ALGORITHM AND MECHANISM DESIGN IN DECISION-MAKING ENVIRONMENTS

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

Makis Arsenis

August 2023

ALGORITHM AND MECHANISM DESIGN IN DECISION-MAKING ENVIRONMENTS

Makis Arsenis, Ph.D.

Cornell University 2023

This dissertation studies algorithm and mechanism design for decision-making environments with specific characteristics regarding the input generation process and the desired output. We focus particularly on addressing the challenges posed by online input, stochastic input, input distributed among strategic and prone-to-failure agents, as well as considerations regarding the fairness of the algorithmic decisions made. Our contributions amount to making progress in the field of algorithmic decision-making by studying new versions of standard problems in Combinatorial Optimization, Optimal Stopping Theory, and Mechanism Design better tailored for such decision-making environments in one or more aspects. Each problem is accompanied by some theoretical performance guarantees appropriate for each setting, offering insights into the trade-offs between different assumptions and constraints.

In the first part of this thesis we focus on *Prophet Inequalities*, a collection of results for online decision-making problems under uncertainty and we explore two aspects of those problems. The first has to do with the power of the decision-maker to decide the order in which to inspect parts of the input, or in other words, the order in which to eliminate uncertainty about the input. We introduce a model interpolating between the two extremes of having no such power to having full order-selection power and quantify the performance vs. freedom trade-offs. The second aspect has to do with the fairness of the decision-making process. We define two notions of Individual Fairness in the context of Prophet Inequality problems, discuss techniques for designing optimal

algorithms under fairness constraints and prove competitiveness guarantees.

We then study the emblematic Mechanism Design problem of *Revenue-maximizing Auctions* under the novel assumption that a subset of the bidders are "Byzantine", a term borrowed from the area of Computer Systems, meaning they can behave in arbitrary ways, departing from the assumptions under which the mechanism was designed. In this setting, we prove a revenue monotonicity theorem characterizing exactly the settings in which the presence of those Byzantine, or misspecified bidders, does not hurt the performance (revenue) of the auction compared to an auction designed in the absence of them.

Finally, we study the classic Combinatorial Optimization problem of *Maximum Flow* from a new, online perspective where sources arrive one at a time and request to send a unit of flow while the algorithm is required to serve each request, maintaining an optimal flow after each source arrival. To achieve this, the algorithm might need to modify the flow maintained at each step which is assumed to incur a cost. We analyze a natural algorithm for this problem and bound its worst-case cost using techniques developed for a similar online version of the Bipartite Matching problem.

BIOGRAPHICAL SKETCH

Makis grew up in Athens, Greece. At a young age, Makis developed a passion for computer programming, which lead him to participate and excel in national and international programming competitions throughout his high school years. He received a diploma in Electrical and Computer Engineering (ECE) from the National Technical University of Athens (NTUA). During his undergraduate studies, he had the opportunity to collaborate with Dimitris Fotakis on a research project that played a pivotal role in shaping his research interests and fueling his intellectual curiosity. He continued his education by joining the Computer Science Ph.D. program at Cornell University in 2017, where he had the privilege of being advised by Robert Kleinberg. Complementing his doctoral studies, Makis also gained valuable industry experience as a Software Engineer during two consecutive summers, working as an intern at Google.

This page intentionally left blank.

ACKNOWLEDGEMENTS

I am deeply grateful for the invaluable support and unwavering guidance provided by my advisor, Bobby Kleinberg, during my studies. Without him, the completion of this manuscript would have been an insurmountable task. During the course of my Ph.D., Bobby consistently prioritized my professional development and mental well-being. When I approached him as a first-year graduate student with a general interest in theoretical computer science, I had yet to settle on a specific research direction. Bobby generously encouraged my exploration of various areas at the intersection of our research interests, allowing me to pursue the subjects that excited me the most. His mentorship encompassed every aspect of the research process, from formulating intriguing questions and crafting conjectures to writing, motivating, and publishing our results. Equally significant was the fact that he instilled in me his work ethic, meticulous attention to detail, and optimistic approach to research. I am also thankful to the other members of my committee, Éva Tardos and Rachit Agarwal, for their gentle feedback, advice, and support at every critical moment of my Ph.D.

A special appreciation is owed to my friend and collaborator, Odysseas Drosis. His time as an M.Eng. student at Cornell in the academic year 2019–2020 proved pivotal in my research direction. Odysseas' infectious enthusiasm and dedication served as powerful motivators, propelling us to work diligently and resulting in the completion of a significant portion of the work presented in this dissertation. Our distinct research styles complemented each other, acting as an essential catalyst to accelerating our progress.

I would also like to express my appreciation to my hosts at Google during my internships as a Software Engineer in the summers of 2021 and 2022: Jagan Sankaranarayanan, Vijayshankar Raman and Elaine Ang for their mentorship, support and career advice.

It has been an honor and absolute pleasure to be a part of Cornell's CS Theory Lab. I've had a wonderful time interacting with all the past and present members of the

lab. Among them, I wish to especially acknowledge Shijin Rajakrishnan, Jesse Goodman, Drishti Wali, Giannis Fikioris, Jyun-Jie Liao, Sloan Nietert, Spencer Peters, Jason Gaitonde, Ayush Sekhari and Thodoris Lykouris.

My six years at Cornell University and in Ithaca were enriched by the presence of remarkable individuals who have become lifelong friends. I want to extend my heartfelt gratitude particularly to Richard Bowen and Shijin Rajakrishnan, for their invaluable support. I want to also thank Drishti Wali who, along with Shijin, provided a much-needed anchor during the COVID pandemic. Additionally, I cannot fail to mention the other Greeks with whom I (re)united here including Giannis Fikioris, Victor Giannakouris, Vasilis Charisopoulos and Marios Papachristou.

My studies at Cornell would not have been possible without the support of mentors and teachers from NTUA. I want to particularly thank Dimitris Fotakis, Stathis Zachos and Nikos Papaspyrou for their academic as well as personal support. Moving to the other side of the globe for my studies was not an easy endeavor, but I was able to tackle all hardships with the invaluable support of many friends including George Papadimitriou, Dionysis Zindros, Lydia Zakynthinou and Thodoris Lykouris. Special appreciation goes to Sophia Georgiakaki and her family for their steadfast support and assistance when I first arrived in Ithaca.

Finally, I wish to express my heartfelt appreciation to my family for their love and support: my grandparents, Aggeliki and Makis Arsenis; my sister, Aggeliki Arseni; my cousin, Ioanna Vlanti; and especially my father, Panagiotis Arsenis, whose encouragement, especially during my high school years, has been the driving force behind my pursuit of mathematics, engineering, and the sciences.

TABLE OF CONTENTS

CHAPTER 1

INTRODUCTION

Computer Science (CS) is the field that studies *computation*, a pervasive concept relevant to many other disciplines outside the narrow scope of computing technologies. In particular, the notion of an *algorithm* has been fundamental to numerous disciplines throughout history despite not being formally defined until the early 20th century in the groundbreaking works of Alonzo Church and Alan Turing. Fields such as Operations Research, Economics/Finance and Statistics/Data Science, to name a few, have benefited from incorporating a computational approach to their methods but have also enriched Computer Science by providing tools and new perspectives for approaching problems and given fruitful grounds for new applications of classic methods/results from within CS.

Traditionally, CS has focused primarily on efficiency, recognizing that fast, scalable algorithms are essential to driving progress in the field. However, in certain applications, efficiency is not always the primary concern or the bottleneck. With the pervasive presence of interconnected computer systems in our everyday life, algorithms frequently play the critical role of a *decision maker* in the sense that they have more responsibilities than just processing data with the eventual goal of presenting them to a human. Rather, they are asked to make decisions in place of humans, decisions which can have significant impact on people's lives. In such cases, the design process needs focus on the *quality* and *robustness* of the decisions as well as adapt to the special *characteristics of the environment* in which they are deployed, for example by adapting the way in which input is acquired.

Take for instance a *ride-sharing platform*. Users make requests for rides and the platform decides how to match them with available drivers. Naturally, those requests, as

well as the availability of drivers, cannot be known in advance and only arrive sequentially as needed, in contrast to a standard algorithm design environment where it is typically assumed that the algorithm has access to the input all at once. More importantly, the algorithm is making a decision that will affect the user's experience, reflected through, for example, the amount of time it will take them to reach their destination or the amount of money they will be charged for the ride. Another important aspect of a setting like this is how the algorithm's optimization objective (typically maximizing revenue for the platform) is usually not aligned with the users' objectives. As a result, users might be incentivized to lie about their preferences. For instance, [91] describe a scenario where drivers are colluding and misreporting their availability by turning their phone apps off in a round-robin fashion in order to cause an artificial shortage of drivers which prompts the platform to surge prices. Another consequence of this misalignment between the optimization objective of the decision-maker and the users, is that the outcome might end up favoring some users over others when it is beneficial to the platform owner. In other words, *optimal* decisions from the perspective of the platform might be *unfair* from the perspective of the users.

In this thesis we study *Algorithm Design* for environments with characteristics commonly encountered in *algorithmic decision-making* and largely having to do with how the *input* is generated and accessed and what constitutes a valid and/or desired *output*. The unique features of those environments can usually be categorized along the following broad, and sometimes overlapping dimensions[1]:

- **Online Input**: Decision-making often takes place in dynamic environments where decisions have to be made even in the absence of some of the relevant in-

[1]It is important to note here that each of these have been individually studied and are established as independent research areas within Computer Science. In this thesis we are merely taking a unified approach to algorithmic decision-making by recognizing the importance of those areas and through our work we drive progress within each area.

formation. The framework of *Online Algorithms* [16] models such situations, by assuming sequential revelation of the input to the algorithm. The goal is typically to design "competitive" algorithms in the sense that they should perform well in such an online environment compared to an optimal algorithm with access to the entirety of the input (offline algorithm), under certain assumptions on the input generation process — usually some form of adversarially generated input. Many standard combinatorial problems have been studied in this framework including Graph Matchings and Job Scheduling.

Commonly in Online Algorithm Design, the algorithm is required to make *irrevocable* decisions at the time new information arrives. For instance, in the Online Bipartite Matching problem with vertex arrival, the decision of which vertex v to match (if any) with the newly arrived vertex u has to happen immediately after u's arrival and cannot be modified later on. Therefore, the final matching produced by the online algorithm might be sub-optimal due to poor decisions made during the early stages where little information about the graph was available. Alternatively, one might allow the online algorithm to alter decisions made in the past by incurring some cost. Some authors refer to this as *online algorithms with recourse* and in such settings the notion of "competitiveness" reduces to cost minimization (see Section 2.1 for more details).

- **Stochastic Input**: Traditionally, Computer Science has excelled at proving worst-case results with respect to the algorithm's input instance. These results are highly robust since they guarantee good performance regardless of how the input is generated. However, in cases where bounded worst-case performance is unattainable, or simply when worst-case bounds are too pessimistic to be of any practical value, more appropriate models for input generation need to be considered. One such model, which is common in decision-making is to assume the input is *stochastic*,

generated according to some probability distribution which is either known or can be learned by the algorithm designer.

Besides assuming stochastically generated input, other approaches for replacing worst-case analysis with more realistic kinds of analysis methods exist as well; the interested reader is referred to an excellent survey and corresponding book by Tim Roughgarden [85, 1].

- **Distributed Input**: In certain cases, the input is is not owned by the algorithm but instead is distributed among different entities. There are two prominent complications when designing algorithms for such settings.

  First, the entities might be strategic, self-interested agents who may misreport the information they hold in order to influence the outcome. *Mechanism Design* [78], drawing on ideas from Game Theory and Economics, provides the framework for designing algorithms (called "mechanisms" in this context) under the presence of *incentives* and has received considerable attention within the Computer Science community in recent decades. A typical goal is to design *truthful* Mechanisms, meaning algorithms with the property that each input-owning entity is incentivized to report their true input.

  Additionally, a small fraction of the entities involved might be corrupted or malfunctioning in which case they may report arbitrary information to the algorithm. Designing algorithms robust against such agents is a common objective in the area of *Distributed Systems* where it is known by the term *Byzantine fault tolerance* [66].

- **Fairness**: *Algorithmic Fairness* [59] is an active area of research which tries to quantify the abstract concept of fairness of an algorithm's output across various settings. The area has garnered significant attention recently, leading to numerous attempts to formulate precise mathematical definitions of fairness suitable for

different contexts. Most of those definitions fall into one of two broad categories: *individual fairness* and *group fairness*. The former, quoting Dwork et al. [38], tries to capture the notion that "similar individuals [should be] treated similarly" and the latter tries to ensure that protected groups of people (e.g. historically marginalized groups) are treated similarly to the population as a whole in a statistical sense [92].

The main contribution of this thesis is to make progress in the field of Algorithmic Decision-Making through the introduction of *new versions* of standard problems in Combinatorial Optimization, Optimal Stopping Theory and Mechanism Design better tailored for decision-making environments by adjusting the power of the algorithm and/or the adversary in one or more of the four dimensions described above. For each problem studied in the following chapters of this thesis, we provide theoretical performance guarantees appropriate for each context. These guarantees typically involve bounding a competitive ratio, minimizing a cost metric or establishing robustness against model misspecifications and can often give insight on the trade-offs between competing objectives and assumptions that the algorithm designer should be aware of.

While some of our contributions are applicable to a few different combinatorial optimization problems, we focus our attention on some representative problems chosen either for being better suited to studying the respective aspect of interest or simply because they provide a fruitful ground for interesting theoretical results. Those problems are: *Prophet Inequalities*, *Revenue-maximizing Auctions* and the *Maximum Flow* problem. In the next section we briefly introduce those problems and summarize our contributions and results. We defer a more formal discussion of the problems to Chapter 2 and a more accurate description of the results to the introduction sections of each subsequent chapter.

The results described in the chapters of this thesis are based on joint work involving a subset of the authors: Robert Kleinberg, Odysseas Drosis and Makis Arsenis. Chapter 3 appears in SODA '21 under the title "Constrained-Order Prophet Inequalities". Chapter 4 extends on the work published as extended abstract in EC '22 with the title "Individual Fairness in Prophet Inequalities". Chapter 5 is based on "Revenue Monotonicity under Misspecified Bidders" which appears in WINE '20 and Chapter 6 is based on "Online Flow Computation on Unit-Vertex-Capacitated Networks" appearing in APoCS '20.

1.1 Overview of Results

We start by considering a classic problem of Optimal Stopping Theory. *Prophet Inequalities* bound the ratio between the expected value obtained by two parties each selecting one value from a set of independent random variables: a "prophet" (used as a benchmark) who knows the value of each variable and may select the maximum one, and the algorithm who observes the values according to some adversarially chosen order and can make the decision to select (at most) one of them immediately after observing it, without knowing what values will be sampled for the unobserved variables. It is a simple model that captures some of the intricacies of a typical decision-making scenario where the algorithm is constrained to making sequential, irrevocable decisions in an online and stochastic environment.

In Chapter 3 we explore the power of order-selection, i.e. the ability of the algorithm to control the order in which the random variables are observed (i.e. their value is revealed). Both extremes of the spectrum correspond to problems that have been considered before. On one end, the original (adversarial order) Prophet Inequalities problem assumes no

control on the order of inspection (adversarial order). On the other end, the free-order variant gives the algorithm total control over the order of inspection by allowing it to adaptively choose the next random variable for inspection. However, we are the first to ask the question of how much freedom in the selection process is really required to achieve the same performance guarantees as total freedom and further, what does the landscape look like when interpolating between the two extremes. We answer this question through a series of results which asymptotically describe the trade-offs between the amount of freedom and the competitive ratio achievable.

In Chapter 4, we again focus on the Prophet Inequality problem (in its original, adversarial order version) but this time focusing on the fairness properties of the algorithm. We introduce two independent notions of individual fairness that might be desirable in certain applications. We show how to efficiently design optimal, fair algorithms and prove competitiveness guarantees between all combinations of online vs. offline and fair vs. unfair algorithms.

In Chapter 5 we switch gears and consider the standard Mechanism Design problem of Revenue-maximizing Auctions. This is the problem of allocating resources to agents whose valuations are private, with the goal of maximizing the revenue generated by the algorithm. It is a standard assumption that the Mechanism Designer has accurate knowledge of the distributions from which the valuations of the agents are sampled. In this chapter we relax this assumption in a binary way, allowing the existence of agents for whom the distributions that the Mechanism Designer holds are completely inaccurate (red bidder) while the rest of them are totally accurate (green bidder). The mechanism designer crucially does not know the type (color) of each bidder. Another way of interpreting this divide between red and green bidders is that the red bidders are "Byzantine" players i.e. malfunctioning or rogue agents who are not behaving strategically but in-

stead can report arbitrary valuations to the mechanism. We manage to exactly characterize the settings in which the existence of the red agents does not harm the revenue of the optimal mechanism (designed under potentially wrong assumptions due to the fact that the mechanism designer doesn't know who the red bidders are) when compared to the same mechanism running on the green bidders alone. Stated in another way, we prove that there is a sense in which the optimal mechanism is robust against such Byzantine agents.

Finally, in Chapter 6 we study the Maximum Flow problem in an online setting. Motivated by a similar online version of the Bipartite Matching problem appearing in [12], we propose a related formulation of the Maximum Flow problem with sources arriving over time and requesting to send one unit of flow to any one among a set of static sinks. The algorithm is required to maintain on maximum flow after each source arrival and is allowed to update the flow in an arbitrary way between arrivals, however they incur a cost, called a *flow-switching cost*, each time the flow is modified. The goal is to maintain a maximum flow at all times while minimizing the total flow-switching cost. An online formulation like this is usually referred to as online algorithms with recourse. We manage to transfer the competitiveness guarantees proved in [12] in the context of the Bipartite Matching problem to a subclass of instances of our Maximum Flow problem by presenting a reduction between the two settings.

CHAPTER 2

BACKGROUND

In this chapter we formally introduce the relevant background on which later chapters build. The material presented here consists of standard analysis frameworks, problems and combinatorial structures relevant to online decision-making. It is structured as a collection of largely independent sections and the reader can treat it as a reference when studying later chapters.

2.1 Online Algorithms in Adversarial Environments

We start by introducing a standard competitive analysis framework for adversarial, online environments known in the literature as *Request-answer Games* [16, Chapter 7]. It is general enough to be able to capture a variety of online problems including some discussed in this thesis. The material presented here becomes relevant in Chapter 6 but it also serves as a precursor to Section 2.2. Note that the settings described in this section do not capture any stochastic aspects of the environment. We defer that for Section 2.2.

Problem 1 (Request-answer Game)**.** Two parties are involved in this game: a deterministic[1] *algorithm* and an *adversary*. There is a request set R of possible requests the adversary can issue on each time step. The input to the algorithm consists of a finite sequence of requests $\mathbf{r} = (r_1, r_2, \ldots, r_n) \in R^n$ revealed to the algorithm one element at a time over a period of n discrete time-steps. The request sequence is generated by an adversary who knows the algorithm. For each time-step $i \in [n] := \{1, 2, \ldots, n\}$, there is an associated, finite answer set (or action set) A of possible actions that the algorithm

[1] More generally, one can consider randomized online algorithms. In fact, we do make this assumption in the slightly different setting presented in Section 2.2. To keep the discussion simple here, we focus exclusively on deterministic algorithms which is the case of interest for Chapter 6.

can take as a result of processing the i-th request[2].

Algorithms for request-answer games A deterministic, *online algorithm* ALG for a request-answer game is a sequence of functions $g_i : R^i \to A$ mapping a prefix of a request sequence that has been revealed up to time-step i to an action to be taken at that time-step. Request-answer games are usually associated with *optimization objectives* which can either be of maximizing a payoff or minimizing a cost. We introduce both frameworks below.

Payoff-maximization request-answer games A function $\mathrm{P} : R^n \times A^n \to [0, +\infty)$ expresses an algorithm's *payoff* (or *reward*) in terms of the request sequence and the actions taken. Abusing notation, we use $\mathrm{ALG}(\mathbf{r})$ to denote the algorithm's payoff for the request sequence $\mathbf{r}$ and will sometimes omit $\mathbf{r}$ if it is obvious from context:

$$\mathrm{ALG} := \mathrm{ALG}(\mathbf{r}) := \mathrm{P}(r_1, r_2, \ldots, r_n, g_1(r_1), g_2(r_1, r_2), \ldots, g_n(r_1, r_2, \ldots, r_n)).$$

We also define OPT to be the (deterministic) offline algorithm computing the optimal (payoff maximizing) action sequence in hindsight. Abusing notation again, this algorithm's payoff is denoted by

$$\mathrm{OPT} := \mathrm{OPT}(\mathbf{r}) := \max_{\mathbf{a} \in A^n} \mathrm{P}(\mathbf{r}, \mathbf{a}).$$

Cost-minimization request-answer games A function $\mathrm{C} : R^n \times A^n \to [0, +\infty)$ expresses an algorithm's *cost* in terms of the request sequence and the actions taken. Abusing notation, we use $\mathrm{ALG}(\mathbf{r})$ to denote the algorithm's *cost* for the request sequence $\mathbf{r}$

[2]In principle, one can relax some of the assumptions of the model such as the finiteness of the sets or the constraint that actions sets are equal among different time-steps. See [16, Chapter 7] for a more in-depth treatment.

and will sometimes omit $\mathbf{r}$ if it is obvious from context:

$$\text{ALG} := \text{ALG}(\mathbf{r}) := \text{C}(r_1, r_2, \ldots, r_n, g_1(r_1), g_2(r_1, r_2), \ldots, g_n(r_1, r_2, \ldots, r_n)).$$

We also define OPT to be the (deterministic) offline algorithm computing the optimal (cost minimizing) action sequence in hindsight. Abusing notation again, this algorithm's cost is denoted by

$$\text{OPT} := \text{OPT}(\mathbf{r}) := \min_{\mathbf{a} \in A^n} \text{C}(\mathbf{r}, \mathbf{a}).$$

Competitive Ratio To evaluate the performance of an online algorithm, it is typical to compare its objective function value (payoff or cost) with that of OPT. A typical metric developed for this purpose is the *competitive ratio*[3]. Below we give the relevant definitions for both payoff-maximization and cost-minimization frameworks.

Definition 1 (Competitive ratio for payoff-maximization games)**.** An online algorithm ALG for a *payoff-maximization* request-answer game is said to be c_n-competitive[4], or have a competitive ratio of *at least* $c_n \in [0, 1]$ if for all request sequences $\mathbf{r} \in R^n$ of length n,

$$\text{ALG}(\mathbf{r}) \geq c_n \cdot \text{OPT}(\mathbf{r}).$$

The largest c_n for which ALG is c_n-competitive is called the *competitive ratio* of ALG. Equivalently, assuming no request sequence induces an optimal payoff of zero, one can define the competitive ratio of an algorithm as,

$$\text{competitive ratio of ALG} = \inf_{\mathbf{r} \in R^n} \frac{\text{ALG}(\mathbf{r})}{\text{OPT}(\mathbf{r})}.$$

[3]This is not the only way one can compare the performance of an online algorithm to an offline counterpart. Other metrics, for example the notion of *regret* have been developed, particularly useful for some stochastic environments. In the problems we study in this dissertation, the competitive ratio is the more well-established metric so we use it almost exclusively. See [16, Chapter 15] for an in-depth discussion.

[4]Notice how we allow the competitive ratio to be a function of the size of the input n.

The competitive ratio of a particular request-answer game $\mathcal{G}$ is defined to be the competitive ratio of the best online algorithm for $\mathcal{G}$,

$$\text{competitive ratio of } \mathcal{G} = \sup_{\text{ALG}} \inf_{\mathbf{r} \in R^n} \frac{\text{ALG}(\mathbf{r})}{\text{OPT}(\mathbf{r})}.$$

Definition 2 (Competitive ratio for cost-minimization games)**.** An online algorithm ALG for a *cost-minimization* request-answer game is said to be c_n-competitive, or have a competitive ratio of *at least* $c_n \in [1, +\infty)$ if for all request sequences $\mathbf{r} \in R^n$ of length n,

$$\text{ALG}(\mathbf{r}) \leq c_n \cdot \text{OPT}(\mathbf{r}).$$

The smallest c_n for which ALG is c_n-competitive is called the *competitive ratio* of ALG. Equivalently, assuming no request sequence induces an optimal cost of zero, one can define the competitive ratio of an algorithm as,

$$\text{competitive ratio of ALG} = \sup_{\mathbf{r} \in R^n} \frac{\text{ALG}(\mathbf{r})}{\text{OPT}(\mathbf{r})}.$$

The competitive ratio of a particular cost-minimization request-answer game $\mathcal{G}$ is defined to be the competitive ratio of the best online algorithm for $\mathcal{G}$,

$$\text{competitive ratio of } \mathcal{G} = \inf_{\text{ALG}} \sup_{\mathbf{r} \in R^n} \frac{\text{ALG}(\mathbf{r})}{\text{OPT}(\mathbf{r})}.$$

Online Algorithms Typically, an algorithmic problem can be studied from an online perspective by setting up a request-answer game as follows: the input is partitioned into requests in a natural way, specific to the problem, and the objective is to build the final solution step-by-step where each action is able to *append* an element to a currently maintained solution but *not* otherwise modify the solution. In other words, each action is an irrevocable decision about part of the final solution. The payoff function is defined to match exactly the maximization objective of the original problem and so the competitive ratio in this setting is a natural way to evaluate how close the online algorithm can get

to an offline counterpart which is not constrained to make committing decisions before seeing the entirety of the input. This is a realistic model for situations where updating the solution is prohibitively expensive or impossible.

Online Algorithms with Recourse Another way of framing a problem as a (cost-minimization) request-answer game which can be more appropriate for some applications is to allow actions to modify a currently maintained solution in an arbitrary way, but each modification incurs a positive cost, sometimes called a *switching cost*. In a setting like this, it is reasonable to require the algorithm to maintain a solution after each request that is *optimal* (or near-optimal) for the part of the input that has been revealed so far, and as a result, build a final solution that is optimal (resp. near optimal) for the entire input instance. The competitive ratio is still a reasonable metric to optimize in this scenario for the following reason: typically, for an input instance whose optimal solution is of size m, an offline algorithm can build it by appending one element at a time at a total cost of $\Theta(m)$. Therefore, the denominator in the definition of the competitive ratio is constant among all instances and optimizing the competitive ratio reduces to designing an algorithm that minimizes the worst-case switching cost.

Among the problems studied in this framework, of particular interest to this thesis are Bipartite Matching problems (Section 2.5) and Network Flow problems (Chapter 6). We study both in an incremental setting where vertices arrive in an online manner and an optimal solution needs to be maintained while minimizing the cost of updating the solution dynamically. Similar problems are considered in [46] where they study the trade-offs between the approximation ratio and switching cost. In [81], they also consider problems involving flow computation on bipartite graphs in a decremental model where edges and nodes disappear over time. Other problems include Job Scheduling [81, 87, 94], Maximal Independent Set [4, 26, 45] and Minimum Set Cover [45].

2.2 Online Algorithms in Stochastic Environments

We now turn our attention to a set of results colloquially known as *Prophet Inequalities*. These are competitive ratio guarantees for online decision-making problems under uncertainty traditionally studied by Optimal Stopping Theory, but more recently also by the Mechanism Design community due to their applications in sequential auction design [69]. Chapters 3 and 4 rely heavily on the material presented here.

To help develop intuition for the problem defined below, consider the following informal description of an online decision-making game which serves as the basis for all problems later formalized in this section.

Prophet Problem (informal) You are presented with n opaque boxes numbered 1 through n. Inside each box is a piece of paper with a non-negative real number written on it. You are promised that the number in box i was sampled from a distribution $\mathcal{F}_i$ whose description is written on a visible spot on top of box i. You are allowed to start inspecting the boxes in the order given, i.e. first box 1, then box 2 etc. Inspecting box i means opening it and looking at the number X_i written on the piece of paper inside it. At this point you have to make an *immediate* and *irrevocable* decision to either *accept* an amount of $\$X_i$ as a reward and end the game there, or reject it and proceed to inspect the next box (if there is any remaining). If you reject all amounts, your reward is \$0. How should you play this game to maximize your expected reward?

Perhaps surprisingly, it turns out there is a way for you to guarantee an expected reward *at least half* of the expected reward of a "prophet", i.e. someone able to see the future and predict the values in the boxes not opened yet and who can trivially choose to accept the maximum among all values in the boxes, hence the name of the problem.

The framework of request-answer games introduced earlier is not adequate to capture the stochastic aspects of prophet problems. In this section we adapt the concepts introduced previously to accommodate the study of such kinds of stochastic problems under a unified framework by introducing three entities: a *decision-maker*, an *adversary* and *nature*. We then formalize three versions of the prophet problem depending on which of the three entities controls the order of inspection of the boxes.

Before moving on, let us introduce the notion of an *instance* of a problem and the notion of a *stopping rule* which is the name used in Optimal Stopping Theory literature for an online (perhaps randomized) algorithm for this setting.

Definition 3 (Instance)**.** An instance of a prophet problem is a vector $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$ of n probability distributions for the values of n non-negative, independent random variables $X_1, X_2, \ldots, X_n$. Depending on the context, the distributions might be discrete, continuous or a mix of the two. For continuous $\mathcal{F}_i$, we will usually denote the associated probability density function (p.d.f.) by f_i and the cumulative distribution function (c.d.f) by F_i.

Definition 4 (Stopping rule)**.** A *stopping rule* $\mathrm{ALG}_{\mathcal{I},\pi}$ (or simply ALG) adapted to the instance $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$ and the permutation $\pi : [n] \to [n]$ is an online (perhaps randomized) algorithm which inspects the random variables in order $X_{\pi(1)}, X_{\pi(2)}, \ldots, X_{\pi(n)}$ and, after inspecting the value of $X_{\pi(i)}$ takes one of two actions, *accept* or *reject*, based on the values $X_{\pi(1)}, X_{\pi(2)}, \ldots, X_{\pi(i)}$ inspected so far. Accepting an element terminates the process assigns a *payoff/reward* equal to the value of the accepted element. Rejecting an element means proceeding to inspect the next one (if any).

We overload the notation $\mathrm{ALG}_{\mathcal{I},\pi}$ (or simply ALG) to also denote the random variable of the reward obtained by the stopping rule. Sometimes, we also associate with the stopping rule a random variable $\tau \in [n] \cup \{\perp\}$ indicating the *time-step* at which an

element was accepted (or $\perp$ if no element was accepted[5]), i.e. if $X_{\pi(i)}$ was accepted then $\tau = i$.

The following special kind of stopping rule will be be of interest later due to its simplicity and power.

Definition 5 (Single-threshold Stopping Rule)**.** A stopping rule adapted to an instance $\mathcal{I}$ and permutation π is a *single-threshold stopping rule* with threshold T, denoted by $\mathrm{ALG}_{\mathcal{I},\pi,T}$, if it never accepts a value strictly less than T, and it always accepts the first value strictly greater than T. In other words, the time-step $\tau \in [n] \cup \{\perp\}$ where an acceptance decision was made by the stopping rule must satisfy the following constraints for all $i \in [n]$:

$$X_{\pi(i)} < T \Rightarrow \tau \neq i, \qquad X_{\pi(i)} > T \Rightarrow \tau \leq i.$$

Note that the tie-breaking behavior of a single-threshold stopping rule is unconstrained: when $X_{\pi(i)} = T$, then $\tau = i$ is allowed but not required[6]. However, when the distributions $\mathcal{F}_1, \ldots, \mathcal{F}_n$ are continuous and have no point-masses, the event $[X_{\pi(i)} = T]$ has probability zero and therefore the choice of a tie-breaking rule is inconsequential. There is a general way of adapting the definition of a single-threshold stopping rule to distributions which possibly contain point-masses which we describe extensively in Appendix A.2.2.

We are now ready to formally define the three variants of interest for the prophet problem.

Problem 2 (Adversarial-order Prophet Problem)**.** The **adversary** chooses an *instance* $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$ *and* a permutation $\pi : [n] \to [n]$. **Nature** samples the value of each X_i

[5]We impose the ordering $i < \perp$ for all integers i when comparing elements of the set $[n] \cup \{\perp\}$.

[6]This tie-breaking convention is not universal. For example [40] adopt the stricter convention that $X_{\pi(i)} = T$ implies $\tau = i$.

from the respective distribution $\mathcal{F}_i$. The **decision-maker** receives $\mathcal{I}, \pi$ and decides on a *stopping rule* $\text{ALG}_{\mathcal{I},\pi}$ adapted to $\mathcal{I}, \pi$.

Problem 3 (Free-order Prophet Problem)**.** The **adversary** chooses an *instance* $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$. **Nature** samples the value of each X_i from the respective distribution $\mathcal{F}_i$. The **decision-maker** receives the $\mathcal{I}$, decides on a permutation[7] $\pi : [n] \to [n]$ and a *stopping rule* $\text{ALG}_{\mathcal{I},\pi}$ adapted to $\mathcal{I}, \pi$.

Problem 4 (Random-order Prophet Problem — Prophet Secretary)**.** The **adversary** chooses an *instance* $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$. **Nature** chooses a permutation $\pi : [n] \to [n]$ *uniformly* at random from the symmetric group S_n of all permutations on $[n]$ and also samples the value of each X_i from the respective distribution $\mathcal{F}_i$. The **decision-maker** receives the $\mathcal{I}, \pi$ and decides on a *stopping rule* $\text{ALG}_{\mathcal{I},\pi}$ adapted $\mathcal{I}, \pi$.

Remark 1. Notice how, in contrast to the previous section, the **adversary** only has control over the "offline" portion of the input[8] that is immediately visible to the decision-maker. In other words, the adversary controls the setup of the problem with respect to which we want to obtain worst-case bounds. **Nature** on the other hand controls the "online" part of the input that is revealed sequentially and in this sense, the results we prove later belong to what we described in the introduction as beyond-worst-case bounds.

In each of the above problems, we can define a "prophet" as an optimal offline algorithm with access to all the sampled values X_i and which can trivially select the maximum among them. We denote by $\text{OPTOFF}_{\mathcal{I}}$, or simply OPTOFF when the instance is implicit,

[7] In principle one could also consider a model in which the permutation π is specified adaptively, i.e. the value $\pi(k)$ is allowed to depend on $X_{\pi(1)}, \ldots, X_{\pi(k-1)}$. One of the main results of [52], Theorem 3.11, shows that the optimal adaptive ordering is no better than the optimal non-adaptive one.

[8] As a consequence, there is no need to distinguish between *oblivious* and *adaptive* adversaries which is usually the case with randomized online algorithms.

the random variable of the value selected by the prophet, in other words, $\text{OPTO}_{\text{FF}_{\mathcal{I}}} = \max_{i\in[n]} X_i$.

The notion of the "competitive ratio" introduced in the previous section can be adapted to this setting and it still serves as a good measure for evaluating the performance of an online algorithm for a prophet problem. The most common way of defining it is as a ratio of expected values: the decision-maker's expected payoff over a prophet's expected payoff.

Definition 6 (Adversarial-order Competitive Ratio)**.** Fix a positive integer n. The *competitive ratio* of the *adversarial-order prophet problem* (Problem 2) over instances of size n is defined as,

$$\mathcal{R}_n^{\text{adv}} = \inf_{\mathcal{I}=(\mathcal{F}_i)_{i\in[n]},\pi\in S_n} \frac{\sup_{\text{ALG}_{\mathcal{I},\pi}} \mathbb{E}_{\mathcal{I}}[\text{ALG}_{\mathcal{I},\pi}]}{\mathbb{E}_{\mathcal{I}}[\text{OPTO}_{\text{FF}_{\mathcal{I}}}]}.$$

The *single-threshold competitive ratio* $\overline{\mathcal{R}}_n^{\text{adv}}$ for instances of size n is similarly defined with the supremum being over all single-threshold stopping rules. The *asymptotic* competitive ratio is defined as

$$\mathcal{R}^{\text{adv}} = \liminf_{n\to\infty} \mathcal{R}_n^{\text{adv}},$$

and respectively for the asymptotic single-threshold competitive ratio $\overline{\mathcal{R}}^{\text{adv}}$.

Definition 7 (Free-order Competitive Ratio)**.** Fix a positive integer n. The *competitive ratio* of the *free-order prophet problem* (Problem 3) over instances of size n is defined as,

$$\mathcal{R}_n^{\text{free}} = \inf_{\mathcal{I}=(\mathcal{F}_i)_{i\in[n]}} \frac{\sup_{\pi\in S_n,\text{ALG}_{\mathcal{I},\pi}} \mathbb{E}_{\mathcal{I}}[\text{ALG}_{\mathcal{I},\pi}]}{\mathbb{E}_{\mathcal{I}}[\text{OPTO}_{\text{FF}_{\mathcal{I}}}]}.$$

The definitions for single-threshold and/or asymptotic competitive ratios are adapted accordingly.

Definition 8 (Random-order Competitive Ratio)**.** Fix a positive integer n. The *competitive ratio* of the *random-order prophet problem* (Problem 4) over instances of size n is

defined as,

$$\mathcal{R}_n^{\text{rand}} = \inf_{\mathcal{I}=(\mathcal{F}_i)_{i\in[n]}} \frac{\mathbb{E}_{\pi\sim S_n}\left[\sup_{\text{ALG}_{\mathcal{I},\pi}} \mathbb{E}_{\mathcal{I}}[\text{ALG}_{\mathcal{I},\pi}]\right]}{\mathbb{E}_{\mathcal{I}}[\text{OPTOFF}_{\mathcal{I}}]}.$$

The definitions for single-threshold and/or asymptotic competitive ratios are adapted accordingly.

A *Prophet Inequality* is a result of the form of Theorem 1 that follows, bounding the competitive ratio of a prophet problem, first introduced and proven by Krengel and Sucheston in [62, 63] by analyzing the optimal online algorithm which is derived using backwards-induction — essentially, Dynamic Programming. Appendix A.2.1 includes a simpler proof based on [86, 61].

Theorem 1 (Prophet Inequality)**.** *For any n, any instance $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$ of distributions on n independent, non-negative random variables $X_1, X_2, \ldots, X_n$, and any permutation $\pi : [n] \to [n]$, there exists a stopping rule* ALG *for the* adversarial-order prophet problem *adapted to the instance and the ordering such that*

$$\mathbb{E}[\text{ALG}] \geq \frac{1}{2} \cdot \mathbb{E}\left[\max_{i\in[n]} X_i\right]. \tag{2.1}$$

Additionally, the prophet inequality above is tight in the sense that there exists a permutation and a family of instances $\{\mathcal{I}_m\}_{m=1}^{\infty}$ such that for all m, there is no stopping rule for $\mathcal{I}_m$ satisfying an inequality like (2.1) with a constant greater than $\frac{1}{2} + \frac{1}{m}$.. We describe such a family of instances with $n = 2$ distributions in Appendix A.2.1. A consequence of Theorem 1 is that the competitive ratio $\mathcal{R}_n^{\text{adv}}$ of Problem 2 is at least $1/2$ for all n. Combined with the tight sequence of instances presented in the Appendix which can be trivially extended to any number $n \geq 2$ of distributions, we can conclude that $\mathcal{R}_n^{\text{adv}} = 1/2$ for all $n \geq 2$ and therefore the asymptotic competitive ratio of Problem 2 is $\mathcal{R}^{\text{adv}} = 1/2$.

It is noteworthy that there exist single-threshold stopping rules satisfying the same

Prophet Inequality as in (2.1)! In other words, if in the definition of the competitive ratio of the prophet problem we restrict the decision-maker to using single-threshold stopping rules ALG_T, we recover the same competitive ratio of $1/2$. Samuel-Cahn in [86] was the first to notice this and proved a prophet inequality for the single threshold stopping rule using the median of the distribution of $\max_{i\in[n]} X_i$ as a threshold, in other words choosing a T such that $\Pr[\max_{i\in[n]} X_i \geq T] = 1/2$. Later, Kleinberg and Weinberg in [61] noticed that another single-threshold stopping rule, the one using $T = \frac{1}{2}\mathbb{E}[\max_{i\in[n]} X_i]$ as a threshold, also satisfies the same prophet inequality.

The free-order prophet problem was first introduced by [52]. The extra power given to the decision-maker in Problem 3 allows them to cross the $1/2$ competitive ratio barrier presented earlier. Computing the exact (asymptotic) competitive ratio $\mathcal{R}^{\text{free}}$ of Problem 3 is an open problem at the time of writing but we know it lies in the interval $[0.725\ldots, 0.732\ldots]$ with the lower-bound corresponding to an algorithm analyzed recently in [80] and the upper bound of $\sqrt{3} - 1 = 0.732\ldots$ is due to a tight instance appearing in [32].

When confined to single-threshold stopping rules, the power of order selection diminishes compared to more general stopping rules, but still surpasses the $1/2$ barrier of the adversarial-order version. Indeed the single-threshold competitive ratio of the free-order prophet problem is,

$$\overline{\mathcal{R}}_n^{\text{free}} = 1 - \frac{1}{e} + o(1),$$

or, asymptotically equal to $\overline{\mathcal{R}}^{\text{free}} = 1 - 1/e = 0.632\ldots$[9]. A proof is discussed in Appendix Appendix A.2.3.

This prophet secretary problem (or random-order prophet problem) was introduced by

[9] For this result, our assumption that the single-threshold algorithm is allowed to either accept or reject in the special case when $X_i = T$ is crucial, otherwise counterexamples are known [40].

[40] who proved a prophet inequality for general stopping rules with a constant of $1 - 1/e$, later to be marginally improved by [7]. The best known bound for the asymptotic competitive ratio of the Prophet Secretary problem at the time of writing is $0.669\ldots$ due to [32]. The same upper bound of $\sqrt{3} - 1 = 0.732\ldots$ presented in [39] applies in this setting as well. For single-threshold stopping rules, the competitive ratio of Problem 4 is again $\overline{\mathcal{R}}^{\text{rand}} = 1 - 1/e$; see the last paragraph of the introduction to Chapter 3 for a discussion.

2.3 Matroids

Matroids offer a way of extending the notion of *linear independence* of Linear Algebra to any finite set of elements — not necessarily ones that form a vector space. The most prominent connection of Matroids to Computer Science is in the characterization of the problems that admit a greedy solution (see Lemma 1). Matroids have also found applications in Mechanism Design and Online Decision-Making as they turn out to provide enough structure to the underlying problem to guarantee interesting properties. For example [37] exactly characterize, in the context of Mechanism Design, the markets for which optimal auctions satisfy a certain revenue-submodularity property by proving that they are exactly the markets with Matroid constraints. In Chapter 5 we discover a similar characterization for a revenue-monotonicity property, this time under the assumption that we have inaccurate priors on the valuations of a subset of participants.

We proceed with a brief introduction to the theory of Matroids along with important results. We refer the reader to the classic text of Oxley [79] for a more in-depth treatment of the subject.

Given a finite ground set E and a collection $\mathcal{I} \subseteq 2^E$ of subsets of E , we call $\mathcal{M} = (E, \mathcal{I})$

a *set system*. $\mathcal{M}$ is an *independence system* or *a downward-closed set system* if $\emptyset \in \mathcal{I}$ and $\mathcal{I}$ is *downward-closed* as defined below:

(I1) **(downward-closed axiom)** If $A \in \mathcal{I}$ and $B \subseteq A$ then $B \in \mathcal{I}$.

