Alea Iacta Est: Auctions, Persuasion, Interim Rules, and Dice

Shaddin Dughmi
University of Southern California, Los Angeles, CA, USA
shaddin@usc.edu

David Kempe
University of Southern California, Los Angeles, CA, USA
David.M.Kempe@gmail.com

Ruixin Qiang
University of Southern California, Los Angeles, CA, USA
rqiang@usc.edu

Abstract

To select a subset of samples or "winners" from a population of candidates, order sampling [20] and the k-unit Myerson auction [17] share a common scheme: assign a (random) score to each candidate, then select the k candidates with the highest scores. We study a generalization of both order sampling and Myerson's allocation rule, called *winner-selecting dice*. The setting for winner-selecting dice is similar to auctions with feasibility constraints: candidates have random types drawn from independent prior distributions, and the winner set must be feasible subject to certain constraints. Dice (distributions over scores) are assigned to each type, and winners are selected to maximize the sum of the dice rolls, subject to the feasibility constraints. We examine the existence of winner-selecting dice that implement prescribed probabilities of winning (i.e., an interim rule) for all types.

Our first result shows that when the feasibility constraint is a matroid, then for any feasible interim rule, there always exist winner-selecting dice that implement it. Unfortunately, our proof does not yield an efficient algorithm for constructing the dice. In the special case of a 1-uniform matroid, i.e., only one winner can be selected, we give an efficient algorithm that constructs winner-selecting dice for any feasible interim rule. Furthermore, when the types of the candidates are drawn in an i.i.d. manner and the interim rule is symmetric across candidates, unsurprisingly, an algorithm can efficiently construct symmetric dice that only depend on the type but not the identity of the candidate.

One may ask whether we can extend our result to "second-order" interim rules, which not only specify the winning probability of a type, but also the winning probability conditioning on each other candidate's type. We show that our result does not extend, by exhibiting an instance of *Bayesian persuasion* whose optimal scheme is equivalent to a second-order interim rule, but which does not admit any dice-based implementation.

2012 ACM Subject Classification Theory of computation → Algorithmic game theory

Keywords and phrases Interim rule, order sampling, virtual value function, Border's theorem

Digital Object Identifier 10.4230/LIPIcs.ITCS.2019.31

Related Version A full version of the paper is available at `https://arxiv.org/abs/1811.11417`.

Funding Work supported in part by NSF Grant CCF-1423618.

1 Introduction

Economic design often features scenarios in which choices must be made based on stochastic inputs. In auction design, bidders drawn from a population interact with an auction mechanism, and the mechanism must then choose winners and losers of the auction. In information structure design, a principal observes information pertinent to the various actions available to one or more decision makers, and must use this information to recommend actions to the decision makers. In such settings, an important framing device is the notion of an *interim rule* (also called a *reduced form*) of an (ex-post) winner selection rule, summarizing the probability for each candidate to be selected.

We focus on the simplest and most natural class of such decision making scenarios, one which includes auctions and Bayesian persuasion [12] as special cases. In a *winner selection* environment, there is a set of *candidates* $\mathcal{C}$, each equipped with a random attribute known as its *type*. A *winner selection rule* is a randomized function (or algorithm) which maps each profile of types, one per candidate, to a choice of winning candidates, subject to the requirement that the set of winners must belong to a specified family $\mathcal{I} \subseteq 2^{\mathcal{C}}$ of feasible sets. The winner selection rule is also referred to as an ex-post rule since it specifies the winning probabilities conditioned on every realized type profile. In auctions, candidates correspond to bidders, and a winner selection rule is an allocation rule of the auction. In Bayesian persuasion, candidates correspond to actions available to a decision maker, and a winner selection rule corresponds to a persuasion scheme used by a principal to recommend one of the actions to the decision maker. We restrict our attention to winner selection scenarios in which the types of different candidates are independently distributed.

We distinguish two classes of interim rules: *first-order* and *second-order*. The former is the traditional notion from auction theory, while the latter is the notion better suited for persuasion. A *first-order interim rule* specifies, for each candidate i and type t of candidate i, the conditional probability of i winning given that his type is t. A *second-order interim rule* specifies more information: for each pair of candidates i, j and type t of candidate j, it specifies the conditional probability of i winning given that j has type t. First-order interim rules, when combined with a payment rule, suffice for evaluating the welfare, revenue, and incentive-compatibility of a single-item auction. For Bayesian persuasion, second-order interim rules are needed for evaluating the incentive constraints of a persuasion scheme.

Our motivation for studying winner selection at this level of generality stems from the success of Myerson's [17] famous and elegant characterization of revenue-optimal single-item auctions when bidder type distributions are independent. In that special case, Myerson showed that the optimal *single-item* auction features a particularly structured winner-selection rule: each type is associated with a virtual value, and given a profile of reported types, the rule selects the bidder with the highest (non-negative) virtual value as the winner.

