a penalty for using this information. The dualized problem (with the penalties) is an upper bound on the performance of the best online policy.

The main distinctions with our approach are:

1. Information Relaxation requires to identify OFFLINE's filtration and penalties to build a proxy for OFFLINE's value function. This proxy can then be used to assess the performance of specific online policies.

   The proxy that is developed—as the true OFFLINE value in our framework—may be difficult to compute. To overcome this difficulty, [18] proposes an approximation through which penalties can be computed and hence an upper bound can be obtained.

2. Our framework requires, as well, identifying a suitable information structure (a filtration) and a relaxation φ. Because we allow for a Bellman Loss, we can develop φ $\hat{\varphi}$ that are computationally tractable. In most cases, a linear program. The framework explicitly then provides a mechanism, the RABBI algorithm, to derive a good online policy.

There is also an explicit mathematical connection. To state it, we first present a weaker version of our Bellman Inequalities, called thus because it is easier to find an object φ under this definition. Recall that, for a given non-anticipatory policy π, we denote v_π^{on} the expected value. Observe that the distinction with Definition 5.3.3 is in the initial ordering condition; we now require ϕ to upper bound the online value instead of the best offline.

Definition D.3.1 (Weak Bellman Inequalities). The sequence of r.v. $\{\varphi(t,s)\}_{t\in T, s\in\mathcal{S}}$ satisfies the Weak Bellman Inequalities w.r.t. filtration $\mathcal{G}$ and events $\mathcal{B}(t,s)\subseteq\Omega$ if $\varphi(t,s)$ is $\mathcal{G}_t$-measurable for all t,s and the following holds:

1. Initial ordering: $\max_\pi v_\pi^{\text{on}} \leq \mathbb{E}[\varphi(T, S^T|\mathcal{G}_T)]$, where S^T is the initial state.

2. Monotonicity: $\forall s \in \mathcal{S}, t \in [T], \omega \notin \mathcal{B}(t,s)$,

$$\varphi(t,s|\mathcal{G}_t) \leq \max_{u\in\mathcal{U}}\{\mathcal{R}(s,\xi^t,u) + \mathbb{E}[\varphi(t-1,\mathcal{T}(s,\xi^t,u)|\mathcal{G}_{t-1})|\mathcal{G}_t]\}. \tag{D.5}$$

In Proposition 2.1 in [18] it is shown that if φ is some function that satisfies the Bellman equation for OFFLINE with the penalized immediate rewards function, then, in particular, it satisfies the initial ordering above. Since such φ satisfies, by construction, the Bellman inequality the following is an immediate corollary.

Proposition D.3.2 (Proposition 2.1 in [18]). *Given feasible penalties z_t, the penalized value function satisfies Definition 5.3.3 with exclusion sets $\mathcal{B}(t,s) = \varnothing$.*

Our framework is a structured approach for building a computationally tractable φ, and deriving an online policy is bounded regret, without pre-computing penalties.

D.4 Extension to Random Rewards and Random Transitions

We assume for simplicity that, given a random input, rewards and transitions are deterministic. This assumption is w.l.o.g. since we can always add additional periods where extra inputs are revealed. We give a formal reduction in Section 5.4.2.

On the other hand, we can define Bellman Inequalities for the case where transitions and rewards are random. All of our results continue to hold, the only modification is that we have to take expectations everywhere. To be more precise, the loss L_B can be both random and dependent on the action u. Additionally, since now rewards and transitions are unknown at t given ξ^t, the monotonicity requirement (5.5) takes the following form:

$$\varphi(t,s|\mathcal{G}_t) \leq \max_{u\in\mathcal{U}}\{\mathbb{E}[\mathcal{R}(s,\xi^t,u) + \varphi(t-1,\mathcal{T}(s,\xi^t,u)|\mathcal{G}_{t-1})|\mathcal{G}_t] + \mathbb{E}[L_B(t,s,u)|\mathcal{G}_t]\}.$$

BIBLIOGRAPHY

[1] Ittai Abraham, Daniel Delling, Amos Fiat, Andrew V Goldberg, and Renato F Werneck. Vc-dimension and shortest path algorithms. In *International Colloquium on Automata, Languages, and Programming*, pages 690–699. Springer, 2011.