Furthermore, $\mathcal{M}$ is called a *matroid* if it satisfies both (I1) and:

(I2) **(exchange axiom)** If $A, B \in \mathcal{I}$ and $|A| > |B|$ then there exists $x \in A \backslash B$ such that $B + x \in \mathcal{I}$[10].

In the context of independece systems and matroids, sets in (resp. not in) $\mathcal{I}$ are called *independent* (resp. *dependent*). An (inclusion-wise) maximal independent set is called a *basis*. A fundamental consequence of axioms (I1), (I2) is that all bases of a matroid have equal cardinality and this common quantity is called the *rank* of the matroid. A *circuit* is a minimal dependent set. The set of all circuits of a matroid will be denoted by C. The following is a standard property of C.

Proposition 1 ([79, Proposition 1.4.11])**.** *For any C which is the circuit set of a matroid $\mathcal{M}$, let $C_1, C_2 \in C, e \in C_1 \cap C_2$ and $f \in C_1 \backslash C_2$. Then there exists $C_3 \in C$ such that $f \in C_3 \subseteq (C_1 \cup C_2) - e$.*

For any set system $\mathcal{M} = (E, \mathcal{I})$ and any given $S \subseteq E$, define $\mathcal{I}_{|S} = \mathcal{I} \cap 2^S$ and call $\mathcal{M}_{|S} = (S, \mathcal{I}_{|S})$ the *restriction* of $\mathcal{M}$ on S. Notice that restrictions maintain properties (I1), (I2) if they were satisfied already in $\mathcal{M}$.

Here are some examples of common matroids that can provide some intuition behind the definitions. The reader is invited to check those structures indeed satisfy axioms (I1), (I2).

[10]Throughout this thesis, we use the shorthand notation $B + x$ (resp. $B - x$) to mean $B \cup \{x\}$ (resp. $B \backslash \{x\}$).

Uniform matroids When $\mathcal{I} = \{S \subseteq E : |S| \leq k\}$ for some given positive integer $k \leq |E|$, $(E, \mathcal{I})$ is called a *uniform (rank-k)* matroid.

Graphic matroids Given a multi-graph $G = (V, E)$ (a graph which possibly contains parallel edges and self-loops) let $\mathcal{I}$ include all and only those subsets of edges which do *not* form a cycle, i.e. the subgraph $G[S] = (V, S)$ is a forest. Then $(E, \mathcal{I})$ forms a matroid called a *graphic* matroid. Graphic matroids capture many of the properties of general matroids and notions like bases and circuits have their graphic counterparts of spanning trees and cycles respectively.

Transversal matroids Let $G = (A \cup B, E)$ be a simple, bipartite graph. Define $\mathcal{I}$ to include all and only those subsets $S \subseteq A$ of vertices for which the induced subgraph on $S \cup B$ contains a matching covering all vertices in S. Then $(A, \mathcal{I})$ is called a *transversal* matroid.

If $\mathcal{M} = (E, \mathcal{I})$ is equipped with a weight function $w : E \to \mathbb{R}^+$ it is called a *weighted* matroid. In what follows we're going to assume without loss of generality that the function w is one-to-one, meaning that no two elements have the same weight. All proofs can be adapted to work in the general case using any deterministic tie-breaking rule. The problem of finding an independent set of maximum sum of weights is central to the study of matroids. A very simple greedy algorithm is guaranteed to find the optimal solution and in fact matroids are exactly the independence systems for which that greedy algorithm is always guaranteed to find the optimal solution.

Lemma 1 ([79, Lemma 1.8.3.])**.** *Let $\mathcal{M} = (E, \mathcal{I})$ be a weighted downward-closed set system. Then* Greedy *is guaranteed to return an independent set of maximum total weight for every weight function $w : E \to \mathbb{R}^+$ if and only $\mathcal{M}$ is a matroid.*

Algorithm 1: GREEDY

Input : Weighted matroid $\mathcal{M} = (E, \mathcal{I}, w)$
Output: $I \in \operatorname{argmax}_{I \in \mathcal{I}} \sum_{i \in I} w(e_i)$
Sort the elements of E in non-increasing order by weight and relabel them s.t. $w(e_1) \geq w(e_2) \geq \ldots \geq w(e_n)$.
$I \leftarrow \emptyset$. `// Initialize empty solution.`
for $i = 1, 2, \ldots, n$ **do**
  if $I \cup \{e_i\} \in \mathcal{I}$ **then** `// If appending` e_i `maintains independence,`
    $I \leftarrow I \cup \{e_i\}$ `// add` e_i `to current solution.`
  end
end
return I

The following proposition provides a convenient way for updating the solution to an optimization problem under matroid constraints when new elements are added. The proof is standard in Matroid Theory but is included in Appendix A.1 for completeness.

Proposition 2. *Let $\mathcal{M} = (E, \mathcal{I})$ be a weighted matroid with weight function $w : E \to \mathbb{R}^+$. Consider the max-weight independent set I of the restricted matroid $\mathcal{M}_{|E-x}$. Then the max-weight independent set I^* of $\mathcal{M}$ can be obtained from I as follows: if $(I + x) \in \mathcal{I}$ then $I^* = I + x$, otherwise, $I^* = (I + x) - y$ where y is the minimum-weight element in the unique[11] circuit C of $I + x$.*

2.4 Mechanism Design

This section provides a brief introduction to Mechanism Design [49, 78] focusing on single-parameter settings with a revenue-maximizing objective. This material becomes relevant in Chapter 5 where we extend the setting presented here to allow for "Byzantine" bidders.

[11] It can be shown that $I + x$ contains a unique circuit C when I is an independent set of matroid and x is any element such that $I + x$ is dependent. See [79, Proposition 1.1.6].

2.4.1 Bayesian, single-parameter environments

A *Bayesian, single-parameter environments* is a standard Mechanism Design setting in which a *seller* (or mechanism designer) holds many *identical copies* of an item they want to sell. Specifically, in such a setting, a set of n bidders (or players), numbered 1 through n, participate in the auction and each bidder i has a private, non-negative value $v_i \sim \mathcal{F}_i$, sampled (independently across bidders) from a distribution $\mathcal{F}_i$[12] known to the seller. The value v_i, sometimes also referred to as the i-th player's "valuation" denotes the maximum amount bidder i is willing to pay to win one item. Let V_i be the support of distribution $\mathcal{F}_i$ and define $V = V_1 \times \ldots \times V_n$. For a valuation vector $\mathbf{v} \in V$, we use the standard notation $\mathbf{v}_{-i} = (v_1, \ldots, v_{i-1}, v_{i+1}, \ldots, v_n)$ to express the vector of valuations of all bidders *except* bidder i. When the index set $[n]$ is partitioned into two sets A, B and we have vectors $\mathbf{v}_A \in \mathbb{R}^A$, $\mathbf{w}_B \in \mathbb{R}^B$, we will abuse notation and let $(\mathbf{v}_A, \mathbf{w}_B)$ denote the vector obtained by interleaving $\mathbf{v}_A$ and $\mathbf{w}_B$, i.e. $(\mathbf{v}_A, \mathbf{w}_B)$ is the vector $\mathbf{u} \in \mathbb{R}^n$ such that $u_i = v_i$ for $i \in A$ and $u_i = w_i$ for $i \in B$. Similarly, when $\mathbf{v} \in V$, $i \in [n]$, and $z \in \mathbb{R}$, $(z, \mathbf{v}_{-i})$ will denote the vector obtained by replacing the i^{th} component of $\mathbf{v}$ with z.

A *feasibility constraint* $\mathcal{I} \subseteq 2^{[n]}$ defines all subsets of bidders that can be simultaneously declared winners of the auction. We will interchangeably denote elements of $\mathcal{I}$ both as subsets of $[n]$ and as vectors in $\{0,1\}^n$. Of special interest are feasibility constraints which define the independent sets of a *matroid*. We will sometimes use the phrase *matroid market* to indicate this fact. Matroid markets model many real world applications. For example when selling k identical copies of an item, the market is a uniform rank-k matroid. Another example is kidney exchange markets which can be modeled using transversal matroids ([84]).

[12] In line with the notation used earlier, we'll use F_i to denote the cumulative distribution function and f_i to denote the probability density function of the respective distribution $\mathcal{F}_i$.

In a *sealed-bid auction*, each bidder i submits a *bid* $b_i \in V_i$ simultaneously to the mechanism. Formally, a *mechanism* $\mathcal{A}$ is a pair (x, p) of an allocation rule $x : V \to \mathcal{I}$ accepting the bids and choosing a feasible outcome and a *payment rule* $p : V \to \mathbb{R}^n$ assigning each bidder a monetary payment they need to make to the mechanism. We denote by $x_i(\mathbf{b})$ (or just x_i when clear from the context) the i-th component of the 0-1 vector $x(\mathbf{b})$ and similarly for p. An allocation rule is called *monotone* if the function $x_i(z, \mathbf{b}_{-i})$ is monotone non-decreasing in z for any vector $\mathbf{b}_{-i} \in V_{-i}$ and any bidder i.

We assume bidders have *quasilinear utilities* meaning that bidder's i utility for winning the auction and having to pay a price p_i is $u_i = v_i - p_i$ and 0 if they do not win and pay nothing. Bidders are selfish agents aiming to maximize their own utility.

A mechanism is called *truthful* if bidding $b_i = v_i$ is a *dominant strategy* for each bidder, i.e. no bidder can increase their utility by reporting $b_i \neq v_i$ regardless of the values and bids of the other bidders. An allocation rule x is called *implementable* if there exists a payment rule p such that (x, p) is truthful. Such mechanisms are well understood and easy to reason about since we can predict how the bidders are going to behave. In what follows we focus our attention only on truthful mechanisms and thus use the terms value and bid interchangeably.

A well known result of Myerson ([76]) states that a given allocation rule x is implementable if and only if x is monotone. In case x is monotone, Myerson gives an explicit formula for the unique[13] payment rule such that (x, p) is truthful. In the single-parameter setting we're studying, the payment rule can be informally described as follows: p_i is equal to the minimum b_i that bidder i has to report such that they are included in the set of winners — we'll refer to such a b_i as the *critical bid* of bidder i.

[13] Up to the normalizing assumption that $p_i = 0$ whenever $b_i = 0$.

2.4.2 Revenue-maximizing Auctions

The mechanism designer, who is collecting all the payments, commonly aims to maximize her *expected revenue* which for a mechanism $\mathcal{A}$ is defined as $\text{Rev}(\mathcal{A}) = \mathbb{E}_{b_i \sim \mathcal{F}_i}\left[\sum_{i \in [n]} p_i\right]$.

Lemma 2 ([76])**.** *For any truthful mechanism* (x, p) *and any bidder* $i \in [n]$*:*

$$\mathbb{E}[p_i] = \mathbb{E}[\phi_i(b_i) \cdot x_i(b_i, \mathbf{b}_{-\mathbf{i}})]$$

where the expectations are taken over $b_1, \ldots, b_n \sim \mathcal{F}_1, \ldots, \mathcal{F}_n$*, the function* $\phi_i(\cdot)$ *is defined as*

$$\phi_i(z) = z - \frac{1 - F_i(z)}{f_i(z)}$$

and $\phi_i(b_i)$ *is called the* virtual value *of bidder i.*

The importance of this lemma is that it reduces the problem of revenue maximization to that of virtual welfare maximization. More specifically, consider a sequence of distributions $\mathcal{F}_1, \ldots, \mathcal{F}_n$ which have the property that all ϕ_i are monotone non-decreasing (such distributions are called *regular*). In this case, the allocation rule that chooses a set of bidders with the maximum total virtual value (subject to feasibility constraints) is monotone (a consequence of the regularity condition) and thus implementable. We'll frequently denote this revenue-maximizing mechanism (Algorithm 2) by MyerOPT.

Remark 2. The "critical bid" mentioned in Algorithm 2 can be more formally defined as follows: consider a bidder $i \in [n]$ and fix a bid profile $\mathbf{b}$. Let $\theta_i(\mathbf{b}_{-i})$ be the minimum

virtual value which would allow bidder i to be included in the optimal solution S^*; in other words,

$$\theta_i(\mathbf{b}_{-i}) = \inf\left\{ x \geq 0 \;\middle|\; \exists S^* \in \operatorname{argmax}_{\substack{S \subseteq [n], \\ S \in \mathcal{I}}} \left\{ x \cdot \mathbb{I}[i \in S] + \sum_{j \in S \setminus \{i\}} \phi_j(b_j) \right\} \text{ s.t. } i \in S^* \right\}.$$

This quantity would have been the price charged to bidder i if bidders were reporting virtual values directly. However, bidders are reporting quantities from their valuation space, so we need to apply the inverse of the virtual valuation transformation to get the price that bidder i needs to pay. This price is

$$p_i(\mathbf{b}) = \phi_i^{-1}(\theta_i(\mathbf{b}_{-i})).$$

When $\theta_i(\mathbf{b}_{-i})$ is not in the image of the function $\phi_i(\cdot)$, we define the inverse as follows

$$\phi_i^{-1}(z) = \inf\{v \in \mathcal{S}_i \mid \phi_i(v) \geq z\},$$

where $\mathcal{S}_i$ is the support of $\mathcal{F}_i$, bidder i's valuation distribution.

Algorithm 2: MyerOPT

Parameters: Matroid $\mathcal{M} = ([n], \mathcal{I})$, Regular distributions $\{\mathcal{F}_i\}_{i \in [n]}$
Collect bid b_i from each bidder $i \in [n]$.
Compute virtual values $\phi_i(b_i)$ and discard from $\mathcal{M}$ all bidders i with $\phi_i(b_i) < 0$.
Solve the optimization problem

$$S^* = \operatorname{argmax}_{S \in \mathcal{I}} \sum_{i \in S} \phi_i(b_i).$$

Allocate the items to S^* and charge each bidder $i \in S^*$ their critical bid.

Handling non-regular distributions is possible using the standard technique of *ironing*. Very briefly, it works as follows. So far, we've been expressing x, p and ϕ as a function of the random vector $\mathbf{v}$. It is convenient to switch to the quantile space and express them as a function of a vector $\mathbf{q} \in [0, 1]^n$ where for a given sample z from $\mathcal{F}_i$ we let $q_i = \Pr_{b_i \sim F_i}[b_i \geq z]$. Another way to think of this is, instead of sampling values, we

sample quantiles q_i distributed uniformly at random in the interval $[0, 1]$ which are then transformed into values $v_i(q_i) = F_i^{-1}(1 - q_i)$[14]. Let $R_i(q_i) = q_i \cdot v_i(q_i)$ and notice that $\phi_i(v_i(q_i)) = \left.\frac{dR_i}{dq}\right|_{q=q_i}$. Now, since $v_i(\cdot)$ is a non-increasing function we have that $\phi_i(\cdot)$ is monotone if and only if R is concave.

Now, suppose that F_i is such that R_i is not concave. One can consider the concave hull $\overline{R_i}$ of R_i which replaces R_i with a straight line in every interval that R_i was not following that concave hull. The corresponding function $\overline{\phi_i}(\cdot) = \frac{d\overline{R}_i}{dq}$ is called the *ironed virtual value function*.

Lemma 3 ([49, Theorem 3.18])**.** *For any monotone allocation rule x and any virtual value function ϕ_i of bidder i, the expected virtual welfare of i is upper bounded by their expected ironed virtual value welfare.*

$$\mathbb{E}\left[\phi_i(v_i(q_i)) \cdot x_i(v_i(q_i), \mathbf{v}_{-i}(\mathbf{q}))\right] \leq \mathbb{E}\left[\overline{\phi_i}(v_i(q_i)) \cdot x_i(v_i(q_i), \mathbf{v}_{-i}(\mathbf{q}))\right]$$

Furthermore, the inequality holds with equality if the allocation rule x is such that for all bidders i, $x_i'(q) = 0$ whenever $\overline{R}_i(q) > R_i(q)$.

As a consequence, consider the monotone allocation rule which allocates to a feasible set of maximum total ironed virtual value. On the intervals where $\overline{R_i}(q) > R_i(q)$, $\overline{R_i}$ is linear as part of the concave hull so the ironed virtual value function, being a derivative of a linear function, is a constant. Therefore, the allocation rule is not affected when q ranges in such an interval.

A crucial property of any (ironed) virtual value function ϕ corresponding to a distribution $\mathcal{F}$ is that $z \geq \phi(z)$ for all z in the support of $\mathcal{F}$. This is obvious for ϕ as defined in Lemma 2. We claim it also holds for ironed virtual value functions: if z lies in an interval

[14] In general, $v_i(q_i) = \min\{v \mid F_i(v) \geq q_i\}$.

where $\overline{\phi} = \phi$ it holds trivially. Otherwise, if $z \in [a, b]$ for some interval where ϕ needed ironing (i.e. $\overline{R}(q) > R(q)$ in the quantile space), we have: $z \geq a \geq \phi(a) = \overline{\phi}(a) = \overline{\phi}(z)$. We've thus proven:

Proposition 3. *Any (possibly non-regular) valuation distribution $\mathcal{F}$ having an ironed virtual value function $\overline{\phi}$ satisfies $z \geq \overline{\phi}(z)$ for any z in the support of $\mathcal{F}$.*

Remark 3. For simplicity from now on we'll use ϕ and $\overline{\phi}$ interchangeably and we will refer to ϕ as virtual value function. The reader should keep in mind that if the associated distribution is non-regular, then *ironed* virtual value functions should be used instead.

2.5 Matchings and Flows

Two of the most well-studied problems in Combinatorial Optimization are the *Maximum (Bipartite) Matching* and the *Maximum Flow* problems [95]. More generally, the frameworks of graph matchings and network flows are capable of accurately modeling a plethora of real-world problems from many fields including Computer Science, Operations Research, Economics and more. In Chapter 6 we study an online version of the Maximum Flow problem with recourse[15]. Here we give the basic definitions and background needed for the reader to follow along.

2.5.1 Bipartite Matching Problems

Bipartite Graphs An (undirected) graph G is *bipartite* if its vertices can be partitioned into two sets L, R — sometimes referred to as left and right side respectively — such that no edges exist between vertices of the same side. To specify the two sides explicitly, we

[15] See Section 2.1 for a definition of Online Algorithms with Recourse.

denote the graph as $G = (L \cup R, E)$. Motivated by applications in Computer Networks, we'll sometimes refer to the vertices of the left side as *clients* and the vertices of the right side as *servers*. We'll also sometimes denote the number of clients by $n_c = |L|$ and the total number of vertices by $n = |L| + |R|$.

Matchings A *matching* M is a subset of the edges of G such that no two edges in M share a vertex. The cardinality of the matching is the number of edges in M. The classic *Maximum Bipartite Matching* problem is that of computing a maximum cardinality matching for a given bipartite graph $G = (L \cup R, E)$. In this context, we'll call a graph *matching-admissible* if a matching of size $|L|$ exists.

Online Bipartite Matching with Replacements We consider the following *online* variant of the bipartite matching problem in the context of online algorithms with recourse: the vertices of the right side (servers) are fixed and their identity is known ahead of time. The rest of the graph (including all edges) are unknown to the algorithm at the start. The vertices of the left side (clients) arrive one per time step along with the edges incident to them. In the language of Section 2.1, each vertex arrival is a request. The goal is to come up with an online algorithm $\mathcal{A}$ which maintains for each time-step τ a perfect matching M_τ between the set of clients that have arrived so far and a subset of the servers, hence an action involves matching the newly arrived client with a free server (at a cost of zero) after perhaps a series of modifications which involve switching the match of previously arrived clients for a cost of 1 per switch. The goal is to minimize the total *vertex-switching cost* (denoted by $\mathrm{VSC}^G(\mathcal{A})$) which is defined as the total number of times a client gets reassigned to a different server between consecutive time-steps.

Augmenting Paths in Bipartite Graphs Given a bipartite graph $G = (L \cup R, E)$ and a matching M of G, an *alternating path* is a sequence of edges which form a path in G such that those edges are alternating between not belonging to the matching and belonging to it. A vertex is *free* under a matching M if it's not an endpoint of any edges in M. An *augmenting path* is an alternating path from a free client to a free server. Augmenting paths are a central concept in the design of algorithms for bipartite matching problems both in the offline and online setting.

Bipartite-SAP Algorithm A natural algorithm for solving the Online Bipartite Matching with Replacements problem is the *Bipartite Shortest Augmenting Path* (Bipartite-SAP) Algorithm: when a client $l \in L$ arrives, update the matching by switching the state (membership in the matching) of every edge along a shortest augmenting path from l to a free server (ties broken arbitrarily). Notice that augmenting paths in bipartite graphs are of odd length and an augmenting path of length $2k + 1$ contributes exactly k to the switching cost. Consequently, the total vertex-switching cost is at most half the total length of the augmenting paths used during the run of the algorithm and thus in the analysis of the algorithm it suffices to bound the total length of all augmenting paths as a proxy to bounding the vertex-switching cost.

Worst-case bounds for the switching cost of the Bipartite-SAP algorithm for online bipartite matching were derived in many prior works [12, 17, 18, 44]. Obtaining a tight analysis for this algorithm in general bipartite graphs is still an important open problem. Letting n_c denote the number of dynamic vertices (clients) in the graph, it was shown in [44] that any algorithm must incur a total switching cost of at least $\Omega(n_c \log n_c)$ in the worst case, even when the graph is a path. A matching upper bound of $O(n_c \log n_c)$ is known for trees [18], for bipartite graphs whose dynamic vertices have maximum degree 2 [44], and for general bipartite graphs whose dynamic vertices arrive in random

order [27]. For general bipartite graphs the best known upper bound is $O(n_c \log^2 n_c)$, obtained in a breakthrough result by Bernstein et al. [12].

Interestingly, the techniques of [12] bound the vertex-switching cost of Bipartite-SAP at each time-step τ independently of which algorithm was used to pick a matching on previous time-steps. This allows us to generalize their result in the following way which is going to be useful in Chapter 6. We defer the proof to the Appendix C.

Theorem 2 (Generalization of [12, Theorem 1, Lemma 6])**.** *Let $G = (L \cup R, E)$ be a matching-admissible bipartite graph and let M_0 be a matching covering every vertex of some set $L_0 \subseteq L$. Suppose we run Bipartite-SAP on G with an initial matching M_0 and the vertices in $L \backslash L_0$ arriving in an online fashion one at a time. The total vertex-switching cost of the algorithm is at most $O(n_c \log^2 n_c)$ where $n_c = |L|$.*

2.5.2 Network Flows

Networks A (directed) *network* is a directed graph $G = (V, E)$ with two special disjoint sets of vertices $S, T \subseteq V$ called *sources* and *sinks* respectively along with a capacity function $c : (V \cup E) \to \mathbb{R}_+$ on edges **and** vertices[16]. Vertices in $V \backslash (S \cup T)$ will sometimes be referred to as *internal*. The sets V, E will sometimes be denoted as $V(G), E(G)$. In this chapter we'll only consider *simple* networks which contain no parallel edges or self-loops.

Note that *undirected networks* can also be modeled in this framework under the following transformation: replace each undirected edge with two directed edges of the same capacity, one for each direction. Then delete all incoming edges to the sources and all

[16]Networks are usually defined to have capacities only on edges. The model presented here can be easily reduced to the standard one by replacing each vertex with two vertices connected by a directed edge of capacity equal to the original capacity of the vertex.

outgoing edges from the sinks.

Flows A *flow* on a network G is a function $f : E \to \mathbb{R}_{\geq 0}$. Given f, we define $f_{\text{in}}, f_{\text{out}} : V \to \mathbb{R}_{\geq 0}$ as $f_{\text{in}}(v) = \sum_{u:(u,v)\in E} f(u,v)$ and $f_{\text{out}}(v) = \sum_{u:(v,u)\in E} f(v,u)$. Abusing notation we'll denote by $f(v)$ for a vertex v the net flow imbalance at v, $f(v) = f_{\text{in}}(v) - f_{\text{out}}(v)$. We call f a *valid flow* if:

1. $0 \leq f(e) \leq c(e)$ for every edge $e \in E$.
2. $f_{\text{in}}(v) \leq c(v)$ and $f_{\text{out}}(v) \leq c(v)$ for every vertex $v \in V$.
3. $f(v) = 0$ for all $v \in V \backslash \{S, T\}$
4. $-1 \leq f(v) \leq 0$ for $v \in S$ and $0 \leq f(v) \leq 1$ for $v \in T$.

The *value* of a flow f is defined as $|f| = \sum_{v\in T} f(v)$. We'll call a network G *flow-admissible* if there exists a valid flow of value $|S|$, i.e. where every source sends one unit of flow. We'll denote by G_f the *residual graph* of G based on flow f defined the standard way . A vertex $v \in S \cup T$ is free if $f(v) = 0$. An augmenting path is a path in G_f from a free client to a free server.

A directed network $G = (V, E)$ for which $c(x) = 1$ for every $x \in V \cup E$ will be called *unit-vertex-capacitated*. For those networks, we'll make the additional assumption that sources have 0 in-degree and sinks have 0 out-degree. The assumption is without loss of generality, in the sense that edges into sources and out of sinks can be deleted from the network without changing the maximum flow value. That is because every flow path that passes through a source (respectively, sink) can be truncated to begin at that source (respectively, end at that sink) without changing the flow value.

CHAPTER 3

CONSTRAINED-ORDER PROPHET INEQUALITIES

When one is planning under uncertainty, the order in which decisions must be made can powerfully influence the quality of the eventual outcome. It is therefore advantageous for a decision maker to be able to control the order of the decisions they will face. For example, a new Ph.D. student might prefer to know if they will be offered a position in the research group of their top-choice advisor before having to decide whether to accept a position with their second-choice advisor, but they might not have the freedom to dictate the order in which those two decisions are made.

As discussed in Chapter 2, Prophet Inequalities, and more generally Optimal Stopping Theory furnishes a theoretical basis for quantifying the benefit of exercising control over the order of one's decisions in uncertain environments. Recall that in the standard, adversarial order prophet problem there is a sequence of independent random variables $X_1, \ldots, X_n$ with known distributions, and one aims to design and analyze an online algorithm (or stopping rule) ALG that maximizes the expected value of the sample selected, namely $\mathbb{E}[\text{ALG}]$[1]. This problem models a setting where the decision maker has no control over the order of inspection. On the other end of the spectrum, the free-order prophet problem (see Problem 3) posits that the decision-maker has total control over the order in which to observe the random variables *and* the time at which to stop the permuted sequence.

For the adversarial order prophet problem, we've seen how to achieve a competitive ratio of $1/2$ against an omniscient "prophet" and further, $1/2$ is the best possible constant. In fact, the optimal ratio $1/2$ remains attainable even if the online algorithm is constrained

[1] As a reminder, we are abusing notation and use ALG both to refer to the algorithm and as a random variable indicating the value selected by the algorithm.

to using a *single-threshold stopping rule*, which sets a fixed threshold T and commits to reject every sample less than T and to stop whenever it encounters a sample strictly greater than T. With order selection, an online algorithm can better compete against the prophet and higher competitive ratios are attainable. At the time of writing, the best competitive ratio known to be achievable by an online algorithm for the free-order prophet problem is $0.725\ldots$ due to [80]. When constrained to single-threshold stopping rules, the best achievable ratio is $1 - 1/e = 0.632\ldots$.

To summarize the foregoing discussion, the gap between the competitive ratios attainable with or without order selection formalizes, and quantifies, the advantage that a decision maker gains by being able to control the order in which decisions are made under uncertainty. But how much control over the ordering is needed to gain this advantage? This chapter initiates an in-depth exploration of that question. We introduce *constrained-order prophet inequalities*[2], in which there is a predefined subset $\Pi \subseteq S_n$ of the set of all permutations of $[n] := \{1, 2, \ldots, n\}$, and the decision-maker is allowed to reorder the random variables $X_1, \ldots, X_n$ using any permutation in Π before running a stopping rule.

Problem 5 (Constrained-order Prophet Problem)**.** Given a positive integer n and a non-empty set $\Pi \subseteq S_n$ of permutations on n elements, the constrained-order prophet problem on instances of size n parameterized by the set of allowable permutations Π is set up as follows. The **adversary** chooses an *instance* $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$. **Nature** samples the value of each X_i from the respective distribution $\mathcal{F}_i$. The **decision-maker** receives the $\mathcal{I}$, decides on a $\pi \in \Pi$ among the allowed permutations and a *stopping rule* $\mathrm{ALG}_{\mathcal{I},\pi}$ adapted to $\mathcal{I}, \pi$.

The concept of competitive ratio is adapted as follows.

[2]The work of [15] introducing order constraints in the Pandora's box problem partly served as inspiration for the authors to start exploring this type of prophet inequalities.

Definition 9 (Constrained-order Competitive Ratio)**.** Fix a positive integer n and a non-empty set $\Pi \subseteq S_n$ of permutations on n elements. The *competitive ratio* of the *constrained-order prophet problem* parameterized by Π (Problem 5) over instances of size n is defined as,

$$\mathcal{R}_n(\Pi) = \inf_{\mathcal{I}=(\mathcal{F}_i)_{i\in[n]}} \frac{\sup_{\pi\in\Pi, \mathrm{ALG}_{\mathcal{I},\pi}} \mathbb{E}_{\mathcal{I}}[\mathrm{ALG}_{\mathcal{I},\pi}]}{\mathbb{E}_{\mathcal{I}}[\mathrm{OPTO}\textsc{ff}_{\mathcal{I}}]}.$$

The *single-threshold competitive ratio* $\overline{\mathcal{R}}_n(\Pi)$ for instances of size n is similarly defined with the supremum being over all single-threshold stopping rules.

In the extreme cases where Π has only one element or Π is the entire permutation group S_n, one recovers the definitions of the adversarial-order prophet problem and free-order prophet problem, respectively, defined in Section 2.2. Constrained-order prophet inequalities interpolate between these two extremes and allow us to gain deeper insight into how and why optimizing the order of decisions leads to better outcomes for optimal stopping rules. Interestingly, our first main result shows that much of the benefit of order selection can be gained by choosing the better of just *two* permutations, corresponding to the forward and reverse orderings.

Theorem 3. *Let* $\Pi = \{\iota, \rho\}$*, where* ι *and* ρ *are the permutations of* $[n]$ *that place its elements in forward and reverse order, respectively. The single-threshold competitive ratio of* Π *is* $\overline{\mathcal{R}}_n(\Pi) = \varphi^{-1} = \frac{1}{2}\left(\sqrt{5}-1\right) = 0.618\ldots$.

One may interpret this result as lending support to the intuition that most of the benefit of order selection stems from the ability to schedule a high-risk high-reward option (e.g. a value that is $1/\varepsilon$ with probability ε, else zero) in the first half of the decision sequence, leaving safer options for afterward.

Given the large quantitative gain in $\overline{\mathcal{R}}_n(\Pi)$ when Π is enlarged to contain the reverse ordering of the input sequence, one might expect to extract additional significant gains

when Π is allowed to contain three permutations, or an even larger constant number of them. Surprisingly, we show this is not the case: to exceed the golden-ratio bound at all one needs a super-constant number of permutations, and to exceed it by any constant $\varepsilon > 0$ one needs a logarithmic number of them.

Theorem 4. *If $n \geq 3$ and $|\Pi| < \sqrt{\log n}$ then $\overline{\mathcal{R}}_n(\Pi) \leq \varphi^{-1}$. For any $\varepsilon > 0$, if $|\Pi| < \log_{1/\varepsilon}(n)$ then $\overline{\mathcal{R}}_n(\Pi) \leq \varphi^{-1} + O(\varepsilon)$.*

Recall that if the algorithm is allowed to order the elements arbitrarily, the best possible single-threshold competitive ratio is $\overline{\mathcal{R}}_n(S_n) = 1 - \frac{1}{e} + o(1)$. Our third and final main result shows that a logarithmic number of permutations are sufficient to get within ε of this bound, and that a quadratic number of permutations suffice to match the $1 - \frac{1}{e}$ bound exactly.

Theorem 5. *For every $n \in \mathbb{N}$ and $\varepsilon > 0$, there is a set Π consisting of $O(\mathrm{poly}(\varepsilon^{-1}) \cdot \log n)$ permutations such that $\overline{\mathcal{R}}_n(\Pi) > 1 - \frac{1}{e} - \varepsilon$. There is also a set Π consisting of $O(n^2)$ permutations such that $\overline{\mathcal{R}}_n(\Pi) \geq 1 - \frac{1}{e}$.*

Taken together, our results give a nearly complete answer to the question: for a given α, what is the smallest m such that there exists an m-element set of permutations whose threshold competitive ratio is at least α? The answer is:

1. $m = 1$ for $\alpha \leq \frac{1}{2}$ [62, 63];
2. $m = 2$ for $\frac{1}{2} < \alpha \leq \varphi^{-1}$ (Theorems 3 and 4);
3. $m = \Theta(\log n)$ for $\varphi^{-1} < \alpha < 1 - \frac{1}{e}$, with the constant inside the $\Theta(\cdot)$ depending on the value of α (Theorem 5);
4. $m = O(n^2)$ for $\alpha = 1 - \frac{1}{e}$ (Theorem 5);

5. there is no set of permutations whose single-threshold competitive ratio is greater than $1 - \frac{1}{e}$; see also Proposition 4 in the Appendix.

It would be desirable, of course, to gain a similarly comprehensive understanding of the smallest set of permutations needed to achieve a given competitive ratio (rather than single-threshold competitive ratio), but at present this seems out of reach: even determining the competitive ratio of the full permutation group, $\mathcal{R}_n(S_n)$, is a major open problem as it amounts to determining the best possible constant in the free order prophet inequality. For now, the most we can say is that our results imply lower bounds on the competitive ratio $\mathcal{R}_n(\Pi)$ for the permutation sets Π we study, due to the trivial observation that $\mathcal{R}_n(\Pi) \geq \overline{\mathcal{R}}_n(\Pi)$ for every set Π.

In summary, then, the message of this chapter is as follows: an online algorithm solving an the prophet problem using a single-threshold stopping rule can achieve significant gains if they are allowed to control the order in which they observe the values in the sequence, but most of these gains can be achieved just by choosing the better of the forward or reverse ordering, and nearly all of the gains can be achieved by choosing among a logarithmic number of predefined permutations.

Lastly, although we have motivated and presented the results of this chapter in terms of free-order and constrained-order prophet inequalities, i.e. a setting in which the decision-maker chooses the order in which values are observed, our results also have a bearing on the prophet secretary problem (see Section 2.2) due to their method of proof. The constrained-order prophet inequalities asserted in Theorems 3 to 5 are all proven by constructing a small set of permutations, Π, and analyzing the performance of a single-threshold stopping rule when the order in which values are observed is drawn uniformly at random from Π. Thus, all of our results can also be interpreted as constructing "pseudo-random" distributions over permutations that have small support size, but

ensure a single-threshold competitive ratio that is nearly as good as what can be achieved in the prophet secretary problem when the order of observation is sampled uniformly at random.

3.1 Using the forward and reverse permutations

In this section, we let $\Pi = \{\iota, \rho\}$ where ι is the identity permutation mapping each $i \in [n]$ to $\iota(i) = i$ and ρ is the *reverse* permutation mapping $i \in [n]$ to $\rho(i) = n - i + 1$. We prove that $\overline{\mathcal{R}}_n(\Pi) = \varphi^{-1}$.

Theorem 6. *For every n-tuple of independent random variables $X_1, \ldots, X_n$, there exists a threshold T such that*

$$\mathbb{E}_\pi \left[\mathbb{E}[\mathrm{ALG}_{\pi,T}]\right] \geq \varphi^{-1} \cdot \mathbb{E}[\mathrm{OPT}], \tag{3.1}$$

where the outer expectation on the left side is over a uniformly random choice of $\pi \in \Pi = \{\iota, \rho\}$.

Proof.

For a given threshold T, let $p = \Pr[\max_{i=1}^n X_i \geq T]$. We have

$$\mathbb{E}[\mathrm{OPT}] \leq T + \mathbb{E}[(\mathrm{OPT} - T)^+] \leq T + \sum_{i=1}^n \mathbb{E}[(X_i - T)^+]. \tag{3.2}$$

For the random variable $\mathrm{ALG} = \mathrm{ALG}_{\pi,T}$, we have

$$\mathbb{E}[\mathrm{ALG}] = p \cdot T + \mathbb{E}[(\mathrm{ALG} - T)^+] = p \cdot T + \sum_{i=1}^n c_i \cdot \mathbb{E}[(X_i - T)^+] \tag{3.3}$$

where

$$c_i = \frac{1}{2}\left(\prod_{j=1}^{i-1} \Pr[X_j < T] + \prod_{j=i+1}^{n} \Pr[X_j < T]\right) \tag{3.4}$$

denotes the probability that no element is selected before the stopping rule observes X_i, given that the sequence is observed in forward or reverse order with equal probability. Letting

$$a_i = \prod_{j=1}^{i-1} \Pr[X_j < T], \qquad b_i = \prod_{j=i+1}^{n} \Pr[X_j < T],$$

we have

$$a_i b_i \geq \prod_{j=1}^{n} \Pr[X_j < T] = 1 - p \tag{3.5}$$

$$c_i = \frac{1}{2}(a_i + b_i) \geq (a_i b_i)^{1/2} \geq \sqrt{1-p}, \tag{3.6}$$

where the second line follows from the arithmetic-geometric mean inequality. Letting $q = \sqrt{1-p}$ and substituting $c_i \geq q$ and $p = 1 - q^2$ back into Eq. (3.3) we obtain

$$\mathbb{E}[\mathrm{ALG}] \geq (1-q^2)T + q \sum_{i=1}^{n} \mathbb{E}[(X_i - T)^+]. \tag{3.7}$$

Choosing T so that $q = \varphi^{-1}$, which implies also $1 - q^2 = q = \varphi^{-1}$, we find that

$$\mathbb{E}[\mathrm{ALG}] \geq \varphi^{-1} \cdot \left(T + \sum_{i=1}^{n} \mathbb{E}[(X_i - T)^+] \right) \geq \varphi^{-1} \cdot \mathbb{E}[\mathrm{OPT}], \tag{3.8}$$

where the second inequality follows from (3.2). □

Next we prove that the golden-ratio bound in Theorem 6 is the best possible.

Lemma 4. *For all $\varepsilon > 0$ there exists a sequence of three independent random variables X_1, X_2, X_3 such that X_1 and X_3 are identically distributed, and for every single-threshold stopping rule* ALG *we have*

$$\mathbb{E}[\mathrm{ALG}] < (\varphi^{-1} + \varepsilon) \cdot \mathbb{E}[\mathrm{OPT}]. \tag{3.9}$$

Proof. Fix a parameter $\delta > 0$ to be determined later, and suppose X_1 and X_3 are uniformly distributed in $[1-\delta, 1]$ while X_2 is distributed as follows:

$$X_2 \sim \begin{cases} \dfrac{\sqrt{5}-1}{\delta}, & \text{w.p. } \delta \\ 0, & \text{otherwise} \end{cases}$$

Then

$$\mathbb{E}[\mathrm{OPT}] > \delta \cdot \frac{\sqrt{5}-1}{\delta} + (1-\delta)\cdot(1-\delta) > \sqrt{5} - 2\delta \tag{3.10}$$

If ALG is a single-threshold stopping rule with threshold T there are four cases to consider. When $T > (\sqrt{5}-1)/\delta$, no element is ever accepted and $\mathbb{E}[\mathrm{ALG}] = 0$. When $1 < T \le (\sqrt{5}-1)/\delta$ we have $\mathbb{E}[\mathrm{ALG}] = \sqrt{5}-1$. On the other end of the spectrum, when $T < 1-\delta$, we have $\mathbb{E}[\mathrm{ALG}] = 1 - \frac{\delta}{2}$. In all of these cases, $\mathbb{E}[\mathrm{ALG}] < (\varphi^{-1}+\varepsilon)\cdot\mathbb{E}[\mathrm{OPT}]$ provided δ is sufficiently small. The remaining case is when $1-\delta \le T \le 1$. In this case, let $r = (1-T)/\delta$, so that $\Pr[X_1 > T] = r$. Then

$$\begin{aligned}
\mathbb{E}[\mathrm{ALG}] &\le r\cdot 1 + (1-r)\delta\cdot\frac{\sqrt{5}-1}{\delta} + (1-r)(1-\delta)r\cdot 1 \\
&\le r + (1-r)(\sqrt{5}-1) + (r-r^2) \\
&= (\sqrt{5}-1) + (3-\sqrt{5})r - r^2.
\end{aligned} \tag{3.11}$$

The right side of (3.11) is maximized when $r = \frac{1}{2}(3-\sqrt{5})$, when it equals $(\sqrt{5}-1) + \frac{1}{4}(14-6\sqrt{5}) = \frac{1}{2}(5-\sqrt{5})$. Hence,

$$\mathbb{E}[\mathrm{ALG}] \le \frac{1}{2}(5-\sqrt{5}) = \varphi^{-1}\cdot\sqrt{5}. \tag{3.12}$$

Combining (3.10) with (3.12) we see that the conclusion of the lemma holds, as long as δ is chosen small enough that $(\varphi^{-1}+\varepsilon)\cdot(\sqrt{5}-2\delta) \ge \varphi^{-1}\cdot\sqrt{5}$. □

Corollary 1. *Suppose* Π *is a non-empty set of permutations of* $[n]$ *and* $i, j, k \in [n]$ *are three distinct indices such that* $\pi^{-1}(j)$ *is between* $\pi^{-1}(i)$ *and* $\pi^{-1}(k)$ *for all* $\pi \in \Pi$. *Then* $\overline{\mathcal{R}}_n(\Pi) \le \varphi^{-1}$.

Proof. Define an n-tuple of independent random variables $X_1, \ldots, X_n$ by specifying that the distributions of X_i, X_j, X_k are identical to the distributions of X_1, X_2, X_3 specified in Lemma 4 above, and for $\ell \notin \{i, j, k\}$ let the distribution of X_ℓ be identically zero. For any $\pi \in \Pi$, a single-threshold stopping rule adapted to π with threshold $T > 0$ will skip X_ℓ for

every $\ell \notin \{i, j, k\}$, and it will observe the values X_i, X_j, X_k at times $\pi^{-1}(i), \pi^{-1}(j), \pi^{-1}(k)$. Note that the distributions of these three variables and the order in which they are observed are identical to the distributions X_1, X_2, X_3 specified in Lemma 4, so the inequality (3.9) will be satisfied. As ALG is an arbitrary single-threshold stopping rule, and $\varepsilon > 0$ is arbitrarily small, we conclude that Π does not satisfy a constrained-order prophet inequality with factor α for any $\alpha > \varphi^{-1}$. □

Theorem 7. *For the permutation set $\Pi = \{\iota, \rho\}$, if $n \geq 3$, we have $\overline{\mathcal{R}}_n(\Pi) = \varphi^{-1}$.*

Proof. A threshold stopping rule that is allowed to select the better of ι and ρ can do no worse than one which samples one of the two permutations uniformly at random. Therefore, Theorem 6 implies $\overline{\mathcal{R}}_n(\Pi) \geq \varphi^{-1}$. On the other hand, any three distinct indices $i < j < k$ in $[n]$ satisfy the condition in Corollary 1, implying that $\overline{\mathcal{R}}_n(\Pi) \leq \varphi^{-1}$. □

3.2 Beating the golden-ratio bound requires many permutations

In this section we show that $\overline{\mathcal{R}}_n(\Pi) \leq \varphi^{-1}$ whenever $|\Pi| < \sqrt{\log n}$ and that $\overline{\mathcal{R}}_n(\Pi) \leq \varphi^{-1} + O(\varepsilon)$ whenever $\varepsilon > 0$ and $|\Pi| < \log_{1/\varepsilon}(n)$. This was stated as Theorem 4 earlier, and is proven by combining Lemmas 5 and 6 below.