Interestingly, order sampling [20] works in a similar way as Myerson's auction, in the special case when the feasible sets are sets of k or fewer candidates, and each candidate has only one type. Order sampling assigns each candidate a random score variable (a die). The winners are the candidates with the k highest score variables. In the language of order sampling, virtual value functions define a single-sided die for each type. Unlike Myerson's auction, there is no notion of "revenue" or "incentive constraints" in order sampling. The task is simply to find dice that will induce a prescribed first-order interim rule.

As a generalization of order sampling and Myerson's virtual-value approach, a *dice-based winner-selection rule* assigns each type a die, and selects the feasible set of winners maximizing the sum of the dice rolls. *We explore the extent to which dice-based rules are applicable beyond*

single-parameter auctions or single-type environments, to winner selection with independent type distributions under more complex constraints. In particular, we examine whether dice-based winner-selection rules exist for winner selection subject to matroid constraints and for Bayesian persuasion.

1.1 Our Results

As mentioned previously, all of our results are restricted to settings in which the candidates' type distributions are independent. It follows from Myerson's characterization that every first-order interim rule *corresponding to some optimal auction* admits a dice-based implementation.[1] Our main result (in Section 3) is an existential proof showing that *every* feasible first-order interim rule with respect to a matroid constraint admits a dice-based implementation. This illustrates that the structure revealed by Myerson's characterization is more general, and applies to other settings in which only first-order interim information is relevant. For example, single-item auctions with (public or private) budgets are such a setting (see, e.g., [19]). Our result also provides a generalization of order sampling from the k-winner setting to the general matroid, multi-type setting.

Beyond the existential proof of dice-based implementations, we show (in Section 4) that for single-winner environments, an algorithm can construct the dice-based rule *efficiently*. When the types are identically distributed, we also constructively show (in the full version) that every first-order interim rule which is symmetric across candidates admits a symmetric dice implementation; i.e., different candidates have the same die for the same type. This is consistent with Myerson's symmetric characterization of optimal single-item auctions with i.i.d. bidders, and generalizes it to any other first-order single-winner selection setting in which candidates are identical. Single-item auctions with identically distributed budgeted bidders are such a setting, and a symmetric dice-based implementation of the optimal allocation rule was already known from [19].

For single-winner cases, we also show the converse direction: how to efficiently compute the first-order interim rule of a given dice-based winner selection rule. In effect, these results show that collections of dice are a computationally equivalent description of single-winner first-order interim information. This implies a kind of equivalence between the two dominant approaches for mechanism design: the Myersonian approach based on virtual values (i.e., dice), and the Borderian approach based on optimization over interim rules.

In an attempt to leverage the same kinds of insights for Bayesian persuasion, we examine (in Sections 5 and 6) the dice implementability of second-order interim rules. When the candidate type distributions are non-identical, we show an impossibility result. We construct an instance of Bayesian persuasion with independently distributed non-identical actions, and show that no optimal persuasion scheme for this instance can be implemented by dice. Since second-order interim rules are sufficient for evaluating the objective and constraints of Bayesian persuasion, this implies that there exist second-order interim rules which are not dice-implementable. This rules out the Myersonian approach for characterizing and computing optimal schemes for Bayesian persuasion with independent non-identical actions, complementing the negative result of [8] which rules out the Borderian approach for the same problem.

[1] As we show in the full version, there are interim rules which are not optimal for any auction.

Our impossibility result disappears when the actions are i.i.d., since second-order interim rules collapse to first-order interim rules in symmetric settings. In particular, as we show in Section 6, our results for first-order interim rules, combined with those of [8], imply that Bayesian persuasion with i.i.d. actions admits an optimal dice-based scheme, which can be computed efficiently.

1.2 Additional Discussion of Related Work

Myerson [17] was the first to characterize revenue-optimal single-item auctions; this characterization extends to single-parameter mechanism design settings more generally (see, e.g., [11]). The (first-order) interim rule of an auction, also known as its reduced form, was first studied by Maskin and Riley [14] and Matthews [15]. The inequalities characterizing the space of feasible interim rules were described by Border [3, 4]. Border's analytically tractable characterization of feasible interim rules has served as a fruitful framework for mechanism design, since an optimal auction can be viewed as the solution of a linear program over the space of interim rules. Moreover, this characterization has enabled the design of efficient algorithms for recognizing interim rules and optimizing over them, by Cai et al. [5] and Alaei et al. [2]. This line of work has served as a foundation for much of the recent literature on Bayesian algorithmic mechanism design in multi-parameter settings.

It is important to contrast our dice-based rule with the characterization of Cai et al. [5]. In particular, the results of Cai et al. [5] imply that every first-order interim rule can be efficiently implemented as a distribution over virtual value maximizers. In our language, this implies the existence of an efficiently computable dice-based implementation *in which the dice may be arbitrarily correlated*. Our result, in contrast, efficiently computes a family of *independent dice* implementing any given first-order interim rule in single-winner settings, and shows the existence of a dice-based rule in matroid settings. This is consistent with Myerson's characterization, in which virtual values are drawn independently.