[2] Ittai Abraham, Daniel Delling, Amos Fiat, Andrew V Goldberg, and Renato F Werneck. Highway dimension and provably efficient shortest path algorithms. *Journal of the ACM (JACM)*, 63(5):41, 2016.

[3] Ittai Abraham, Daniel Delling, Andrew V Goldberg, and Renato F Werneck. A hub-based labeling algorithm for shortest paths in road networks. In *International Symposium on Experimental Algorithms*, 2011.

[4] Ittai Abraham, Amos Fiat, Andrew V Goldberg, and Renato F Werneck. Highway dimension, shortest paths, and provably efficient algorithms. In *Proceedings of the twenty-first annual ACM-SIAM symposium on Discrete Algorithms*, 2010.

[5] Shipra Agrawal and Nikhil Devanur. Linear contextual bandits with knapsacks. In *Advances in Neural Information Processing Systems*, pages 3450–3458, 2016.

[6] Shipra Agrawal and Nikhil R Devanur. Bandits with concave rewards and convex knapsacks. In *Proceedings of the fifteenth ACM conference on Economics and computation*, pages 989–1006. ACM, 2014.

[7] Shipra Agrawal and Nikhil R Devanur. Fast algorithms for online stochastic convex programming. In *Proceedings of the twenty-sixth annual ACM-SIAM symposium on Discrete algorithms*, pages 1405–1424. SIAM, 2014.

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

[9] Saeed Alaei. Bayesian combinatorial auctions: Expanding single buyer mechanisms to many buyers. *SIAM Journal on Computing*, 43(2):930–972, 2014.

[10] Alessandro Arlotto and Itai Gurvich. Uniformly bounded regret in the multi-secretary problem. *Stochastic Systems*, 2018. Forthcoming.

[11] John Augustine, Sandy Irani, and Chaitanya Swamy. Optimal power-down strategies. In *45th Annual IEEE Symposium on Foundations of Computer Science*, pages 530–539. IEEE, 2004.

[12] Pablo D Azar, Robert Kleinberg, and S Matthew Weinberg. Prophet inequalities with limited information. In *Proceedings of the twenty-fifth annual ACM-SIAM symposium on Discrete algorithms*, pages 1358–1377. SIAM, 2014.

[13] Moshe Babaioff, Shaddin Dughmi, Robert Kleinberg, and Aleksandrs Slivkins. Dynamic pricing with limited supply. *ACM Transactions on Economics and Computation (TEAC)*, 3(1):4, 2015.

[14] Maxim Babenko, Andrew V Goldberg, Haim Kaplan, Ruslan Savchenko, and Mathias Weller. On the complexity of hub labeling. In *International Symposium on Mathematical Foundations of Computer Science*, 2015.

[15] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. Bandits with knapsacks. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 207–216. IEEE, 2013.

[16] Ashwinkumar Badanidiyuru, Robert Kleinberg, and Aleksandrs Slivkins. Bandits with knapsacks. *Journal of the ACM (JACM)*, 65(3):13, 2018.

[17] Ashwinkumar Badanidiyuru, John Langford, and Aleksandrs Slivkins. Resourceful contextual bandits. In *Conference on Learning Theory*, pages 1109–1134, 2014.

[18] Santiago Balseiro and David Brown. Approximations to stochastic dynamic programs via information relaxation duality. *Operations Research*, page to appear, 2016.

[19] Chris Barrett, Keith Bisset, Martin Holzer, Goran Konjevod, Madhav Marathe, and Dorothea Wagner. Engineering label-constrained shortest-path algorithms. In *International Conference on Algorithmic Applications in Management*, 2008.

[20] Hannah Bast, Daniel Delling, Andrew Goldberg, Matthias Müller-Hannemann, Thomas Pajor, Peter Sanders, Dorothea Wagner, and Renato F Werneck. Route planning in transportation networks. In *Algorithm engineering*, pages 19–80. Springer, 2016.

[21] Amariah Becker, Philip N Klein, and David Saulpic. Polynomial-time approximation schemes for k-center, k-median, and capacitated vehicle routing in bounded highway dimension. In *26th Annual European Symposium on Algorithms (ESA 2018)*. Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[22] Alberto Bemporad, Francesco Borrelli, Manfred Morari, et al. Model predictive control based on linear programming˜ the explicit solution. *IEEE transactions on automatic control*, 47(12):1974–1985, 2002.