Corollary 1 presented a necessary condition for a permutation set Π to satisfy $\overline{\mathcal{R}}_n(\Pi) > \varphi^{-1}$: for every three indices i, j, k there exists $\pi \in \Pi$ such that $\pi^{-1}(j)$ does not lie between $\pi^{-1}(i)$ and $\pi^{-1}(k)$. It turns out this condition guarantees $|\Pi| > \log\log n$ [58] but it does not imply any stronger lower bound than that doubly-logarithmic one. To prove stronger lower bounds on $|\Pi|$, we begin by generalizing Corollary 1.

Definition 10. If Π is a set of permutations of $[n]$, we say an index $j \in [n]$ is *ε-centered with respect to* Π if there exists a probability distribution p on $[n] \setminus \{j\}$ such that for every

$\pi \in \Pi$ with inverse permutation $\sigma = \pi^{-1}$, the sets $\{i \mid \sigma(i) < \sigma(j)\}$ and $\{i \mid \sigma(i) > \sigma(j)\}$ both have measure greater than $\frac{1}{2} - \varepsilon$ under p. In the special case $\varepsilon = 0$, we shall say that j is *centered with respect to* Π.

Lemma 5. *If* Π *is a non-empty set of permutations of* $[n]$ *and there exists an index* $j \in [n]$ *that is* ε*-centered with respect to* Π*, then* $\overline{\mathcal{R}}_n(\Pi) \leq \varphi^{-1} + O(\varepsilon)$.

The proof, which is deferred to Appendix B.1, generalizes the proof of Corollary 1. The value of X_j is defined to be $(\sqrt{5} - 1)/\delta$ with probability δ, else $X_j = 0$, whereas the remaining distributions are all tightly concentrated around 1. Single-threshold stopping rules face a dilemma: if the threshold is set high enough that there is a reasonable probability of not stopping before X_j, then there must be a reasonable probability that the algorithm also doesn't stop after X_j and comes away empty-handed. The prophet faces no such dilemma: she can pick X_j when it is non-zero, and otherwise she can pick a value that is nearly equal to 1.

Lemma 6. *If* Π *is a set of fewer than* $\sqrt{\log n}$ *permutations of* $[n]$ *then there exists an index* j *that is centered with respect to* Π*. If* Π *is a set of fewer than* $\log_{1/\varepsilon}(n)$ *permutations of* $[n]$ *for some* $\varepsilon > 0$ *then there exists an index* j *that is* ε*-centered with respect to* Π.

Proof. Let $\sigma_1, \sigma_2, \ldots, \sigma_m$ be an enumeration of the permutations whose inverse belongs to Π. For any pair of distinct indices $i \neq j$ in $[n]$ define a vector $\mathbf{v}_{ij} \in \{\pm 1\}^m$ by specifying that for all $k \in [m]$

$$(\mathbf{v}_{ij})_k = \begin{cases} -1 & \text{if } \sigma_k(i) < \sigma_k(j) \\ 1 & \text{if } \sigma_k(i) > \sigma_k(j). \end{cases} \tag{3.13}$$

For any probability distribution p on $[n] \setminus j$, if we use $q_k(p)$ to denote the measure of the set $\{i \mid \sigma_k(i) < \sigma_k(j)\}$ under p and $\mathbf{q}(p)$ to denote the vector $(q_1(p), \ldots, q_m(p))$, then we

have

$$\sum_{i \neq j} p(i)\mathbf{v}_{ij} = \mathbf{1} - 2\mathbf{q}(p) \tag{3.14}$$

where $\mathbf{1}$ denotes the vector $(1, 1, \ldots, 1)$. The criterion that j is centered with respect to Π is equivalent to the existence of a distribution p such that $\mathbf{q}(p) = \frac{1}{2} \cdot \mathbf{1}$. Similarly, j is ε-centered with respect to Π if and only if there is a distribution p such that $\mathbf{q}(p) \in \left(\frac{1}{2} - \varepsilon, \frac{1}{2} + \varepsilon\right)^m$. Using (3.14) we now see that *j is centered with respect to* Π *if and only if* $\mathbf{0}$ *is a convex combination of the vectors in the set* $V_j = \{\mathbf{v}_{ij} \mid i \neq j\}$. *Similarly, j is ε-centered with respect to* Π *if and only if the convex hull of* V_j *intersects the hypercube* $(-2\varepsilon, 2\varepsilon)^m$.

To finish proving the first part of the lemma, we must show that if $m < \sqrt{\log n}$, then for some index j, $\mathbf{0}$ is a convex combination of the vectors in V_j. Contrapositively, we will assume that for every j, $\mathbf{0}$ is not in the convex hull of V_j, and we will deduce from this assumption that $m^2 \geq \log n$. By the separating hyperplane theorem, our assumption that $\mathbf{0}$ is not in the convex hull of V_j implies there is a vector $\mathbf{w}_j$ such that $\langle \mathbf{w}_j, \mathbf{v}_{ij} \rangle > 0$ for all $i \neq j$. Note that $\mathbf{v}_{ij} = -\mathbf{v}_{ji}$, so for all $i \neq j$,

$$\langle \mathbf{w}_j, \mathbf{v}_{ij} \rangle > 0 > \langle \mathbf{w}_i, \mathbf{v}_{ij} \rangle, \tag{3.15}$$

where the second inequality follows because $\langle \mathbf{w}_i, \mathbf{v}_{ji} \rangle > 0$ by our assumption on $\mathbf{w}_i$. Now consider the linear threshold functions defined by $f_i(\mathbf{x}) = \operatorname{sgn}(\langle \mathbf{w}_i, \mathbf{x} \rangle)$ for each $i \in [n]$. Equation (3.15) says that $f_i(\mathbf{v}_{ij}) \neq f_j(\mathbf{v}_{ij})$. Recalling that $\mathbf{v}_{ij} \in \{\pm 1\}^m$, this means the restrictions of $f_1, \ldots, f_n$ to $\{\pm 1\}^m$ are pairwise distinct. There are fewer than 2^{m^2} distinct linear threshold functions on $\{\pm 1\}^m$ [33], hence $m^2 \geq \log n$.

To finish proving the second part of the lemma, we must show that if $m < \log_{1/\varepsilon}(n)$, then for some index j, the convex hull of V_j and the hypercube $(-2\varepsilon, 2\varepsilon)^m$ intersect. Contrapositively, we will assume that for every j, the minimum ∞-norm of the vectors in the convex hull of V_j is at least 2ε. From this assumption we will deduce that $(1/\varepsilon)^m \geq n$.

Minimizing the ∞-norm of vectors in the convex hull of V_j is equivalent to solving the following linear program, whose dual is presented below.

$$\begin{aligned}
\min \quad & r \\
\text{s.t.} \quad & r - \sum_{i \neq j} v_{ij,k} p_i \geq 0 && \forall k \in [m] \\
& r + \sum_{i \neq j} v_{ij,k} p_i \geq 0 && \forall k \in [m] \\
& \sum_{i \neq j} p_i = 1 \\
& p_i \geq 0 && \forall i \in [n] \setminus \{j\}
\end{aligned}$$

$$\begin{aligned}
\max \quad & z \\
\text{s.t.} \quad & z + \sum_{k=1}^{m} v_{ij,k}(y_k - x_k) \leq 0 && \forall i \in [n] \setminus \{j\} \\
& \sum_{k=1}^{m} (y_k + x_k) = 1 \\
& x_k, y_k \geq 0 && \forall k \in [m]
\end{aligned}$$

Our assumption is that for each j, the optimum of the primal LP is at least 2ε, which means that the optimum of the dual LP is also at least 2ε. Let $\mathbf{x}, \mathbf{y}, z$ denote a feasible dual solution with $z \geq 2\varepsilon$, and let $\mathbf{w}_j = \mathbf{x} - \mathbf{y}$. The first dual constraint, combined with the inequality $z \geq 2\varepsilon$, implies $\langle \mathbf{w}_j, \mathbf{v}_{ij} \rangle \geq 2\varepsilon$ for all $i \in [n] \setminus j$. Using the fact that $x_k, y_k \geq 0$ implies $x_k + y_k \geq |x_k - y_k|$, we see that the second dual constraint implies $\|\mathbf{w}_j\|_1 \leq 1$. Thus, there exist vectors $\mathbf{w}_1, \ldots, \mathbf{w}_n$ in the L_1 unit ball, such that for all $i \neq j$ in $[n]$, $\langle \mathbf{w}_j, \mathbf{v}_{ij} \rangle \geq 2\varepsilon$. Recalling that $\mathbf{v}_{ji} = -\mathbf{v}_{ij}$, we may combine the inequalities

$$2\varepsilon \leq \langle \mathbf{w}_j, \mathbf{v}_{ij} \rangle, \qquad 2\varepsilon \leq \langle \mathbf{w}_i, \mathbf{v}_{ji} \rangle$$

to obtain

$$\begin{aligned}
4\varepsilon \;&\leq\; \langle \mathbf{w}_j - \mathbf{w}_i, \mathbf{v}_{ij} \rangle \\
&\leq\; \|\mathbf{w}_j - \mathbf{w}_i\|_1 \cdot \|\mathbf{v}_{ij}\|_\infty \;=\; \|\mathbf{w}_j - \mathbf{w}_i\|_1
\end{aligned} \tag{3.16}$$

Summarizing, the L_1 unit ball contains n vectors $\mathbf{w}_1, \ldots, \mathbf{w}_n$, and the pairwise distances between these vectors (in L_1) are at least 4ε. The L_1 balls of radius 2ε centered at these vectors are pairwise disjoint, and all of them are contained in the ball of radius $1 + \varepsilon$ centered at $\mathbf{0}$, so the combined volume of the n balls of radius 2ε must not exceed the volume of the radius-$(1 + \varepsilon)$ ball. Since $1 + \varepsilon < 2$, this implies $(2\varepsilon)^m \cdot n < 2^m$, hence $n < (1/\varepsilon)^m$ as claimed. □

3.3 Achieving the optimal threshold prophet ratio

Recall that when one is free to order the elements arbitrarily, single-threshold stopping rules satisfy a prophet inequality with factor $1 - \frac{1}{e}$ but (asymptotically) no greater, i.e. $\overline{\mathcal{R}}_n(S_n) = 1 - \frac{1}{e} + o(1)$ (see Appendix A.2.3). In this section we construct a small set of permutations that achieves this bound, and an even smaller set that comes arbitrarily close. To do that, we follow a proof[3] for deriving that bound in the prophet secretary version of the problem[4] and identify weaker properties that a distribution over a set of permutations need to satisfy in order for the proof to (approximately) go through. Those properties are captured by the notions of *pairwise independence* and *almost pairwise independence* which we now define.

Definition 11. A distribution over permutations $\sigma \in S_n$ is *pairwise independent* if for every pair of distinct indices $i \neq j$ in $[n]$, the pair $(\sigma(i), \sigma(j))$ is distributed uniformly over $\{(a, b) \in [n] \times [n] \mid a \neq b\}$. It is (ε, δ)-*almost pairwise independent* if for every pair of distinct indices $i \neq j$, the distribution of the pair $\left(\left\lceil \frac{\sigma(i)}{\varepsilon n} \right\rceil, \left\lceil \frac{\sigma(j)}{\varepsilon n} \right\rceil\right)$ is δ-close, in total variation distance, to the uniform distribution on $\left[\frac{1}{\varepsilon}\right] \times \left[\frac{1}{\varepsilon}\right]$, where $\left[\frac{1}{\varepsilon}\right]$ denotes the set

[3] A slight variation of the proof technique used in Theorem 8 also appears in [32] in which they use Schur Convexity instead of the AM-GM inequality to derive similar bounds for the prophet secretary problem.

[4] Recall, in the prophet secretary problem (Problem 4), the order of inspection is chosen uniformly at random by nature from $\Pi = S_n$.

$\left\{1, 2, \ldots, \left\lceil \frac{1}{\varepsilon} \right\rceil\right\}$.

Notice that the uniform distribution over S_n is indeed pairwise independent. More importantly, as the following lemma shows, there exist distributions with much smaller support that (approximately) achieve the same property.

Lemma 7. *For prime n, there exists a set Π of $n(n-1)$ permutations in S_n such that the uniform distribution over Π is pairwise independent. For any $\varepsilon, \delta > 0$ such that $1/\varepsilon$ is an integer, if n is an integer multiple of $1/\varepsilon$ and $\varepsilon n \geq 2/\delta$, then there exists a set Π of $O((\frac{1}{\delta\varepsilon})^2 \log n)$ permutations in S_n such that the uniform distribution over Π is (ε, δ)-almost pairwise independent.*

The proof is deferred to Appendix B.2. The first part is proven using the permutations $\sigma(k) = ak + b \pmod{n}$ for all $a \in [n-1]$ and $b \in [n]$. The second part is proven using the probabilistic method to show that a random $\Pi \subseteq S_n$ of the given cardinality has positive probability of being (ε, δ)-almost pairwise independent. Explicit constructions using ϵ-biased sets [77, 89] can achieve $|\Pi| = O\left(\left(\frac{1}{\delta\varepsilon}\right)^{2+o(1)} \log n\right)$.

Theorem 8. *Suppose σ is a random permutation of $[n]$ and $\pi = \sigma^{-1}$. If σ is drawn from a pairwise independent distribution then there exists a threshold T such that*

$$\mathbb{E}_\pi\left[\mathbb{E}[\mathrm{ALG}_{\pi,T}]\right] \geq \left(1 - \tfrac{1}{e}\right) \cdot \mathbb{E}[\mathrm{OPT}]. \tag{3.17}$$

If σ is drawn from an $(\varepsilon, \varepsilon^2)$-almost pairwise independent distribution then there exists a threshold T such that

$$\mathbb{E}_\pi\left[\mathbb{E}[\mathrm{ALG}_{\pi,T}]\right] \geq \left(1 - \tfrac{1}{e} - O(\varepsilon)\right) \cdot \mathbb{E}[\mathrm{OPT}]. \tag{3.18}$$

Proof. For a given threshold T, let $p = \Pr[\max_{i=1}^n X_i \geq T]$. We have

$$\mathbb{E}[\mathrm{OPT}] \leq T + \mathbb{E}[(\mathrm{OPT} - T)^+] \leq T + \sum_{i=1}^n \mathbb{E}[(X_i - T)^+]. \tag{3.19}$$

For the random variable $\mathrm{ALG} = \mathrm{ALG}_{\pi,T}$, we have

$$\mathbb{E}[\mathrm{ALG}] = p \cdot T + \mathbb{E}[(\mathrm{ALG} - T)^+] = p \cdot T + \sum_{i=1}^{n} c_i \mathbb{E}[(X_i - T)^+] \tag{3.20}$$

where

$$c_i = \sum_{k=1}^{n} \Pr[\pi(k) = i] \cdot \prod_{\ell=1}^{k-1} \Pr[X_{\pi(\ell)} < T \mid \pi(k) = i] \tag{3.21}$$

denotes the probability that no element is selected before the stopping rule observes X_i. We can bound the product occurring in the formula for c_i using the arithmetic mean-geometric mean inequality, similarly to the proof of Theorem 6. Let $q_j = \Pr[X_j < T]$ for each $j \in [n]$. For any set $S \subseteq [n]$, let $p_{k,i}(S) = \Pr[\pi([k-1]) = S \mid \pi(k) = i]$ denote the conditional probability that S is exactly equal to the set of elements observed before X_i, given that $\pi(k) = i$.

$$\begin{aligned}
\prod_{\ell=1}^{k-1} \Pr[X_{\pi(\ell)} < T \mid \pi(k) = i] &= \sum_{S \subseteq [n]} p_{k,i}(S) \prod_{j \in S} q_j \\
&\geq \prod_{S \subseteq [n]} \left(\prod_{j \in S} q_j \right)^{p_{k,i}(S)} \\
&= \prod_{j \in [n] \setminus \{i\}} q_j^{\sum_{S \subseteq [n], j \in S} p_{k,i}(S)}
\end{aligned} \tag{3.22}$$

The exponent $\sum_{S \subseteq [n], j \in S} p_{k,i}(S)$ on the right side is equal to $\Pr[\sigma(j) < k \mid \sigma(i) = k]$. Hence,

$$c_i \geq \sum_{k \in [n]} \Pr[\pi(k) = i] \cdot \prod_{j \neq i} q_j^{\Pr[\sigma(j) < k \mid \sigma(i) = k]}. \tag{3.23}$$

If σ is pairwise independent then $\Pr[\pi(k) = i] = \frac{1}{n}$ and for all j, $\Pr[\sigma(j) < k \mid \sigma(i) = k] = \dfrac{k-1}{n-1}$. Hence, if we let $q = \prod_{j=1}^{n} q_j$, then

$$\begin{aligned}
c_i &\geq \frac{1}{n} \cdot \sum_{k=1}^{n} \left(\prod_{j \neq i} q_j \right)^{\frac{k-1}{n-1}} \\
&\geq \frac{1}{n} \left(1 + q^{1/(n-1)} + \cdots + q^{(n-2)/(n-1)} + q \right)
\end{aligned} \tag{3.24}$$

If we set T so that $q = 1/e$, then Lemma 28 in Appendix B.2 shows that the right side of (3.24) is greater than $1 - \frac{1}{e}$. Also, note that

$$p = \Pr[\mathrm{OPT} \geq T] = 1 - \Pr[\mathrm{OPT} < T] = 1 - \prod_{j=1}^{n} q_j = 1 - q = 1 - \tfrac{1}{e}.$$

Having shown that $p = 1 - \frac{1}{e}$ and that $c_i > 1 - \frac{1}{e}$ for all i, we may substitute these bounds into (3.20) and conclude that

$$\begin{aligned} \mathbb{E}[\mathrm{ALG}] &\geq \left(1 - \frac{1}{e}\right) T + \left(1 - \frac{1}{e}\right) \sum_{i=1}^{n} \mathbb{E}[(X_i - T)^+] \\ &\geq \left(1 - \frac{1}{e}\right) \cdot \mathbb{E}[\mathrm{OPT}] \end{aligned} \tag{3.25}$$

Now we turn to the case that the distribution of σ is (ε, δ)-almost pairwise independent for $\delta = \varepsilon^2$. In that case, we group the time steps $k \in [n]$ into "buckets" of εn consecutive steps; the bucket containing the time step when X_i is observed is numbered $b(i) = \left\lceil \frac{\sigma(i)}{\varepsilon n} \right\rceil$. The counterpart of (3.21) is the following inequality:

$$c_i \geq \sum_{u=1}^{1/\varepsilon} \Pr[b(i) = u] \cdot \prod_{\ell=1}^{\varepsilon n u} \Pr[X_{\pi(\ell)} < T \mid b(i) = u] \tag{3.26}$$

The inequality is valid because the left side is the probability that no value greater than T is observed before X_i is observed, and the right side is the probability no value greater than T is observed in buckets $1, 2, \ldots, b(i)$. For any set $S \subseteq [n]$, let $\tilde{p}_{u,i}(S) = \Pr[\pi([\varepsilon n u]) = S \mid b(i) = u]$ denote the conditional probability that S is exactly equal to the set of elements observed in buckets $1, 2, \ldots, u$, given that $b(i) = u$. Analogously to (3.22), we can use the AM-GM inequality to derive

$$\prod_{\ell=1}^{\varepsilon n u} \Pr[X_{\pi(\ell)} < T \mid b(i) = u] \;\geq\; \prod_{j \in [n]} q_j^{\sum_{S \subseteq [n], j \in S} \tilde{p}_{u,i}(S)} . \tag{3.27}$$

The exponent of q_j on the right side is equal to $\Pr[b(j) \leq u \mid b(i) = u]$. If the distribution of σ is (ε, δ)-almost pairwise independent, this probability is at most $\varepsilon u + \delta$ and $\Pr[b(i) =$

$u] \geq \varepsilon - \delta$. Therefore,

$$c_i \geq (\varepsilon - \delta) \cdot \sum_{u=1}^{1/\varepsilon} q^{\varepsilon u + \delta} \tag{3.28}$$
$$= \left(1 - \tfrac{\delta}{\varepsilon}\right) \cdot q^{\varepsilon+\delta} \cdot \left[\varepsilon\left(1 + q^{\varepsilon} + q^{2\varepsilon} + \cdots + q^{1-\varepsilon}\right)\right].$$

If $q = \frac{1}{e}$ and $\delta = \varepsilon^2$ then the factors $1 - \frac{\delta}{\varepsilon}$ and $q^{\varepsilon+\delta}$ are both $1 - O(\varepsilon)$ and the factor $\varepsilon\left(1 + q^{\varepsilon} + q^{2\varepsilon} + \cdots + q^{1-\varepsilon}\right)$ is at least $1 - \frac{1}{e}$, again by Lemma 28. Hence, $c_i \geq 1 - \frac{1}{e} - O(\varepsilon)$. As before, $p = 1 - q = 1 - \frac{1}{e}$, and the lemma follows by substituting these bounds for p and c_i into (3.20) and comparing with (3.19). □

We now restate and prove Theorem 5.

Theorem 9. *For every $n \in \mathbb{N}$ and $\varepsilon > 0$, there is a set Π consisting of $O(\mathrm{poly}(\varepsilon^{-1}) \cdot \log n)$ permutations such that $\overline{\mathcal{R}}_n(\Pi) > 1 - \frac{1}{e} - \varepsilon$. There is also a set Π consisting of $O(n^2)$ permutations such that $\overline{\mathcal{R}}_n(\Pi) \geq 1 - \frac{1}{e}$.*

Proof. If n is prime, Lemma 7 combined with Theorem 8 show that there exists a set Π of $n(n-1)$ permutations such that $\overline{\mathcal{R}}_n(\Pi) \geq 1 - \frac{1}{e}$. If n is composite, we make use of a "padding lemma" (Lemma 29 in Appendix B.2) which says that for any $N \geq n$, a set of m permutations $\Pi_N \subseteq S_N$ can be transformed into a set of at most m permutations, $\Pi_n \subseteq S_n$, such that $\overline{\mathcal{R}}_n(\Pi_n) \geq \overline{\mathcal{R}}_N(\Pi_N)$. Taking N to be a prime between n and $2n$ [90], the padding lemma implies there is a set Π_n of fewer than $4n^2$ permutations satisfying $\overline{\mathcal{R}}_n(\Pi_n) \geq 1 - \frac{1}{e}$. Similarly, for $\varepsilon > 0$, take $k \in \mathbb{N}$ such that $\varepsilon/2 < \frac{1}{k} \leq \varepsilon$. If $2 \leq n \leq 2k^3$ then $n^2 \leq 4k^6 \leq 256\varepsilon^{-6}$, and we have already shown the existence of $\Pi \subseteq S_n$ with $|\Pi| = O(n^2) = O(\varepsilon^{-6})$ and $\overline{\mathcal{R}}_n(\Pi) \geq 1 - \frac{1}{e}$. Otherwise, $n > 2k^3$ so there is a multiple of k between n and $2n$. Denoting this number by N, and observing that the constraint $\varepsilon' N \geq 2/\delta'$ is satisfied when $\varepsilon' = 1/k$ and $\delta' = (\varepsilon')^2$, we apply Lemma 7 and Theorem 8 to deduce the existence of a set Π_N of $O(k^6 \log N)$ permutations of $[N]$ such

that $\overline{\mathcal{R}}_N(\Pi_N) \geq 1 - \frac{1}{e} - \varepsilon$, then we use the padding lemma to deduce the existence of $\Pi_n \subseteq S_n$ such that $|\Pi_n| = O(k^6 \log N) = O(\varepsilon^{-6} \log n)$ and $\overline{\mathcal{R}}_n(\Pi_n) \geq 1 - \frac{1}{e} - \varepsilon$. □

CHAPTER 4

INDIVIDUAL FAIRNESS IN PROPHET INEQUALITIES

In Algorithmic Decision-making there is a tension between optimality — making a decision maximizing some aggregate objective — and fairness, — ensuring equal opportunity for all alternatives — especially when multiple parties are involved and affected by the decisions. In this chapter we investigate optimal stopping problems from a fairness perspective. In contrast to prior work that studies optimal stopping in the context of *group fairness* constraints [30], in this work we initiate the study of how *individual fairness constraints* influence the performance of stopping rules.

In this chapter we work with an abstract formulation of individual fairness in optimal stopping that is as simple and free of application-specific details as possible. However, to motivate the fairness constraints we will consider, it will be helpful to contemplate some motivating applications in which the potential for stopping rules to yield unfair outcomes is evident.

Example 1. Consider a firm interviewing a sequence of candidates for a job opening. Assume that the firm may hire at most one candidate, and that the decision whether or not to hire a candidate must be made immediately after their interview, without waiting to judge the quality of candidates scheduled to be interviewed later in the hiring season. If we assume that the firm wishes to maximize the expected quality of the candidate hired, and that the candidates' qualities are independent and identically distributed[1] random variables, then an optimal stopping rule would calculate a decreasing sequence of thresholds and hire the first candidate whose quality exceeds the corresponding threshold. Thus, although the candidates are *a priori* identical, the optimal stopping rule dis-

[1]The i.i.d. assumption is for the sake of this example. In general we will be considering independent but non-identical distributions in this work.

criminates according to arrival time: a candidate of low quality has no chance of being hired if they interview early (when the threshold will be above their quality level) but stands a chance of being hired if they interview late. On the other hand, the opposite type of unfairness exists for candidates of high quality: they have a high probability of being hired if they interview early, and a lower probability of being hired if they interview late because of the possibility that an earlier candidate has already been hired. Replacing the optimal stopping rule with a threshold stopping rule — i.e., one which uses a constant threshold over time and hires the first candidate whose quality exceeds this threshold — eliminates the first type of unfairness but not the second.

In the preceding example, the selection rule treated individuals differently due to differences in their arrival time. Another potential type of unfairness occurs when an individual's probability of being selected (conditional on their value) depends on the individual's identity. This type of unfairness can arise even in offline selection problems, where a decision maker is choosing one of n elements and observes the value (or quality) of each element before making a choice.

Example 2. Consider a ride sharing platform in a city where demand exceeds supply: in each time interval, the platform receives a set of requests and must decide which ones will receive service. A seemingly fair selection rule is to choose the request of maximum value, breaking ties at random. To illustrate the potential for unfairness, suppose there is only one driver and two users, Odysseus and Penelope. Odysseus is a frequent traveler who requests a ride every day, whereas Penelope mostly stays home and only needs a ride once every m days for some $m > 1$. Let us model the value of selecting a user to be 1 if they requested a ride and 0 otherwise. If the platform uses random tie-breaking then half of Penelope's requests are denied whereas the fraction of Odysseus' requests that are denied is only $\frac{1}{2m}$. A fairer rule would select Penelope with probability $\frac{m}{m+1}$ when she makes a request, and otherwise it would select Odysseus. Under this rule, both users

would find that $\frac{1}{m+1}$ of their requests are denied and the rest are served.

The foregoing two examples motivate two different notions of individual fairness. We formalize these two properties in Section 4.2 and label them as *time-independent fairness* (TIF) and *identity-independent fairness* (IIF). Depending on the application, one may wish for selection rules to satisfy one of these properties, or the other, or both. A few natural questions arise about such fairness-constrained selection rules.

Can one efficiently optimize over fairness-constrained stopping rules? Absent fairness constraints, one can generally compute optimal stopping rules using dynamic programming, but it appears difficult to incorporate fairness constraints into the dynamic programming formulation since they impose dependencies among the decisions made at different times. We show in Section 4.4 that optimal fairness-constrained stopping rules can be computed by solving a polynomial-sized linear program.

How well can fairness-constrained selection rules approximate optimal selection rules, in the worst case? There are many versions of this question, depending whether the fairness constraint is TIF or IIF (or both), and whether the fairness-constrained selection rule or the optimal selection rule (or both) is allowed to make decisions offline (i.e., observe the full sequence of values before making its selection). In Section 4.5 we answer almost all versions of this question; our results are summarized in Figure 4.1. In Section 4.6 we investigate how the answers to these questions differ when one imposes an additional constraint that the decision maker must select one of the n alternatives. In the traditional setting of prophet inequalities one can assume this property without loss of generality: if no hire is made before the last interview, a firm loses nothing by hiring the final candidate. In contrast, we show that requiring the firm to hire a candidate, while also imposing either the TIF or IIF fairness constraint, can make an enormous qualitative difference: the expected value of the candidate selected by the optimal TIF

(or IIF) stopping rule may differ from the expected value of the best candidate by an unbounded factor.

In cases when the precise input distribution is not known, can approximately optimal fairness-constrained selection rules be learned from data? Since the fairness criteria in this paper are distribution-dependent, one might anticipate that there is no way for a decision maker to ensure that their selection rule is fair without knowing the input distribution. In Section 4.7 we show that this intuition is false: if the decision maker has access to a single sample from each distribution, that is enough to implement a constant-competitive offline selection rule that satisfies both TIF and IIF. With access to two independent samples from each distribution, the decision maker can implement a constant-competitive online stopping rule that satisfies both TIF and IIF. The question of whether fairness-constrained stopping rules can be constant-competitive with just one sample from each distribution, rather than two, is an enticing open question that we leave for future work.

4.1 Related Work

Among the prophet inequalities literature, fairness has been very recently considered in [30] but unlike our work which deals with individual fairness (from the perspective of each candidate), [30] deal exclusively with a form of group-fairness constraint for the secretary problem and prophet inequality settings. In their prophet model, individuals are partitioned into disjoint groups, and fairness constraints are expressed as lower bounds on the expected number of members selected from each group. They prove a tight competitive ratio of $1/2$ between a fair, online algorithm and a fair, offline one, akin to some of our results. Additionally they compare fair, online algorithms to fair, offline ones in some restricted settings (e.g. group hiring priorities being proportional to

group size, distributions being i.i.d. and more) and provide tight competitive ratios there as well.

Somewhat closer to our setting, a similar notion to our TIF fairness has been studied in the context of the secretary problem under the name of "Incentive Compatibility" in [22]. This is the property of a stopping rule for the secretary problem which requires the probability of selecting a candidate at the t-th step to be equal for all $t \in [n]$, hence the name incentive compatible since any candidate i has no incentive to arrive at any different time. Interpreted within our framework, the incentive compatibility of [22] can be thought of as an ex-ante kind of time-independent fairness, whereas our TIF is an ex-interim kind since our definition conditions on the sampled value of the i-th candidate when considering their hiring probability.

In a broader sense, our work adds to a line of research exploring fairness in dynamic settings such as dynamic resource allocation problems (e.g. [73, 67, 88]) or dynamic fair division (e.g. [93, 57]), settings in which agents arrive dynamically and some allocation needs to be computed that is both maximizing some performance metric but also satisfying certain fairness criteria.

4.2 Fairness

In the introduction of this chapter we studied examples of different kinds of unfairness that manifest in online decision-making environments and pinned down two particular kinds of fairness criteria that were violated in each case. Here we give formal definitions for those fairness properties in the context of the adversarial-order prophet problem (Problem 2).

Important remarks on chapter-specifc assumptions To accommodate the treatment of the material in this chapter, we will be making the simplifying assumption that all distributions of a given instance are supported on the *same*, *discrete* set $\mathcal{S}$, meaning for all $x \in \mathcal{S}$ and all $i \in [n] : \Pr[X_i = x] > 0$. This is done to avoid some technically pathological cases where one needs to condition on events with probability zero. Additionally, the fairness definitions studied here are relevant to both offline algorithms and stopping rules alike. Previously, the only offline algorithm we considered was the trivial algorithm of the prophet, OPTOFF. In this chapter, we consider broader classes of offline algorithms, ones that are not necessarily making optimal choices. When defining fairness, we refer to the decision-making process simply as an *algorithm* and this includes both offline algorithms with the ability to see the values of all X_i's before making a decision as well as stopping rules adapted to a given inspection order dictated by a permutation π.

We begin by considering the situation in Example 2. The observation there was that conditional on candidate i having value x, the probability of them getting hired should only depend on the value x and not on the identity i of the particular candidate. This leads naturally to a fairness definition adapted to the framework of the adversarial-order prophet problem (Problem 2).

Definition 12 (Identity-Independent Fairness (IIF))**.** An algorithm $\text{ALG}_{\mathcal{I},\pi}$ for an instance $\mathcal{I} = (\mathcal{F}_i)_{i\in n}$ supported on $\mathcal{S}$ and inspection order dictated by permutation $\pi : [n] \to [n]$ (the latter is relevant only for stopping rules) is said to satisfy *Identity-Independent Fairness* if there exists a function $p : \mathcal{S} \to [0, 1]$ such that:

$$\Pr[\text{ALG}_{\mathcal{I},\pi} \text{ accepts } X_i \mid X_i = x] = p(x), \quad \forall i \in [n], x \in \mathcal{S}.$$

In other words, the probability of selecting X_i conditional on its value being x is independent of its *identity* i (but potentially dependent on its value x).

When an algorithm satisfies this property, we say it is "IIF with probability function

$p(x)$".

The above definition is given for a particular inspection order π but it will be useful to extend it in the case of stopping rules to *families* of algorithms $\{\text{ALG}_{\mathcal{I}}^{\pi}\}_{\pi\in S_n}$, all for the same instance $\mathcal{I}$ but each for a different inspection order, and say that a family like that is IIF if each stopping rule in the family is IIF with some probability function $p_\pi(x)$ (potentially different for each permutation).

Similarly, we can formalize the fairness notion demonstrated in Example 1 as follows.

Definition 13 (Time-Independent Fairness (TIF))**.** A family of algorithms $\{\text{ALG}_{\mathcal{I},\pi}\}_{\pi\in S_n}$, all for the same instance $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$ supported on $\mathcal{S}$, each for a different inspection order π is said to satisfy *Time-Independent Fairness* if there exists a function $p : [n] \times \mathcal{S} \to [0, 1]$ such that:

$$\Pr[\text{ALG}_{\mathcal{I},\pi} \text{ accepts } X_i \mid X_i = x] = p(i, x), \quad \forall i \in [n], x \in \mathcal{S}, \pi \in S_n.$$

In other words, the probability of accepting X_i conditional on its value being x is independent of its inspection time across different inspection orders (but is allowed to depend on its identity i and its value x).

Unlike Definition 12, this definition of fairness is meaningful only in the online setting. When considering the behavior of an offline algorithm with access to all r.v.s at once, the definition is trivially satisfied.

We claim that the two definitions capture different notions of fairness and indeed they are independent of one another. The following examples demonstrate that fact by giving instances and families of stopping rules which satisfy either Definition 12 or Definition 13 but not both.

- Consider an instance of two independent r.v.s X_1, X_2, arbitrarily distributed on a common support set $\mathcal{S}$. For arrival permutation $\pi_1 = (1, 2)$, define ALG_{π_1} as the stopping rule which selects one of X_1, X_2 uniformly at random. On the other hand, for $\pi_2 = (2, 1)$, define ALG_{π_2} as the stopping rule that terminates without hiring.

  Each algorithm in the family $\{\text{ALG}_{\pi_i}\}_{i=1}^2$ satisfies the IIF definition with $p_{\pi_1}(x) = 1/2, p_{\pi_2}(x) = 0$ respectively for all $x \in \mathcal{S}$ but it does **not** satisfy the TIF definition because of the dependence on π.

- Consider again an instance X_1, X_2 of two independent r.v.s, arbitrarily distributed on a common support set $\mathcal{S}$. For either arrival permutation, define a stopping rule ALG_{π_i} which always selects variable $i = 1$ regardless of the realization of its value and regardless of its arrival time.

  Then, the family $\{\text{ALG}_{\pi_i}\}_{i=1}^n$ satisfies the TIF property with $p(i, x) = p(i) = \mathbb{I}[i = 1]$, for all $x \in \mathcal{S}$. On the other hand, none of the algorithms in the family satisfy the IIF property because of the dependence on i.

More importantly, the definitions are **not** mutually exclusive and there exist families of algorithms that satisfy both. A trivial example is an algorithm that never selects any value. This is obviously IIF with $p(x) = 0$ for all x as well as TIF with $p(i, x) = 0$ and indeed can be described as fair even if it performs poorly. Another example is an algorithm which selects a variable X_i uniformly at random among all the choices regardless of the permutation π and regardless of the realization of its value. This algorithm is both IIF and TIF but again has poor performance tending to zero as $n \to \infty$.

Before moving on, let's also verify what was implied earlier, that common stopping rules for the prophet problem indeed fail to satisfy both of our fairness definitions. Consider a simple instance with two 0-1 random variables distributed as follows:

$$X_1 = \begin{cases} 1, & \text{w.p. } 1/2 \\ 0, & \text{w.p. } 1/2 \end{cases}, \quad X_2 = \begin{cases} 1, & \text{w.p. } 2/3 \\ 0, & \text{w.p. } 1/3 \end{cases}.$$

Let $\text{ALG}_{\rightarrow}$ be any single-threshold stopping rule which uses a threshold $T \in (0, 1)$ (e.g. the [61] or the [86] stopping rule) on the forward arrival order, i.e. $\pi = (1, 2)$. In this instance, such a rule would select a variable after inspecting it if and only if its value is equal to 1. Similarly, $\text{ALG}_{\leftarrow}$ is the same stopping rule but applied on the reverse inspection order, i.e. $\pi = (2, 1)$. Consider now the selection probabilities of each r.v. and on each arrival ordering conditional on sampling the higher support point:

$$\begin{aligned} &\Pr[\text{ALG}_{\rightarrow} \text{ accepts } X_1 \mid X_1 = 1] = 1, \\ &\Pr[\text{ALG}_{\rightarrow} \text{ accepts } X_2 \mid X_2 = 1] = \Pr[X_1 = 0] = \frac{1}{2}, \\ &\Pr[\text{ALG}_{\leftarrow} \text{ accepts } X_1 \mid X_1 = 1] = \Pr[X_2 = 0] = \frac{1}{3}, \\ &\Pr[\text{ALG}_{\leftarrow} \text{ accepts } X_2 \mid X_2 = 1] = 1. \end{aligned}$$

Fixing an arrival ordering, we see that neither stopping rule is IIF because the conditional probabilities depend on the identity i of the variable considered. Further, the family $\{\text{ALG}_{\rightarrow}, \text{ALG}_{\leftarrow}\}$ fails to satisfy the TIF property since there is a dependence on the inspection order.

As we observed earlier, the IIF property can be desirable even in an offline version of prophet problem. It is therefore possible to think of fairness as an extra dimension of complexity on top of the online dimension of the problem. Each dimension (offline vs. online and unfair vs. fair) constrains the set of algorithms that can be used in different ways and degrades performance as measured by the expected value of the element accepted. Akin to a prophet inequality, one can ask what is the best competitive ratio achievable when comparing algorithms that have to satisfy different kinds of constraints.

For example, how much more powerful are offline IIF algorithms (IIF prophets) compared to IIF stopping rules? Or, how much more powerful are general stopping rules without fairness considerations compared to IIF stopping rules? What about compared to TIF stopping rules?

In the sections that follow, we address those questions for interesting combinations of settings in order to understand the trade-offs in performance that arise when different kinds of constraints are imposed on the decision maker.

4.3 Designing Simple, Fair Stopping Rules

As noted earlier, some of the standard stopping rules typically used in prophet problems fail to satisfy the fairness properties we introduce in this chapter because they assign different interim acceptance probabilities to variables with different identities or variables with different inspection order, for the IIF or TIF properties respectively. In this section we present a way of modifying one of the standard, single-threshold stopping rules to get a $1/4$-competitive stopping rule which satisfies *both* the IIF and the TIF properties. While the sections that follow present instance-optimal, fair stopping rules with strictly better competitive ratios, the algorithm presented here can be of interest due to its simplicity as well as serving as an instructional example for developing intuition on the use of randomness to achieve fairness in the context of decision-making under uncertainty. The idea is to attempt to equalize the interim probabilities by introducing randomness and allowing the stopping rule to reject a variable — even if its value is above the threshold — with some carefully calibrated, variable-specific probability.

Consider Algorithm 3 below which uses a single threshold T but the decision to accept a variable X_i under consideration requires both the variable having a high enough value

$(X_i \geq T)$ as well as an independent b_i-biased coin coming up heads.

Algorithm 3: IIF and TIF single-threshold-with-coins stopping rule

Parameters: Permutation $\pi : [n] \to [n]$, threshold T, biases $b_1, \ldots, b_n \in [0, 1]$.
for $t = 1, \ldots, n$ **do**
  $i \leftarrow \pi(t)$. `// Index of variable` X_i `inspected on time-step` t.
  if $X_i \geq T$ **then**
    Toss a b_i-biased coin.
    if coin comes up *Heads* **then**
      Accept X_i and halt.
    else
      Reject X_i.
    end
  else
    Reject X_i.
  end
end

We use the threshold of [86] and the following biases, computed so as to guarantee the fairness properties:

$$T := \text{median of } \max_{i \in [n]} X_i, \tag{4.1}$$

$$b_{\pi(t)} := \frac{1}{2 \cdot \left(1 - \frac{1}{2} \sum_{s<t} \Pr[X_{\pi(s)} \geq T]\right)}, \ \forall t \in [n]. \tag{4.2}$$

The lemmas that follow establish that the parameters are well-defined, they induce the IIF and the TIF properties and that the resulting algorithm remains constant-competitive when compared to a prophet.

Lemma 8. *Equations* (4.1) *and* (4.2) *define valid biases* $b_i \in [0, 1]$ *for all* $i \in [n]$.

Proof. To show $b_i \in [0, 1]$, it suffices to show $\sum_{j=1}^{n} \Pr[X_j \geq T] \leq 1$ since $1/(2 \cdot (1 - x/2)) \in [0, 1]$ for $x \leq 1$. Indeed,

$$\begin{aligned}\sum_{j=1}^{n} \Pr[X_j \geq T] &= n - \sum_{j=1}^{n} \Pr[X_j < T] \overset{\text{AM-GM}}{\leq} n - n\left(\prod_{j=1}^{n} \Pr[X_j < T]\right)^{1/n} \\ &= n\left(1 - \left(\frac{1}{2}\right)^{1/n}\right) \leq n\left(1 - e^{-1/n}\right) \leq n\left(1 - \left(1 - \frac{1}{n}\right)\right) = 1.\end{aligned}$$

□

Lemma 9. *Algorithm 3 parameterized by the threshold and biases shown in Equations* (4.1) *and* (4.2)*, defines a family of algorithms (one for each arrival ordering* $\pi \in S_n$*) satisfying* both *the IIF and the TIF properties.*

Proof. Let ALG_π refer to Algorithm 3 parameterized as described in the lemma statement for a particular arrival ordering $\pi : [n] \to [n]$. We claim the following equality,

$$\Pr[\text{ALG}_\pi \text{ accepts } X_i \mid X_i = x] = \begin{cases} \frac{1}{2}, & x \geq T \\ 0, & \text{otherwise} \end{cases}. \tag{4.3}$$

Notice how the right side is independent of both i and π, meaning that each ALG_π is individually IIF and further, the family $\{\text{ALG}_\pi\}_{\pi \in S_n}$ is TIF.

Let us now prove Equation (4.3). For $x < T$, it's easy to see that,

$$\Pr[\text{Accept } X_i \mid X_i = x < T] = 0, \ \forall i \in [n].$$

Now suppose $x \geq T$. We proceed inductively on t. For $t = 1$, using Equation (4.2),

$$\Pr[\text{Accept } X_{\pi(1)} \mid X_{\pi(1)} = x \geq T] = \Pr[\text{Coin 1 comes up Heads}] = b_{\pi(1)} = \frac{1}{2}.$$

Assuming, $\Pr[\mathrm{ALG}^{\pi} \text{ accepts } X_{\pi(s)} \mid X_{\pi(s)} \geq T] = 1/2$, for all $s = 1, 2, \ldots, t$, we have:

$$
\begin{aligned}
&\Pr[\mathrm{ALG}^{\pi} \text{ accepts } X_{\pi(t+1)} \mid X_{\pi(t+1)} = x \geq T] \\
&\qquad = \Pr[\text{Coin } t+1 \text{ comes up Heads}] \cdot \Pr[\text{Inspect } X_{\pi(t+1)}] \\
&\qquad = b_{\pi(t+1)} \cdot \Pr[\text{Inspect } X_{\pi(t+1)}] \\
&\qquad = b_{\pi(t+1)} \cdot (1 - \Pr[\text{Accept } X_{\pi(s)} \text{ for some } s < t]) \\
&\qquad = b_{\pi(t+1)} \cdot \left(1 - \sum_{s<t} \Pr[\text{Accept } X_{\pi(s)}]\right) \\
&\qquad = b_{\pi(t+1)} \cdot \left(1 - \sum_{s<t} \Pr[\text{Accept } X_{\pi(s)} \mid X_{\pi(s)} \geq T] \cdot \Pr[X_{\pi(s)} \geq T]\right) \\
&\qquad \overset{\text{Ind. hyp.}}{=} b_{\pi(t+1)} \cdot \underbrace{\left(1 - \frac{1}{2} \sum_{s<t} \Pr[X_{\pi(s)} \geq T]\right)}_{= \frac{1}{2b_{\pi(t+1)}}} = \frac{1}{2}
\end{aligned}
$$

Hence, eq. (4.3) follows. □

Lemma 10. *Algorithm 3 parameterized with the threshold and biases defined in Equations* (4.1) *and* (4.2) *is* $1/4$*-competitive.*

Proof. For simplicity, and without loss of generality, in this proof we assume the arrival ordering is the identity permutation, i.e. variables are to be inspected in order $X_1, X_2, \ldots, X_n$. Denote by ALG the the value selected by Algorithm 3 (or zero if no element was accepted) and let h be the probability of accepting exactly one element. We proceed similarly to Appendix A.2.1 by proving a lower bound on the expected

performance on the algorithm and prophet separately.

$$\begin{aligned}
\mathbb{E}[\text{ALG}] &= \mathbb{E}\left[\sum_{i=1}^{n} X_i \cdot \mathbb{I}[\text{Accept } X_i]\right] \\
&= \mathbb{E}\left[\sum_{i=1}^{n} T \cdot \mathbb{I}[\text{Accept } X_i] + \sum_{i=1}^{n} (X_i - T) \cdot \mathbb{I}[\text{Accept } X_i]\right] \\
&= T \cdot h + \sum_{i=1}^{n} \mathbb{E}[(X_i - T) \cdot \mathbb{I}[\text{Accept } X_i]] \\
&= hT + \sum_{i=1}^{n} \mathbb{E}[(X_i - T) \cdot \mathbb{I}[\text{Accept } X_i] \mid X_i \geq T] \cdot \Pr[X_i \geq T] \\
&= hT + \sum_{i=1}^{n} \mathbb{E}[(X_i - T) \mid X_i \geq T] \cdot \Pr[X_i \geq T] \cdot \Pr[\text{Accept } X_i \mid X_i \geq T] \\
&= hT + \sum_{i=1}^{n} \mathbb{E}[(X_i - T)^+] \cdot \Pr[\text{Accept } X_i \mid X_i \geq T] \\
&\geq hT + b \sum_{i=1}^{n} \mathbb{E}[(X_i - T)^+],
\end{aligned}$$

where $b := \min_{i=}^{n} b_i$ is the minimum bias.

As for the prophet, a similar upper bound to Inequality (A.11) can be used here as well:

$$\mathbb{E}\left[\max_{i \in [n]} X_i\right] \leq T + \sum_{i=1}^{n} \mathbb{E}\left[(X_i - T)^+\right].$$

Combining the two bounds, we get an inequality

$$\mathbb{E}[\text{ALG}] \geq \min(h, b) \cdot \mathbb{E}\left[\max_{i \in [n]} X_i\right].$$

It remains to show that $\min(h, b) \geq 1/4$. In fact, $b = \min_{i \in [n]} b_i \geq 1/2$ by definition of b_i. As for h,

$$\begin{aligned}
h &= \Pr[\text{Accept some element}] \\
&= \Pr\left[\text{Accept some element} \;\middle|\; \max_{i \in [n]} X_i \geq T\right] \cdot \Pr\left[\max_{i \in [n]} X_i \geq T\right] \\
&\geq \Pr\left[\text{Accept the maximum element } X_{i^*} \mid X_{i^*} \geq T\right] \cdot \frac{1}{2} \\
&= p_{\geq T} \cdot \frac{1}{2} = \frac{1}{4},
\end{aligned}$$

where $p_{\geq T}$ denotes the upper branch of the right side of Equation (4.3). □

4.4 A Characterization of Fair Stopping Rules

We now turn our attention to designing optimal, fair (IIF/TIF) stopping rules, which will eventually lead us to a unified approach to designing algorithms with good competitive ratios in multiple settings. The reason this is a good starting point is that, as we shall see shortly, the combined requirement for fairness along with the online nature of the setting induces a good structure to the set of available algorithms allowing us to compute the optimal by solving a simple polynomial-size Linear Program (LP).

4.4.1 IIF Constraints

The following lemma formalizes the observation that the probability functions in the definition of the IIF property characterize the performance of the stopping rule satisfying that property.

Lemma 11. *Consider an instance $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$ and inspection order permutation π. All stopping rules $\mathrm{ALG}_{\mathcal{I},\pi}$ for this instance satisfying the IIF property with probability function $p(x)$ have the* same *expected performance which is given by the expression:*

$$\mathbb{E}[\mathrm{ALG}_{\mathcal{I},\pi}] = \sum_{i=1}^{n} \sum_{x\in\mathcal{S}} x \cdot f_i(x) \cdot p(x) \tag{4.4}$$

Proof.

$$\begin{aligned}
\mathbb{E}[\text{ALG}] &= \mathbb{E}\left[\sum_{i=1}^{n} X_i \cdot \mathbb{I}[\text{ALG accepts } X_i]\right] \\
&= \sum_{i=1}^{n} \sum_{x \in \mathcal{S}} \mathbb{E}[X_i \cdot \mathbb{I}[\text{ALG accepts } X_i] \mid X_i = x] \cdot \Pr[X_i = x] \\
&= \sum_{i=1}^{n} \sum_{x \in \mathcal{S}} x \cdot \Pr[\text{ALG accepts } X_i \mid X_i = x] \cdot \Pr[X_i = x] \\
&= \sum_{i=1}^{n} \sum_{x \in \mathcal{S}} x \cdot p(x) \cdot f_i(x)
\end{aligned}$$

□

Further, the function p associated with an IIF stopping rule characterizes its behavior in the sense that we can parameterize Algorithm 4 below with any eligible p to induce the same interim acceptance probabilities as the ones dictated by p. Lemma 12 below states this observation formally and identifies a necessary and sufficient condition for an arbitrary function p to correspond to interim acceptance probabilities of a stopping rule for the prophet problem.

Algorithm 4: IIF Stopping Rule

Parameters: $\mathcal{I} = (\mathcal{F}_i)_{i \in [n]}, \pi, p(\cdot)$
for $t = 1, \ldots, n$ **do**
  $i \leftarrow \pi(t)$.
  Inspect X_i and let $x \leftarrow X_i$.
  $Q_t \leftarrow 1 - \sum_{k=1}^{t-1} \sum_{y \in \mathcal{S}} f_{\pi(k)}(y) \cdot p(y)$
  Flip a coin with Heads probability equal to $q_t(x) = \dfrac{p(x)}{Q_t}$.
  if *coin comes up* Heads **then**
    Select i and *halt*.
  else
    Reject and proceed.
  end
end

Lemma 12 (IIF Structural Lemma)**.** *There exists an online algorithm* $\mathrm{ALG}_{\mathcal{I},\pi}$ *for an instance* $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$ *and inspection order* π *satisfying the IIF property with probability function* $p(x)$ *if and only if* p *satisfies*

$$p(x) + \sum_{k=1}^{n-1} \sum_{y\in\mathcal{S}} p(y) \cdot f_{\pi(k)}(y) \leq 1, \quad \forall x \in \mathcal{S}. \tag{4.5}$$

Moreover, if p *is a function satisfying the above inequality, then Algorithm 4 (parameterized by* p*) satisfies the IIF property with the same probability function* p*.*

Proof. For the "only if" direction, the key observation is that for the probabilities $p(x)$ to be valid, they have to be such that the probability of selecting $X_{\pi(t)}$ conditional on sampling any value $x \in \mathcal{S}$ is bounded above by the probability of reaching time step t, because you cannot select a variable with probability greater than that of reaching it in the first place. More formally, let Q_t be the probability of reaching time step t computed as follows:

$$\begin{aligned} Q_t &= \Pr[\text{ALG rejects } X_{\pi(1)}, \ldots, X_{\pi(t-1)}] = 1 - \Pr\left[\bigcup_{k=1}^{t-1} \text{ALG accepts } X_{\pi(k)}\right] \\ &= 1 - \sum_{k=1}^{t-1} \sum_{y\in\mathcal{S}} \Pr[\text{ALG accepts } X_{\pi(k)} \mid X_{\pi(k)} = y] \cdot \Pr[X_{\pi(k)} = y] \\ &= 1 - \sum_{k=1}^{t-1} \sum_{y\in\mathcal{S}} p(y) \cdot f_{\pi(k)}(y), \end{aligned}$$

where we used the fact that a stopping rule hires at most one candidate to assert that the events in the union are disjoint.

We thus need to require $Q_t \geq p(x)$ for all $t \in [n], x \in \mathcal{S}$. Notice however that Q_t is non-negative, decreasing in $t \in [n]$ and independent of x therefore it suffices to require the inequality for $t = n$, which gives the required condition in the statement of the Lemma:

$$p(x) \leq 1 - \sum_{k=1}^{n-1} \sum_{y\in\mathcal{S}} p(y) \cdot f_{\pi(k)}(y), \quad \forall x \in \mathcal{S}.$$

For the "if" direction, it suffices to prove that Algorithm 4 satisfies the IIF property with probability function $p(x)$ if we are given that p satisfies inequalities (4.5). We do this by induction on the time step $t \in [n]$. For the base case, Q_1 is set to 1 in the algorithm, therefore $q_1(x) = p(x)$ and we're done. For the inductive step, consider any time step $t > 1$. The conditional probability of accepting $X_{\pi(t)}$ is,

$$\begin{aligned}
&\Pr[\text{Algorithm 4 accepts } X_{\pi(t)} \mid X_{\pi(t)} = x] \\
=&\Pr[\text{Algorithm 4 accepts } X_{\pi(t)} \mid X_{\pi(t)} = x, \text{ have reached time step } t] \\
&\cdot \Pr[\text{Algorithm 4 rejects } X_{\pi(1)}, \ldots, X_{\pi(t-1)}].
\end{aligned}$$

The first factor in the product is exactly the probability $q_t(x)$ of the coin used by Algorithm 4. Using the inductive hypothesis, and a similar computation to the "only if" direction, we can express the probability of Algorithm 4 rejecting the first $t-1$ variables in terms of the p.d.f.s of each X_i and the probabilities $p(x)$, getting $\Pr[\text{Algorithm 4 rejects } X_{\pi(1)}, \ldots, X_{\pi(t-1)}] = Q_t$, where Q_t is exactly the quantity computed in the for-loop. Hence,

$$\Pr[\text{Algorithm 4 accepts } X_{\pi(t)} \mid X_{\pi(t)} = x] = q_t(x) \cdot Q_t = \frac{p(x)}{Q_t} \cdot Q_t = p(x)$$

□

A consequence of the structural lemma is that it allows us to reduce the problem of designing a fair stopping rule into a simple LP.

Indeed, for IIF stopping rules, Lemma 12 states that the set of all functions $p(x)$ for which there exist an IIF stopping rule with that probability function matches exactly the set of all functions $p(x)$ that satisfy inequalities (4.5). Further, Lemma 11 states that the optimal stopping rule would be the one maximizing a linear function of the probabilities $p(x)$. Therefore, the solution to the following Linear Program gives the conditional

hiring probabilities of the *optimal, IIF* stopping rule for a given instance $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$ and inspection order π:

$$\begin{cases} \max & \displaystyle\sum_{i=1}^{n}\sum_{x\in S} x\cdot f_i(x)\cdot p(x) & \\ \text{s.t.} & \displaystyle p(x)+\sum_{k=1}^{n-1}\sum_{y\in S} f_{\pi(k)}(y)\cdot p(y)\le 1, & \forall x\in S. \\ & p(x)\in[0,1] & \forall x\in S. \end{cases} \tag{OPT Online IIF}$$

After solving (OPT Online IIF), one only has to use the solution to parameterize Algorithm 4 which becomes an optimal, fair stopping rule. We have thus proved the following theorem.

Theorem 10. *The optimal IIF stopping rule for a given instance $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$ and inspection order π can be computed in time polynomial in $n, |\mathcal{S}|$ by solving (OPT Online IIF).*

4.4.2 TIF Constraints

Similarly to IIF fairness, we proceed to prove similar structural lemmas for TIF stopping rules.

Lemma 13. *Consider an instance $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$. Any stopping rule $\mathrm{ALG}_{\mathcal{I},\pi}$ for $\mathcal{I}$ with inspection order π satisfying the TIF property with probability function $p(i,x)$ has the expected performance given by the expression*

$$\mathbb{E}[\mathrm{ALG}_{\mathcal{I},\pi}] = \sum_{i=1}^{n}\sum_{x\in\mathcal{S}} x\cdot f_i(x)\cdot p(i,x),$$

which is independent of the specific permutation π and the specifics of the stopping rule.

The proof of the lemma is omitted as it uses the same probabilistic calculations as Lemma 11, the only difference being that $p(i,x)$ is now in place of $p(x)$. We now pro-

ceed to describe a TIF online algorithm and prove the analogue of Lemma 12 for this setting.

Algorithm 5: TIF Stopping Rule

Data: $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}, \pi : [n] \to [n], p(\cdot,\cdot)$
for $t = 1, \ldots, n$ **do**
  $i \leftarrow \pi(t)$.
  Inspect X_i and let $x \leftarrow X_i$.
  $Q_t^\pi \leftarrow 1 - \sum_{k=1}^{t-1} \sum_{y\in\mathcal{S}} f_{\pi(k)}(y) \cdot p(\pi(k), y)$
  Flip a coin with Heads probability equal to $q_t^\pi(x) = \dfrac{p(i,x)}{Q_t^\pi}$.
  if *coin comes up* Heads **then**
    Accept i and halt.
  else
    Reject i and proceed.
  end
end

Lemma 14 (TIF Structural Lemma)**.** *There exists a family of stopping rules* $\{\mathrm{ALG}_{\mathcal{I},\pi}\}_{\pi\in S_n}$ *— all for the same instance* $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$ *but for different inspection orders — which satisfies the TIF property with probability function* $p(i,x)$ *if and only if* p *satisfies*

$$p(i,x) + \sum_{k\neq i} \sum_{y\in\mathcal{S}} p(k,y) \cdot f_k(y) \leq 1, \quad \forall i \in [n], x \in \mathcal{S}. \tag{4.6}$$

Moreover, if p *is a function satisfying the above inequality, then the family defined by Algorithm 5 for all permutations* $\pi \in S_n$ *satisfies the TIF property with the same probability function* p*.*

Proof. For the "only if" direction we proceed as in the IIF case by first expressing the

probability of rejecting the first $t-1$ elements under some arrival ordering π.

$$\begin{aligned} Q_t^\pi &= \Pr[\mathrm{ALG}_{\mathcal{I},\pi} \text{ rejects } X_{\pi(1)}, \ldots, X_{\pi(t-1)}] = 1 - \Pr\left[\bigcup_{k=1}^{t-1} \mathrm{ALG}_{\mathcal{I},\pi} \text{ accepts } X_{\pi(k)}\right] \\ &= 1 - \sum_{k=1}^{t-1} \sum_{y \in \mathcal{S}} \Pr[\mathrm{ALG}_{\mathcal{I},\pi} \text{ accepts } X_{\pi(k)} \mid X_{\pi(k)} = y] \cdot \Pr[X_{\pi(k)} = y] \\ &= 1 - \sum_{k=1}^{t-1} \sum_{y \in \mathcal{S}} p(\pi(k), y) \cdot f_{\pi(k)}(y) \end{aligned}$$

where we used the fact that a stopping rule selects at most one variable to assert that the events in the union are disjoint. We then require that the conditional hiring probability is bounded by the arrival probability at time step t,

$$p(\pi(t), x) \leq Q_t^\pi, \quad \forall t \in [n], \forall x \in \mathcal{S} \text{ and } \forall \pi \in S_n. \tag{4.7}$$

Unlike the IIF case which considered a single inspection order, the TIF property applies to the whole family of stopping rules and the constraint we just described has to apply for every such permutation. However, we can simplify it and prove that the set of inequalities in (4.6) is satisfied if and only if the set of inequalities (4.7) is satisfied. The direction "(4.7) $\Rightarrow$ (4.6)" follows easily by taking π to be any permutation such that $\pi(n) = i$. For the other direction, assume inequalities (4.6) are all satisfied, let $\pi \in S_n$ be arbitrary permutation, fix arbitrary $t \in [n], x \in \mathcal{S}$ and denote $i = \pi(t)$. Define permutation π' which is derived from π by swapping the t-th and n-th elements. More precisely,

$$\pi'(j) = \begin{cases} \pi(n), & j = t \\ i = \pi(t), & j = n \\ \pi(j), & \text{otherwise} \end{cases}$$

Inequality (4.6) for i, x states that

$$p(i, x) \leq 1 - \sum_{j \neq i} \sum_{y \in \mathcal{S}} p(j, y) \cdot f_j(y),$$

or, equivalently expressed using permutation π' as

$$p(\pi'(n), x) \leq 1 - \sum_{k=1}^{n-1} \sum_{y \in \mathcal{S}} p(\pi'(k), y) \cdot f_{\pi'(k)}(y).$$

The right side of the above inequality cannot decrease if we make the outer summation range from 1 to $t-1$ instead so,

$$p(\pi'(n), x) \leq 1 - \sum_{k=1}^{t-1} \sum_{y \in \mathcal{S}} p(\pi'(k), y) \cdot f_{\pi'(k)}(y).$$

Since π' agrees with π for all $j < t$, the above is equivalent to

$$p(\pi(t), x) \leq 1 - \sum_{k=1}^{t-1} \sum_{y \in \mathcal{S}} p(\pi(k), y) \cdot f_{\pi(k)}(y),$$

which is exactly inequality (4.7).

Now for the "if" direction of the lemma, we prove that the family defined by Algorithm 5 for all $\pi \in S_n$ satisfies the TIF property with probability function $p(i, x)$ again using induction on the time step $t \in [n]$. The base case is again trivial with $Q_1^\pi = 1$ therefore $q_1^\pi(x) = p(i, x)$. For the inductive step, consider time step $t > 1$. The conditional probability of $\text{ALG}_{I,\pi}$ selecting variable $X_{\pi(t)}$ is

$$\begin{aligned}
&\Pr[\text{Algorithm 5 accepts } X_{\pi(t)} \mid X_{\pi(t)} = x] \\
&= \Pr[\text{Algorithm 5 accepts } X_{\pi(t)} \mid X_{\pi(t)} = x, \text{ have reached time step } t] \\
&\quad \cdot \Pr[\text{Algorithm 5 rejects } X_{\pi(1)}, \ldots, X_{\pi(t-1)}].
\end{aligned}$$

The first factor in the product is exactly the probability $q_t^\pi(x)$ of the coin used by Algorithm 5. Using the inductive hypothesis, we can express the probability of Algorithm 5 rejecting the first $t-1$ elements in terms of the p.d.f.s of each X_i and the probabilities $p(i, x)$, getting

$$\Pr[\text{Algorithm 5 rejects } X_{\pi(1)}, \ldots, X_{\pi(t-1)}] = Q_t^\pi,$$

where Q_t^π is exactly the quantity computed in the for-loop. Hence,

$$\Pr[\text{Algorithm 5 accepts } X_{\pi(t)} \mid X_{\pi(t)} = x] = q_t^\pi(x) \cdot Q_t^\pi = \frac{p(i, x)}{Q_t^\pi} \cdot Q_t^\pi = p(i, x).$$

□

Similarly to the IIF case, we can express the probability function $p(i, x)$ of the optimal family of TIF stopping rules as the solution to the following LP

$$\begin{cases} \max & \displaystyle\sum_{i=1}^{n} \sum_{x \in \mathcal{S}} x \cdot p(i,x) \cdot f_i(x) & \\ \text{s.t.} & \displaystyle p(i,x) + \sum_{k \neq i} \sum_{y \in \mathcal{S}} f_k(y) \cdot p(k,y) \leq 1, & \forall i \in [n], \forall x \in \mathcal{S}. \\ & p(i,x) \in [0,1] & \forall i \in [n], \forall x \in \mathcal{S}. \end{cases} \tag{OPT Online TIF}$$

Theorem 11. *The optimal TIF family of stopping rules for a given instance $\mathcal{I} = (\mathcal{F}_i)_{i \in [n]}$ can be computed in time polynomial in $n, |\mathcal{S}|$ by solving (OPT Online TIF).*

4.4.3 Offline Relaxation

The following LP relaxation of the offline prophet problem is going to be useful in later sections.

Lemma 15. *Let $\mathcal{I} = (\mathcal{F}_i)_{i \in [n]}$ be an instance of the prophet problem. Let* OPTO_{FF} *be the value of the selected element by the optimal offline algorithm for $\mathcal{I}$ and let C^* be the optimal solution to the following LP:*

$$\begin{cases} \max & \displaystyle\sum_{i=1}^{n} \sum_{x \in S} x \cdot p_{ix} \cdot f_i(x) & \\ \textit{s.t.} & \displaystyle\sum_{i=1}^{n} \sum_{x \in S} p_{ix} \cdot f_i(x) \leq 1 & \\ & p_{ix} \in [0,1] & \forall i \in [n], \forall x \in S. \end{cases} \tag{Offline Relaxation}$$

Then,

$$\mathbb{E}[\text{OPTO}_{\text{FF}}] \leq C^*.$$

Proof. Let $p_{ix} = \Pr[\text{OPTO}_{\text{FF}} \text{ accepts } i \mid X_i = x]$. Since any algorithm accepts at most one variable, it follows that

$$\mathbb{E}[\#\text{ of variables accepted by OPTO}_{\text{FF}}] \leq 1.$$

The expectation can be expressed in terms of p_{ix} as

$$\begin{aligned}\mathbb{E}[\#\text{ of variables accepted by OPTO}_{\text{FF}}] &= \mathbb{E}\left[\sum_{i=1}^{n} \mathbb{I}[\text{OPTO}_{\text{FF}} \text{ accepts } X_i]\right] \\ &= \sum_{i=1}^{n} \sum_{x \in \mathcal{S}} \Pr[\text{OPTO}_{\text{FF}} \text{ accepts } X_i \mid X_i = x] \cdot \Pr[X_i = x] \\ &= \sum_{i=1}^{n} \sum_{x \in \mathcal{S}} p_{ix} \cdot f_i(x).\end{aligned}$$

Therefore, the probabilities p_{ix} of OPTO$_{\text{FF}}$ constitute a feasible solution to the LP and further the objective value C of the LP for p_{ix} is equal to the expected performance of OPTO$_{\text{FF}}$. Hence, the optimal solution p^*_{ix} with objective value C^* cannot be any worse,

$$C^* \geq C = \mathbb{E}[\text{OPTO}_{\text{FF}}].$$

□

The special form of this LP allows to prove that there are optimal solutions with special structure that will allow us later to transform them into solutions to Online IIF/TIF LPs presented previously.

Lemma 16. *Among the set of optimal solutions to (Offline Relaxation), there is one that satisfies $p_{ix} = p_{jx}$ for all $i, j \in [n]$ and all $x \in \mathcal{S}$.*

Proof. Let $p = (p_{ix})$ be any optimal solution to (Offline Relaxation). We will explain how to construct another solution $p' = (p'_{ix})$ that obeys the LP constraints, has the same objective value and further satisfies the condition required by the lemma. For any $x \in S$, such that $\sum_{i=1}^{n} p_{ix} f_{ix} = 0$ we set $p'_{ix} = 0$. For any $x \in S$ such that $\sum_{i=1}^{n} p_{ix} f_i(x) = w_x > 0$,

we let $z_x = \sum_{i=1}^{n} f_i(x)$ and set $p'_{ix} = w_x/z_x$ for all i. Note that $0 \leq p'_{ix} \leq 1$ because w_x/z_x is the weighted average of the numbers p_{ix}, weighted by $f_i(x)/z_x$, and each p_{ix} belongs to $[0, 1]$.

The only constraint of (Offline Relaxation) is satisfied by p' because

$$\sum_{i=1}^{n} p'_{ix} f_i(x) = \frac{w_x}{z_x} \sum_{i=1}^{n} f_i(x) = \frac{w_x}{z_x} \cdot z_x = w_x = \sum_{i=1}^{n} p_{ix} f_i(x) \tag{4.8}$$

and summing the above equations over x, we get that the left side of the LP constraint is the same for p' as for p. Finally, to verify the equality of the objective function values, scale Equation (4.8) by x, then sum over all $x \in S$. □

4.5 Competitive Fair Algorithms

In the previous sections we described how to reduce the problem of computing the optimal IIF/TIF algorithm for a given instance to solving an LP. Here, we use this reduction to show tight competitive ratios between different settings (online vs. offline, unfair vs. fair etc.). In what follows, we present the results separately for each kind of fairness. The results are summarized in Figure 4.1 as competitive ratios between different settings depicted on top of arcs connecting those settings and our theorems indicate which arc they correspond to.

4.5.1 Competitive IIF

For the IIF property, there are four settings of interest that correspond to all combinations of offline/online and non-fair/fair settings according to the IIF property. We pictorially represent each of those settings as vertices on a square in the left side of Figure 4.1.

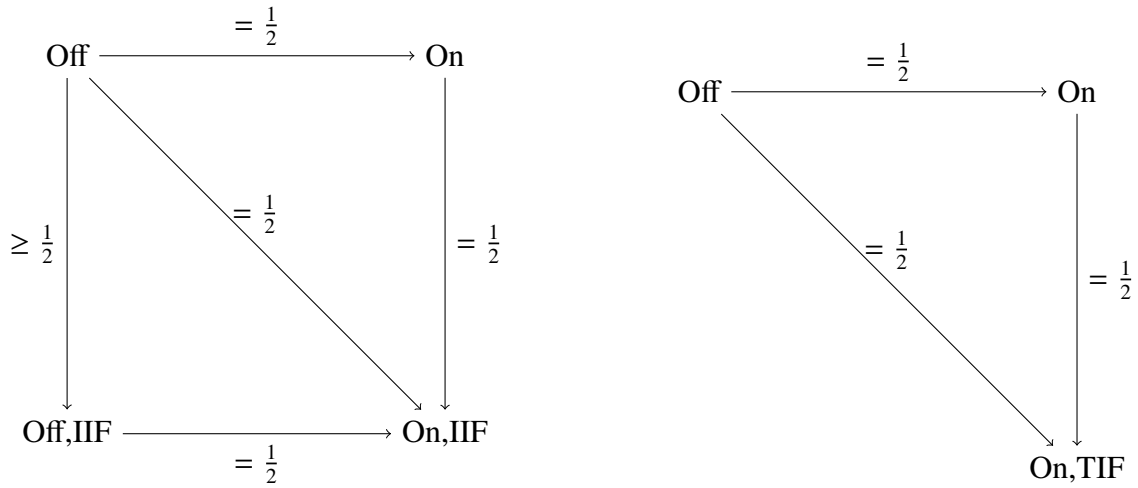


Figure 4.1: Summary of different *settings* we consider. Labels on arcs represent competitive ratios: an arc from setting A to setting B labeled "$\geq r$" (resp. "$= r$") means a competitive ratio bound of the form: for any instance $(\mathcal{F}_i)_{i=1}^n$, let $\mathrm{OPT}^A, \mathrm{OPT}^B$ be the optimal algorithms on this instance for the respective settings, then $\mathbb{E}\left[\mathrm{OPT}^B\right] / \mathbb{E}\left[\mathrm{OPT}^A\right] \geq r$ (resp. the bound is tight). The settings are named in the form $X[, Y]$, where $X \in \{\text{Off}, \text{On}\}$ denotes whether it is an offline or online setting and $Y \in \{\text{IIF}, \text{TIF}\}$ (if present) represents the kind of fairness property required (if any).

Edges on this diagram represent competitive ratios between the corresponding settings (either already known or proven in this chapter).

The top edge of the IIF diagram is the original prophet inequality giving a tight competitive ratio of $1/2$ between the unconstrained (in terms of fairness) offline setting and an unconstrained online setting.

Next, we focus of the diagonal edge comparing the offline non-fair setting (i.e. a prophet) with an online and IIF setting for which we manage to recover a $1/2$-competitive ratio that is tight! This means that despite the fact that a decision maker in the Online IIF setting is more constrained as to what decisions they can make in order to remain fair, they nevertheless can perform at least half as well as an omniscient prophet who makes no effort in maintaining fairness.

Theorem 12 (IIF Diagonal Arc)**.** *Let $\mathcal{I} = (F_i)_{i\in[n]}$ be an instance and π be an inspection*

order for the prophet problem. Let $\text{OPTO}_{\text{FF}_{\mathcal{I}}}$ *be the optimal, offline algorithm on* $\mathcal{I}$ *(i.e. the prophet selecting* $i^* \in \text{argmax}_i X_i$*), and* $\text{OPTO}_{\text{N}}\text{IIF}_{\mathcal{I},\pi}$ *be the optimal IIF stopping rule adapted to the instance* $\mathcal{I}$ *and inspection order* π*. Then,*

$$\mathbb{E}[\text{OPTO}_{\text{N}}\text{IIF}_{\mathcal{I},\pi}] \geq \frac{1}{2} \cdot \mathbb{E}[\text{OPTO}_{\text{FF}_{\mathcal{I}}}].$$

Moreover, the inequality is tight.

Proof. We start by considering an optimal solution p^*_{ix} to (Offline Relaxation) presented in Section 4.4.3 which satisfies $p^*_{ix} = p^*_{jx}$ for all i, j and x and denote that common value by p^*_x. The existence of such a solution is guaranteed by Lemma 16. Let C^* be the objective value of that solution. Based on this, we define $p(x) = p^*_x/2$ for all $x \in \mathcal{S}$.

We claim that $p(x)$ is a feasible solution to (OPT Online IIF) with objective value $C^*/2$. Once we prove this, the theorem follows since C^* is an upper bound to $\mathbb{E}[\text{OPTO}_{\text{FF}}]$. Indeed, consider the left side of a constraint of (OPT Online IIF) for some $x \in \mathcal{S}$:

$$p(x) + \sum_{k=1}^{n-1} \sum_{y \in \mathcal{S}} f_{\pi(k)}(y) \cdot p(y) = \frac{1}{2} p^*_x + \frac{1}{2} \sum_{k=1}^{n-1} \sum_{y \in \mathcal{S}} f_{\pi(k)}(y) \cdot p^*_y. \tag{4.9}$$

Recall that p^*_x as a solution to (Offline Relaxation) it satisfies the feasibility constraint

$$\sum_{i=1}^{n} \sum_{x \in \mathcal{S}} f_i(x) \cdot p^*_x \leq 1.$$

Also, $p^*_x \in [0, 1]$, therefore, we can bound the right side of Equation (4.9) as follows:

$$\begin{aligned} \frac{1}{2} p^*_x + \frac{1}{2} \sum_{k=1}^{n-1} \sum_{y \in \mathcal{S}} f_{\pi(k)}(y) \cdot p^*_y &\leq \frac{1}{2} + \frac{1}{2} \sum_{k=1}^{n} \sum_{x \in \mathcal{S}} f_{\pi(k)}(x) \cdot p^*_x \\ &= \frac{1}{2} + \frac{1}{2} \sum_{i=1}^{n} \sum_{x \in \mathcal{S}} f_i(x) \cdot p^*_x \leq \frac{1}{2} + \frac{1}{2} = 1. \end{aligned}$$

As for the objective value of (OPT Online IIF) for $p(x)$,

$$\sum_{i=1}^{n} \sum_{x \in \mathcal{S}} x \cdot f_i(x) \cdot p(x) = \frac{1}{2} \cdot \sum_{i=1}^{n} \sum_{x \in \mathcal{S}} x \cdot f_i(x) \cdot p^*_x = \frac{C^*}{2}.$$

The tightness of the bound follows from the tightness of the standard prophet inequality with the tight instance we presented in Appendix A.2.1. To be precise though, we need to adapt that example to our framework which requires that all distributions share the same support. We therefore consider the following δ-perturbation of the standard tight instance where we add probability mass $\delta < \varepsilon^2$ to the support points missing on the distribution of each random variable,

$$X_1 = \begin{cases} 1/\varepsilon, & \text{w.p. } \delta \\ 1, & \text{w.p. } 1-2\delta \\ 0, & \text{w.p. } \delta \end{cases}, \quad X_2 = \begin{cases} 1/\varepsilon, & \text{w.p. } \varepsilon \\ 1, & \text{w.p. } \delta \\ 0, & \text{w.p. } 1-\delta-\varepsilon \end{cases}. \tag{4.10}$$

In this instance, a prophet gets $\mathbb{E}[\text{OPTO}_{\text{FF}}] = 2 - O(\varepsilon)$. On the other hand, any online algorithm (IIF or otherwise) on this instance with arrival ordering $\pi_1 = (1, 2)$ gets no more than $\mathbb{E}[\text{Online}_{\pi_1}] = 1 + O(\varepsilon)$. □

Regarding the remaining arcs on the IIF side of Figure 4.1, notice that the $\geq 1/2$ part of each bound is a consequence of Theorem 12. This is because the optimal algorithm for any setting in {Off,IIF, On} performs no better than the optimal algorithm in the Off setting (in expectation) and no worse than the optimal in the On,IIF setting (in expectation). Having shown that Off and On,IIF have a gap of $1/2$, it directly follows that the competitive ratio between any two other setting is at least $1/2$. The formal results appear in the following theorems.

Theorem 13 (IIF Bottom Arc)**.** *Let $\mathcal{I} = (F_i)_{i\in[n]}$ be an instance of the prophet problem and π be an inspection order. Let* $\text{OPTO}_{\text{FF}}\text{IIF}_{\mathcal{I}}$ *be the optimal, offline* and IIF *algorithm on $\mathcal{I}$, and* $\text{OPTO}_{\text{N}}\text{IIF}_{\mathcal{I},\pi}$ *be the optimal, IIF stopping rule adapted to $\mathcal{I}$ and π. Then,*

$$\mathbb{E}[\text{OPTO}_{\text{N}}\text{IIF}_{\mathcal{I},\pi}] \geq \frac{1}{2} \cdot \mathbb{E}[\text{OPTO}_{\text{FF}}\text{IIF}_{\mathcal{I}}].$$

Moreover, the inequality is tight.

Proof. We already argued how the inequality follows from Theorem 12. To show tightness, consider again the instance in (4.10). We already argued that for $\pi_1 = (1, 2)$, no stopping rule can do better than $1+O(\varepsilon)$, therefore the same holds for the any Online, IIF stopping rule. We now prove that there is an Offline, IIF rule with expected performance at least $2 - O(\varepsilon)$. To do this, it is easier to design an online, IIF stopping rule for the reverse permutation $\pi_2 = (2, 1)$, which implies the existence of an Offline, IIF algorithm with the same expected performance. Indeed let ALG_{π_2} work as follows: inspect X_2 and if $X_2 > 0$, accept it with probability $1 - \varepsilon$ (by tossing a biased coin). Never accept the zero value for X_2. If X_2 was not selected, proceed to inspect X_1. If $X_1 < 0$ again do not select it. Otherwise, for any value $X_1 > 0$, accept it with probability $r = \frac{1-\varepsilon}{1-q}$ where

$$q = \Pr[\text{ALG}_{\pi_2} \text{ accepts } X_2 \text{ at time step } t = 1] = (1 - \varepsilon) \cdot (\varepsilon + \delta).$$

If δ is sufficiently small, e.g. $\delta < \varepsilon^2$ then $q \leq \varepsilon$ and so r is a well-defined probability. To verify the IIF property, consider the probability

$$p(i, x) = \Pr[\text{ALG}_{\pi_2} \text{ accepts } X_i \mid X_i = x].$$

If $x = 0$ then $p(i, x) = 0$ regardless of i. For $x > 0$, it is $p(2, x) = 1 - \varepsilon$ by definition and $p(1, x) = \frac{1-\varepsilon}{1-q} \cdot (1 - q) = 1 - \varepsilon$. Therefore $p(i, x) = p(x)$ confirming the definition of IIF. Finally, the expected value of this IIF stopping rule applied on π_2 can be computed as follows:

$$\begin{aligned}
\sum_{i=1}^{2} \sum_{x \in \{0,1,1/\varepsilon\}} x \cdot p(x) \cdot f_i(x) &= (1 - \varepsilon) \cdot \mathbb{E}[X_1] + (1 - \varepsilon) \cdot \mathbb{E}[X_2] \\
&= (1 - \varepsilon)c \cdot [1 + \delta(1/\varepsilon - 2)] + (1 - \varepsilon) \cdot [1 + \delta] \\
&> 2 - 2\varepsilon
\end{aligned}$$

□

Theorem 14 (IIF Right Arc)**.** *Let $\mathcal{I} = (F_i)_{i \in [n]}$ be an instance of the prophet problem and π be an inspection order. Let* $\text{OPTON}_{\mathcal{I},\pi}$ *be the optimal stopping rule adapted to*

instance $\mathcal{I}$ arrival order π, and $\text{OPTO}_{\text{N}}\text{IIF}_{\mathcal{I},\pi}$ *be the optimal, IIF stopping rule on same instance and arrival order. Then,*

$$\mathbb{E}[\text{OPTO}_{\text{N}}\text{IIF}_{\mathcal{I},\pi}] \geq \frac{1}{2} \cdot \mathbb{E}[\text{OPTO}_{\text{N}}{}_{\mathcal{I},\pi}].$$

Moreover, the inequality is tight.

Proof. Again, we focus only on the upper bound and provide a tight instance. Consider the following instance[2]:

$$X_1 = 1, \quad X_2 = \begin{cases} 1, & \text{w.p. } \varepsilon \\ 0, & \text{w.p. } 1-\varepsilon \end{cases},$$

with arrival ordering $\pi_1 = (1, 2)$. Clearly, $\mathbb{E}[\text{OPTO}_{\text{N}_{\pi_1}}] = 1$ by always selecting the first variable. To compute the optimal, we solve (OPT Online IIF), which in this case reduces to

$$\begin{aligned} \max \quad & p_1 + \varepsilon \cdot p_1 \\ \text{s.t.} \quad & p_0 + p_1 \leq 1 \\ & p_1 + p_1 \leq 1 \\ & p_0, p_1 \in [0, 1] \end{aligned}$$

whose optimal solution as $\varepsilon \to 0$ approaches $1/2$, meaning that $\mathbb{E}[\text{OPTO}_{\text{FF}}\text{IIF}] \to 1/2$.

□

Similarly, Theorem 12 implies a lower bound on the competitive ratio for the arc on the left, comparing fairness exclusively on an offline setting. For this, we do not have a tight example and it is an interesting open question whether this gap can be improved.

Theorem 15 (IIF Left Arc)**.** *Let $\mathcal{I} = (F_i)_{i\in[n]}$ be an instance of the prophet problem. Let* $\text{OPTO}_{\text{FF}}{}_{\mathcal{I}}$ *be the optimal offline rule on $\mathcal{I}$ (i.e. a prophet), and* $\text{OPTO}_{\text{FF}}\text{IIF}_{\mathcal{I}}$ *be the*

[2]For the IIF property to be well-defined, technically we need a common support across r.v.s. This can be amended by considering a δ-perturbation of the example just like we did with previous tight examples.

optimal, IIF stopping rule on the same instance. Then,

$$\mathbb{E}[\text{OPTO}_{\text{FF}}\text{IIF}_{\mathcal{I}}] \geq \frac{1}{2} \cdot \mathbb{E}[\text{OPTO}_{\text{FF}_{\mathcal{I}}}].$$

4.5.2 Competitive TIF Algorithms

We now move our attention to competitive ratios between settings where the fairness criterion is the TIF property. Here the picture is simpler since the TIF property only makes sense in the online setting. The three interesting settings we will be working with in this section are shown on the right side of Figure 4.1. The top arc again corresponds to the well-known, standard prophet inequality. Here we prove the bounds for the other two arc using the tools we developed in the previous sections.

We begin with the diagonal arc where again we manage to prove a $1/2$-competitive ratio. In fact, the theorem that follows is stronger as it implies the existence of a family of algorithms that is at the same time TIF and each of the algorithms individually is IIF achieving the desired ratio of $1/2$.