[23] Dimitri P Bertsekas. *Dynamic programming and optimal control*. Athena scientific Belmont, MA, 1995.

[24] Hans-Georg Beyer and Bernhard Sendhoff. Robust optimization–a comprehensive survey. *Computer methods in applied mechanics and engineering*, 196(33-34):3190–3218, 2007.

[25] Francesco Borrelli. *Constrained optimal control of linear and hybrid systems*, volume 290. Springer, 2003.

[26] Francesco Borrelli, Alberto Bemporad, and Manfred Morari. Geometric algorithm for multiparametric linear programming. *Journal of optimization theory and applications*, 118(3):515–540, 2003.

[27] Stéphane Boucheron, Gábor Lugosi, and Pascal Massart. *Concentration inequalities: A nonasymptotic theory of independence*. Oxford university press, 2013.

[28] E Andrew Boyd and Ioana C Bilegan. Revenue management and e-commerce. *Management science*, 49(10):1363–1386, 2003.

[29] David Brown, James Smith, and Peng Sun. Information Relaxations and Duality in Stochastic Dynamic Programs. *Operations Research*, 58(4):785–801, 2010.

[30] David B Brown and James E Smith. Optimal sequential exploration: Bandits, clairvoyants, and wildcats. *Operations Research*, 61(3):644–665, 2013.

[31] David B Brown and James E Smith. Information relaxations, duality, and convex stochastic dynamic programs. *Operations Research*, 62(6):1394–1415, 2014.

[32] Brian Brubach, Karthik Abinav Sankararaman, Aravind Srinivasan, and Pan Xu. Online stochastic matching: New algorithms and bounds. *arXiv preprint arXiv:1606.06395*, 2016.

[33] Sébastien Bubeck, Nicolo Cesa-Bianchi, et al. Regret analysis of stochastic and nonstochastic multi-armed bandit problems. *Foundations and Trends® in Machine Learning*, 2012.

[34] Sébastien Bubeck, Vianney Perchet, and Philippe Rigollet. Bounded regret in stochastic multi-armed bandits. In *Conference on Learning Theory*, pages 122–134, 2013.

[35] Niv Buchbinder and Joseph Naor. Online primal-dual algorithms for covering and packing. *Mathematics of Operations Research*, 34(2):270–286, 2009.

[36] Niv Buchbinder, Joseph Seffi Naor, et al. The design of competitive online algorithms via a primal–dual approach. *Foundations and Trends® in Theoretical Computer Science*, 3(2–3):93–263, 2009.

[37] Pornpawee Bumpensanti and He Wang. A re-solving heuristic with uniformly bounded loss for network revenue management. *Management Science*, 2019. Forthcoming.

[38] Niangjun Chen, Anish Agarwal, Adam Wierman, Siddharth Barman, and Lachlan LH Andrew. Online convex optimization using predictions. In *ACM SIGMETRICS Performance Evaluation Review*, volume 43, pages 191–204. ACM, 2015.

[39] Niangjun Chen, Joshua Comden, Zhenhua Liu, Anshul Gandhi, and Adam Wierman. Using predictions in online optimization: Looking forward with an eye on the past. *ACM SIGMETRICS Performance Evaluation Review*, 44(1):193–206, 2016.

[40] Qi Chen, Stefanus Jasin, and Izak Duenyas. Nonparametric self-adjusting control for joint learning and optimization of multiproduct pricing with finite resource capacity. *Mathematics of Operations Research*, 2019.

[41] Ben Chugg and Takanori Maehara. Submodular stochastic probing with prices. In *International Conference on Control, Decision and Information Technologies (CoDIT)*, pages 60–66. IEEE, 2019.

[42] Kai-Min Chung, Henry Lam, Zhenming Liu, and Michael Mitzenmacher. Chernoff-hoeffding bounds for markov chains: Generalized and simplified. In *Symposium on Theoretical Aspects of Computer Science*, volume 14, pages 124–135, 2012.

[43] Dragos Florin Ciocan and Vivek Farias. Model predictive control for dynamic resource allocation. *Mathematics of Operations Research*, 37(3):501–525, 2012.

[44] Zachary Clawson, Xuchu Ding, Brendan Englot, Thomas A Frewen, William M Sisson, and Alexander Vladimirsky. A bi-criteria path planning algorithm for robotics applications. *arXiv preprint arXiv:1511.01166*, 2015.