Theorem 16 (TIF Diagonal Arc)**.** *Let $\mathcal{I} = (F_i)_{i\in[n]}$ be an instance of the prophet problem and let* $\text{OPTO}_{\text{FF}_{\mathcal{I}}}$ *be the optimal, offline algorithm (i.e. the prophet choosing $i^* \in argmax_i X_i$). There exists a family of TIF stopping rules $\{\text{ONTIF}_{\mathcal{I},\pi}\}_{\pi\in S_n}$ one for each inspection order but all adapted to the common instance $\mathcal{I}$, such that,*

$$\mathbb{E}[\text{ONTIF}_{\mathcal{I},\pi}] \geq \frac{1}{2} \cdot \mathbb{E}[\text{OPTO}_{\text{FF}_{\mathcal{I}}}], \quad \forall \pi \in S_n.$$

Moreover, each of $\text{ONTIF}_{\mathcal{I},\pi}$ **also satisfies** *the IIF property. Finally, the inequality is tight.*

Proof. Consider again an optimal solution p^*_{ix} to (Offline Relaxation) presented in Section 4.4.3 and denote the optimal objective value by C^*. Define $p(i, x) = p^*_{ix}/2$. Recall

that p^*_{ix} is such that

$$\sum_{i=1}^{n} \sum_{x \in \mathcal{S}} p^*_{ix} \cdot f_i(x) \leq 1.$$

To show that $p(i, x)$ as defined satisfies the constraints of (OPT Online TIF),

$$p(i,x) + \sum_{k \neq i} \sum_{y \in \mathcal{S}} f_k(y) \cdot p(k,y) = \frac{1}{2} p^*_{ix} + \frac{1}{2} \sum_{k \neq i} \sum_{y \in \mathcal{S}} f_k(y) \cdot p^*_{ky} \leq \frac{1}{2} + \frac{1}{2} \sum_{k=1}^{n} \sum_{y \in \mathcal{S}} f_k(y) \cdot p^*_{ky} \leq 1.$$

Now we compute the objective value of (OPT Online TIF),

$$\sum_{i=1}^{n} \sum_{x \in \mathcal{S}} x \cdot f_i(x) \cdot p(i,x) = \frac{1}{2} \sum_{i=1}^{n} \sum_{x \in \mathcal{S}} x \cdot f_i(x) \cdot p^*_{ix} = \frac{C^*}{2}.$$

In fact, if we assume as before the special solution $p^*_{ix} = p^*_x$ guaranteed by Lemma 16, we can further prove that the constraints of (OPT Online IIF) are satisfied for any fixed inspection order $\pi \in S_n$, meaning that the algorithm we designed in this proof will also be an IIF stopping rule:

$$p(x) + \sum_{k=1}^{n-1} \sum_{y \in \mathcal{S}} f_{\pi(k)}(y) \cdot p(y) \leq \frac{1}{2} p^*_x + \frac{1}{2} \sum_{i=1}^{n} \sum_{y \in \mathcal{S}} f_i(y) \cdot p^*_y \leq 1.$$

As for the tightness of the inequality, the same δ-perturbation of the standard prophet inequality instance presented in Theorem 12 suffices since for permutation $\pi_1 = (1, 2)$, no algorithm (TIF or otherwise) can do better than $1 + O(\varepsilon)$ whereas a prophet gets at least $2 - O(\varepsilon)$. □

Theorem 17 (TIF Right Arc)**.** *Let $\mathcal{I} = (F_i)_{i \in [n]}$ be an instance of the prophet problem. Let $\{\text{OPTON}_{\mathcal{I},\pi}\}_{\pi \in S_n}$ be the family of the optimal stopping rules for each arrival order π of the common instance $\mathcal{I}$. There exists a family of TIF stopping rules $\{\text{ONTIF}_{\mathcal{I},\pi}\}_{\pi \in S_n}$ for the same instance $\mathcal{I}$ such that,*

$$\mathbb{E}[\text{ONTIF}_{\mathcal{I},\pi}] \geq \frac{1}{2} \cdot \mathbb{E}[\text{OPTON}_{\mathcal{I},\pi}], \quad \forall \pi \in S_n.$$

Moreover, the inequality is tight.

Proof. Observe that the $\geq 1/2$ part is a consequence of Theorem 16. This is because no online algorithm can perform better that an offline one in expectation, therefore if an Online,TIF algorithm is $1/2$-competitive compared to an Offline, the bound carries over when comparing to Online algorithms which perform at most as well.

For the tightness, we consider again instance (4.10). We claim that no TIF stopping rule can perform better than $1 + O(\varepsilon)$ and also there exists an online algorithm for the same instance with arrival ordering $\pi_2 = (2, 1)$ which achieves performance $2 - O(\varepsilon)$. The later claim is easy to see: absent of the δ-perturbation (i.e. for the instance shown in (A.16)), a stopping rule performs equally as well as a prophet, achieving an expected reward of $2 - \varepsilon$. The perturbation, for $\delta \ll \epsilon$, does not asymptotically alter the expected performance of the stopping rule. For the former claim, one needs to solve the LP in (OPT Online TIF) for this specific instance to verify that the optimal solution has objective value $1 + O(\varepsilon)$. We omit the computation here. □

4.6 An impossibility result

We've seen how fairness (IIF, TIF or both) is achievable in the online setting while guaranteeing a performance at least half that of a prophet. This means that requiring fairness does not hurt the expected performance of an online decision maker (in the worst case over instances) more than the loss induced purely by the online nature of the setting when compared to an offline one. However, fairness comes at a cost which is not immediately visible when comparing expected performance.

Consider the online IIF $1/2$-competitive algorithm derived in Theorem 12. By defini-

tion, $p(x) = \frac{1}{2}p_x^*$ so the expected number of elements accepted by the algorithm is

$$\sum_{i=1}^{n}\sum_{x\in\mathcal{S}} p(x)\cdot f_i(x) = \frac{1}{2}\sum_{i=1}^{n}\sum_{s\in\mathcal{S}} p_x^*\cdot f_i(x) \le \frac{1}{2},$$

using the constraint of (Offline Relaxation). Since the stopping rule always accepts at most one variable, the probability of accepting some variable at all is at most $1/2$. So there is at least $1/2$ probability of rejecting all variables!

Sometimes, rejecting all random variables could be undesirable even when on average a stopping rule is performing well. It is natural to ask, is there a fair (IIF/TIF) stopping rule which selects exactly one variable with probability 1 and has non-zero competitive ratio when compared to a prophet? A property like that, which requires that a stopping rule always makes some selection has also been studied in the context of the secretary problem by [22] where it is called "must-hire" property.

In this section, we give a negative answer to this question for both IIF and TIF stopping rules. However, our work leaves open the question of whether there exist constant-competitive *offline* algorithms in the Offline, IIF setting which satisfy the must-hire constraint.

Theorem 18. *For any $\varepsilon > 0$, there exists an instance $\mathcal{I}_\varepsilon = (\mathcal{F}_i)_{i\in n}$ and an inspection ordering π_ε of the prophet problem such that for any IIF stopping rule $\mathrm{ALG}_{\mathcal{I}_\varepsilon,\pi_\varepsilon}$ for $\mathcal{I}_\varepsilon$ with*

$$Pr[\mathrm{ALG}\ \textit{accepts exactly one element}] = 1,$$

we have

$$\mathbb{E}[\mathrm{ALG}_{\mathcal{I}_\varepsilon,\pi_\varepsilon}] < \varepsilon\cdot\mathbb{E}[\max_i X_i].$$

Proof. The constraint that a stopping rule always accepts a random variable can be

expressed as a linear constraint in terms of $p(x)$ by letting

$$q_i = \Pr[\text{ALG accepts } X_i] = \sum_{x \in \mathcal{S}} \Pr[\text{ALG accepts } X_i \mid X_i = x] \cdot \Pr[X_i = x] = \sum_{x \in \mathcal{S}} p(x) \cdot f_i(x).$$

Then we require that,

$$\mathbb{E}[\#\text{ of variables accepted}] = \sum_{i=1}^{n} q_i = \sum_{i=1}^{n} \sum_{x \in \mathcal{S}} p(x) \cdot f_i(x) = 1.$$

Append this constraint to (OPT Online IIF) to get an augmented LP. Any solution to this augmented LP can be turned back into a stopping rule which accepts at most one variable as we argued in the previous sections and the extra constraint ensures that the stopping rule derived accepts exactly one. Denoting by OPTONMHIIF the value accepted by the optimal, must-hire, IIF stopping rule, we thus have that the optimal objective value of the augmented LP is exactly $\mathbb{E}[\text{OPTONMHIIF}]$. Re-write the first constraint by using the second constraint as,

$$\begin{aligned}
&\forall x \in \mathcal{S} : p(x) + \sum_{k=1}^{n-1} \sum_{y \in \mathcal{S}} f_{\pi(k)}(y) \cdot p(y) \leq 1 \\
&\qquad \Leftrightarrow p(x) + \underbrace{\sum_{i=1}^{n} \sum_{y \in \mathcal{S}} f_i(y) \cdot p(y)}_{=1} - \sum_{y \in \mathcal{S}} f_{\pi(n)}(y) \cdot p(y) \leq 1 \\
&\qquad \Leftrightarrow p(x) \leq \sum_{y \in \mathcal{S}} f_{\pi(n)}(y) \cdot p(y).
\end{aligned}$$

The last inequality says that all values $p(x)$ (which are real numbers in $[0, 1]$) should be bounded above by strict convex combinations of the set of all $\{p(x)\}_{x \in \mathcal{S}}$. The only way for this to happen is if $p(x) = p(y) = p$ for all $x, y \in \mathcal{S}$. Using now the extra constraint, we can compute p as

$$\sum_{i=1}^{n} \sum_{x \in \mathcal{S}} p \cdot f_i(x) = 1 \Rightarrow p \cdot \sum_{i=1}^{n} 1 = 1 \Rightarrow p = 1/n.$$

Substituting back in the objective we get,

$$\mathbb{E}[\text{OPTONMHIIF}] = \sum_{i=1}^{n} \sum_{x \in \mathcal{S}} x \cdot f_i(x) \cdot p = \frac{1}{n} \sum_{i=1}^{n} \mathbb{E}[X_i].$$

Therefore, we can take any instance such that $\sum_{i=1}^{n} \mathbb{E}[X_i]$ is very close to $\mathbb{E}[\max_{i=1}^{n} X_i]$ and make n large enough. More precisely, given $\varepsilon > 0$ define $\mathcal{I}_\varepsilon$ to contain

$$n = \left\lceil \frac{2 \ln 2}{\ln\left(\frac{1}{1-\varepsilon/2}\right)} \right\rceil$$

i.i.d. random variables distributed as

$$\forall i \in [n] : X_i = \begin{cases} 2/\varepsilon, & \text{w.p. } \varepsilon/2 \\ 0, & \text{w.p. } 1 - \varepsilon/2 \end{cases}.$$

Since all the variables are identically distributed, an arbitrary inspection order π_ε can be chosen, for example the identity permutation. Then $\mathbb{E}[\text{OPTONMHIIF}_{\mathcal{I}_\varepsilon,\pi_\varepsilon}] = 1$ and it can be shown that $\mathbb{E}[\max_i X_i] > 1/\varepsilon$. □

Theorem 19. *For any $\varepsilon > 0$, there exists an instance $\mathcal{I}_\varepsilon = (\mathcal{F}_i)_{i\in[n]}$ of the prophet problem such that for any TIF family of stopping rules $\{\text{ALG}_{\mathcal{I}_\varepsilon,\pi}\}_{\pi\in S_n}$ with*

$$Pr[\text{ALG}_{\mathcal{I}_\varepsilon,\pi} \text{ accepts exactly one variable}] = 1, \forall \pi \in S_n,$$

we have,

$$\mathbb{E}[\text{ALG}_{\mathcal{I}_\varepsilon,\pi}] \leq \varepsilon \cdot \mathbb{E}[\max_i X_i], \forall \pi \in S_n$$

The proof uses similar tools as the proof of the previous theorem and is omitted.

4.7 Single-Sample and Double-Sample Fair Prophet Inequalities

Suppose our algorithm doesn't know the distributions $\mathcal{F}_1, \ldots, \mathcal{F}_n$ but is given independent samples from them. In this section we present a $\frac{1}{2}$-competitive offline selection rule that satisfies IIF, given one sample from each distribution. Then we present a $\frac{1}{9}$-competitive family of stopping rules satisfying both IIF and TIF, given *two* independent samples from each distribution.

In the following algorithms, the elements to be selected are denoted $X_1, \ldots, X_n$; additional independent samples from the same distributions are denoted as $Y_1, \ldots, Y_n$ and $Z_1, \ldots, Z_n$. The algorithms that we analyze are comparison-based, i.e. their decisions are based on comparing pairs of elements of the multiset $W = \{X_1, Y_1, Z_1, \ldots, X_n, Y_n, Z_n\}$. It will be convenient to assume that all such comparisons are strict, i.e. that no two elements of W are equal. To ensure strict comparisons, we assume that our algorithms sample a uniformly random tie-breaking priority $\psi(w)$ in $[0, 1]$, for each $w \in W$. Then when comparing two elements $w, w' \in W$, if w and w' are of equal value, the outcome of the comparison is determined by comparing $\psi(w)$ with $\psi(w')$. (The probability of $\psi(w) = \psi(w')$ is zero because they are sampled from a distribution with no point masses.)

Algorithm 6: Single-sample offline algorithm

Data: $X_i, Y_i \sim \mathcal{F}_i$
Let $i \in [n]$ be such that $X_i = \max\{X_1, \ldots, X_n\}$.
if $X_i > Y_i$ **then**
  Accept X_i.
else
  Reject every variable.
end

Lemma 17. *Algorithm 6 is IIF and $\frac{1}{2}$-competitive.*

Proof. For any $x > 0$ and $i \in [n]$,

$$\Pr[\text{Select } i \mid X_i = x] = \Pr[(\forall j \neq i : X_j < x) \wedge (Y_i < x)] = \prod_{j=1}^{n} F_j(x).$$

The right side does not depend on i, as required by the definition of IIF.

Let $M = \max\{X_1, Y_1, X_2, Y_2, \ldots, X_n, Y_n\}$. When $M = X_i$ for some i, Algorithm 6 is guaranteed to select X_i. Therefore, if we define

$$I = \begin{cases} 1, & \text{if } M \in \{X_1, \ldots, X_n\} \\ 0, & \text{if } M \in \{Y_1, \ldots, Y_n\} \end{cases}$$

we have ALG $\geq I \cdot M$. Observing that $\mathbb{E}[I \mid M] = \frac{1}{2}$ for all values of M, we find that

$$\mathbb{E}[\text{ALG}] \geq \mathbb{E}[I \cdot M] = \mathbb{E}[\mathbb{E}[I \mid M] \cdot M] = \frac{1}{2} \cdot \mathbb{E}[M] \geq \frac{1}{2} \cdot \mathbb{E}[\max\{X_1, \ldots, X_n\}].$$

□

Next we present and analyze a $\frac{1}{9}$-competitive stopping rule that satisfies TIF and IIF given two independent samples Y_i, Z_i from each distribution. The stopping rule is defined in Algorithm 7 below.

Algorithm 7: Double-sample online algorithm

Data: $Y_i, Z_i \sim \mathcal{F}_i, \pi \in S_n$
Let $Y_* = \max\{Y_1, \ldots, Y_n\}$.
for $t = 1, 2, \ldots, n$ **do**
  Inspect $X_{\pi(t)} \sim \mathcal{F}_{\pi(t)}$.
  if $X_{\pi(t)} > Y_*$ *and* ($X_{\pi(s)} < Y_*$ *for all* $s < t$) *and* ($Z_{\pi(s)} < Y_*$ *for all* $s \geq t$) **then**
    Accept $X_{\pi(t)}$.
  end
end

First, observe that this is a well-defined stopping rule: given the samples $Y_1, \ldots, Y_n, Z_1, \ldots, Z_n$, the criterion for selecting $X_{\pi(t)}$ depends only on the values of $X_{\pi(1)}, \ldots, X_{\pi(t)}$. Furthermore Algorithm 7 never selects more than one element: if it selects $X_{\pi(t)}$ then $X_{\pi(t)} > Y_* = \max_{i \in [n]} Y_i$, which means that for all $t' > t$ the condition for selecting $X_{\pi(t')}$ will not be satisfied.

Lemma 18. *Algorithm 7 satisfies TIF and IIF*

Proof. For any $x, y \in \mathcal{S}$, and any index i, let us calculate the conditional probability of selecting i given that $X_i = x$ and $Y_* = y$. If $x < y$ then this probability is zero. Otherwise, suppose $i = \pi(t)$. We have

$$\begin{aligned}\Pr[\text{Accept } i \mid X_i = x, Y_* = y] &= \Pr[X_{\pi(s)} < y \text{ for all } s < t] \cdot \Pr[Z_{\pi(s)} < y \text{ for all } s \geq t] \\ &= \prod_{s=1}^{n} F_{\pi(s)}(y) = \prod_{j=1}^{n} F_j(y).\end{aligned}$$

Summing over the possible values of y,

$$\begin{aligned}\Pr[\text{Accept } i \mid X_i = x] &= \sum_{y \in \mathcal{S}} \Pr[Y_* = y] \cdot \Pr[\text{Select } i \mid X_i = x, Y_* = y] \\ &= \sum_{y \in \mathcal{S}} \Pr[Y_* = y] \prod_{j=1}^{n} F_j(y).\end{aligned}$$

The right side depends on neither π nor i, so the stopping rule is both TIF and IIF. □

Lemma 19. *Algorithm 7 is $\frac{1}{9}$-competitive.*

Proof. Recall the multiset $W = \{X_1, Y_1, Z_1, \ldots, X_n, Y_n, Z_n\}$, and recall that its elements are totally ordered using random priorities to break ties between elements of W whose values are equal. Let $W_* > W_{**}$ denote the two largest values in W. We will use $\mathcal{E}$ to denote the event that $W_* \in \{X_1, \ldots, X_n\}$ and $W_{**} \in \{Y_1, \ldots, Y_n\}$. Conditional on $\mathcal{E}$, Algorithm 7 is assured of selecting the largest element of W. Conditional on the contents of the set W — but not the partition of its elements into $X = \{X_1, \ldots, X_n\}$, $Y = \{Y_1, \ldots, Y_n\}, Z = \{Z_1, \ldots, Z_n\}$ — the probability that $W_* \in X$ is $\frac{1}{3}$ and the conditional probability that $W_{**} \in Y$ given $W_* \in X$ is at least $\frac{1}{3}$. (It is $\frac{1}{2}$ if W_{**} and W_* are samples from the same distribution, and $\frac{1}{3}$ if they are from different distributions.) Hence, $\Pr[\mathcal{E} \mid W] \geq \frac{1}{9}$ and

$$\begin{aligned}\mathbb{E}[\text{ALG}] = \mathbb{E}[\mathbb{E}[\text{ALG} \mid W]] &\geq \mathbb{E}[W_* \cdot \Pr[\text{Hire } W_* | W]] \\ &\geq \frac{1}{9} \cdot \mathbb{E}[W_*] \geq \frac{1}{9} \cdot \mathbb{E}[\max\{X_1, \ldots, X_n\}].\end{aligned}$$

□

4.8 Fairly accepting more than one variables.

The prophet problems presented in Section 2.2 naturally generalize to a setting where the decision-maker is allowed to accept more than one elements, adhering to constraints

on which elements can be accepted together, while still maintaining the requirement that acceptance decisions are *immediate* and *irrevocable*. A theoretically interesting variant of the problem in this direction is when the constraints are encoded through a Matroid as follows.

Problem 6 ((Adversarial-order) Matroid Prophet Problem)**.** The **adversary** chooses an instance $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$, an arrival ordering $\pi \in S_n$ *and* a Matroid $\mathcal{M} = ([n], \mathcal{I})$. The **decision-maker** inspects the random variables (r.v.s) in order $X_{\pi(1)}, \ldots, X_{\pi(n)}$ and the value of each X_i being sampled independently from $\mathcal{F}_i$ by **nature**. After each inspection, the decision-maker either accepts or rejects the corresponding r.v. with the constraint that the set I_t of accepted elements up to time t remains independent at all times, i.e. $I_t \in \mathcal{I}$, for all $t \in [n]$. We associate with a stopping rule ALG^π for this problem a random variable,

$$\mathrm{ALG} := \mathrm{ALG}^\pi := \sum_{i\in I_n} X_i,$$

denoting its payoff which is defined as the sum of the values accepted. The definition of the competitive ratio is extended in the natural way.

One may recover the original prophet problem by setting $\mathcal{M}$ to be the 1-uniform matroid. Kleinberg and Weinberg in [61] prove a prophet inequality for the matroid prophet problem with the same constant of $1/2$ as in the case of the standard prophet problem. As a result, the competitive ratio of Problem 6 is exactly $1/2$.

A special case of the matroid prophet problem is when $\mathcal{M}$ is the k-uniform matroid, allowing the decision-maker to accept any subset of up to k elements for some given constant k. We will refer to that problem as "k-uniform prophet problem". The optimal stopping rule for this variant can be computed using for example Dynamic Programming, similarly to the $k = 1$ case. The competitive ratio of the k-uniform prophet problem is known to be at most $1 - O\left(\sqrt{\frac{1}{k}}\right)$ due to [47] who also provide a stopping rule

and a respective a prophet inequality with a constant $1 - O\left(\sqrt{\frac{\ln k}{k}}\right)$. Later, [2] improved on the lower-bound on the competitive ratio by presenting a near-optimal stopping rule satisfying a prophet inequality with a constant of $1 - \frac{1}{\sqrt{k+3}}$, asymptotically closing the gap between the upper and lower bounds.

The definitions of Identity-Independent Fairness (IIF) and Time-Independent Fairness (TIF) given in Section 4.2 are applicable as they are in the context of the general matroid prophet problem with the same interpretation as before: from the perspective of each candidate $i \in [n]$, conditional on their value, the probability of belonging to the set I_n of accepted elements should be independent of their identity i or their arrival time (for the IIF or TIF properties respectively). Justifying the need for these properties can be trickier in this setting. The motivation behind both definitions was that similar individuals should be treated similarly by the decision-maker. However, one may argue that the matroid constraints already treat the candidates unequally. For instance in a graphic matroid, any bridge of the graph can always safely be added to the set of accepted elements. In the context of an employee-hiring application, a matroid constraint like that may represent the fact that a candidate has some unique qualifications among the candidate pool and that there is always enough budget to hire them. It is an interesting open question to define fairness in such general settings.

In the case of the k-uniform prophet problem however, our definitions of fairness are well-aligned with our original goal of treating similar individuals similarly due to the symmetry already present in the constraints. In a hiring setting for instance, the k-uniform prophet problem can model situations where there are k open slots for a single role where equally qualified candidates are exchangeable in the first place, therefore it makes sense to require that they are treated in a similar manner.

In what follows, we extend some of the fairness results of the previous sections from

the $k = 1$ case to the general $k > 1$ case of the k-uniform prophet problem using largely similar techniques. For simplicity of presentation, we develop the results exclusively in the context of the IIF property, making the assumption that the arrival order is given by the identity permutation.

4.8.1 Optimal, IIF stopping rules for k-uniform prophets

The ideas we developed in Section 4.4 for characterizing the optimal, fair online algorithm using a Linear Program can be extended to the k-uniform prophet problem. In fact, [3] describe such an LP formulation for the laminar matroid version of the prophet problem — of which the k-uniform is a special case — without any fairness constraints. Incorporating fairness into this formulation can be done trivially by expressing the IIF constraints as a linear equality. In this section we briefly describe how this can be achieved.

Let S_i^r be the event that right before inspecting X_i, there are $0 \leq r \leq k$ spots already occupied, i.e. the decision-maker has accepted exactly r elements before inspecting i and there are $k - r$ spots remaining. For each i, the events $\{S_i^r\}_{r=0}^k$ partition the sample space and characterize the state in which the algorithm is at any time-step $i \in [n]$. We now introduce notation for the following probabilities which will often be used in this

and the following section,

$$
\begin{aligned}
s_i^r &= \Pr[S_i^r] = \Pr[r \text{ elements accepted right before inspecting } X_i] \\
p_{ix}^r &= \Pr[\text{Accept } i \mid X_i = x \cap S_i^r] \\
q_{ix}^r &= \Pr[\text{Accept } i \cap S_i^r \mid X_i = x] = p_{ix}^r \cdot s_i^r \\
r_{ix}^r &= \Pr[\text{Accept } i \cap S_i^r \cap X_i = x] = p_{ix}^r \cdot s_i^r \cdot f_i(x) \\
p_{ix} &= \Pr[\text{Accept } i \mid X_i = x] = \sum_{r=0}^{k} q_{ix}^r = \sum_{r=0}^{k} p_{ix}^r \cdot s_i^r.
\end{aligned}
$$

With this notation, we can express the behavior of an optimal, IIF fair stopping rule using the following LP on the variables q_{ix}^r and s_i^r:

$$
\begin{aligned}
\max \quad & \sum_{i=1}^{n} \sum_{r=0}^{k} \mathbb{E}_{x\sim\mathcal{F}_i}[x \cdot q_{ix}^r] && && (4.11)\\
\text{s.t.} \quad & s_{i+1}^{r+1} = s_i^{r+1} + \mathbb{E}_{x\sim\mathcal{F}_i}[q_{ix}^r] - \mathbb{E}_{x\sim\mathcal{F}_i}[q_{ix}^{r+1}], && r \in \{0\} \cup [k-1], i \in [n-1] && (4.12)\\
& s_{i+1}^0 = s_i^0 - \mathbb{E}_{x\sim\mathcal{F}_i}[q_{ix}^0], && i \in [n-1] && (4.13)\\
& s_1^r = \mathbb{I}[r = 0], && r \in \{0\} \cup [k] && (4.14)\\
& q_{ix}^k = 0, && i \in [n], r \in \{0\} \cup [k] && (4.15)\\
& \sum_{r=0}^{k} q_{ix}^r = \sum_{r=0}^{k} q_{jx}^r, && i, j \in [n], x \in \mathcal{S} && (4.16)\\
& 0 \le q_{ix}^r \le s_i^r \le 1, && i \in [n], x \in \mathcal{S}, r \in \{0\} \cup [k], && (4.17)
\end{aligned}
$$

where $\mathbb{E}_{x\sim\mathcal{F}_i}[g(x)]$ is just a shorthand notation for the linear expression $\sum_{x\in\mathcal{S}} g(x) \cdot f_i(x)$ where $f_i(x) = \Pr[X_i = x]$. Equation (4.12) expresses how the probability of the algorithm reaching "state" S_i^r evolves over time and Equations (4.13) to (4.15) are initial and boundary conditions. Equation (4.17) ensures the variables are valid probabilities and further, ensures the probability of accepting an element is never higher than the probability of reaching it in the first place — trivially a necessary condition for the LP variables to correspond to probabilities of an online algorithm. Equation (4.16) is our

only addition to the LP of [3] which expresses the IIF constraint,

$$\Pr[\text{ALG accepts } X_i \mid X_i = x] = \Pr[\text{ALG accepts } X_j \mid X_j = x], \text{ for all } i, j \in [n], x \in \mathcal{S}.$$

Any feasible solution to the LP above can be plugged into Algorithm 8 below to give an online, IIF algorithm. Further, Theorem 20 ensures that the LP polytope contains the probabilities of the optimal, IIF online algorithm, therefore optimizing over objective (4.11) allows us to derive the optimal, IIF stopping rule for any given instance.

Algorithm 8: IIF Stopping Rule for the k-uniform prophet problem.

Parameters: $\{q^r_{ix}\}_{i\in[n],x\in\mathcal{S},0\le r\le k}, \{s^r_i\}_{i\in[n],0\le r\le k}$.
$p^r_{ix} \leftarrow \frac{q^r_{ix}}{s^r_i}$, for all $i \in [n], x \in \mathcal{S}, 0 \le r \le k$.
$I_0 \leftarrow \emptyset$.
for $i = 1, \ldots, n$ **do**
  Inspect X_i and let $x \leftarrow X_i$.
  $r \leftarrow |I_{i-1}|$.
  Flip a p^r_{ix}-biased coin.
  if *coin comes up* Heads **then**
    Accept $X_i : I_i \leftarrow I_{i-1} \cup \{i\}$.
  else
    Reject $X_i : I_i \leftarrow I_{i-1}$.
  end
  Return I_n.
end

Theorem 20. *The optimal, IIF stopping rule for a given instance $\mathcal{I} = (F_i)_{i\in[n]}$ of the k-uniform prophet problem can be computed in time $O(\text{poly}(n, k, |\mathcal{S}|))$ by solving the LP in eqs.* (4.11) *to* (4.17) *and parameterizing Algorithm 8 with the solution.*

The proof of Theorem 20 is omitted and the reader is referred to [3, Section 2.7] for a treatment of a similar LP absent of fairness constraints.

Remark 4. An analogous result to Theorem 20 can be derived for the TIF property with the important caveat that the trivial way of expressing a TIF constraint results in an exponential number of constraints. At the time of writing, we are not aware of any

poly-size LP formulation of the optimal, TIF stopping rule for the k-uniform prophet problem. However, the next section develops a near-optimal stopping rule satisfying both the IIF and TIF properties.

4.8.2 Competitive, IIF stopping rules for k-uniform prophets

In the previous section we described how to efficiently compute the optimal, IIF fair stopping rule for any given instance of the k-uniform prophet problem. However, we didn't make any claims about its competitiveness compared to an unconstrained prophet. Here we describe an easier-to-analyze, IIF (and TIF) fair stopping rule and prove a prophet inequality (Theorem 21) with a constant $1 - O\left(\sqrt{\frac{\ln k}{k}}\right)$, implying the same bound for Algorithm 8 presented earlier. We do this by adding randomness to the single-threshold stopping rule of [47] — an idea we developed in Section 4.3 for the $k = 1$ case — to achieve the IIF property while preserving the same asymptotic competitive ratio guarantee.

Consider Algorithm 9 below which is parameterized by a threshold T and a sequence of biases $b_1, \ldots, b_n$. We start by proving a general, parametric lower bound on the expected performance of any such algorithm.

Lemma 20. *Let $k \leq n$ be positive integers and $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$ be any instance of the k-uniform prophet problem. Consider running Algorithm 9 on $\mathcal{I}$ for some choice of threshold T and biases $b_1, \ldots, b_n$. Denote by H the random variable counting the number of elements accepted (equal to the value of s on termination of the algorithm) and let*

$$h := \mathbb{E}[H] = \mathbb{E}[\#\text{ of elements accepted by ALG}], \tag{4.18}$$

$$p := \min_{i\in[n]} Pr[\text{ALG accepts } X_i \mid X_i \geq T]. \tag{4.19}$$

Algorithm 9: Single-threshold with coins k-uniform prophet problem.

Parameters: Threshold T, biases $b_1, \ldots, b_n \in [0, 1]$.

$s \leftarrow 0$; `// # of occupied slots`

for $i = 1, \ldots, n$ **do**

  if $X_i \geq T$ and $s < k$ **then**

    Toss a b_i-biased coin.

    if *Coin comes up Heads* **then**

      Accept X_i.

      $s \leftarrow s + 1$.

    else

      Reject X_i.

    end

  else

    Reject X_i.

  end

end

Then,

$$\mathbb{E}[\text{ALG}] \geq h \cdot T + p \cdot \sum_{i=1}^{n} \mathbb{E}[(X_i - T)^+].$$

Proof.

$$\begin{aligned}
\mathbb{E}[\text{ALG}] &= \mathbb{E}\left[\sum_{i=1}^{n} X_i \cdot \mathbb{I}[\text{ALG accepts } X_i]\right] \\
&= \mathbb{E}\left[\sum_{i=1}^{n} T \cdot \mathbb{I}[\text{ALG accepts } X_i] + \sum_{i=1}^{n} (X_i - T) \cdot \mathbb{I}[\text{ALG accepts } X_i]\right] \\
&= T \cdot \mathbb{E}[\#\text{ of elements accepted}] + \sum_{i=1}^{n} \mathbb{E}[(X_i - T) \cdot \mathbb{I}[\text{ALG accepts } X_i]] \\
&= h \cdot T + \sum_{i=1}^{n} \mathbb{E}[(X_i - T) \cdot \mathbb{I}[\text{ALG accepts } X_i] \mid X_i \geq T] \cdot \Pr[X_i \geq T] \\
&= h \cdot T + \sum_{i=1}^{n} \mathbb{E}[(X_i - T) \mid X_i \geq T] \cdot \Pr[X_i \geq T] \cdot \Pr[\text{ALG accepts } X_i \mid X_i \geq T] \\
&= h \cdot T + \sum_{i=1}^{n} \mathbb{E}[(X_i - T)^+] \cdot \Pr[\text{ALG accepts } X_i \mid X_i \geq T] \\
&\geq h \cdot T + p \cdot \sum_{i=1}^{n} \mathbb{E}[(X_i - T)^+]
\end{aligned}$$

□

We also prove the following upper bound on the value of the prophet which is a generalization of Inequality (A.11) for the k-uniform setting.

Lemma 21. *Let $k \leq n$ be positive integers and let $\mathcal{I} = (\mathcal{F}_i)_{i\in[n]}$ be any instance of the k-uniform prophet problem. Define*

$$\mathrm{OPT} := \max_{S\subseteq[n],|S|=k} \sum_{i\in S} X_i$$

to be the optimal value achievable by a prophet on instance $\mathcal{I}$. Then, for all $T \geq 0$,

$$\mathbb{E}[\mathrm{OPT}] \leq k \cdot T + \sum_{i=1}^{n} \mathbb{E}[(X_i - T)^+].$$

Proof.

$$\begin{aligned}\mathbb{E}[\mathrm{OPT}] &= \mathbb{E}\left[\max_{S\subseteq[n],|S|=k} \sum_{i\in S} X_i\right] = k \cdot T + \mathbb{E}\left[\max_{S\subseteq[n],|S|=k} \sum_{i\in S} (X_i - T)\right] \\ &\leq k \cdot T + \mathbb{E}\left[\max_{S\subseteq[n],|S|=k} \sum_{i\in S} (X_i - T)^+\right] \leq k \cdot T + \sum_{i=1}^{n} \mathbb{E}[(X_i - T)^+]\end{aligned}$$

□

Lemmas 20 and 21 imply the prophet inequality of the form

$$\mathbb{E}[\mathrm{ALG}] \geq \min\left(\frac{h}{k}, p\right) \cdot \mathbb{E}[\mathrm{OPT}] \tag{4.20}$$

for Algorithm 9 for the k-uniform prophet problem where h, p are defined as in Equations (4.18) and (4.19) and depend on the choice of threshold T and biases $b_1, \ldots, b_n$.

Consider now the following parameterization of Algorithm 9:

$$T \text{ defined such that: } \sum_{i=1}^{n} \Pr[X_i \geq T] = k - \sqrt{2k \ln k} \tag{4.21}$$

$$(b_i)_{i\in[n]} \text{ defined such that: } b_i = \frac{1 - \sqrt{\frac{2\ln k}{k}}}{1 - \Pr[S_i^k]},\ \forall i \in [n], \tag{4.22}$$

where S_i^r is the event defined in Section 4.8.1 as,

$$S_i^r = [\text{ALG accepted exactly } r \text{ elements before inspecting } X_i].$$

Notice that T is well-defined assuming continuous[3] distributions: the expression on the left is a continuous, non-increasing function of T taking the value n for $T = 0$ and approaching 0 as $T \to \infty$, and since $k - \sqrt{2k \ln k} < k \leq n$, the intermediate value theorem implies the existence of a T satisfying Equation (4.21).

Lemma 22. *The system of equations (4.22) has a unique solution which defines valid biases $b_i \in [0, 1]$ for all $i \in [n]$ as long as k is large enough to satisfy $k \geq 8 \ln k$ (e.g. $k \geq 27$).*

Proof. Each of the events S_i^k depends only on the coins tossed before inspecting X_i and therefore $\Pr[S_i^k]$ is a function of only the first $i - 1$ biases $b_1, \ldots, b_{i-1}$ — as well as the value of the threshold T and the distributions $\mathcal{F}_1, \ldots, \mathcal{F}_{i-1}$. Hence, the unique solution to the equations (4.22) is given by computing the values in order: $0 = \Pr[S_1^k], b_1, \Pr[S_2^k], b_2, \ldots, \Pr[S_n^k], b_n$.

For the second part of the lemma, notice that the expression on right side of Equation (4.22) is in the interval $[0, 1]$ precisely when

$$\Pr[S_i^k] \leq \sqrt{\frac{2 \ln k}{k}} \leq 1. \tag{4.23}$$

The assumptions that $k \geq 8 \ln k$ implies the second inequality as follows:

$$\sqrt{\frac{2 \ln k}{k}} \leq \frac{1}{2} < 1.$$

For the first inequality in line (4.23), since the event S_i^k implies S_j^k for all $j \geq i$, we have $\Pr[S_n^k] = \max_{i \in [n]}\{\Pr[S_i^k]\}$ so we just need to show

$$\Pr[S_n^k] \leq \sqrt{\frac{2 \ln k}{k}}.$$

[3]For discrete distributions, a trick similar to the one described in Appendix A.2.2 can be used to reduce the problem to the continuous case.

We now use the fact that S_n^k is a subset of the event $[N \geq k]$ where $N = |\{j \in [n] \mid X_j \geq T\}|$. Reasoning about the latter event is more convenient because the Chernoff bound applies to it, and it is not influenced by any of the algorithm's choices of parameters other than T. In particular, for our choice of T defined in Equation (4.21) we have

$$\mu = \mathbb{E}[N] = \sum_{j=1}^{n} \Pr[X_j \geq T] = k - \sqrt{2k \ln k}.$$

By the Chernoff bound [74, Inequality (4.2)],

$$\Pr[N \geq k] = \Pr[N \geq (1+\delta)\mu] \leq \exp\left(-\tfrac{1}{3}\delta^2\mu\right),$$

where δ is such that $(1+\delta)\mu = k$,

$$\delta = \frac{k}{k - \sqrt{2k \ln k}} - 1 = \frac{\sqrt{2k \ln k}}{k - \sqrt{2k \ln k}}.$$

Notice that $\delta \in (0, 1]$ where the upper bound is a consequence of the assumption that $k \geq 8 \ln k$. Rewriting the bound, we have

$$\begin{aligned}
\Pr[S_i^k] \leq \Pr[N \geq k] &\leq \exp\left(-\frac{1}{3} \cdot \frac{2k \ln k}{(k - \sqrt{2k \ln k})^2} \cdot (k - \sqrt{2k \ln k})\right) \\
&= \exp\left(-\frac{2}{3} \frac{k}{k - \sqrt{2k \ln k}} \cdot \ln k\right) \\
&\leq \exp\left(-\frac{2}{3} \ln k\right) = k^{-2/3} \leq \sqrt{\frac{2 \ln k}{k}},
\end{aligned}$$

where the last inequality holds for large enough k as needed. □

Theorem 21. *For large enough k, Algorithm 9 parameterized as shown in Equations* (4.21) *and* (4.22) *satisfies the IIF property and further satisfies the following prophet inequality,*

$$\mathbb{E}[\mathrm{ALG}] \geq \left(1 - O\left(\sqrt{\frac{\ln k}{k}}\right)\right) \cdot \mathbb{E}[\mathrm{OPT}].$$

Proof. The IIF property easily follows from Equation (4.22). For any $x \geq T$,

$$\begin{aligned}
\Pr[\text{ALG accepts } X_i \,|X_i = x] &= \sum_{s=0}^{k-1} \underbrace{\Pr[\text{ALG accepts } X_i \mid X_i = x \cap S_i^s]}_{b_i} \cdot \Pr[S_i^s] \\
&= b_i \cdot \sum_{s=0}^{k-1} \Pr[S_i^s] \\
&= b_i \cdot (1 - \Pr[S_i^k]) \\
&= 1 - \sqrt{\frac{2 \ln k}{k}}.
\end{aligned}$$

Therefore,

$$\Pr[\text{ALG accepts } X_i \mid X_i = x] = \begin{cases} 1 - \sqrt{\frac{2 \ln k}{k}}, & x \geq T \\ 0, & \text{otherwise} \end{cases}. \tag{4.24}$$

Recall we defined $p = \min_{i \in [n]} \Pr[\text{ALG accepts } X_i \mid X_i \geq T]$ in Lemma 20. In this case,

$$p = 1 - \sqrt{\frac{2 \ln k}{k}}. \tag{4.25}$$

To bound h, the expected number of elements accepted by Algorithm 9, we use the definition of the threshold T in Equation (4.21),

$$\begin{aligned}
h &= \mathbb{E}[\text{\# of elements accepted}] \\
&= \sum_{i=1}^{n} \Pr[X_i \geq T] \cdot \Pr[\text{ALG accepts } X_i \mid X_i \geq T] \\
&= p \cdot \sum_{i=1}^{n} \Pr[X_i \geq T] \\
&= p \cdot \mu,
\end{aligned}$$

where $\mu = k - \sqrt{2k \ln k}$.

Using Equation (4.20),

$$\begin{aligned}\mathbb{E}[\mathrm{ALG}] &\geq \min\left(\frac{h}{k}, p\right) \cdot \mathbb{E}[\mathrm{OPT}] = \min\left(\frac{\mu}{k}, 1\right) \cdot p \cdot \mathbb{E}[\mathrm{OPT}] \\ &\overset{\mu<k}{=} \frac{\mu \cdot p}{k} \cdot \mathbb{E}[\mathrm{OPT}] = \left(1 - \sqrt{\frac{2\ln k}{k}}\right)^2 \cdot \mathbb{E}[\mathrm{OPT}] \\ &= \left(1 - O\left(\sqrt{\frac{\ln k}{k}}\right)\right) \cdot \mathbb{E}[\mathrm{OPT}].\end{aligned}$$

□

Remark 5. Algorithm 9 can be adapted to any arrival order π without affecting Equation (4.24), which means that the family of algorithms defined by Algorithm 9 when π ranges over all arrival orders, satisfies both the IIF and TIF properties.

CHAPTER 5

REVENUE MONOTONICITY UNDER MISSPECIFIED BIDDERS

In this chapter we focus on Mechanism Design and we examine revenue guarantees for optimal mechanisms when a subset of bidders' value distributions are misspecified, but the auctioneer doesn't know which of the distributions are incorrect. Our model is inspired by the literature on *semi-random adversaries* in the theoretical computer science literature, particularly the work of Bradac et al. [19] on robust algorithms for the secretary problem, as well as research in the systems community of Computer Science studying *Byzantine fault-tolerance*. In the model we investigate here, the auctioneer is given (not necessarily identical) distributions for each of n bidders. An unknown subset of the bidders, called the *green bidders*, draw their values independently at random from these distributions. The other bidders, called the *red bidders*, draw their values from distributions other than the given ones.

We provide two interpretations for a setting like this. One is that the red bidders are standard strategic agents for which the mechanism designer holds inaccurate information and thus is designing the mechanism under the wrong assumptions. Another way of looking at this setting is that the red bidders are "Byzantine" in the sense that they behave in an arbitrary way. For example, in an automated auction where the bidders are computer programs, a faulty bidder might submit bids going against the bidder's incentives.

The question we ask is, "When can one guarantee that the expected revenue of the optimal mechanism for the given distributions is at least as great as the expected revenue that would be obtained by excluding the red bidders and running an optimal mechanism on the green subset of bidders?" In other words, can the presence of bidders with misspecified distributions in a market be worse (for the auctioneer's expected revenue) than

if those bidders were absent? Or does the increased competition from incorporating the red bidders always offset the revenue loss due to ascribing the wrong distribution to them?

We give a precise answer to this question, for the Bayesian single-parameter environments. We show that the answer depends on the structure of the feasibility constraint that defines which sets of bidders may win the auction. For matroid feasibility constraints, the revenue of the optimal mechanism is always greater than or equal to the revenue obtained by running the optimal mechanism on the set of green bidders. For any feasibility constraint that is not a matroid, the opposite holds true: there is a way of setting the specified distributions and the true distributions such that the revenue of the optimal mechanism for the specified distributions, when bids are drawn from the true distributions, is *strictly less* than the revenue of the optimal mechanism on the green bidders only.

The economic intuition behind this result is fairly easy to explain. The matroid property guarantees that the winning red bidders in the auction can be put in one-to-one correspondence with losing green bidders who would have won in the absence of their red competitors, in such a way that the revenue collected from each winning red bidder offsets the lost revenue from the corresponding green bidder whom he or she displaces. When the feasibility constraint is not a matroid, this one-to-one correspondence does not always exist; a single green bidder might be displaced by two or more red bidders each of whom pays almost nothing. The optimal mechanism allows this to happen at some bid profiles, because the low revenue received on such bid profiles is compensated by the high expected revenue that would be received if the red bidders had sampled values from elsewhere in their distributions. However, since the red bidders' distributions are misspecified, the anticipated revenue from these more favorable bid profiles may never

materialize.

Our result can be interpreted as a type of revenue monotonicity statement for optimal mechanisms in Bayesian single-parameter matroid environments. However it does not follow from other known results on revenue monotonicity, and it is illuminating to draw some points of distinction between our result and earlier ones. Let us begin by distinguishing *pointwise* and *setwise* revenue monotonicity results: the former concern how the revenue earned on individual bid profiles varies as the bids are increased, the latter concern how (expected) revenue varies as the set of bidders is enlarged.

- VCG mechanisms are neither pointwise nor setwise revenue monotone in general, but in Bayesian single-parameter matroid environments, VCG revenue satisfies both pointwise and setwise monotonicity. In fact, Dughmi, Roughgarden, and Soundararajan [37] observed that VCG revenue obeys setwise monotonicity *if and only if* the feasibility constraint is a matroid.
- Myerson's optimal mechanism is not pointwise revenue monotone, even for single-item auctions. For example, consider using Myerson's optimal mechanism to sell a single item to Alice whose value is uniformly distributed in $[0, 4]$ and Bob whose value is uniformly distributed in $[0, 8]$. When Alice bids 0 and Bob bids 5, Bob wins and pays 4. If Alice increases her bid to 4, she wins but pays only 3.
- However, Myerson's optimal mechanism is *always* setwise revenue monotone in Bayesian single-parameter environments with downward-closed feasibility constraints, regardless of whether the feasibility constraint is a matroid. This is because the mechanism's expected revenue is equal to the expectation of the maximum, over all feasible sets of winners, of the winners' combined ironed virtual value. Enlarging the set of bidders only enlarges the collection of sets over which this maximization is performed, hence it cannot decrease the expectation of the

maximum.

Our main result is analogous to the setwise revenue monotonicity of Myerson revenue, except that we are considering monotonicity with respect to the operation of enlarging the set of bidders *by adding bidders whose value distributions are potentially misspecified.* We show that the behavior of Myerson revenue with respect to this stricter notion of setwise revenue monotonicity holds under matroid feasibility constraints *but not under any other feasibility constraints*, in contrast to the traditional setwise revenue monotonicity that is satisfied by Myerson mechanisms under arbitrarily downward-closed constraints.

5.1 Related Work

Semi-random models are a class of models studied in the theoretical computer science literature in which the input data is partly generated by random sampling, and partly by a worst-case adversary. Initially studied in the setting of graph coloring [14] and graph partitioning [41, 72], the study of semi-random models has since been broadened to statistical estimation [36, 65], multi-armed bandits [70], and secretary problems [19]. Our work extends semi-random models into the realm of Bayesian mechanism design. In particular, our model of green and red bidders resembles in a sense that of Bradac el al. [19] for the secretary problem which served as inspiration for this work. In both settings, green players/elements behave randomly and independently while red players/elements behave adversarially. In the secretary model of [19], red elements can choose arbitrary arrival times while green elements' arrival times are i.i.d. uniform in $[0, 1]$ and independent of the red arrival times. Similarly, in our setting red bidders can set their bids arbitrarily whereas green bidders sample their bids from known distribu-

tions, independently of the red bidders and one another.

Our work can be seen as part of a general framework of *robust mechanism design*, a research direction inspired by Wilson [96], who famously wrote,

> Game theory has a great advantage in explicitly analyzing the consequences of trading rules that presumably are really common knowledge; it is deficient to the extent it assumes other features to be common knowledge, such as one agent's probability assessment about another's preferences or information. I foresee the progress of game theory as depending on successive reductions in the base of common knowledge required to conduct useful analyses of practical problems. Only by repeated weakening of common knowledge assumptions will the theory approximate reality.

This *Wilson doctrine* has been used to justify more robust solution concepts such as dominant strategy and ex post implementation. The question of when these stronger solution concepts are required in order to ensure robustness was explored in a research program initiated by Bergemann and Morris [9] and surveyed in [10]. Robustness and the Wilson doctrine have also been used to justify prior-free [42] and prior-independent [50] mechanisms as well as mechanisms that learn from samples [21, 29, 35, 54, 75, 23, 20]. A different approach to robust mechanism design assumes that, rather than being given the bid distributions, the designer is given constraints on the set of potential bid distributions and aims to optimize a minimax objective on the expected revenue. For example Azar and Micali [6] assume the seller knows only the mean and variance of each bidder's distribution, Carrasco et al. [24] generalize this to sellers that know the first N moments of each bidder's distribution, Azar et al. [5] consider sellers that know the median or other quantiles of the distributions, and Carroll [25] introduced a model in which bids are correlated but the seller only knows each bidder's marginal distribution (see [43, 8] for fur-

ther work in this correlation-robust model). Bergemann and Schlag [11] develop mechanisms for the single-item/single-bidder setting with robust revenue guarantees when the seller is assumed to be given a distribution which lies in a small neighborhood of the true distribution. Brustle, Cai and Daskalakis [20] and Cai and Daskalakis [23], as part of their work, derive mechanism robustification results (where robustness is defined in a similar sense to [11]) for more general settings.

Another related subject is that of *revenue monotonicity* of mechanisms — regardless of the existence of adversarial bidders. Dughmi et al. [37] prove a result very close in spirit to ours. They consider the VCG mechanism in a Bayesian single-parameter downward-closed environment and prove that it is revenue monotone if and only if the environment is a matroid akin to our Theorems 22 and 23. Devanur et al. [34] prove that optimal auction revenue is monotone under first-order stochastic dominance, a result which they apply to the study of sample complexity in auction revenue maximization. Rastegari et al. [83] study revenue monotonicity properties of mechanisms (including VCG) for Combinatorial Auctions. Under some reasonable assumptions, they prove that no mechanism can be revenue monotone when bidders have single-minded valuations. Chen, Li, and Li [28] use a type of *reverse* setwise revenue monotonicity — the revenue that optimal mechanisms extract from a fixed set of bidders is non-increasing as other bidders join the auction — to derive revenue approximation guarantees in an information elicitation setting where knowledge about the players' distributions is scattered among the players and the seller is trying to both elicit this knowledge and sell (multiple) item(s).

5.2 Revenue Monotonicity on Matroid Markets

We extend the standard single-parameter environment to allow for bidders with misspecified distributions. Formally, the n bidders are partitioned into sets G and R; the former are called *green* and the latter *red*. The color of each bidder (green or red) is not revealed to the mechanism designer at any point. Green bidders sample their values from their respective distribution $\mathcal{F}_i$ but red bidders are sampling $v_i \sim \mathcal{F}_i'$ for some $\{\mathcal{F}_i'\}_{i \in R}$ which are completely unknown to the mechanism designer and can be adversarially chosen.

In this section we are interested in studying the behavior of Myerson's optimal mechanism when designed under the (wrong) assumption that $v_i \sim \mathcal{F}_i$ for all $i \in [n]$. Specifically, we ask the question of whether the existence of the red bidders could harm the expected revenue of the seller compared to the case where the seller was able to identify and exclude the red bidders, thus designing the optimal mechanism for the green bidders alone. The following definition makes this notion of revenue monotonicity more precise.

Definition 14 (RMMB)**.** Consider a single-parameter, downward-closed market $\mathcal{M} = (E, \mathcal{I})$ of $|E| = n$ bidders. A mechanism $\mathcal{A}$ is *Revenue Monotone under Misspecified Bidders (RMMB)* if for any distributions $\mathcal{F}_1, \ldots, \mathcal{F}_n$, any number $1 \leq k \leq n$ of green bidders and any fixed misspecified bids $\mathbf{b}_R \in \mathbb{R}^R$ of the red bidders

$$\mathbb{E}\left[\mathrm{Rev}(\mathcal{A}(\mathbf{b}_G, \mathbf{b}_R))\right] \geq \mathbb{E}\left[\mathrm{Rev}(\mathcal{A}(\mathbf{b}_G))\right], \tag{5.1}$$

where both expectation are taken over $\mathbf{b}_G \sim \prod_{i \in G} \mathcal{F}_i$.

An alternative definition of the revenue monotonicity property allows red bidders to have stochastic valuations drawn from distributions $\mathcal{F}_i' \neq \mathcal{F}_i$ instead of fixed bids. We note that the two definitions are equivalent: if $\mathcal{A}$ is RMMB according to Definition 14

then inequality (5.1) holds point-wise for any fixed misspecified bids and thus would also hold in expectation. For the other direction, if inequality (5.1) holds in expectation over the red bids, regardless of the choice of distributions $\{\mathcal{F}'_i \mid i \in R\}$ then we may specialize to the case when each $\mathcal{F}'_i$ is a point-mass distribution with a single support point b_i for each $i \in R$, and then Definition 14 follows.

In what follows we assume bidders always submit bids that fall within the support of their respective distribution. Green bidders obviously follow that rule and red bidders should do as well, otherwise the mechanism could recognize they are red and just ignore them.

Consider first the simpler case of selling a single item. This corresponds to a uniform rank-1 matroid market. Intuitively when the item is allocated to a green bidder, the existence of the red bidders is not problematic and in fact could help increase the critical bid and thus the payment of the winner. On the other hand, when a red bidder wins one has to prove that they are not charged too little and thus risk bringing the expected revenue down.

Let $m = \max(\max_{i \in G} \phi_i(b_i), 0)$ be the random variable denoting the highest non-negative virtual value in the set of green bidders. Let also X be an indicator a random variable which is 1 if the winner belongs to G and Y denote an indicator random variable which is 1 if the winner belongs to R. For the optimal mechanism MyerOPT (see Algorithm 2 defined in Section 2.4.2), we have

$$\mathbb{E}\left[\text{revenue from green bidders}\right] = \mathbb{E}\left[m \cdot X\right] \tag{5.2}$$

$$\mathbb{E}\left[\text{revenue from red bidders}\right] \geq \mathbb{E}\left[m \cdot Y\right], \tag{5.3}$$

where (5.2) follows from Myerson's lemma and (5.3) follows from the observation that the winner of the optimal auction never pays less than the second-highest virtual value.

To see why the latter holds, let ϕ_s be the second highest virtual value, r be the red winner and g is the green player with the highest virtual value. The critical bid of the red winner is at least

$$\phi_r^{-1}(\phi_s) \geq \phi_r^{-1}(\phi_g(b_g)) \geq \phi_g(b_g),$$

where we applied the fact that $x \geq \phi(x)$ to the virtual value function $\phi = \phi_r$ and the value $x = \phi_r^{-1}(\phi_g(b_g))$. Summing (5.2) and (5.3) and using the fact that $X + Y = 1$ whenever $m > 0$, we find

$$\begin{aligned}\mathbb{E}\left[\text{Revenue from all bidders } 1, \ldots, n\right] &\geq \mathbb{E}\left[m \cdot X\right] + \mathbb{E}\left[m \cdot Y\right] \\ &= \mathbb{E}\left[m\right] \\ &= \mathbb{E}\left[\text{revenue of MyerOPT on } G\right].\end{aligned}$$

We therefore conclude that Myerson's optimal mechanism is RMMB in the single-item case. We are now ready to generalize the above idea to any matroid market.

Theorem 22. *Let $\mathcal{M} = (E, \mathcal{I})$ be any matroid market. Then MyerOPT in $\mathcal{M}$ is RMMB.*

Proof. Call G the set of green bidders and R the set of red bidders. Let (x, p) denote the allocation and payment rules for the mechanism MyerOPT that runs Myerson's optimal mechanism on all n bidders, using the given distribution of each. Let (x', p') denote the allocation and payment rules for the mechanism MyerOPT_G that runs Myerson's optimal mechanism in the bidder set G only. For a set $S \subseteq [n]$, let T_S be the random variable denoting the independent subset of S that maximizes the sum of ironed virtual values. In other words, T_S is the set of winners chosen by Myerson's optimal mechanism on bidder set S. By Myerson's Lemma, the revenue of MyerOPT_G satisfies

$$\mathbb{E}\left[\sum_{i \in G} p_i'(\mathbf{b})\right] = \mathbb{E}\left[\sum_{i \in G} x_i'(\mathbf{b}) \cdot \phi_i(b_i)\right]. \tag{5.4}$$

By linearity of expectation, we can break up the expected revenue of MyerOPT into two terms as

$$\mathbb{E}\left[\sum_{i\in[n]} p_i(\mathbf{b})\right] = \mathbb{E}\left[\sum_{i\in G} p_i(\mathbf{b})\right] + \mathbb{E}\left[\sum_{i\in R} p_i(\mathbf{b})\right]. \tag{5.5}$$

The first term on the right side of (5.5) expresses the revenue originating from the green bidders. Using Myerson's Lemma, we can equate this revenue with the expectation of the green winners' combined virtual value

$$\mathbb{E}\left[\sum_{i\in G} p_i(\mathbf{b})\right] = \mathbb{E}\left[\sum_{i\in G} x_i(\mathbf{b}) \cdot \phi_i(b_i)\right]. \tag{5.6}$$

To express the revenue coming from the red bidders in terms of virtual valuations, we provide the argument that follows. One way to derive T_{G+R} from T_G is to start with T_G and sequentially add elements of $T_{G+R} \cap R$ in arbitrary order while removing at each step the least weight element in the circuit that potentially forms (repeated application of Proposition 2). Let e be the i-th red element we're adding. If no circuit forms after the addition, then e pays its critical bid $\phi_e^{-1}(0)$ which is some non-negative quantity. Otherwise, let C be the unique circuit that forms after that addition. Let f be the minimum weight element in C and let b_f be the associated bid made by player f. Notice that f must be green; by assumption, every red element we're adding is part of the eventual optimal solution so it cannot be removed at any stage of this process. The price charged to e is their critical bid which we claim is at least $\phi_e^{-1}(\phi_f(b_f))$. The reason is that e is part of circuit C and f is the min-weight element of that circuit. The min-weight element of a circuit is never in the max-weight independent set[1] so if bidder e bids any value v such that $\phi_e(v) < \phi_f(b_f)$ they will certainly *not* be included in the set of winners, T_{G+R}. By Proposition 3 it follows that

$$\phi_e^{-1}(\phi_f(b_f)) \geq \phi_f(b_f),$$

[1]This is a consequence of the optimality of the Greedy algorithm since the min-weight element of a circuit is the last to be considered among the elements of the circuit and its inclusion will violate independence.

thus

$$p_e(\mathbf{b}) \geq \phi_f(b_f).$$

The above reasoning allows us "charge" each red bidder's payment to a green player's virtual value in $T_G \setminus T_{G+R}$:

$$\mathbb{E}\left[\sum_{i\in R} p_i(\mathbf{b})\right] \geq \mathbb{E}\left[\sum_{i\in T_G\setminus T_{G+R}} \phi_i(b_i)\right] = \mathbb{E}\left[\sum_{i\in G}(x_i'(\mathbf{b}) - x_i(\mathbf{b}))\cdot\phi_i(b_i)\right]. \tag{5.7}$$

The equality is justified by observing that for $i \in G$, $x_i'(\mathbf{b}) = x_i(\mathbf{b})$ unless $i \in T_G \setminus T_{G+R}$, in which case $x_i'(\mathbf{b}) - x_i(\mathbf{b}) = 1$. Combining Inequalities/Equations (5.4)–(5.7) we get

$$\begin{aligned}\mathbb{E}\left[\sum_{i\in[n]} p_i(\mathbf{b})\right] &\geq \mathbb{E}\left[\sum_{i\in G} x_i(\mathbf{b})\cdot\phi_i(b_i)\right] + \mathbb{E}\left[\sum_{i\in G}(x_i'(\mathbf{b}) - x_i(\mathbf{b}))\cdot\phi_i(b_i)\right] \\ &= \mathbb{E}\left[\sum_{i\in G} x_i'(\mathbf{b})\cdot\phi_i(b_i)\right] = \mathbb{E}\left[\sum_{i\in G} p_i'(\mathbf{b})\right].\end{aligned}$$

In other words, the expected revenue of MyerOPT is greater than or equal to that of MyerOPT_G. □

5.3 General Downward-Closed Markets

When the market is not a matroid, the existence of red bidders can do a lot of damage to the revenue of the mechanism as shown in the following simple example.

Example 3. Consider a 3-element downward-closed set system on $E = \{a, b, c\}$ with maximal feasible sets: $\{a, b\}$ and $\{c\}$. Let c be a green bidder with a deterministic value of 1 and a, b be red bidders each with a specified value distribution given by the following cumulative distribution function

$$F(x) = 1 - (1 + x)^{1-\theta},$$

for some parameter $\theta \geq 3$. Note that the associated virtual value function is

$$\phi(x) = \left(1 - \frac{1}{\theta - 1}\right)x - \frac{1}{\theta - 1}.$$

For this virtual value function we have

$$\phi^{-1}(0) = \frac{1}{\theta - 2},$$
$$\phi^{-1}(1) = \frac{\theta}{\theta - 2}.$$

Consider the revenue of Myerson's mechanism when the red bidders, instead of following their specified distribution, each bid $\phi^{-1}(1)$ — and the green bidder bids 1, the only support point of their distribution. The set $\{a, b\}$ wins over $\{c\}$ since the former sums to a total virtual value of 2 over the latter's virtual value 1 so bidders a, b pay their critical bid.

To compute that, notice that each of the bidders a, b could unilaterally decrease their bid to any $\varepsilon > \frac{1}{\theta-2}$ and they would still win the auction since the set $\{a, b\}$ would still have a total virtual value greater than 1. Therefore, each of a, b pays $\frac{1}{\theta-2}$ for a total revenue of $\frac{2}{\theta-2}$. On the other hand, the same mechanism when run on the set $\{c\}$ of only the green bidder, always allocates an item to c and collects a total revenue of 1.

Letting $\theta \to \infty$ we see that the former revenue tends to zero while the latter remains 1, violating the revenue monotonicity property of Definition 14 by an unbounded multiplicative factor.

To generalize the above idea to any non-matroid set system we need the following lemma.

Lemma 23. *A downward-closed set system $\mathcal{S} = (E, \mathcal{I})$ is* not *a matroid if and only if there exist $I, J \in \mathcal{I}$ with the following properties:*

1. *For every $K \in \mathcal{I}|_{I \cup J}$, if $|K| \geq |I|$ then $K \supseteq I \backslash J$.*
2. *$|J \backslash I| \geq 1$.*

3. *I is a maximum cardinality element of $\mathcal{I}|_{I\cup J}$.*

Proof. For the forward direction, suppose $\mathcal{S}$ is *not* a matroid and let V be a minimum-cardinality subset of E that is not a matroid. Since $\mathcal{I}|_V$ is downward-closed and non-empty, it must violate the exchange axiom. Hence, there exist sets $I, J \in \mathcal{I}|_V$ such that $|I| > |J|$ but $J + x \notin \mathcal{I}$ for all $x \in I\backslash J$. Note that $V = I \cup J$, since otherwise $I \cup J$ is a strictly smaller subset of E satisfying the property that $(I \cup J, \mathcal{I}|_{I\cup J})$ is not a matroid.

Observe that J is a maximal element of $\mathcal{I}_V$. The reason is that $V = I \cup J$, so every element of $V\backslash J$ belongs to I. By our assumption on the pair I, J, there is no element $y \in I$ such that $J + y \in \mathcal{I}|_V$. Since $\mathcal{I}|_V$ is downward-closed, it follows that no strict superset of J belongs to $\mathcal{I}|_V$.

We now proceed to prove that I, J satisfy the required properties of the lemma:

(1) Let $K \in \mathcal{I}_V$ with $|K| \geq |I|$. It follows that $|K| > |J|$, but J is maximal in $\mathcal{I}|_V$, so K and J must violate the exchange axiom. Thus, $\mathcal{I}|_{K\cup J}$ is not a matroid. By the minimality of V, this implies $K \cup J = V$ hence $K \supseteq I\backslash J$.

(2) If $J\backslash I = \emptyset$ then $J \subseteq I$ which contradicts the fact that I, J violate the exchange axiom.

(3) Suppose there exists $I' \in \mathcal{I}|_V$ with $|I'| > |I|$, then by property (1) we have $I' \supseteq I\backslash J$. Remove elements of $I\backslash J$ from I' one by one, in arbitrary order, until we reach a set $K \in \mathcal{I}_V$ such that $|K| = |I|$. This is possible because after the entire set $I\backslash J$ is removed from I', what remains is a subset of J, hence has strictly fewer elements than I. The set K thus constructed has $|K| = |I|$ but $K \not\supseteq I\backslash J$, violating property (1).

For the "only if" direction, supposing that $\mathcal{S}$ is a matroid, we must show that no $I, J \in \mathcal{I}$

satisfy all three properties. To this end, suppose I and J satisfy (2) and (3). Since $\mathcal{S}|_{I\cup J}$ is a matroid, there exists $K \supseteq J$ such that $K \in \mathcal{I}|_{I\cup J}$ and $|K| = |I|$. By property (2), we know that no $|I|$-element superset of J contains $I - J$ as a subset. Therefore, the set K violates property (1). □

We are now ready to generalize Example 3 to every non-matroid set system.

Theorem 23. *For any $\mathcal{M} = (E, \mathcal{I})$ which is* not *a matroid, MyerOPT is* not *RMMB.*

Proof. Consider a downward-closed $\mathcal{M} = (E, \mathcal{I})$ which is *not* a matroid. We are going to show there exists a partition of players into green and red sets and a choice of valuation distributions and misspecified red bids such that the RMMB property is violated.

Let $I, J \subseteq E$ be the subsets whose existence is guaranteed by Lemma 23. Define $G = J$ to be the set of green bidders, $R = I\backslash J$ to be the set of red bidders. All other bidders are irrelevant and can be assumed to be bidding zero. Set the value of each green bidder to be deterministically equal to 1. For each red bidder r, the specified value distribution has the same cumulative distribution function

$$F(x) = 1 - (1 + x)^{1-\theta},$$

for some $\theta > 0$ as defined in Example 3. Now let's consider the expected revenue of Myerson's mechanism when every bidder in R bids $\phi^{-1}(1)$.[2] Every bidder's virtual value is 1, so the mechanism will choose any set of winners with maximum cardinality which, according to Lemma 23, property (3), is $|I|$. For example, the set of winners could be I.

A consequence of Lemma 23, property (1) is that for every red bidder r there is no feasible set of bidders disjoint from $\{r\}$ with combined virtual value greater than $|I| - 1$.

[2]Members of G bid as well, but it hardly matters, because their bid is always 1 — the only support point of their value distribution — so the auctioneer knows their value without their having to submit a bid.

Thus each red bidder pays $\phi^{-1}(0)$. Elements of $I \cap J$ correspond to green bidders who win the auction and pay 1, because a green bidder pays 1 whenever they win. There are $|I \cap J|$ such bidders. Thus, the Myerson revenue is

$$|I \cap J| + \frac{1}{\theta - 2}|I \backslash J|.$$

The optimal auction on the green bidders alone charges each of these bidders a price of 1, receiving revenue

$$|J| = |I \cap J| + |J \backslash I|.$$

This exceeds $|I \cap J| + \frac{1}{\theta-2}|I \backslash J|$ as long as

$$(\theta - 2) \cdot |J \backslash I| > |I \backslash J|.$$

This inequality is satisfied, for example, when

$$\theta = |I \backslash J| + 3,$$

because $J \backslash I$ has at least one element (Lemma 23, property (2)). □

CHAPTER 6

ONLINE FLOW COMPUTATION

Consider the following online decision-making scenario in the context of computer networks. A social media platform allows its users to broadcast a video live-stream using their phone's camera. The platform owns a set of servers that can be used to receive the live-stream data before it is broadcast to the other users. As requests for initiating a live-stream arrive dynamically, the platform needs to make online decisions on how to match them with available servers in the geographic vicinity of the client while respecting the capacity constraints of the servers and the network. The primary consideration is to ensure that no client is ever denied the ability to start a live stream — assuming the total capacity of the servers/network can accommodate all the clients. To achieve this, the platform may need to dynamically reassign clients to different servers. But this comes at a cost which is experienced by the client as a short interruption in the live-stream so it is desirable for the number of such reassignments to be minimized.

Making a simplifying assumption that a user who starts a live-stream remains in the network indefinitely, we can model this situation with the *Online Bipartite Matching with Replacements problem* introduced in Section 2.5. The right side of the bipartite graph is the static set of servers, the left side is the set of clients arriving sequentially and an edge (u, v) indicates that client u is in the vicinity of server v and therefore a connection between them can be efficiently established.

In this chapter we take a natural next step in modeling this problem by incorporating the structure of the network in the model and allowing for multi-hop routes between clients and servers, while retaining the other key features of the online bipartite matching setting: servers are always present in the network, clients arrive over time and seek to send a unit of flow which persists permanently after their arrival, and this unit of flow

may be routed to any server. To stay consistent with network flow notation, we refer to the vertices that arrive dynamically as *sources* who are requesting to send one unit of flow to any vertex among a set of *sinks*. Other vertices besides sources and sinks may be present in the network, and we refer to them as *internal vertices*.

We quantify the cost of a sequence of flows by summing the lengths of the augmenting paths used to modify each flow to its successor, a quantity that we call the *flow switching cost*. This cost measure — which matches (up to a factor-of-two rescaling) the switching cost used in the analysis of the online bipartite matching with replacements problem — is motivated by two considerations. First, shifting flow from one routing path to another imposes overhead on the network: messages must be sent to each node whose next-hop changes and those nodes must change their internal state accordingly. The sum of augmenting path lengths quantifies the total number of such control messages and state changes. Second, as we discussed in the live-streaming application it is particularly costly for a source vertex (client) to change its next hop, as this may entail a disruption in the stream itself. The number of such disruptions is bounded above by the *total* number of vertices in the network that change their next hop, and hence upper bounds on the sum of augmenting path lengths imply upper bounds on the number of such disruptions.

6.1 Our results and techniques

The most natural algorithm for online matching, and more generally for online flow computation, is the greedy algorithm that modifies the flow in each time step so as to minimize the cost of switching from the preceding flow. Specialized to online matching, this corresponds to the Bipartite Shortest Augmenting Path (Bipartite-SAP) algorithm presented in Section 2.5. Here we study the aforementioned greedy algorithm that generalizes Bipartite-SAP to networks. To distinguish between these two algorithms, we

henceforth refer to the greedy algorithm for online flow computation as Network-SAP and we'll define it formally in Section 6.3.

Our main result, Theorem 25, pertains to a special case of online flow computation in which there is a (directed or undirected) network consisting of static vertices present at time zero, and dynamic vertices which arrive one by one. Edges between static vertices are also present at time zero, edges from a dynamic vertex to a static vertex arrive at the same time as the arrival of their dynamic endpoint, and edges between dynamic vertices are assumed not to exist in our model. There is a fixed set of static vertices known as *sinks*; each dynamic vertex is a *source* that requests, upon arriving into the network, to send one unit of flow which may be routed to any sink. We prove a $O(n \log^2 n)$ bound on the total flow switching cost in the unit-capacitated case when all vertices and edges have capacity 1, where n is the total number of vertices in the network.

The main tool that enables us to obtain this bound is a reduction from online flow computation in unit-vertex-capacitated graphs to online bipartite matching. Bernstein et al. [12] prove a $O(n_c \log^2 n_c)$ bound for the online bipartite matching, where n_c is the number of dynamic vertices (clients). This bound translates into a similar bound (up to a constant factor) for flow switching cost in our model. The reduction consists of creating a sequence of auxiliary bipartite graphs and matchings, one per time step, such that flow-augmenting paths in the online flow computation problem are in one-to-one correspondence with matching-augmenting paths in the auxiliary graphs. This correspondence, which preserves augmenting path length up to a factor-of-two rescaling, implies that there is a one-to-one correspondence between executions of the Network-SAP algorithm for our problem and executions of the Bipartite-SAP algorithm for the auxiliary problem. One wrinkle that arises in the reduction is that in order for the correspondence to hold at time zero, the auxiliary bipartite graph sequence must be initialized

with a non-empty matching. We observe in Section 6.4 below that the results of [12] continue to hold when one runs the SAP algorithm on a bipartite graph initialized with a non-empty matching.

With an eye toward extending our result beyond unit-vertex-capacitated networks, we devote Section 6.5 to presenting a generalization to a class of bipartite directed networks that includes the bipartite networks obtained by applying the gadget reduction in Section 6.4 to unit-vertex-capacitated networks. The generalization involves introducing structures that we call *semi-matchings* which generalize matchings, and showing that the $O(n_c \log^2 n_c)$ bound of Bernstein et al. [12] generalizes to executions of the SAP algorithm initialized with a semi-matching rather than a matching.

6.2 Related work

Reductions from flow problems to matching problems have appeared in prior works [68, 71] on offline algorithms for maximum flow and maximum matching. Our reduction, like Lin's reduction [68], uses a simple gadget that replaces each vertex with two vertices joined by an edge and initializes a bipartite matching problem with this set of "gadget edges". Our analysis of this reduction in the online setting, which involves coupling the executions of the online flow and matching algorithms and proving a suitable invariant of the coupling which ensures that augmenting paths remain in one-to-one correspondence as nodes join the network, is novel to the best of our knowledge.

Our theoretical investigation of minimizing reconfiguration changes in network routing is also partially inspired by, and intended to complement, the investigation of the same topic in the systems literature. Following the introduction of systems such as B4 [55] and SWAN [53] that use a centralized controller in conjunction with a solver to repeatedly

re-optimize the distribution of traffic over routing paths in a network, a few papers have investigated methods for reducing the rate of "churn" in such a system. SMORE [64] proposes to do so by restricting flows to a statically selected set of routing paths chosen using Räcke's [82] oblivious routing algorithm, and only using the solver to re-optimize the distribution of traffic over this limited set of paths. It is known that in the worst case, no such "semi-oblivious" routing scheme can achieve a constant-factor approximation to the optimum congestion unless it uses exponentially many paths [48]. SOL [51] supports generating new routing paths each time the solver runs, but it allows the user to specify constraints that bound (or minimize) the logical distance between the new configuration and its predecessor, to ensure that churn remains at a tolerable level. The same idea of limiting churn by bounding the logical distance between two consecutive configurations is used by Niagara [56] but at the level of entries in an individual switch's rule-table, rather than at the level of paths in a network.

6.3 Online Flow Computation

We are going to work on an *incremental online setting* in the framework of request-answer games presented in Section 2.1. There is an underlying, directed, capacitated network $G = (V, E)$ with two special sets of vertices $S, T \subseteq V$ called sources and sinks respectively. Sources arrive one at a time, requesting to send a unit amount of flow through the network to an arbitrary sink. The part of the graph containing the vertices $V \backslash S$ and the edges between them is known in advance. The new information that becomes available when a source arrives is the set of edges originating from that source. The *actions* available to the online algorithm allow modifying the currently maintained flow in an arbitrary way as long as the flow remains valid and all sources that have arrived so far are sending one unit of flow.

More formally, denote by $S = \{x_1, x_2, \ldots, x_{|S|}\}$ the set of sources. Then, for any integer $\tau \in [0 \ldots |S|] := \{0, 1, \ldots, |S|\}$, let G^τ be the induced subgraph of G on the vertex set $S^\tau \cup (V \backslash S)$ where $S^\tau = \{x_j \mid 1 \leq j \leq \tau\}$ is the set of the sources that have arrived by time τ. At any point, we have to maintain a valid flow of maximum value (*max flow*) in G^τ — assuming G is flow-admissible[1]. The cost incurred by an online algorithm for this problem is defined as follows.

Definition 15 (Flow-switching cost)**.** Let f_τ be the maximum flow maintained after the first $\tau < |S|$ sources have arrived and $f_{\tau+1}$ the flow maintained on the next iteration. We define the cost of switching from flow f_τ to flow $f_{\tau+1}$ as

$$\mathrm{SC}^G(f_\tau, f_{\tau+1}) = \sum_{e \in E(G)} |f_\tau(e) - f_{\tau+1}(e)|.$$

The total *flow-switching cost* of an algorithm $\mathcal{A}$ which maintains flows $f_0, f_1, \ldots, f_{|S|}$ is defined as

$$\mathrm{FSC}^G(\mathcal{A}) = \sum_{\tau=0}^{|S|-1} \mathrm{SC}^G(f_\tau, f_{\tau+1})$$

Our goal is to design competitive, online algorithms which in this setting reduces to minimizing the total flow-switching cost (see discussion on Online Algorithms with Recourse in Section 2.1).

Paths A *path* in G is represented as a sequence of edges. In the case of simple graphs (e.g. unit-vertex-capacitated ones), sequences of vertices uniquely identify a path. We'll sometimes assume P is represented as a sequence of vertices instead and we'll try to make it clear, based on the context, which definition of path we adopt at each point. In either case, the *length* of a path is defined as the number of *edges* in the edge-based representation.

[1] Recall the definition of a flow-admissible network given in Section 2.5 which is a flow network with a maximum flow value equal to $|S|$.

Network-SAP We are interested in analyzing the performance of the *Shortest Augmenting Path* algorithm (denoted as Network-SAP to distinguish it from Bipartite-SAP introduced in Chapter 2 in the context of online bipartite matching). The algorithm works as follows: when the τ-th source arrives, choose a shortest augmenting path in the residual graph $G^{\tau}_{f_{\tau-1}}$ from x_τ to a free sink (breaking ties arbitrarily) and update f by pushing 1 unit of flow along that path.

6.4 From Matchings to Flows

We now describe a reduction of the online flow computation problem on unit-vertex-capacitated networks to the online bipartite matching problem which will allow us to transfer the bound on the vertex-switching cost of the Bipartite-SAP algorithm to the flow-switching cost of the Network-SAP algorithm. This is achieved by proving an equivalence between augmenting paths in the two settings, meaning that Network-SAP and Bipartite-SAP are essentially the same algorithm on slightly different graphs.

The reduction is achieved in two steps. First, we apply a standard gadget reduction which replaces every internal vertex with a pair of vertices connected with an edge effectively replacing vertex capacities with edge capacities. A consequence of this is that the resulting graph H has a bipartite structure in which all sources are on one side and all sinks on the other. The second step is to notice that paths in H correspond to alternating paths in the undirected version $\overline{H}$ of H under an appropriate initial matching M_0, a correspondence that has also been noted by Lin in the context of reducing between *offline* flow and matching problems [68]. Furthermore this correspondence continues to hold at any time τ (as new sources arrive) between the residual graph of the modified network and the undirected version of the modified network under an appropriate matching M_τ.

Remark 6. For the rest of the section, we annotate some properties of the graph H with labels in the range **(P1)**–**(P6)**. The annotations are irrelevant to this section and can be ignored by the reader on a first reading. They become relevant in Section 6.5 when we generalize the reduction to any graph satisfying properties labeled **(P1)**–**(P6)**; we will then use the annotations to cross-reference arguments presented in this section.

Let G be a flow-admissible unit-vertex-capacitated network with $S, T \subseteq V(G)$ its sets of sources and sinks respectively. For the first step of the reduction, we define $H = (V, E)$ by substituting each vertex u with a "left" and a "right" counterpart (u, l) and (u, r)[2]. Edges are either "internal" (those within the gadget) or "external", corresponding to edges already existing in G. If (u, v) was an edge in G then $((u, l), (v, r))$ becomes an edge of H. The reduction is illustrated in Figure 6.1 and formalized as follows

$$S' = (S \times \{l\}), T' = (T \times \{r\}) \tag{6.1}$$

$$V(H) = S' \cup T' \cup ((V(G) \backslash (S \cup T)) \times \{l, r\}) \tag{6.2}$$

$$E_i = \{((u, r), (u, l)) : u \in V(G) \backslash (S \cup T)\} \tag{6.3}$$

$$E_o = \{((u, l), (v, r)) : (u, v) \in E(G)\} \tag{6.4}$$

$$E(H) = E_i \cup E_o \tag{6.5}$$

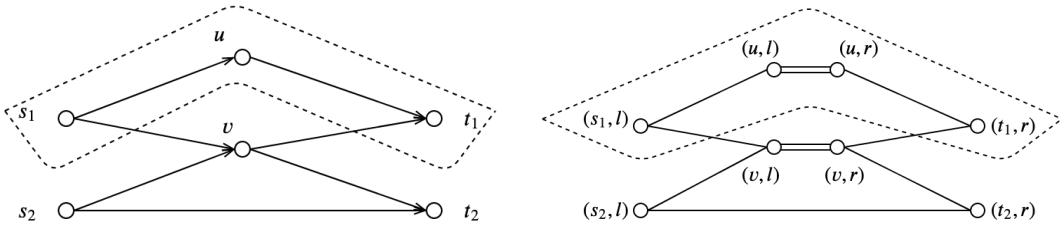
$$c(x) = \begin{cases} 1, & x \in S' \cup T' \cup E \\ +\infty, & \text{otherwise} \end{cases} . \tag{6.6}$$

For the second step, we define the undirected analogue of a graph as follows.

Definition 16. Given a directed network $H = (V, E)$, define $\overline{H} = (V, \overline{E})$ where $\overline{E} = \{\{u, v\} \mid (u, v) \in E\}$

The undirected analogue of a graph H as defined above satisfies some key properties which the rest of this section relies on. We list them below:

[2] l, r in this context are just labels we introduce to make the construction easier to define.



(a) G (b) $\overline{H}$

Figure 6.1: Here's an example of the gadget reduction. (6.1a) G is a directed flow network (edge and vertex capacities are assumed to be 1). $S = \{s_1, s_2\}, T = \{t_1, t_2\}$. An augmenting path in G is shown within the dotted set. (6.1b) $\overline{H}$ is the resulting undirected graph after we applied the gadget reduction and discarded the directions of the edges. Edges in M_0 are shown with double lines. The corresponding augmenting path is shown in a dotted set.

(H1) There is a bijection between edges of $\overline{H}$ and edges of H since H contains no 2-cycles **(P4)** and no parallel edges (by assumption that G is simple). Indeed, suppose there exist vertices $(u, l), (v, r)$ in H that form a 2-cycle. Then $((u, r), (v, l))$ must have been introduced in E_i. But that means $u = v$ and since the reverse edge is introduced in E_o it must be that $(u, u) \in E$ which contradicts the assumption that G is simple.

(H2) The residual graph of H under any valid, integral flow f has a special structure as a consequence of the unit edge capacities **(P3)** and the absence of parallel edges and 2-cycles **(P4)**. Namely, for every $e = (u, v) \in E(H)$, either $f(e) = 1$ in which case in place of e there is a backwards edge (v, u) in H_f or $f(e) = 0$ in which case e is in H_f. In either case there is only one edge between u, v in H_f at any time.

(H3) $\overline{H}$ is bipartite under the partition $V = L \cup R$ where $L = V(H) \cap (V(G) \times \{l\}), R = V(H) \cap (V(G) \times \{r\})$ and $S' \subseteq L, T' \subseteq R$ **(P1)**, **(P2)**. In the context of bipartite matchings, vertices in L play the role of clients, vertices in R the role of servers and S' is the set of clients arriving online.

As a reminder, we use the notation H^τ and $\overline{H}^\tau$ to denote the respective subgraphs at the

time when the first τ sources have arrived. The next step is to define the initial matching

$$M_0 = \{\{u, v\} \,|u \in L, v \in R, (v, u) \in E(H^0)\}. \tag{6.7}$$

To verify that M_0 is indeed a matching, notice that $M_0 = \overline{E}_i$ (the undirected analogue of the set E_i) which is a matching by construction. An example of the transformations described so far is shown in Figure 6.1.

Consider running Network-SAP starting on H^0 with an empty flow f_0 (i.e. $f_0(e) = 0$ for every $e \in E(H^0)$) and denote by f_τ the flow maintained by Network-SAP on H^τ. At the same time, consider running Bipartite-SAP starting on $\overline{H}^0$ with the initial matching M_0. Denote by M_τ the matching maintained by the algorithm on $\overline{H}^\tau$ at the end of iteration τ. Before we proceed any further, let us introduce some helpful notation.

Definition 17. Let $\phi_\tau, \psi_\tau, \chi_\tau, f_\tau : V \times V \to \{0, 1\}$ be defined on pairs (u, v) of adjacent vertices at time τ as follows

$$\phi_\tau(u, v) = \mathbb{I}[\{u, v\} \in M_\tau]$$
$$\psi_\tau(u, v) = \mathbb{I}[(u \in L) \text{ and } (v \in R)]$$
$$\chi_\tau(u, v) = \mathbb{I}[(u, v) \in E(H^\tau)],$$

where $\mathbb{I}[P]$ is the indicator function of property P.

For f_τ, abusing notation, we define it as the symmetric extension of the flow function $f_\tau : E(H^\tau) \to \{0, 1\}$, i.e. if $e = (u, v) \in E(H^\tau)$, define $f_\tau(u, v) = f_\tau(v, u) = f_\tau(e)$.

Definition 18 (Invariant (I))**.** We say that invariant (I) holds between a flow f_τ and matching M_τ at time τ if

$$\phi_\tau(u, v) \oplus \psi_\tau(u, v) \oplus \chi_\tau(u, v) \oplus f_\tau(u, v) = 0,$$

for all $u, v \in V$ such that $\{u, v\} \in E(\overline{H}^\tau)$.

The significance of the above invariant will become apparent from the following lemma which establishes an equivalence between augmenting paths in the network H and augmenting paths in the bipartite graph $\overline{H}$.

Lemma 24. *At any point τ, if invariant (I) holds between f_τ and M_τ then any sequence of vertices $P = v_0, \ldots, v_k$ that forms an augmenting path in $H^\tau_{f_\tau}$ also forms an augmenting path in $\overline{H}^\tau$ under M^τ and vice versa.*

Proof. Observe that a sequence of vertices $v_0, v_1, \ldots, v_k$ with $v_0 \in S'$ and $v_k \in T'$ constitutes an augmenting path in $\overline{H}^\tau$ if and only if

(i) v_0 and v_k are free with respect to M_τ,

(ii) for every consecutive pair $(u, v) = (v_i, v_{i+1})$ in the sequence, either $(u \in L, v \in R, \{u, v\} \notin M_\tau)$ or $(v \in L, u \in R, \{u, v\} \in M_\tau)$.

Similarly, $v_0, \ldots, v_k$ constitutes an augmenting path in $H^\tau_{f_\tau}$ if and only if

(iii) v_0 and v_k are free with respect to f_τ,

(iv) for every consecutive pair $(u, v) = (v_i, v_{i+1})$ in the sequence, either

$$e = (u, v) \in E(H^\tau) \text{ and } f_\tau(e) = 0,$$

or

$$e = (v, u) \in E(H^\tau) \text{ and } f_\tau(e) = 1.$$

Property (ii) is equivalent to $\phi_\tau(u, v) \oplus \psi_\tau(u, v) = 1$, while property (iv) is equivalent to $\chi_\tau(u, v) \oplus f_\tau(u, v) = 1$. Thus, assuming invariant (I), an ordered pair of vertices satisfies (ii) if and only if it satisfies (iv).