[45] Edith Cohen, Eran Halperin, Haim Kaplan, and Uri Zwick. Reachability and distance queries via 2-hop labels. *SIAM Journal on Computing*, 32(5), 2003.

[46] William Cook, Albertus MH Gerards, Alexander Schrijver, and Éva Tardos. Sensitivity theorems in integer linear programming. *Mathematical Programming*, 34(3):251–264, 1986.

[47] José Correa, Patricio Foncea, Ruben Hoeksma, Tim Oosterwijk, and Tjark Vredeveld. Posted price mechanisms for a random stream of customers. In *Conference on Economics and Computation*, pages 169–186. ACM, 2017.

[48] José Correa, Tobias Harks, Vincent J. C. Kreuzen, and Jannik Matuschke. Fare evasion in transit networks. *Operations Research*, 65(1):165–183, 2017.

[49] Camil Demetrescu, Andrew V Goldberg, and David S Johnson. *The Shortest Path Problem: Ninth DIMACS Implementation Challenge*, volume 74. American Mathematical Soc., 2009.

[50] Vijay V Desai, Vivek F Farias, and Ciamac C Moallemi. Pathwise optimization for optimal stopping problems. *Management Science*, 58(12):2292–2308, 2012.

[51] Nikhil R Devanur, Kamal Jain, Balasubramanian Sivan, and Christopher A Wilkens. Near optimal online algorithms and fast approximation algorithms for resource allocation problems. *Journal of the ACM*, 66(1):7, 2019.

[52] Luc Devroye. The equivalence of weak, strong and complete convergence in l1 for kernel density estimates. *The Annals of Statistics*, 11(3):896–904, 1983.

[53] Yann Disser, Andreas Emil Feldmann, Max Klimm, and Jochen Könemann. Travelling on graphs with small highway dimension. *Graph-Theoretic Concepts in Computer Science LNCS 11789*, page 175, 2019.

[54] Devdatt P Dubhashi and Alessandro Panconesi. *Concentration of measure for the analysis of randomized algorithms*. Cambridge University Press, 2009.

[55] Paul Düetting, Michal Feldman, Thomas Kesselheim, and Brendan Lucier. Prophet inequalities made easy: Stochastic optimization by pricing non-stochastic inputs. In *IEEE FOCS*, 2017.

[56] Atilla Eryilmaz and Rayadurgam Srikant. Asymptotically tight steady-state queue length bounds implied by drift conditions. *Queueing Systems*, 72(3-4):311–359, 2012.

[57] Guy Even, Dror Rawitz, and Shimon Moni Shahar. Hitting sets when the vc-dimension is small. *Information Processing Letters*, 95(2), 2005.

[58] Jianqing Fan, Bai Jiang, and Qiang Sun. Hoeffding's lemma for markov chains and its applications to statistical learning. *arXiv preprint arXiv:1802.00211*, 2018.

[59] YY Fan, RE Kalaba, and JE Moore II. Arriving on time. *Journal of Optimization Theory and Applications*, 127(3), 2005.

[60] Andreas Emil Feldmann. Fixed-parameter approximations for k-center problems in low highway dimension graphs. *Algorithmica*, 81(3):1031–1052, 2019.

[61] Andreas Emil Feldmann, Wai Shing Fung, Jochen Könemann, and Ian Post. A $(1+\varepsilon)$-embedding of low highway dimension graphs into bounded treewidth graphs. *SIAM Journal on Computing*, 47(4):1667–1704, 2018.

[62] Andreas Emil Feldmann and Dániel Marx. The parameterized hardness of the k-center problem in transportation networks. In *16th Scandinavian Symposium and Work-*

shops on Algorithm Theory (SWAT 2018). Schloss Dagstuhl-Leibniz-Zentrum fuer Informatik, 2018.

[63] P Festa. Constrained shortest path problems: state-of-the-art and recent advances. In *Transparent Optical Networks (ICTON), 2015 17th International Conference on*, 2015.

[64] Tomáš Gal. Linear parametric programming—a brief survey. In *Sensitivity, Stability and Parametric Analysis*, pages 43–68. Springer, 1984.

[65] Guillermo Gallego and Garrett Van Ryzin. Optimal dynamic pricing of inventories with stochastic demand over finite horizons. *Management science*, 40(8):999–1020, 1994.