To conclude the proof we must show that a vertex v is free with respect to f_τ if and only if it is free with respect to M_τ. This, too, follows from invariant (I). We will present the argument assuming $v \in T'$, omitting the symmetric argument that pertains when $v \in S'$. If $v \in T'$ is not free in M_τ it means that M_τ contains an edge $\{u, v\}$. Observe that $u \in L$ and $v \in R$. This means $\phi_\tau(u, v) = \psi_\tau(u, v) = \chi_\tau(u, v) = 1$. Invariant (I) then implies $f_\tau(u, v) = 1$, from which it follows that v is not free in f_τ. Conversely, if v is not free in f_τ then $f_\tau(v) \neq 0$, hence there exists an edge $(u, v) \in H^\tau$ such that $f_\tau(u, v) = 1$. Observing that $u \in L$ and $v \in R$ we have $\psi_\tau(u, v) = \chi_\tau(u, v) = 1$. Invariant (I) then implies $\phi_\tau(u, v) = 1$, hence v is not free in M_τ. □

Now we just need to prove that (I) indeed holds for every time $\tau \in [0..|S|]$ and the equivalence we claimed will follow.

First, let's notice some useful properties of the functions defined earlier. ϕ_τ and f_τ are symmetric on their arguments: $\phi_\tau(u, v) = \phi_\tau(v, u)$, $f_\tau(u, v) = f_\tau(v, u)$ and ψ_τ, χ_τ are skew-symmetric in the boolean sense: $\psi_\tau(u, v) = 1 - \psi_\tau(v, u)$, $\chi_\tau(u, v) = 1 - \chi_\tau(v, u)$ for every pair u, v of adjacent vertices at time τ. Hence when arguing that (I) holds, it suffices to prove it for only one of the pairs $(u, v), (v, u)$. Furthermore, for every τ_0, τ_1 such that $u, v \in V(H^{\min(\tau_0,\tau_1)})$: $\psi_{\tau_0}(u, v) = \psi_{\tau_1}(u, v)$ and $\chi_{\tau_0}(u, v) = \chi_{\tau_1}(u, v)$, i.e. the direction of the edges as well as which endpoints belong to L vs R remains a constant throughout the run.

We are now ready to prove (I) in the following theorem.

Theorem 24. *Let f_τ, M_τ be the flow and matching maintained by the two algorithms. Then invariant (I) holds between them for all $\tau \in [0..|S|]$.*

Proof. We proceed by induction on τ. For $\tau = 0$, let u, v be arbitrary adjacent vertices in $\overline{H}^0$. Initially the flow is empty so $f_0(u, v) = 0$. By definition of $M_0, \{u, v\} \in M_0$ (or

equivalently $\phi_0(u, v) = 1$) if and only if $\psi_0(u, v) \oplus \chi_0(u, v) = 1$. Hence (I) holds.

Suppose the invariant holds at time $\tau < |S|$. Now after the arrival of source $x_{\tau+1}$, a shortest augmenting path is chosen by the Network-SAP algorithm leading to a free sink — such a path is guaranteed to exist by the flow-admissibility assumption. By the inductive hypothesis and Lemma 24, we can assume without loss of generality that the Bipartite-SAP algorithm chooses the corresponding augmenting path on $\overline{H}^{\tau}$. Consider two adjacent vertices $u, v \in E(\overline{H}^{\tau})$. If $u, v \neq x_{\tau+1}$ then those vertices were already present in H^{τ} and as noted $\psi_{\tau+1}(u, v) = \psi_{\tau}(u, v), \chi_{\tau+1}(u, v) = \chi_{\tau}(u, v)$.

Now either $\{u, v\}$ was an edge of the augmenting path in which case Network-SAP modified its flow value and Bipartite-SAP modified its presence in the matching resulting in $\phi_{\tau+1}(u, v) = 1 - \phi_{\tau}(u, v), f_{\tau+1}(u, v) = 1 - f_{\tau}(u, v)$, or it was not on the augmenting path in which case the values of ϕ, f remain the same. In either case it's easy to see that (I) holds at $\tau + 1$.

It remains to show the invariant in the case that either u or v is the new source. As mentioned before, it suffices to show it for the case $u = x_{\tau+1}$ as the other one is symmetric. So in this case $\{u, v\}$ is a new edge and $\psi_{\tau+1}(u, v) = 1$ because all sources are in L **(P1)** and $\overline{H}$ is bipartite **(P2)** and $\chi_{\tau+1}(u, v) = 1$ because sources only have outgoing edges. If (u, v) was an edge of the augmenting path then $\phi_{\tau+1}(u, v) = f_{\tau+1}(u, v) = 1$. Otherwise $\phi_{\tau+1}(u, v) = f_{\tau+1}(u, v) = 0$. So in either case (I) holds. □

To complete the reduction, we need to associate the flow-switching cost in G with the vertex-switching cost in $\overline{H}$. Lemma 24 and Theorem 24 guarantee that the two algorithms follow corresponding paths on graphs $G, \overline{H}$. It's also easy to see that there is a one-to-one correspondence between paths in G^{τ} and H^{τ} under which paths of length k become paths of length $2k-1$. Summing up over all those contributions for every source

we get

$$\mathrm{FSC}^{G}(\text{Network-SAP}) = \frac{1}{2}(\mathrm{FSC}^{H}(\text{Network-SAP}) + |S|). \tag{6.8}$$

Now, a path of length k in H^{τ} contributes $\frac{1}{2}(k-1)$ to the vertex switching cost in $\overline{H}^{\tau}$. Summing again over all sources we get

$$\mathrm{FSC}^{H}(\text{Network-SAP}) = 2\mathrm{VSC}^{\overline{H}}(\text{Bipartite-SAP}) + |S|. \tag{6.9}$$

Combining the above and noticing that $|S| \leq n$ we conclude

$$\mathrm{FSC}^{G}(\text{Network-SAP}) \leq \mathrm{VSC}^{\overline{H}}(\text{Bipartite-SAP}) + n. \tag{6.10}$$

The final step is to argue that Theorem 2 applies on $\overline{H}$ which amounts to showing that $\overline{H}$ is matching-admissible. The existence of M_0 is not enough since it doesn't cover all vertices in L. The assumption that G is flow-admissible allows us to prove the matching-admissibility of $\overline{H}$ as follows.

Lemma 25. *If G is a flow-admissible unit-vertex-capacitated network then $\overline{H}$ is matching-admissible.*

Proof. Suppose f is a maximum flow in G which accommodates all sources. Due to the vertex capacity constrains, this flow must be decomposable into vertex-disjoint paths $\mathcal{Q} = \{Q_1, \ldots, Q_{|S|}\}$ in G. Let $\{P_1, \ldots, P_{|S|}\}$ denote the corresponding set of paths in H, obtained by inflating each vertex u in the interior of one of the paths into the pair of "gadget vertices" $(u, r), (u, l)$. We claim that the symmetric difference $M = \left(\bigcup_{\tau=1}^{|S|} \overline{P_\tau}\right) \oplus M_0$ is a matching which covers all vertices in L: M_0 is a matching and $\left(\bigcup_{\tau=1}^{|S|} \overline{P_\tau}\right)$ is a set of vertex-disjoint M_0-augmenting paths, one for each source in S. □

Theorem 2 thus implies a bound of $O(|L| \log^2 |L|)$ to $\mathrm{VSC}^{\overline{H}}$(Bipartite-SAP) and $|L| \leq |V(G)| = n$. Combining with equation 6.10 we get the theorem that follows.

Theorem 25. *For any flow-admissible unit-vertex-capacitated network $G = (V, E)$, the flow-switching cost of the Network-SAP algorithm on G when sources arrive in an online manner is at most $O(n \log^2 n)$ where $n = |V|$.*

6.5 More general networks

The reduction described in the previous section can in fact be generalized. The unit vertex capacities were easy to handle since the gadget of the first step transformed the network in a way that not only obviated the vertex capacity constraints, but also provided the network with the special bipartite structure that enabled us to argue that a certain equivalence holds between augmenting paths in the directed and undirected case. In this section we identify the properties that a directed graph needs to possess in order for the reduction to go through. These properties turn out to define a more general family of directed graphs than the ones produced by the construction (6.1)–(6.6) specified in the preceding section.

Let $H = (V, E)$ be a directed network with a set $S \subseteq V$ of sources and a set $T \subseteq V$ of sinks. Let L, R be a partition of its vertices. We claim that the essential properties that H must have in order for the reduction to go through are the following:

(P1) $S \subseteq L,\ T \subseteq R$.

(P2) **[Bipartite]** $E \subseteq (L \times R) \cup (R \times L)$

(P3) **[Unit capacities]** $\forall x \in (E \cup S \cup T) : c(x) = 1$ and $c(x) = +\infty$ otherwise.

(P4) **[Oriented graph]** The graph is *simple* (no loops or parallel edges) and contains *no* 2*-cycles* (for every $u, v \in V$ either $(u, v) \notin E$ or $(v, u) \notin E$).

(P5) **[Contains a matching]** The undirected analogue $\overline{H}$ (see Definition 16) of H contains a matching that covers all vertices in L.

(P6) The set $\{\{u, v\} \mid u \in R, v \in L, (u, v) \in E(H)\}$ is a matching in H.

Before proving it, notice that the graph H defined in the preceding section by (6.1)-(6.6) indeed satisfies all these properties. **(P1)**–**(P4)** hold by construction as noted in the previous section when defining $\overline{H}$. Property **(P5)** is what Lemma 25 asserts and **(P6)** holds by definition of H.

In fact, we can generalize **(P6)** to:

(P6') **[Bounded in-degree]** $\forall l \in L : |E \cap (R \times \{l\})| \leq 1$, i.e. vertices in L have in-degree at most 1.

For a network satisfying **(P6')**, the set M_0 as defined in 6.7 is now a *semi-matching*, i.e. vertices in R are allowed to be matched to multiple vertices in L but each vertex in L is matched to at most one vertex in R (see Definition 19). As we shall see shortly, Theorem 24 still holds when M_τ is a semi-matching and furthermore, as we prove in Appendix C, the bound of [12] extends to when the graph is initialized with a semi-matching.

Theorem 26. *For any flow-admissible network $H = (V, E)$ for which there exists a partition $V = L \cup R$ of its vertices that satisfies **(P1)**–**(P5)** and **(P6')**, the flow-switching cost of the Network-SAP algorithm on H when sources arrive in an online manner is at most $O(n \log^2 n)$ where $n = |V|$.*

Proof. The proof is presented in Section 6.4. That proof relied on three properties of H that were denoted (H1)–(H3); the justifications for those properties presented in Section 6.4 are annotated with labels indicating how the justifications can be derived from

properties **(P1)**–**(P4)**. The proofs of Lemma 24 and Theorem 24 rely only on (H1)–(H3) and on the fact that M_τ is a matching. As discussed, (H1)–(H3) follow from **(P1)**–**(P4)**, and M_0 is a semi-matching due to **(P6')**. The fact that M_τ is a semi-matching for $\tau > 0$ is proven in Lemma 30. Thus, the proofs of Lemma 24 and Theorem 24 remain valid if we replace "matching" with "semi-matching" throughout. Property **(P5)** replaces Lemma 25: $\overline{H}$ is matching-admissible according to the original definition of that term (not modified to incorporate semi-matchings), and the property of matching-admissibility is needed for the application of Theorem 27 to come.

Equation (6.9) still holds and expresses the flow-switching cost in H in terms of the vertex-switching cost in $\overline{H}$:

$$\mathrm{FSC}^{G}(\text{Network-SAP}) \leq 2\mathrm{VSC}^{\overline{G}}(\text{Bipartite-SAP}) + n.$$

Theorem 27 now applies on $\overline{H}$ because of **(P5)** and **(P6')**, hence

$$\mathrm{FSC}^{H}(\text{Network-SAP}) = O(|L| \log^2 |L|) = O(n \log^2 n).$$

□

APPENDIX A

ADDITIONAL MATERIAL FROM CHAPTER 2

A.1 Matroids

Here we restate and prove Proposition 2 which provides a way of updating the optimal solution of a weighted matroid.

Proposition. *Let $\mathcal{M} = (E, \mathcal{I})$ be a weighted matroid with weight function $w : E \to \mathbb{R}^+$. Consider the max-weight independent set I of the restricted matroid $\mathcal{M}_{|E-x}$. Then the max-weight independent set I^* of $\mathcal{M}$ can be obtained from I as follows: if $(I + x) \in \mathcal{I}$ then $I^* = I + x$, otherwise, $I^* = (I + x) - y$ where y is the minimum-weight element in the unique circuit C of $I + x$.*

Proof. Consider running Greedy (Algorithm 1) in parallel on both $\mathcal{M}_{|E-x}$ and $\mathcal{M}$ and call these executions $\mathcal{E}^-, \mathcal{E}$ respectively.

In case $(I + x) \in \mathcal{I}$, the downward-closed property of $\mathcal{M}$ guarantees that both executions will make identical decisions on elements other than x and element x will be included in the optimal solution of $\mathcal{M}$, hence $I^* = I + x$.

For the other case, suppose first that $x = y$, i.e. x is the min-weight element on C. At the time x is inspected, all other elements of C have already been inspected and added to the current solution, hence x is not included since it would violate independence. Therefore, both executions proceed making identical decisions in every step and arrive to the same solution $I^* = (I + x) - x = I$.

Now suppose that $x \neq y$. At the time element x is considered in $\mathcal{E}$, it can be safely included in the solution. The reason is that if adding x resulted in a circuit C', then

$C' \subsetneq C$ violating the minimality of C. The next step at which the two executions will diverge again is when considering y — if they diverged at a previous step it would again mean that x is part of a circuit $C' \subsetneq C$ — at this point $\mathcal{E}$ ignores y.

Finally, suppose that the two executions diverge at a later step on an element e with $w(e) < w(y)$. Denote by J the current solution $\mathcal{E}^-$ is maintaining and thus $(J + x) - y$ is the current solution of $\mathcal{E}$. There are two reasons the executions might diverge:

- $(J + e) \in \mathcal{I}$ but $((J + x) - y) + e \notin \mathcal{I}$.

  In this case, there must exist circuit $C' \subseteq ((J + x) - y) + e$ such that $x, e \in C'$ and $y \notin C'$. Therefore, by Proposition 1 there exists circuit C'' such that $e \in C'' \subseteq (C' \cup C) - x$. This is a contradiction because C'' is a circuit of $J + e$ which was assumed to be independent.

- $(J + e) \notin \mathcal{I}$ but $((J + x) - y) + e \in \mathcal{I}$.

  This case is similar and the proof is omitted.

□

A.2 Prophet Inequalities

A.2.1 Proof

Here we present a proof of the standard Prophet Inequality (Theorem 1) for the adversarial-order prophet problem by considering two closely related single-threshold stopping rules. The first uses as a threshold a T such that $\Pr[\max_{i\in[n]} X_i > T] = 1/2$[1]

[1] See Appendix A.2.2 for a discussion on how to achieve this if the distributions are discrete or if they have point masses.

proposed by [86]. The second, uses a threshold $T = \frac{1}{2}\mathbb{E}[\max_{i\in[n]} X_i]$ proposed by [61]. Both stopping rules satisfy a prophet inequality with a factor $1/2$ as we shall see shortly.

The proof relies on a generic lower bound on the expected reward of any single-threshold stopping rule and an upper bound on the expected reward of the prophet, both as functions of some parameters of the instance. More specifically, let T be the single threshold used by a stopping rule and $p = \Pr[\max_{i\in[n]} X_i > T]$ be the probability that at least one of the X_i's exceeds the threshold or in other words the probability that the decision-maker accepts exactly one. In what follows we use the notation $\mathbb{I}[E]$ for the indicator random variable of an event E and for any real number x we denote $(x)^+ = \max(x, 0)$.

First, let us bound the expected reward of the algorithm as follows:

$$\mathbb{E}[\text{ALG}] = \mathbb{E}\left[\sum_{i=1}^{n} X_i \cdot \mathbb{I}[\text{ALG accepts } X_i]\right] \tag{A.1}$$

$$= \mathbb{E}\left[\sum_{i=1}^{n} (T + (X_i - T)) \cdot \mathbb{I}[\text{ALG accepts } X_i]\right] \tag{A.2}$$

$$= T \cdot \sum_{i=1}^{n} \Pr[\text{ALG accepts } X_i] + \sum_{i=1}^{n} \mathbb{E}[(X_i - T) \cdot \mathbb{I}[\text{ALG accepts } X_i]] \tag{A.3}$$

$$= T \cdot p + \sum_{i=1}^{n} \mathbb{E}[(X_i - T) \cdot \mathbb{I}[\text{ALG rejects } X_1, X_2, \ldots, X_{i-1} \text{ and } X_i > T] \tag{A.4}$$

$$= T \cdot p + \sum_{i=1}^{n} \mathbb{E}[(X_i - T) \cdot \mathbb{I}[X_i > T] \cdot \mathbb{I}[\text{ALG rejects } X_1, X_2, \ldots, X_{i-1}]] \tag{A.5}$$

$$= T \cdot p + \sum_{i=1}^{n} \mathbb{E}[(X_i - T)^+] \cdot \Pr[\text{ALG rejects } X_1, X_2, \ldots, X_{i-1}] \tag{A.6}$$

$$\geq pT + \Pr[\text{ALG rejects } X_1, X_2, \ldots, X_n] \cdot \sum_{i=1}^{n} \mathbb{E}[(X_i - T)^+] \tag{A.7}$$

$$= pT + (1 - p) \cdot \mathbb{E}\left[\sum_{i=1}^{n} (X_i - T)^+\right] \tag{A.8}$$

$$\geq pT + (1 - p) \cdot \mathbb{E}\left[\max_{i\in[n]} (X_i - T)^+\right], \tag{A.9}$$

where linearity of expectation is used to exchange summations and expectation and Equation (A.5) follows from the fact that the event [ALG rejects $X_1, X_2, \ldots, X_{i-1}$] is independent of $[X_i \geq T]$. On the other hand, for the expected reward of the prophet we have,

$$\mathbb{E}[\text{OPTO}_{\text{FF}}] = \mathbb{E}\left[\max_{i\in[n]} X_i\right] \tag{A.10}$$

$$\leq T + \mathbb{E}\left[\max_{i\in[n]} (X_i - T)^+\right]. \tag{A.11}$$

To derive the prophet inequality for a stopping rule which uses the median of $\max_{i\in[n]} X_i$ as the threshold T, notice that $p = 1/2$ by definition in this case so combining Inequalities (A.9), (A.11) we derive a prophet inequality:

$$\mathbb{E}[\text{ALG}] \geq \frac{1}{2} \cdot \left(T + \mathbb{E}[\max_{i\in[n]} (X_i - T)^+]\right) \geq \frac{1}{2} \cdot \mathbb{E}[\text{OPTO}_{\text{FF}}].$$

Similarly, when $T = \frac{1}{2}\mathbb{E}[\max_{i\in[n]} X_i]$,

$$\mathbb{E}[\text{ALG}] \geq pT + (1-p) \cdot \mathbb{E}\left[\max_{i\in[n]} (X_i - T)^+\right] \tag{A.12}$$

$$\geq pT + (1-p) \cdot \left(\mathbb{E}[\max_{i\in[n]} X_i] - T\right) \tag{A.13}$$

$$= pT + (1-p)T \tag{A.14}$$

$$= T = \frac{1}{2}\mathbb{E}[\text{OPTO}_{\text{FF}}]. \tag{A.15}$$

We now show that the prophet inequality above is tight. Consider an instance with the two random variables shown below, where the first one is deterministically equal to 1 and the second is almost always equal to 0 except with probability $\varepsilon \in (0, 1)$ it takes a value of $1/\varepsilon > 1$. For simplicity, assume the permutation chosen by the adversary is the identity permutation (i.e. the variables have to be inspected in the order given):

$$X_1 = 1, \quad X_2 = \begin{cases} \frac{1}{\varepsilon}, & \text{w.p. } \varepsilon \\ 0, & \text{w.p. } 1-\varepsilon \end{cases}. \tag{A.16}$$

A prophet able to see both values can always pick the maximum of the two for an expected reward of

$$\mathbb{E}[\max(X_1, X_2)] = \varepsilon \cdot \frac{1}{\varepsilon} + (1 - \varepsilon) \cdot 1 = 2 - \varepsilon.$$

On the other hand, any deterministic algorithm receives no new information after inspecting X_1, hence every possible algorithm falls into one of two categories: (a) always accepting X_1 or (b) always rejecting X_1 and proceeding to consider X_2. Algorithms of the first kind receive an expected reward exactly equal to 1. Algorithms of the second kind, receive a reward of at most $\mathbb{E}[X_2] = 1$. Either way, for any deterministic algorithm ALG we have,

$$\mathbb{E}[\text{ALG}] \leq 1.$$

To see how this result extends to randomized algorithms, notice that a randomized algorithm is a probability distribution on deterministic algorithms. By linearity of expectation, the reward of any randomized algorithm will be a linear combination of the rewards of deterministic algorithms, all of which have expected reward at most 1, therefore no online algorithm (randomized or otherwise) can get an expected reward better than 1. Taking $\varepsilon \to 0$, the claim that inequality (2.1) is tight follows.

A.2.2 Single-threshold stopping rules

We continue the discussion of single-threshold stopping rules initiated in Section 2.2 with a stricter treatment to encompass a wider range of distributions. Recall the definition of a single-threshold stopping rule with threshold T: the time-step $\tau \in [n] \cup \{\perp\}$ when an acceptance decision was made by the algorithm must satisfy the following constraints for all $i \in [n]$,

$$X_{\pi(i)} < T \Rightarrow \tau \neq i, \qquad X_{\pi(i)} > T \Rightarrow \tau \leq i.$$

When $X_{\pi(i)} = T$, a tie-breaking rule is needed. If the distributions of $\mathcal{F}_1, \ldots, \mathcal{F}_n$ have no point-masses, the event $X_{\pi(i)} = T$ has probability zero so the tie-breaking convention is inconsequential. To reduce from the general case to the case where tie-breaking is inconsequential, we adopt the following artificial but convenient convention. We assume that in addition to the random variables $X_1, \ldots, X_n$, there is an auxiliary sequence of random variables $\tilde{X}_1, \ldots, \tilde{X}_n$, each uniformly distributed in $[0, 1]$, independent of $X_1, \ldots, X_n$ and mutually independent of one another. These auxiliary values are used for tie-breaking as follows. The pairs $\{(X_j, \tilde{X}_j)\}_{j=1}^n$ are regarded as elements of $\mathbb{R} \times [0, 1]$ under the lexicographic ordering. For every $\tilde{T} \in [0, 1]$, the single-threshold stopping rule with threshold $(T, \tilde{T})$ is defined as

$$\tau = \min\{i \in [n] \mid (X_{\pi(i)}, \tilde{X}_{\pi(i)}) \geq (T, \tilde{T})\}$$

where, again, the relation $<$ is interpreted lexicographically. We make the following observations.

1. For all $i \in [n]$, the event $(X_{\pi(i)}, \tilde{X}_{\pi(i)}) = (T, \tilde{T})$ has probability zero.
2. For any $i \in [n]$ and $p \in (0, 1)$, we can find $(T, \tilde{T})$ such that

$$\Pr[(X_{\pi(i)}, \tilde{X}_{\pi(i)}) > (T, \tilde{T})] = p.$$

3. Similarly, if $(X_*, \tilde{X}_*)$ denotes the $(\mathbb{R} \times [0, 1])$-valued random variable that is the lexicographic maximum of $\{(X_{\pi(i)}, \tilde{X}_{\pi(i)})\}_{i=1}^n$, then for any $p \in (0, 1)$ we can find $(T, \tilde{T})$ such that

$$Pr[(X_*, \tilde{X}_*) > (T, \tilde{T})] = p.$$

In short, by treating the sequence of n random variables as taking values in $\mathbb{R} \times [0, 1]$ (lexicographically ordered) and treating the threshold as belonging to the same set, we can make the same continuity and no-tie-breaking assumptions that are always justified

in the case of point-mass-free distributions, but without having to assume the distributions of $X_1, \ldots, X_n$ have no point masses.

To avoid cumbersome notation in what follows, when discussing threshold stopping rules we will still denote the threshold by T rather than $(T, \tilde{T})$ and we'll use the notation $X_j < T$ or $X_* < T$ as shorthand for $(X_j, \tilde{X}_j) < (T, \tilde{T})$ or $(X_*, \tilde{X}_*) < (T, \tilde{T})$. In short, we'll treat the distributions of $X_1, \ldots, X_n$ as if they were free of point masses, depending on the conventions set forth in this section to justify that the results derived under the point-mass-free assumption extend to the case of general distributions.

A.2.3 Single-threshold competitive ratio for free-order prophets

The fact that the single-threshold competitive ratio of the free-order prophet problem (Problem 3) is $\overline{\mathcal{R}}_n^{\text{free}} = 1 - \frac{1}{e} + o(1)$ appears in [39] and is implicit in [31, 60]; we also provide a proof below using a theorem proved in Chapter 3.

Proposition 4. *The single-threshold competitive ratio of the free-order prophet problem is* $\overline{\mathcal{R}}_n^{\text{free}} = 1 - \frac{1}{e} + o(1)$. *As* $n \to \infty$, *the competitive ratio converges to* $\overline{\mathcal{R}}^{\text{free}} = 1 - \frac{1}{e}$ *from above.*

Proof. By Theorem 5, $\overline{\mathcal{R}}_n^{\text{free}} \geq 1 - \frac{1}{e} + o(1)$, so we need only show that $\overline{\mathcal{R}}_n^{\text{free}} \leq 1 - \frac{1}{e} + o(1)$. To this end, let $H \gg 1$ be an arbitrarily large number and consider a sequence of random variables $X_1, \ldots, X_n$ drawn i.i.d. from a distribution whose cumulative distribution

function F is given by

$$F(x) = \begin{cases} 0 & \text{if } x < 1 \\ \left(H - \frac{1}{(e-2)n}\right)(x-1) & \text{if } 1 \le x \le 1 + \frac{1}{H} \\ 1 - \frac{1}{(e-2)nH} & \text{if } 1 + \frac{1}{H} < x < H + 1 \\ 1 - \frac{1}{(e-2)nH} + \frac{x-H-1}{(e-2)n} & \text{if } H + 1 \le x \le H + 1 + \frac{1}{H} \\ 1 & \text{if } x > H + 1 + \frac{1}{H}. \end{cases}$$

In words, each X_i is sampled from a mixture of two uniform distributions, on the intervals $\left[1, 1 + \frac{1}{H}\right]$ and $\left[H + 1, H + 1 + \frac{1}{H}\right]$, with the mixture weights being $1 - \frac{1}{(e-2)nH}$ and $\frac{1}{(e-2)nH}$, respectively.

Since the variables $X_1, \ldots, X_n$ are identically distributed, reordering them has no effect on the performance of stopping rules, so we merely need to show that if ALG is a single-threshold stopping rule adapted to the sequence $X_1, \ldots, X_n$ then $\mathbb{E}[\text{ALG}] \le \left(1 - \frac{1}{e} + o(1)\right) \cdot \mathbb{E}[\text{OPTO}_{\text{FF}}]$, where the $o(1)$ term vanishes as $n \to \infty$.

Let $p = \frac{1}{(e-2)nH}$ denote the probability that a sample X_i exceeds H. The prophet's expected value satisfies

$$\mathbb{E}[\text{OPTO}_{\text{FF}}] = 1 + (1 - (1-p)^n)H + O\left(\tfrac{1}{H}\right) \ge \frac{e-1}{e-2} - O\left(\tfrac{1}{H}\right) \tag{A.17}$$

where the second inequality follows from:

$$(1-p)^n < 1 - np + \binom{n}{2}p^2 < 1 - \frac{1}{(e-2)H} + \frac{1}{2(e-2)^2H^2},$$
$$(1 - (1-p)^n)H > \frac{1}{e-2} - \frac{1}{2(e-2)^2H}.$$

Now consider a single-threshold stopping rule ALG that sets a threshold T such that $F(T) = 1 - q$; in other words, any given element of the sequence $X_1, \ldots, X_n$ has probability q of exceeding the threshold. Then, ALG rejects all r.v.s with probability

$(1-q)^n$. Furthermore, conditional on ALG accepting some r.v., the events ALG $\in \left[H+1, H+1+\frac{1}{H}\right]$ and ALG $\in \left[1, 1+\frac{1}{H}\right]$ have conditional probabilities $\min\{p/q, 1\}$ and $1-\min\{p/q, 1\}$, respectively. Thus,

$$\mathbb{E}[\text{ALG}] \leq (1-(1-q)^n) \cdot \left(\frac{p}{q}H + 1 + \frac{1}{H}\right).$$

Let $r = q/n$. Ignoring $O(1/H)$ terms that vanish as $H \to \infty$ and $O(1/n)$ terms that vanish as $n \to \infty$ we have

$$\begin{aligned} \mathbb{E}[\text{ALG}] &\approx (1-e^{-r}) \cdot \left(1 + \frac{1}{(e-2)r}\right) \\ \frac{\mathbb{E}[\text{ALG}]}{\mathbb{E}[\text{OPTO}_{\text{FF}}]} &\approx (1-e^{-r}) \cdot \left(\frac{e-2}{e-1} + \frac{1}{(e-1)r}\right). \end{aligned} \tag{A.18}$$

The right side of (A.18) is maximized at $r = 1$, where it equals $1 - \frac{1}{e}$. □

APPENDIX B

ADDITIONAL MATERIAL FROM CHAPTER 3

B.1 Deferred proofs from Section 3.2

In this section we restate and prove Lemma 5.

Lemma 26. *If* Π *is a non-empty set of permutations of* $[n]$ *and there exists an index* $j \in [n]$ *that is* ε*-centered with respect to* Π*, then* $\overline{\mathcal{R}}_n(\Pi) \leq \varphi^{-1} + O(\varepsilon)$.

Proof. Suppose j is ε-centered with respect to Π, and let p be a probability distribution on $[n] \setminus \{j\}$ such that for every permutation σ whose inverse belongs to Π, the sets $\{i \mid \sigma(i) < \sigma(j)\}$ and $\{i \mid \sigma(i) > \sigma(j)\}$ have measure at least $\frac{1}{2} - \varepsilon$ under p. Define the distributions of $X_1, X_2, \ldots, X_n$ as follows. For a small positive number δ to be determined later, the value of X_j is $(\sqrt{5} - 1)/\delta$ with probability δ, and otherwise $X_j = 0$. For every $i \in [n] \setminus \{j\}$, X_i has cumulative distribution function F_i satisfying

$$F_i(1-t) = \Pr[X_i \leq 1 - t] = \begin{cases} 1 & \text{if } t < 0 \\ \exp\left(-\dfrac{p(i)t}{\delta}\right) & \text{if } 0 \leq t \leq 1 \\ 0 & \text{if } t > 1. \end{cases}$$

In other words, $1 - X_i = Y_i \wedge 1$ where Y_i is exponentially distributed with rate parameter $p(i)/\delta$ and the notation $a \wedge b$ denotes the minimum of a and b.

First, observe that OPT is equal to $(\sqrt{5} - 1)/\delta$ with probability δ, and otherwise

$$\mathrm{OPT} = \max_{i \neq j}\{X_i\} = 1 - \min_{i \neq j}\{Y_i \wedge 1\} \geq 1 - \bigwedge_{i \neq j} Y_i.$$

The minimum of independent exponential random variables with rates $r_1, \ldots, r_n$ is exponential with rate $r_1 + \cdots + r_n$. Hence, $\bigwedge_{i \neq j} Y_i$ is exponentially distributed with rate

$\frac{1}{\delta}\sum_{i\neq j} p(i) = \frac{1}{\delta}$, and its expected value is δ. Consequently,

$$\mathbb{E}[\text{OPT} \mid X_j = 0] \geq \mathbb{E}\left[1 - \bigwedge_{i\neq j} Y_i\right] = 1 - \delta$$

and the prophet's unconditional expected value satisfies

$$\begin{aligned}\mathbb{E}[\text{OPT}] &= \delta \cdot \frac{\sqrt{5}-1}{\delta} + (1-\delta)\cdot \mathbb{E}[\text{OPT} \mid X_j = 0] \\ &\geq \sqrt{5} - 1 + (1-\delta)^2 > \sqrt{5} - 2\delta.\end{aligned} \tag{B.1}$$

Now we turn to analyzing the expected value obtained by a single-threshold stopping rule ALG with threshold T, assuming ALG is π-adapted for some $\pi \in \Pi$ with inverse permutation $\sigma = \pi^{-1}$. If $T < 0$ then ALG always chooses the first value observed and ALG $= X_{\pi(1)} \leq 1$ so $\mathbb{E}[\text{ALG}] \leq 1$. If $T > 1$ then ALG either rejects all random variables in which case ALG $= 0$ or it chooses the random variable X_j so $\mathbb{E}[\text{ALG}] = \mathbb{E}[X_j] = \sqrt{5} - 1$. In all of those cases, $\mathbb{E}[\text{ALG}] < \varphi^{-1} \cdot \mathbb{E}[\text{OPT}]$ provided $\delta < 0.07$. The remaining case to consider is when $0 \leq T \leq 1$. In that case let $I_0 = \{i \mid \sigma(i) < \sigma(j)\}$ denote the set indexing the values appearing before X_j in the sequence $X_{\pi(1)}, \ldots, X_{\pi(n)}$, and let $I_1 = \{i \mid \sigma(i) > \sigma(j)\}$ denote the set indexing the values appearing after X_j in that sequence. If q_0, q_1 denote the probabilities of the events $\max\{X_i \mid i \in I_0\} < T$ and $\max\{X_i \mid i \in I_1\} < T$, respectively, then the algorithm's expected value satisfies

$$\begin{aligned}\mathbb{E}[\text{ALG}] \ &\leq (1-q_0)\cdot 1 + q_0 \cdot \delta \cdot \frac{\sqrt{5}-1}{\delta} \\ &\quad + q_0 \cdot (1-\delta)\cdot(1-q_1)\cdot 1 \\ &< 1 - q_0 q_1 + (\sqrt{5}-1)q_0,\end{aligned} \tag{B.2}$$

where the first inequality is justified because ALG ≤ 1 if the algorithm chooses any X_i with $i \in [n] \setminus j$ and ALG $= 0$ if the algorithm rejects every random variable. Writing

$T = 1 - \delta s$, we have $\Pr[X_i < T] = \exp(-sp(i))$ for $i \neq j$ and therefore

$$q_0 = \prod_{i \in I_0} \Pr[X_i < T] = \exp\left(-s \sum_{i \in I_0} p(i)\right) < \exp\left(-\left(\frac{1}{2} - \varepsilon\right) s\right)$$

$$q_0 q_1 = \prod_{i \in I_0 \cup I_1} \Pr[X_i < T] = \exp\left(-s \sum_{i \neq j} p(i)\right) = \exp(-s).$$

Substituting these into (B.2) we find that

$$\mathbb{E}[\mathrm{ALG}] < 1 - \exp(-s) + (\sqrt{5} - 1) \exp\left(-\left(\frac{1}{2} - \varepsilon\right) s\right). \tag{B.3}$$

To complete the proof we must bound the right side of (B.3) above by $\varphi^{-1} + O(\varepsilon)$ when $s \geq 0$. The derivative of the right side is

$$\begin{aligned} &\exp(-s) - (\sqrt{5} - 1)(\tfrac{1}{2} - \varepsilon) \exp\left(-\left(\frac{1}{2} - \varepsilon\right) s\right) \\ &= \left[\exp\left(-\left(\frac{1}{2} + \varepsilon\right) s\right) - \varphi^{-1} + (\sqrt{5} - 1)\varepsilon\right] \cdot \exp\left(-\left(\frac{1}{2} - \varepsilon\right) s\right) \end{aligned}$$

which is negative for all $s \geq 1$, provided that $\varepsilon < \frac{1}{110}$. Assume henceforth that $\varepsilon < \frac{1}{110}$, as otherwise the inequality asserted by the lemma, $\overline{\mathcal{R}}_n(\Pi) \leq \varphi^{-1} + O(\varepsilon)$, is trivially satisfied due to the $O(\varepsilon)$ term. The derivative calculation above shows that the right side of (B.3) is a decreasing function of $s \geq 1$, implying that its maximum value on the interval $s \in [0, \infty)$ is attained when $s \in [0, 1]$. For s in this range, if we let $q = \exp\left(-\frac{1}{2}s\right)$, then

$$\exp\left(-\left(\frac{1}{2} - \varepsilon\right) s\right) = q \cdot e^{\varepsilon s} \leq q \cdot e^{\varepsilon} \leq q + O(\varepsilon).$$

Hence we can rewrite (B.3) as

$$\mathbb{E}[\mathrm{ALG}] < 1 - q^2 + (\sqrt{5} - 1)q + O(\varepsilon). \tag{B.4}$$

The right side of (B.4) is a quadratic function of q that is maximized when $q = \varphi^{-1}$ and $1 - q^2 = q = \varphi^{-1}$. Therefore,

$$\mathbb{E}[\mathrm{ALG}] < \sqrt{5}\varphi^{-1} + O(\varepsilon). \tag{B.5}$$

Combining (B.1) with (B.5), the bound $\mathbb{E}[\mathrm{ALG}] \leq (\varphi^{-1} + O(\varepsilon)) \cdot \mathbb{E}[\mathrm{OPT}]$ follows, which confirms that $\overline{\mathcal{R}}_n(\Pi) \leq \varphi^{-1} + O(\varepsilon)$. □

B.2 Deferred proofs from Section 3.3

This section contains proofs of propositions and lemmas that were mentioned in Section 3.3 whose proofs were deferred.

Lemma 27. *For prime n, there exists a set Π of $n(n-1)$ permutations such that the uniform distribution over Π is pairwise independent. For any $\varepsilon, \delta > 0$ such that $1/\varepsilon$ is an integer, if n is an integer multiple of $1/\varepsilon$ and $\varepsilon n \geq 2/\delta$, then there exists a set Π of $O\left(\left(\frac{1}{\delta\varepsilon}\right)^2 \log n\right)$ permutations such that the uniform distribution over Π is (ε, δ)-almost pairwise independent.*

Proof. If n is equal to a prime number p, for any integers a, b such that a is not divisible by p, the function $x \mapsto ax + b \pmod{p}$ is a permutation of $[p]$. If (a, b) are sampled uniformly at random from $[p-1] \times [p]$, this defines a pairwise independent permutation distribution. The reason is that for any (i, j) and (k, ℓ) in $[p] \times [p]$ such that $i \neq j$, $k \neq \ell$, the system of linear congruences

$$ai + b \equiv k, \quad aj + b \equiv \ell \pmod{p}$$

has the unique solution $a \equiv (k-\ell)(i-j)^{-1}$, $b \equiv k - ai \pmod{p}$.

For $\varepsilon, \delta > 0$ such that $1/\varepsilon$ and εn are integers and $\varepsilon n \geq 2/\delta$, we use the probabilistic method to prove the existence of a multiset Π of $m = O((\varepsilon\delta)^{-2} \log n)$ permutations such that the uniform distribution on Π is (ε, δ)-almost pairwise independent. In fact, we will prove that the multiset obtained by drawing m uniformly-random samples, with replacement, satisfies this property with positive probability. Define the function $b(k) = \left\lceil \frac{k}{\varepsilon n} \right\rceil$, mapping $[n]$ to $[\frac{1}{\varepsilon}]$. The (ε, δ)-almost pairwise independence property asserts that when σ is a random sample from the distribution, for any $i \neq j$ the distribution of the pair $(b(\sigma(i)), b(\sigma(j)))$ is δ-close to the uniform distribution on $\left\lceil \frac{1}{\varepsilon} \right\rceil \times \left\lceil \frac{1}{\varepsilon} \right\rceil$ in total variation

distance. For any pair $(u, v) \in \left[\frac{1}{\varepsilon}\right] \times \left[\frac{1}{\varepsilon}\right]$, a uniformly random $\sigma \in S_n$ satisfies:

$$\Pr\big((b(\sigma(i)), b(\sigma(j))) = (u, v)\big) \;=\; \begin{cases} \frac{\varepsilon n}{n} \cdot \frac{\varepsilon n - 1}{n-1} & \text{if } u = v \\ \frac{\varepsilon n}{n} \cdot \frac{\varepsilon n}{n-1} & \text{if } u \neq v \end{cases}$$

In both cases we have

$$\begin{aligned} \Pr\big((b(\sigma(i)), b(\sigma(j))) = (u, v)\big) &> \frac{\varepsilon n}{n} \cdot \frac{\varepsilon n - 1}{n} \\ &= \varepsilon\left(\varepsilon - \frac{1}{n}\right) \geq \varepsilon^2\left(1 - \frac{\delta}{2}\right) \end{aligned} \tag{B.6}$$

where the last inequality is a consequence of $\varepsilon n \geq 2/\delta$.

Now, suppose $\sigma_1, \ldots, \sigma_m$ are i.i.d. random draws from the uniform distribution on S_n. Fix an arbitrary $i \neq j$ in $[n]$ and an arbitrary pair of buckets $(u, v) \in \left[\frac{1}{\varepsilon}\right] \times \left[\frac{1}{\varepsilon}\right]$. For $s = 1, 2, \ldots, m$ let

$$Y_s = \begin{cases} 1 & \text{if } (b(\sigma(i)), b(\sigma(j))) = (u, v) \\ 0 & \text{otherwise} \end{cases}$$

$$Z_s = \mathbb{E}[Y_s] - Y_s.$$

In light of (B.6), we have

$$\mathbb{E}\left[\sum_{s=1}^{m} Y_s\right] > \varepsilon^2\left(1 - \frac{\delta}{2}\right)m,$$

so the event $\sum_{s=1}^{m} Y_s \leq \varepsilon^2(1 - \delta)m$ only happens when $\sum_{s=1}^{m} Z_s > \frac{1}{2}m\varepsilon^2\delta$. Bernstein's Inequality [13] ensures that

$$\begin{aligned} \Pr\left[\sum_{s=1}^{m} Z_s > \tfrac{1}{2}m\varepsilon^2\delta\right] \;&<\; \exp\left(-\frac{\frac{1}{8}m^2\varepsilon^4\delta^2}{m\varepsilon^2 + \frac{1}{6}m\varepsilon^2\delta}\right) \\ &=\; \exp\left(-\frac{m\varepsilon^2\delta^2}{8(1 + \delta/6)}\right). \end{aligned} \tag{B.7}$$

Since $\delta < 1$, the right side can be made less than $(\varepsilon/n)^2$ by setting $m \geq 18(\varepsilon\delta)^{-2}\log(n/\varepsilon)$. Note that our assumption that n is divisible by $1/\varepsilon$ implies $n/\varepsilon \leq n^2$, hence

$\log(n/\varepsilon) \leq 2\log n$. Thus, drawing $m = 36(\varepsilon\delta)^{-2}\log n$ samples suffices to make $\Pr\left[\sum_{s=1}^{m} Z_s > \frac{1}{2}m\varepsilon^2\delta\right]$ less than $(\varepsilon/n)^2$.