[66] Guillermo Gallego and Garrett Van Ryzin. A multiproduct dynamic pricing problem and its applications to network yield management. *Operations research*, 45(1):24–41, 1997.

[67] Robert Geisberger, Peter Sanders, Dominik Schultes, and Daniel Delling. Contraction hierarchies: Faster and simpler hierarchical routing in road networks. In *International Workshop on Experimental and Efficient Algorithms*, 2008.

[68] Anupam Gupta and Marco Molinaro. How experts can solve lps online. In *European Symposium on Algorithms*, pages 517–529. Springer, 2014.

[69] Anupam Gupta and Viswanath Nagarajan. A stochastic probing problem with applications. In *International Conference on Integer Programming and Combinatorial Optimization*, 2013.

[70] Anupam Gupta, Viswanath Nagarajan, and Sahil Singla. Algorithms and adaptivity gaps for stochastic probing. *SIAM ACM Symposium on Discrete Algorithms*, 2016.

[71] Bernhard Haeupler, Vahab S Mirrokni, and Morteza Zadimoghaddam. Online stochastic weighted matching: Improved approximation algorithms. *Internet and Network Economics*, 2011.

[72] Pascal Van Hentenryck and Russell Bent. *Online stochastic combinatorial optimization*. The MIT Press, 2009.

[73] Theodore P Hill, Robert P Kertz, et al. Comparisons of stop rule and supremum expectations of iid random variables. *The Annals of Probability*, 10(2):336–345, 1982.

[74] Darrell Hoy and Evdokia Nikolova. Approximately optimal risk-averse routing policies via adaptive discretization. In *AAAI*, 2015.

[75] Longbo Huang. Receding learning-aided control in stochastic networks. *Performance Evaluation*, 91:150–169, 2015.

[76] Timothy Hunter, Pieter Abbeel, and Alexandre M Bayen. The path inference filter: model-based low-latency map matching of probe vehicle data. In *Algorithmic Foundations of Robotics X*. 2013.

[77] Stefanus Jasin. Reoptimization and self-adjusting price control for network revenue management. *Operations Research*, 62(5):1168–1178, 2014.

[78] Stefanus Jasin and Sunil Kumar. A re-solving heuristic with bounded revenue loss for network revenue management with customer choice. *Mathematics of Operations Research*, 37(2):313–345, 2012.

[79] Thomas Kesselheim, Klaus Radke, Andreas Tönnis, and Berthold Vöcking. Primal beats dual on online packing lps in the random-order model. *SIAM Journal on Computing*, 47(5):1939–1964, 2018.

[80] Robert Kleinberg and Seth Matthew Weinberg. Matroid prophet inequalities. In *ACM symposium on Theory of computing*, pages 123–136. ACM, 2012.

[81] Adrian Kosowski and Laurent Viennot. Beyond highway dimension: Small distance labels using tree skeletons. In *Proceedings of the Twenty-Eighth Annual ACM-SIAM Symposium on Discrete Algorithms*, 2017.

[82] Retsef Levi and Ana Radovanović. Provably near-optimal lp-based policies for revenue management in systems with reusable resources. *Operations Research*, 58(2):503–507, 2010.

[83] Siva Theja Maguluri and R Srikant. Heavy traffic queue length behavior in a switch under the maxweight algorithm. *Stochastic Systems*, 6(1):211–250, 2016.

[84] Ovil Mangasarian and Tzong Shiau. Lipschitz Continuity of Solutions of Linear Inequalities, Programs and Complementarity Problems. *SIAM Journal on Control and Optimization*, 25(3):583–595, 1987.

[85] Shie Mannor, Duncan Simester, Peng Sun, and John N Tsitsiklis. Bias and variance approximation in value function estimates. *Management Science*, 53(2):308–322, 2007.

[86] Vahideh H Manshadi, Shayan Oveis Gharan, and Amin Saberi. Online stochastic matching: Online actions based on offline statistics. *Mathematics of Operations Research*, 37(4):559–573, 2012.

[87] Pascal Massart. The tight constant in the dvoretzky-kiefer-wolfowitz inequality. *The annals of Probability*, pages 1269–1283, 1990.

[88] Manfred Morari, CE Garcia, JH Lee, and DM Prett. *Model predictive control*. Prentice Hall Englewood Cliffs, NJ, 1993.

[89] Mehrdad Niknami and Samitha Samaranayake. Tractable pathfinding for the stochastic on-time arrival problem. In *International Symposium on Experimental Algorithms*, 2016.

[90] Evdokia Nikolova, Jonathan A Kelner, Matthew Brand, and Michael Mitzenmacher. Stochastic shortest paths via quasi-convex maximization. In *European Symposium on Algorithms*, 2006.

[91] Daniel Paulin. Concentration inequalities for markov chains by marton couplings and spectral methods. *Electronic Journal of Probability*, 20, 2015.

[92] Erica L. Plambeck and Amy R. Ward. Optimal control of a high-volume assemble-to-order system. *Mathematics of Operations Research*, 31(3):453–477, 2006.

[93] Warren B Powell. *Approximate Dynamic Programming: Solving the Curses of Dimensionality*, volume 842. John Wiley & Sons, 2011.

[94] Martin Reiman and Qiong Wang. An Asymptotically Optimal Policy for a Quantity-Based Network Revenue Management Problem. *Mathematics of Operations Research*, 2008.

[95] Michael Rice and Vassilis J Tsotras. Graph indexing of road networks for shortest path queries with label restrictions. *Proceedings of the VLDB Endowment*, 4(2), 2010.

[96] Aviad Rubinstein, Jack Z Wang, and S Matthew Weinberg. Optimal single-choice prophet inequalities from samples. In *11th Innovations in Theoretical Computer Science Conference (ITCS 2020)*. Schloss Dagstuhl-Leibniz-Zentrum für Informatik, 2020.

[97] Paat Rusmevichientong, Mika Sumida, and Huseyin Topaloglu. Dynamic assortment optimization for reusable products with random usage durations. *Management Science*, 2020. Forthcoming.

[98] Guillaume Sabran, Samitha Samaranayake, and Alexandre Bayen. Precomputation techniques for the stochastic on-time arrival problem. In *2014 Proceedings of the Sixteenth Workshop on Algorithm Engineering and Experiments (ALENEX)*, 2014.

[99] Nathan Shank and Hannah Yang. Coupon collector problem for non-uniform coupons and random quotas. *The electronic journal of combinatorics*, 20(2):33, 2013.

[100] Sahil Singla. The price of information in combinatorial optimization. In *ACM-SIAM SODA'18*, 2018.

[101] Jing-Sheng Song and Paul Zipkin. Supply chain operations: Assemble-to-order systems. *Handbooks in operations research and management science*, 11:561–596, 2003.

[102] Kalyan Talluri and Garrett Van Ryzin. An analysis of bid-price controls for network revenue management. *Management science*, 44(11-part-1):1577–1593, 1998.

[103] Kalyan T Talluri and Garrett J Van Ryzin. *The theory and practice of revenue management*, volume 68. Springer Science & Business Media, 2006.

[104] John N Tsitsiklis and Benjamin Van Roy. Regression methods for pricing complex american-style options. *IEEE Transactions on Neural Networks*, 12(4):694–703, 2001.

[105] Xinshang Wang, Van-Anh Truong, and David Bank. Online advance admission scheduling for services with customer preferences. *arXiv preprint arXiv:1805.10412*, 2018.

[106] Colin White. Lower bounds in the preprocessing and query phases of routing algorithms. In *Algorithms-ESA 2015*, pages 1013–1024. Springer, 2015.

[107] Dawn Woodard, Galina Nogin, Paul Koch, David Racz, Moises Goldszmidt, and Eric Horvitz. Predicting travel time reliability using mobile phone gps data. *Transportation Research Part C: Emerging Technologies*, 75:30–44, 2017.

[108] Huasen Wu, R Srikant, Xin Liu, and Chong Jiang. Algorithms with logarithmic or sublinear regret for constrained contextual bandits. In *Advances in Neural Information Processing Systems*, pages 433–441, 2015.

[109] Huasen Wu, R Srikant, Xin Liu, and Chong Jiang. Algorithms with logarithmic or sublinear regret for constrained contextual bandits. In *Advances in Neural Information Processing Systems*, pages 433–441, 2015.

[110] Huanan Zhang, Cong Shi, Chao Qin, and Cheng Hua. Stochastic regret minimization for revenue management problems with nonstationary demands. *Naval Research Logistics*, 63(6):433–448, 2016.