For $(i,j,u,v) \in [n]\times[n]\times\left[\frac{1}{\varepsilon}\right]\times\left[\frac{1}{\varepsilon}\right]$ with $i \neq j$, define $S(i,j,u,v) \subseteq S_n$ to be the set of permutations σ such that $(b(\sigma(i)), b(\sigma(j))) = (u,v)$. Let $\mathcal{E}(i,j,u,v)$ denote the event that $S(i,j,u,v)$ contains $m\varepsilon^2(1-\delta)$ or fewer of the permutations $\sigma_1,\ldots,\sigma_m$, and let $\mathcal{E} = \bigcup_{i,j,u,v}\mathcal{E}(i,j,u,v)$. Our calculation using Bernstein's Inequality showed that for $m \geq 36(\varepsilon\delta)^{-2}\log n$, we have $\Pr[\mathcal{E}(i,j,u,v)] < (\varepsilon/n)^2$. Taking the union bound over all $n(n-1)/\varepsilon^2$ choices of i,j,u,v, we conclude that $\Pr[\mathcal{E}] < 1$. Therefore, the complementary event $\overline{\mathcal{E}}$ occurs with positive probability. When $\overline{\mathcal{E}}$ occurs, we claim the uniform distribution over $\{\sigma_1,\ldots,\sigma_m\}$ is (ε,δ)-almost pairwise independent. To verify this, consider any $i \neq j$ and let D_{ij} denote the distribution of $(b(\sigma(i)), b(\sigma(j)))$ when σ is drawn randomly from $\{\sigma_1,\ldots,\sigma_m\}$. Let U denote the uniform distribution on $\left[\frac{1}{\varepsilon}\right]\times\left[\frac{1}{\varepsilon}\right]$. We have

$$\begin{aligned}\|D_{ij} - U\|_{\mathrm{TV}} &= \sum_{u=1}^{1/\varepsilon}\sum_{v=1}^{1/\varepsilon}\left(U(S(i,j,u,v)) - D_{ij}(S(i,j,u,v))\right)^{+} \\ &< \left(\tfrac{1}{\varepsilon}\right)^2\left(\varepsilon^2 - \frac{m\varepsilon^2(1-\delta)}{m}\right) = \delta\end{aligned}$$

as required by the definition of (ε,δ)-almost pairwise independence. □

Lemma 28. *If $k \in \mathbb{N}$ and $r^k \geq \frac{1}{e}$ then $\frac{1}{k+1}\left(1 + r + \cdots + r^k\right) \geq 1 - \frac{1}{e}$.*

Proof. The left side is an increasing function of r, so it suffices to prove the inequality when $r^k = \frac{1}{e}$. We begin by noticing that if the hypothesis had been $r^{k+1} = \frac{1}{e}$ rather than $r^k = \frac{1}{e}$, the lemma would follow easily by comparing the sum to an integral.

$$\begin{aligned}\frac{1}{k+1}\left(1 + r + \cdots + r^k\right) &> \frac{1}{k+1}\int_0^{k+1} r^t\,dt \\ &= \frac{1}{(k+1)\ln r}\left[1 - r^{k+1}\right] \\ &= 1 - \frac{1}{e}.\end{aligned}$$

We do not know of any comparably simple argument that works when $r^k = \frac{1}{e}$. Instead, we define the function

$$f(k) = \frac{1}{k+1} \frac{1 - e^{-\frac{k+1}{k}}}{1 - e^{-\frac{1}{k}}}$$

and argue that $f(k) \geq 1 - \frac{1}{e}$ for all $k \geq 1$, by proving that $f(k)$ is monotonically decreasing in k and that $\lim_{k\to\infty} f(k) = 1 - \frac{1}{e}$. The computation of $\lim_{k\to\infty} f(k)$ follows from the facts that $1 - e^{-\frac{k}{k+1}} \to 1 - \frac{1}{e}$ and $(k+1)(1 - e^{-1/k}) \to 1$ as $k \to \infty$. The monotonicity claim follows by expanding the domain of f from the natural numbers to the real numbers and taking the derivative with respect to k.

$$\frac{df}{dk} = \frac{-e^{\frac{k+2}{k}}k^2 - k^2 + e^{\frac{1}{k}}((k+1)(k-2)+1) + e^{\frac{k+1}{k}}(k^2+k+1)}{e\left(e^{\frac{1}{k}} - 1\right)^2 (k+1)^2 k^2}$$

The denominator is positive for all $k \geq 1$ so it suffices to prove that the numerator in non-positive which is equivalent to showing:

$$e^{\frac{1}{k}}((k+1)(k-2)+1+ek(k+1)+e) \leq k^2\left(1 + e^{\frac{k+2}{k}}\right),$$
$$e^{-\frac{1}{k}} + e^{\frac{k+1}{k}} \geq \frac{(e+1)(k+1)^2 - (e+3)(k+1) + (e+1)}{k^2}$$

To prove the last inequality for all $k \geq 1$, we make use of the following quadratic lower bounds on e^x which can be shown using elementary calculus[1].

$$e^x \geq 1 + x + \frac{1}{e}x^2, \text{ for all } x \in [-1, 0]$$
$$e^x \geq \frac{e}{2}(x^2+1) \text{ for all } x \geq 1$$

Notice that $-1 \leq -\frac{1}{k} \leq 0$ and $\frac{k+1}{k} \geq 1$ for $k \geq 1$ so the above inequalities apply.

$$\begin{aligned} e^{-\frac{1}{k}} + e^{\frac{k+1}{k}} &\geq 1 - \frac{1}{k} + \frac{1}{e}\left(\frac{k+1}{k}\right)^2 + \frac{e}{2}\left(\left(\frac{k+1}{k}\right)^2 + 1\right) \\ &= \frac{\left(1 + \frac{1}{e} + e\right)(k+1)^2 - (e+3)(k+1) + \frac{e}{2} + 1}{k^2} \\ &\geq \frac{(1+e)(k+1)^2 - (e+3)(k+1) + (e+1)}{k^2} \end{aligned}$$

where the last inequality follows from $\frac{1}{e}(k+1)^2 + \frac{e}{2} + 2 \geq e + 1$ for $k \geq 1$. □

[1] Both are Taylor expansion approximations of e^x around 0 and 1 respectively.

Lemma 29 (Padding Lemma)**.** *If $N \in \mathbb{N}$ and $\Pi_N \subseteq S_N$ is a set of m permutations such that $\overline{\mathcal{R}}_N(\Pi_N) \geq \alpha$, then for all $n \leq N$ there is a set $\Pi_n \subseteq S_n$ of at most m permutations such that $\overline{\mathcal{R}}_n(\Pi_n) \geq \alpha$.*

Proof. For any permutation $\pi \in S_N$ let $\pi_{\downarrow n}$ denote the unique permutation of $[n]$ that can be expressed as the composition $\pi \circ f$ for some monotonically increasing $f : [n] \to [N]$. (This f maps the elements of $[n]$ to the elements of $\pi^{-1}([n])$ in increasing order.) If $\Pi_N \subseteq S_N$ satisfies $|\Pi_N| = m$ and $\overline{\mathcal{R}}_N(\Pi_N) \geq \alpha$ then the set $\Pi_n = \{\pi_{\downarrow n} \mid \pi \in \Pi_N\}$ certainly satisfies $|\Pi_n| \leq m$; we claim it also satisfies $\overline{\mathcal{R}}_n(\Pi_n) \geq \alpha$. To see why, consider any independent non-negative-valued random variables $X_1, \ldots, X_n$. Extend this to a sequence $X_1, \ldots, X_N$ by defining $X_i \equiv 0$ for $n < i \leq N$. Since $\overline{\mathcal{R}}_N(\Pi_N) \geq \alpha$ there is a $\pi \in \Pi$ and a threshold T such that $\mathbb{E}[\mathrm{ALG}_{\pi,T}] \geq \alpha \cdot \mathbb{E}[\mathrm{OPT}]$. (Note that $\max\{X_1, \ldots, X_N\} = \max\{X_1, \ldots, X_n\}$ so it is immaterial whether OPT is interpreted as referring to the former or the latter quantity.) Let $\pi' = \pi_{\downarrow n}$, and note that $\pi' \in \Pi_n$. For any threshold $(T, \tilde{T}) \in \mathbb{R} \times [0, 1]$ the stopping rule $\mathrm{ALG}(\pi, T, \tilde{T})$ stops at the earliest element in the sequence $X_{\pi(1)}, X_{\pi(2)}, \ldots, X_{\pi(n)}$ such that $(X_{\pi(i)}, \tilde{X}_{\pi(i)}) \geq (T, \tilde{T})$, whereas $\mathrm{ALG}(\pi', T, \tilde{T})$ stops at the earliest element such that $(X_{\pi(i)}, \tilde{X}_{\pi(i)}) \geq (T, \tilde{T})$ *and* $\pi(i) \in [n]$. These two elements are the same unless $T = 0$ and $\mathrm{ALG}_{\pi,T} = 0$, because when $\pi(i) \notin [n]$ we have $X_{\pi(i)} = 0$ by construction. Therefore, $\mathrm{ALG}_{\pi',T} \geq \mathrm{ALG}_{\pi,T}$ pointwise, and we find that

$$\mathbb{E}[\mathrm{ALG}_{\pi',T}] \geq \mathbb{E}[X_{\pi,T}] \geq \alpha \cdot \mathbb{E}[\mathrm{OPT}],$$

as desired. □

APPENDIX C

ADDITIONAL MATERIAL FROM CHAPTER 6

Here we present and slightly generalize the results of [12]. The framework and relevant notation is explained in Section 2.5. We begin by generalizing the definition of a matching to that of *semi-matching* which allows vertices on the server-side of the bipartite graph to be matched to more than one client.

Definition 19 (semi-matching)**.** A subset $M \subseteq E$ of the edges of $G = (L \cup R, E)$ is a *semi-matching* if every vertex $l \in L$ has exactly one incident edge in M.

It's important to note that semi-matchings are only used when initializing an assignment of clients to servers and we're *not* altering the definitions of free servers and augmenting paths nor the behavior of the Bipartite-SAP algorithm. When a fresh client arrives and requests to be matched, the algorithm will still try to find an alternating path from that client to a free server and augment down that path updating the semi-matching. The following lemma guarantees that the algorithm remains well-defined. As a reminder, Bipartite-SAP is applied on a graph where a set L_0 of clients are present at time $\tau = 0$ and clients in $D = L \backslash L_0$ arrive one at a time.

Lemma 30. *If M_0 is an initial semi-matching then the set M_τ maintained at the end of the τ-th iteration of the Bipartite-SAP algorithm remains a semi-matching for any $\tau \in [|D|]$.*

Proof. We proceed by induction on τ. Our assumption on M_0 covers the base case. Suppose $M_{\tau-1}$ is a semi-matching. Let u be a client on the augmenting path P_τ used to update $M_{\tau-1}$ to M_τ.

If u is the vertex x_τ that just arrived then no edge in $M_{\tau-1}$ is incident to it and exactly one edge incident to u enters M_τ.

If u is any other client then there are exactly two edges of P_τ incident on u, one belonging in $M_{\tau-1}$ and the other not. The former is in fact the only edge of $M_{\tau-1}$ incident to u because of the inductive hypothesis. Augmenting down P_τ means switching the state of those two edges so in M_τ there will still be exactly one edge incident to u. □

In what follows we'll denote by $N(v)$ the *neighborhood* of a vertex v and for a subset $A \subseteq L \cup R$ of the vertices, denote $N(A) = \bigcup_{v \in A} N(v)$. To prove their result, [12] introduce the notion of balanced flow. To avoid confusion with the notion of flow on a directed network we'll use the term *balanced function* instead to refer to the same concept.

Definition 20 (Balanced Function)**.** Let $\alpha : R \to \mathbb{R}_{\geq 0}$ be a function. Let $\text{Act}(l) := \text{argmin}_{r \in N(l)} \alpha(r)$ and call this set the *active neighborhood* of a client l. An edge $\{l, r\}$ where $r \in \text{Act}(l)$ is an *active edge*.

The function α is called *balanced* if there exist non-negative $(w_e)_{e \in E}$ such that

$$\begin{aligned} &\textstyle\sum_{r \in N(l)} w_{lr} = 1 && \forall l \in L \\ &\textstyle\sum_{l \in N(r)} w_{lr} = \alpha(r) && \forall r \in R \\ &w_{lr} = 0 && \forall l \in L, r \in N(l) \backslash \text{Act}(l) \end{aligned}$$

Essentially we can think of each element of $l \in L$ as having one unit of some quantity it must entirely distribute to its neighbors in a "balanced" way in the sense that it cannot redistribute its quantity in a way that strictly reduces the maximum load among its neighbors.

The following lemma — the proof of which we omit and can be found in [12] — guarantees that, under a reasonable condition, a balanced function exists and is uniquely defined.

Lemma 31 ([12, Lemma 14])**.** *A* unique *balanced function exists for a graph* $G = (L \cup R, E)$ *if and only if* $|N(l)| \geq 1$ *for all* $l \in L$*.*

In what follows we'll be working with graphs where each client has at least one neighbor (a consequence of the matching-admissibility assumption) and we'll denote by $\alpha(\cdot)$ the unique balanced function associated with that graph. Balanced functions exhibit the following useful properties which we provide without proof:

Lemma 32 ([12, Lemma 21 and 22])**.** *Denote by* $\alpha_\tau(\cdot)$ *the balanced function of the graph after the* τ*-th dynamic client has arrived for* $\tau \in [0..|D|]$*.*[1] *The change* $\Delta^\tau\alpha(r) = \alpha_\tau(r) - \alpha_{\tau-1}(r)$ *of each server* $r \in R$ *obeys the following properties for all* τ*:*

(a) $\Delta^\tau\alpha(r) \geq 0$ *for all* $r \in R$*.*

(b) $\Delta^\tau\alpha(r) = 0$ *for all* $r \in R$ *such that* $\alpha_{\tau-1}(r) < \mu(x_\tau) = \min_{v \in N(x_\tau)} \alpha_{\tau-1}(v)$*.*

Lemma 33 ([12, Lemma 23])**.** *The bipartite graph* $G = (L \cup R, E)$ *contains a matching of size* $|L|$ *if and only if* $\alpha(r) \leq 1$ *for all* $r \in R$*.*

In order to bound the length of augmenting paths originating from a client, it is convenient to bound the length of alternating paths originating from servers in the neighborhood of that client and thus the following definition introduced in [12] is useful: an *augmenting tail* is an alternating path from a server to a free server. Notice that every augmenting path has a unique maximal augmenting tail. Similarly, an *active augmenting tail* (under some semi-matching) is an augmenting tail for which the edges not belonging to the semi-matching are active.

We now proceed to prove a generalization of an essential lemma in [12]. This lemma allows us to assert the existence of an augmenting tail of bounded length from each

[1] For $\tau = 0$, $\alpha_0(\cdot)$ is the balanced function of the graph where the only clients present are the ones in L_0.

server r based on how far $\alpha(r)$ is from 1. Later, using the fact that each client contributes 1 to the sum of the balanced function values and Lemma 33 we'll be able to bound the total length of the augmenting paths used by the Bipartite-SAP.

Lemma 34. (Generalization of [12, Expansion Lemma]) *Let M be an semi-matching of a matching-admissible bipartite graph $G = (L \cup R, E)$ and $r \in R$ with $\alpha(r) \leq 1 - \varepsilon$ for some $\varepsilon > 0$. Then there exists an active augmenting tail from r to a free server of length at most $\frac{2}{\varepsilon} \ln(|L|)$.*

Proof. Notice that any server r' reachable from r by an active augmenting tail will have $\alpha(r') \leq \alpha(r) \leq 1 - \varepsilon$ by the definition of active edges. Let K_i for $i \geq 1$ be the set of all clients reachable from r by an active augmenting tail of length at most $2i - 1$, thus $K_1 \subseteq K_2 \subseteq \ldots \subseteq K_i$, and let $k_i = |K_i|$. Denote $\bigcup_{l \in K_i} \text{Act}(l)$ by $\text{Act}(K_i)$. We have,

$$\begin{aligned} k_i = |K_i| &\leq \sum_{r' \in \text{Act}(K_i)} \alpha(r') \\ &\leq \sum_{r' \in \text{Act}(K_i)} (1 - \varepsilon) \\ &= |\text{Act}(K_i)|(1 - \varepsilon) \end{aligned}$$

Now suppose there is no active augmenting tail of length $\leq 2i$. This means that all servers in $\text{Act}(K_i)$ are matched under M and furthermore, since M is a semi-matching, the function that maps each element of K_{i+1} that belongs to an edge of M to the opposite endpoint of that edge is a surjection from a subset of K_{i+1} to $\text{Act}(K_i)$, so $k_{i+1} \geq |\text{Act}(K_i)|$.

Thus, $k_{i+1} \geq \frac{k_i}{1-\varepsilon}$ i.e. the set of clients reachable by an active augmenting tail of length $2i - 1$ expands by a factor of at least $\frac{1}{1-\varepsilon}$ at each increment of i. Consequently,

$$|L| \geq k_{i+1} \geq \left(\frac{1}{1-\varepsilon}\right)^i k_1 \geq \left(\frac{1}{1-\varepsilon}\right)^i,$$

and thus

$$i \leq \frac{\ln |L|}{\ln \frac{1}{1-\varepsilon}}.$$

Using the inequality $1 - \varepsilon < e^{-\varepsilon}$ we get

$$i < \frac{1}{\varepsilon} \ln |L|.$$

We have shown that the hypothesis that no active augmenting tail has length $\leq 2i$ implies that $i < \frac{1}{\varepsilon} \ln |L|$. Hence, there must exist a free server reachable by an active augmenting tail of length at most $\frac{2}{\varepsilon} \ln |L|$. □

We are now ready to state and prove the main theorem which closely follows the proof techniques in [12].

Theorem 27. (Generalization of [12, Theorem 1, Lemma 6]) *Let $G = (L \cup R, E)$ be a matching-admissible bipartite graph and let M_0 be a semi-matching covering every vertex of some set $L_0 \subseteq L$. Suppose we run Bipartite-SAP on G with an initial semi-matching M_0 and the vertices in $D = L \backslash L_0$ arriving in an online fashion one at a time. The total vertex-switching cost of the algorithm is at most $O(n_c \log^2 n_c)$ where $n_c = |L|$.*

Proof. Recall that

$$\text{VSC}^G(\text{Bipartite-SAP}) \leq \frac{1}{2} \sum_{\tau=1}^{|D|} |P_\tau|,$$

where $|P_\tau|$ is the length of the τ-th augmenting path. Denoting by $m(h)$ the number of augmenting paths among the P_τ whose length is at least h, the sum can alternatively be computed as $\sum_{\tau=1}^{2n_c} m(h)$ — an augmenting path cannot have more than $2n_c$ edges. We're going to bound $m(h+2)$ from above by $\frac{4n_c \ln n_c}{h}$ from which then the result follows:

$$\begin{aligned}
\text{VSC}^G(\text{Bipartite-SAP}) &\leq \frac{1}{2} \sum_{h=1}^{2n_c} m(h) \\
&= \frac{1}{2} \left(m(1) + m(2) + \sum_{h=1}^{2n_c-2} m(h+2) \right) \\
&\leq \frac{1}{2} \left(|D| + |D| + 4n_c \ln n_c \sum_{h=1}^{2n_c} \frac{1}{h} \right) \\
&= O(n_c \log^2 n_c).
\end{aligned}$$

For what follows, fix an h and assume $h > 4 \ln n_c$; otherwise the bound $m(h+2) \leq n_c$ is trivial. Let x_τ be a client whose insertion resulted in an SAP of length $|P_\tau| \geq h+2$ and denote by $\alpha_{\tau-1}$ the balanced function before x_τ arrives. It follows that every server $r \in N(x_\tau)$ must have $\alpha_{\tau-1}(r) > 1 - \frac{2 \ln n_c}{h}$ and thus $\mu(x_\tau) > 1 - \frac{2 \ln n_c}{h}$. (Recall $\mu(x_\tau) = \min_{v \in N(x_\tau)} \alpha_{\tau-1}(v)$.) For supposing $\alpha_{\tau-1}(v) \leq 1 - \frac{2 \ln n_c}{h}$ for some $v \in N(x_\tau)$, by Lemma 34 there must exist an active augmenting tail of length at most h and so an augmenting path from x_τ of length at most $h+1$ which is a contradiction.

Consider the sequence of sets $S_\tau = \{r \in R \mid \alpha_\tau(r) > 1 - \frac{2 \ln n_c}{h}\}$ of clients for $\tau \in [0..|D|]$. By part (a) of Lemma 32, those sets are nested: $S_0 \subseteq S_1 \subseteq \ldots \subseteq S_{|D|}$. Define $\tau_0(r) = \min\{\tau \in [0..|D|] \mid r \in S_\tau\}$. Now we can bound $m(h+2)$ as follows:

$$
\begin{aligned}
m(h+2) &= \sum_{\tau : |P_\tau| \geq h+2} 1 && \text{(C.1)} \\
&= \sum_{\tau : |P_\tau| \geq h+2} \sum_{r \in R} \Delta^\tau \alpha(r) && \text{(C.2)} \\
&= \sum_{\tau : |P_\tau| \geq h+2} \sum_{r \in S_{\tau-1}} \Delta^\tau \alpha(r) && \text{(C.3)} \\
&\leq \sum_{1 \leq \tau \leq |D|} \sum_{r \in S_{\tau-1}} \Delta^\tau \alpha(r) && \text{(C.4)} \\
&= \sum_{r \in S_{|D|-1}} \sum_{\tau_0(r) < \tau \leq |D|} \Delta^\tau \alpha(r) && \text{(C.5)} \\
&= \sum_{r \in S_{|D|-1}} \left(\alpha_{|D|}(r) - \alpha_{\tau_0(r)}(r)\right) && \text{(C.6)} \\
&< \sum_{r \in S_{|D|-1}} \left(1 - \left(1 - \frac{2 \ln n_c}{h}\right)\right) && \text{(C.7)} \\
&\leq |S_{|D|}| \cdot \frac{2 \ln n_c}{h} && \text{(C.8)}
\end{aligned}
$$

Equation (C.2) holds because every new client contributes a total of 1 unit to the sum of the balanced functions. Equation (C.3) follows from part (b) of Lemma 32. Equa-

tion (C.5) is reversing the order of summation. Equation (C.6) is expanding the telescoping sum and eq. (C.7) follows from the observation that $\alpha_{|D|}(r) \leq 1$ for all servers r and $\alpha_{\tau_0(r)}(r) > 1 - \frac{2 \ln n_c}{h}$ as observed in a previous paragraph. The rest of the equations and inequalities follow easily using part (a) of Lemma 32 that all $\Delta^\tau \alpha(r) \geq 0$.

The final step of the proof is to bound $|S_{|L \setminus L_0|}|$. Notice that there are n_c total clients which can contribute to the values $\alpha_{|D|}(r)$ so $n_c \geq |S_{|D|}| \cdot \left(1 - \frac{2 \ln n_c}{h}\right) \geq \frac{|S_{|D|}|}{2}$ where the second inequality follows from the assumption that $h > 4 \ln n_c$. Hence we can conclude,

$$m(h+2) \leq \frac{4 n_c \ln n_c}{h}.$$

□

BIBLIOGRAPHY

[1] *Beyond the Worst-Case Analysis of Algorithms*. Cambridge University Press, 2021.

[2] Saeed Alaei. Bayesian combinatorial auctions: Expanding single buyer mechanisms to many buyers. In *2011 IEEE 52nd Annual Symposium on Foundations of Computer Science*, pages 512–521, 2011.

[3] Nima Anari, Rad Niazadeh, Amin Saberi, and Ali Shameli. Linear programming based near-optimal pricing for laminar bayesian online selection. *Chicago Booth Research Paper*, (20-24), 2019.

[4] Sepehr Assadi, Krzysztof Onak, Baruch Schieber, and Shay Solomon. Fully dynamic maximal independent set with sublinear update time. In *Proceedings of the 50th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2018, pages 815–826, New York, NY, USA, 2018. ACM.

[5] Pablo Azar, Silvio Micali, Constantinos Daskalakis, and S. Matthew Weinberg. Optimal and efficient parametric auctions. In *SODA*, pages 596–604, 2013.

[6] Pablo Daniel Azar and Silvio Micali. Parametric digital auctions. In *ITCS*, ITCS '13, page 231–232, New York, NY, USA, 2013. ACM.

[7] Yossi Azar, Ashish Chiplunkar, and Haim Kaplan. Prophet secretary: Surpassing the 1-1/e barrier. In *Proceedings of the 2018 ACM Conference on Economics and Computation*, EC '18, page 303–318, New York, NY, USA, 2018. Association for Computing Machinery.

[8] Xiaohui Bei, Nick Gravin, Pinyan Lu, and Zhihao Gavin Tang. Correlation-robust analysis of single item auction. In *SODA*, pages 193–208. SIAM, 2019.

[9] Dirk Bergemann and Stephen Morris. Robust mechanism design. *Econometrica*, 73(6):1771–1813, 2005.

[10] Dirk Bergemann and Stephen Morris. An introduction to robust mechanism design. *Foundations and Trends® in Microeconomics*, 8(3):169–230, 2013.

[11] Dirk Bergemann and Karl Schlag. Robust monopoly pricing. *Journal of Economic Theory*, 146(6):2527–2543, 2011.

[12] Aaron Bernstein, Jacob Holm, and Eva Rotenberg. Online bipartite matching with amortized o(log2 n) replacements. In *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '18, pages 947–959, Philadelphia, PA, USA, 2018. Society for Industrial and Applied Mathematics.

[13] Sergei Bernstein. *The theory of probabilities*. Gastehizdat Publishing House, 1946.

[14] Avrim Blum and Joel H. Spencer. Coloring random and semi-random k-colorable graphs. *J. Algorithms*, 19:204–234, 1995.

[15] Shant Boodaghians, Federico Fusco, Philip Lazos, and Stefano Leonardi. Pandora's box problem with order constraints. In *Proceedings of the 21st ACM Conference on Economics and Computation*, EC '20, page 439–458, New York, NY, USA, 2020. Association for Computing Machinery.

[16] A. Borodin and R. El-Yaniv. *Online Computation and Competitive Analysis*. Cambridge University Press, 2005.

[17] Bartłomiej Bosek, Dariusz Leniowski, Piotr Sankowski, and Anna Zych-Pawlewicz. Shortest augmenting paths for online matchings on trees. *Theory of Computing Systems*, 62(2):337–348, Feb 2018.

[18] Bartłomiej Bosek, Dariusz Leniowski, Piotr Sankowski, and Anna Zych-Pawlewicz. A tight bound for shortest augmenting paths on trees. In Michael A. Bender, Martín Farach-Colton, and Miguel A. Mosteiro, editors, *LATIN 2018: Theoretical Informatics*, pages 201–216, Cham, 2018. Springer International Publishing.

[19] Domagoj Bradac, Anupam Gupta, Sahil Singla, and Goran Zuzic. Robust algorithms for the secretary problem. In *ITCS*, 2019.

[20] Johannes Brustle, Yang Cai, and Constantinos Daskalakis. Multi-item mechanisms without item-independence: Learnability via robustness. In *EC*, EC '20, page 715–761, New York, NY, USA, 2020. ACM.

[21] Sébastien Bubeck, Nikhil Devanur, Zhiyi Huang, and Rad Niazadeh. Multi-scale online learning: Theory and applications to online auctions and pricing. *Journal of Machine Learning Research*, 2019.

[22] Niv Buchbinder, Kamal Jain, and Mohit Singh. Secretary problems via linear programming. volume 39, pages 163–176, 06 2010.

[23] Yang Cai and Constantinos Daskalakis. Learning multi-item auctions with (or without) samples. In *FOCS*, pages 516–527. IEEE, 2017.

[24] Vinicius Carrasco, Vitor Farinha Luz, Nenad Kos, Matthias Messner, Paulo Monteiro, and Humberto Moreira. Optimal selling mechanisms under moment conditions. *Journal of Economic Theory*, 177:245 – 279, 2018.

[25] Gabriel Carroll. Robustness and separation in multidimensional screening. *Econometrica*, 85(2):453–488, 2017.

[26] Keren Censor-Hillel, Elad Haramaty, and Zohar Karnin. Optimal dynamic distributed mis. In *Proceedings of the 2016 ACM Symposium on Principles of Distributed Computing*, PODC '16, pages 217–226, New York, NY, USA, 2016. ACM.

[27] K. Chaudhuri, C. Daskalakis, R. D. Kleinberg, and H. Lin. Online bipartite perfect matching with augmentations. In *IEEE INFOCOM 2009*, pages 1044–1052, April 2009.

[28] Jing Chen, Bo Li, and Yingkai Li. Information elicitation for Bayesian auctions. In *SAGT*, pages 43–55. Springer, 2018.

[29] Richard Cole and Tim Roughgarden. The sample complexity of revenue maximization. In *Proceedings of the forty-sixth annual ACM symposium on Theory of computing*, pages 243–252, 2014.

[30] Jose Correa, Andres Cristi, Paul Duetting, and Ashkan Norouzi-Fard. Fairness and bias in online selection. In *International Conference on Machine Learning*, pages 2112–2121. PMLR, 2021.

[31] José Correa, Patricio Foncea, Ruben Hoeksma, Tim Oosterwijk, and Tjark Vredeveld. Posted price mechanisms for a random stream of customers. In *Proceedings of the 2017 ACM Conference on Economics and Computation*, EC '17, page 169–186, New York, NY, USA, 2017. Association for Computing Machinery.

[32] José R. Correa, Raimundo Saona, and Bruno Ziliotto. Prophet secretary through blind strategies. In Timothy M. Chan, editor, *Proceedings of the Thirtieth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2019, San Diego, California, USA, January 6-9, 2019*, pages 1946–1961. SIAM, 2019.

[33] Thomas M Cover. Geometrical and statistical properties of systems of linear in-

equalities with applications in pattern recognition. *IEEE transactions on electronic computers*, (3):326–334, 1965.

[34] Nikhil Devanur, Zhiyi Huang, and Christos-Alexandros Psomas. The sample complexity of auctions with side information. In *STOC*, pages 426–439, 2016.

[35] Peerapong Dhangwatnotai, Tim Roughgarden, and Qiqi Yan. Revenue maximization with a single sample. *Games and Economic Behavior*, 91:318–333, 2015.

[36] Ilias Diakonikolas, Gautam Kamath, Daniel Kane, Jerry Li, Ankur Moitra, and Alistair Stewart. Robust estimators in high-dimensions without the computational intractability. *SIAM Journal on Computing*, 48(2):742–864, 2019.

[37] Shaddin Dughmi, Tim Roughgarden, and Mukund Sundararajan. Revenue submodularity. In *EC '09*, EC '09, page 243–252, New York, NY, USA, 2009. Association for Computing Machinery.

[38] Cynthia Dwork, Moritz Hardt, Toniann Pitassi, Omer Reingold, and Richard Zemel. Fairness through awareness. In *Proceedings of the 3rd innovations in theoretical computer science conference*, pages 214–226, 2012.

[39] Soheil Ehsani, MohammadTaghi Hajiaghayi, Thomas Kesselheim, and Sahil Singla. Prophet secretary for combinatorial auctions and matroids. In Artur Czumaj, editor, *Proceedings of the Twenty-Ninth Annual ACM-SIAM Symposium on Discrete Algorithms, SODA 2018, New Orleans, LA, USA, January 7-10, 2018*, pages 700–714. SIAM, 2018.

[40] Hossein Esfandiari, MohammadTaghi Hajiaghayi, Vahid Liaghat, and Morteza Monemizadeh. Prophet secretary. In Nikhil Bansal and Irene Finocchi, editors, *Algorithms - ESA 2015 - 23rd Annual European Symposium, Patras, Greece, September 14-16, 2015, Proceedings*, volume 9294 of *Lecture Notes in Computer Science*, pages 496–508. Springer, 2015.

[41] Uriel Feige and Joe Kilian. Heuristics for semirandom graph problems. *Journal of Computer and System Sciences*, 63(4):639–671, 2001.

[42] Andrew V. Goldberg, Jason D. Hartline, Anna R. Karlin, Michael Saks, and Andrew Wright. Competitive auctions. *Games and Economic Behavior*, 55(2):242 – 269, 2006. Mini Special Issue: Electronic Market Design.

[43] Nick Gravin and Pinyan Lu. Separation in correlation-robust monopolist problem with budget. In *SODA*, pages 2069–2080. SIAM, 2018.

[44] Edward F. Grove, Ming-Yang Kao, P. Krishnan, and Jeffrey Scott Vitter. Online perfect matching and mobile computing. In Selim G. Akl, Frank Dehne, Jörg-Rüdiger Sack, and Nicola Santoro, editors, *Algorithms and Data Structures*, pages 194–205, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg.

[45] Anupam Gupta, Ravishankar Krishnaswamy, Amit Kumar, and Debmalya Panigrahi. Online and dynamic algorithms for set cover. In *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing*, STOC 2017, pages 537–550, New York, NY, USA, 2017. ACM.

[46] Anupam Gupta, Amit Kumar, and Cliff Stein. Maintaining assignments online: Matching, scheduling, and flows. In *Proceedings of the Twenty-fifth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '14, pages 468–479, Philadelphia, PA, USA, 2014. Society for Industrial and Applied Mathematics.

[47] Mohammad Taghi Hajiaghayi, Robert Kleinberg, and Tuomas Sandholm. Automated online mechanism design and prophet inequalities. In *Proceedings of the 22nd National Conference on Artificial Intelligence - Volume 1*, AAAI'07, page 58–65. AAAI Press, 2007.

[48] MohammadTaghi Hajiaghayi, Robert Kleinberg, and Tom Leighton. Semi-oblivious routing: Lower bounds. In *Proceedings of the Eighteenth Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '07, pages 929–938, Philadelphia, PA, USA, 2007. Society for Industrial and Applied Mathematics.

[49] Jason D. Hartline. Mechanism design and approximation. `http://jasonhartline.com/MDnA/MDnA-ch3.pdf`. Accessed: 2020-07-13.

[50] Jason D Hartline and Tim Roughgarden. Simple versus optimal mechanisms. In *EC*, 2009.

[51] Victor Heorhiadi, Michael K. Reiter, and Vyas Sekar. Simplifying software-defined network optimization using SOL. In *13th USENIX Symposium on Networked Systems Design and Implementation, NSDI 2016, Santa Clara, CA, USA, March 16-18, 2016*, pages 223–237, 2016.

[52] T. P. Hill. Prophet inequalities and order selection in optimal stopping problems. *Proc. AMS*, 88(1):131–137, 1983.

[53] Chi-Yao Hong, Srikanth Kandula, Ratul Mahajan, Ming Zhang, Vijay Gill, Mohan Nanduri, and Roger Wattenhofer. Achieving High Utilization with Software-Driven WAN. In *Proceedings of ACM SIGCOMM 2013*, 2013.

[54] Zhiyi Huang, Yishay Mansour, and Tim Roughgarden. Making the most of your samples. *SIAM Journal on Computing*, 47(3):651–674, 2018.

[55] Sushant Jain, Alok Kumar, Subhasree Mandal, Joon Ong, Leon Poutievski, Arjun Singh, Subbaiah Venkata, Jim Wanderer, Junlan Zhou, Min Zhu, Jonathan Zolla, Urs Hölzle, Stephen Stuart, and Amin Vahdat. B4: Experience with a Globally Deployed Software Defined WAN. In *Proceedings of ACM SIGCOMM 2013*, 2013.

[56] Nanxi Kang, Monia Ghobadi, John Reumann, Alexander Shraer, and Jennifer Rexford. Efficient traffic splitting on commodity switches. In *Proceedings of the 11th ACM Conference on Emerging Networking Experiments and Technologies*, CoNEXT '15, pages 6:1–6:13, New York, NY, USA, 2015. ACM.

[57] Ian Kash, Ariel D. Procaccia, and Nisarg Shah. No agent left behind: Dynamic fair division of multiple resources. In *Proceedings of the 2013 International Conference on Autonomous Agents and Multi-Agent Systems*, AAMAS '13, page 351–358, Richland, SC, 2013. International Foundation for Autonomous Agents and Multiagent Systems.

[58] Thomas Kesselheim, Robert D. Kleinberg, and Rad Niazadeh. Secretary problems with non-uniform arrival order. In *Proceedings of the Forty-Seventh Annual ACM Symposium on Theory of Computing, STOC 2015, Portland, OR, USA, June 14-17, 2015*, pages 879–888. ACM, 2015.

[59] Jon Kleinberg, Jens Ludwig, Sendhil Mullainathan, and Ashesh Rambachan. Algorithmic fairness. *AEA Papers and Proceedings*, 108:22–27, May 2018.

[60] Jon M. Kleinberg and Robert Kleinberg. Delegated search approximates efficient search. In Éva Tardos, Edith Elkind, and Rakesh Vohra, editors, *Proceedings of the 2018 ACM Conference on Economics and Computation, Ithaca, NY, USA, June 18-22, 2018*, pages 287–302. ACM, 2018.

[61] Robert Kleinberg and Seth Matthew Weinberg. Matroid prophet inequalities. In *Proceedings of the Forty-Fourth Annual ACM Symposium on Theory of Computing*, STOC '12, page 123–136, New York, NY, USA, 2012. Association for Computing Machinery.

[62] Ulrich Krengel and Louis Sucheston. Semiamarts and finite values. *Bull. Amer. Math. Soc.*, 83:745–747, 1977.

[63] Ulrich Krengel and Louis Sucheston. On semiamarts, amarts, and processes with finite value. *Adv. in Prob. Related Topics*, 4:197–266, 1978.

[64] Praveen Kumar, Yang Yuan, Chris Yu, Nate Foster, Robert Kleinberg, Petr Lapukhov, Chiun Lin Lim, and Robert Soulé. Semi-oblivious traffic engineering: The road not taken. In *Proceedings of the 15th USENIX Symposium on Networked Systems Design and Implementation (NSDI)*, 2018.

[65] Kevin A Lai, Anup B Rao, and Santosh Vempala. Agnostic estimation of mean and covariance. In *FOCS*, pages 665–674. IEEE, 2016.

[66] Leslie Lamport, Robert Shostak, and Marshall Pease. The byzantine generals problem. *ACM Transactions on Programming Languages and Systems (TOPLAS)*, 4(3):382–401, 1982.

[67] Robert W. Lien, Seyed M. R. Iravani, and Karen R. Smilowitz. Sequential resource allocation for nonprofit operations. *Operations Research*, 62(2):301–317, 2014.

[68] Henry Lin. Reducing directed max flow to undirected max flow, 2009. unpublished manuscript.

[69] Brendan Lucier. An economic view of prophet inequalities. *SIGecom Exch.*, 16(1):24–47, sep 2017.

[70] Thodoris Lykouris, Vahab Mirrokni, and Renato Paes Leme. Stochastic bandits robust to adversarial corruptions. In *STOC*, pages 114–122, 2018.

[71] Aleksander Madry. Navigating central path with electrical flows: From flows to matchings, and back. In *2013 IEEE 54th Annual Symposium on Foundations of Computer Science*, pages 253–262. IEEE, 2013.

[72] Konstantin Makarychev, Yury Makarychev, and Aravindan Vijayaraghavan. Approximation algorithms for semi-random partitioning problems. In *STOC*, pages 367–384, 2012.

[73] Vahideh H. Manshadi, Rad Niazadeh, and Scott Rodilitz. Fair dynamic rationing. In Péter Biró, Shuchi Chawla, and Federico Echenique, editors, *EC '21: The 22nd ACM Conference on Economics and Computation, Budapest, Hungary, July 18-23, 2021*, pages 694–695. ACM, 2021.

[74] Michael Mitzenmacher and Eli Upfal. *Probability and Computing: Randomized Algorithms and Probabilistic Analysis*. Cambridge University Press, USA, 2005.

[75] Jamie Morgenstern and Tim Roughgarden. Learning simple auctions. In *CoLT*, 2016.

[76] Roger B. Myerson. Optimal auction design. *Math. Oper. Res.*, 6(1):58–73, February 1981.

[77] Joseph Naor and Moni Naor. Small-bias probability spaces: Efficient constructions and applications. *SIAM J. Comput.*, 22(4):838–856, 1993.

[78] N. Nisan, T. Roughgarden, E. Tardos, and V.V. Vazirani. *Algorithmic Game Theory*. Cambridge University Press, 2007.

[79] James G. Oxley. *Matroid Theory (Oxford Graduate Texts in Mathematics)*. Oxford University Press, Inc., USA, 2006.

[80] B. Peng and Z. Tang. Order selection prophet inequality: From threshold optimization to arrival time design. In *2022 IEEE 63rd Annual Symposium on Foundations of Computer Science (FOCS)*, pages 171–178, Los Alamitos, CA, USA, nov 2022. IEEE Computer Society.

[81] Steven Phillips and Jeffery Westbrook. Online load balancing and network flow. In *Proceedings of the Twenty-fifth Annual ACM Symposium on Theory of Computing*, STOC '93, pages 402–411, New York, NY, USA, 1993. ACM.

[82] Harald Räcke. Optimal hierarchical decompositions for congestion minimization in networks. In *Proceedings of the Fortieth Annual ACM Symposium on Theory of Computing*, STOC '08, pages 255–264, New York, NY, USA, 2008. ACM.

[83] Baharak Rastegari, Anne Condon, and Kevin Leyton-Brown. Revenue monotonicity in combinatorial auctions. *SIGecom Exch.*, 7(1):45–47, December 2007.

[84] Alvin Roth, Tayfun Sönmez, and Utku Unver. Pairwise kidney exchange. Game theory and information, University Library of Munich, Germany, 2005.

[85] Tim Roughgarden. Beyond worst-case analysis, 2018.

[86] Ester Samuel-Cahn. Comparison of threshold stop rules and maximum for independent nonnegative random variables. *The Annals of Probability*, 12(4):1213–1216, 1984.

[87] Peter Sanders, Naveen Sivadasan, and Martin Skutella. Online scheduling with bounded migration. *Math. Oper. Res.*, 34(2):481–498, May 2009.

[88] Sean R. Sinclair, Gauri Jain, Siddhartha Banerjee, and Christina Lee Yu. Se-

quential fair allocation of limited resources under stochastic demands. *ArXiv*, abs/2011.14382, 2020.

[89] Amnon Ta-Shma. Explicit, almost optimal, epsilon-balanced codes. In Hamed Hatami, Pierre McKenzie, and Valerie King, editors, *Proceedings of the 49th Annual ACM SIGACT Symposium on Theory of Computing, STOC 2017, Montreal, QC, Canada, June 19-23, 2017*, pages 238–251. ACM, 2017.

[90] Pafnuty Tchebychev. Mémoire sur les nombres premiers. *Journal de mathématiques pures et appliquées, Sér. 1*, pages 366–390, 1852.

[91] Manish Tripathy, Jiaru Bai, and H. Sebastian (Seb) Heese. Driver collusion in ride-hailing platforms. *Decision Sciences*, n/a(n/a), 2022.

[92] Sahil Verma and Julia Rubin. Fairness definitions explained. In *Proceedings of the International Workshop on Software Fairness*, FairWare '18, page 1–7, New York, NY, USA, 2018. Association for Computing Machinery.

[93] Toby Walsh. Online cake cutting. In Ronen I. Brafman, Fred S. Roberts, and Alexis Tsoukiàs, editors, *Algorithmic Decision Theory*, pages 292–305, Berlin, Heidelberg, 2011. Springer Berlin Heidelberg.

[94] Jeffery Westbrook. Load balancing for response time. *J. Algorithms*, 35(1):1–16, April 2000.

[95] David P. Williamson. *Network Flow Algorithms*. Cambridge University Press, 2019.

[96] Robert Wilson. *Game-theoretic analyses of trading processes*, page 33–70. Econometric Society Monographs. Cambridge University Press, 1987.