Alaei et al. [2] also studied winner-selection environments, under the different name "service based environments." For single-winner settings, they proposed a mechanism called stochastic sequential allocation (SSA). The mechanism also implements any feasible first-order interim rule, by creating a token of winning and transferring the token sequentially from one candidate to another, with probabilities defined by an efficiently computed transition table. Dice can be considered as the special case of SSA in which the transition probabilities are independent of the current owner of the token.

As another motivation for our focus on dice-based rules, order sampling studies how to sample k winners from n candidates with given inclusion probabilities (i.e., implement an interim rule), by assigning a random score variable (die) to each candidate. Rosén [20] showed that parameterized Pareto distributions can be used to implement a given interim rule asymptotically. Aires et al. [1] proved the existence of an order sampling scheme that exactly implements any feasible interim rule. Our existential proof is a generalization of the proof of [1] to settings with multiple types and matroid constraints.

The Bayesian persuasion model is due to Kamenica and Gentzkow [12], and is the most influential model in the general space of information structure design (see the survey by Dughmi [7] for references). Bayesian persuasion was examined algorithmically by Dughmi and Xu [8], who observed its connection to auction theory and interim rules, and examined the computational complexity of optimal schemes through the lens of optimization over interim rules.

Of particular relevance to our work is the negative result of Dughmi and Xu [8] for Bayesian persuasion with independent non-identical actions: it is #P-hard to compute the interim rule (first- or second-order) of the optimal scheme, or more simply even the sender's optimal utility. Most notable about this result is what it *does not* rule out: an algorithm implementing the optimal persuasion scheme "on the fly," in the sense that it efficiently samples the optimal scheme's (randomized) recommendation when given as input the profile of action types. Stated differently, the negative result of Dughmi and Xu [8] merely rules out the Borderian approach for this problem, leaving other approaches – such as the Myersonian one – viable as a means of obtaining an efficient "on the fly" implementation. This would not be unprecedented: Gopalan et al. [9] exhibit a simple single-parameter auction setting for which the optimal interim rule is #P hard to compute, yet Myerson's virtual values can be sampled efficiently and used to efficiently implement the optimal auction. Our negative result in Section 6 rules out such good fortune for Bayesian persuasion with independent non-identical actions: there does not exist a (Myersonian) dice-based implementation of the optimal persuasion scheme in general.

2 Preliminaries

2.1 Winner Selection

Consider choosing a set of winners from among n *candidates*. Each candidate i has a type $t_i \in T_i$, drawn independently from a distribution f_i. A winner-selection rule $\mathcal{A}$ maps each type profile $\boldsymbol{t} = (t_1, \ldots, t_n)$, possibly randomly, to one of a prescribed family of feasible sets $\mathcal{I} \subseteq 2^{[n]}$. When $i \in \mathcal{A}(\boldsymbol{t})$, we refer to i as a *winning candidate*, and to t_i as his *winning type*. Writing $f = f_1 \times \cdots \times f_n$ for the (independent) joint type distribution, we also refer to $(f, \mathcal{I})$ as the *winner-selection environment*. When $\mathcal{I}$ is the family of singletons, as in the setting of the single item auction, we call $(f, \mathcal{I})$ a *single-winner environment*.

This general setup captures the allocation rules of general auctions with independent unit-demand buyers, albeit without specifying payment rules or imposing incentive constraints. Moreover, it captures Bayesian persuasion with independent action payoffs, albeit without enforcing persuasiveness (also called *obedience*) constraints.

2.2 Matroids

In this paper, we focus on settings in which the feasible sets $\mathcal{I}$ are the independent sets of a matroid. We use the standard definition of a matroid $\mathcal{M}$ as a pair $(E, \mathcal{I})$, where E is the *ground set* and $\mathcal{I} \subseteq 2^E$ is a family of so-called *independent sets*, satisfying the three matroid axioms. We also use the standard definitions of a *circuit* and *rank function* $r_{\mathcal{M}} : 2^E \to \mathbb{N}$. The *restriction* $\mathcal{M}|S$ of $\mathcal{M} = (E, \mathcal{I})$ to some $S \subseteq E$ is the matroid $(E, \mathcal{I} \cap 2^S)$.[2] For details on matroids, we refer the reader to Oxley [18].

A matroid $\mathcal{M} = (E, \mathcal{I})$ is *separable* if it is a *direct sum* of two matroids $\mathcal{M}_1 = (E_1, \mathcal{I}_1)$ and $\mathcal{M}_2 = (E_2, \mathcal{I}_2)$. Namely, $E = E_1 \uplus E_2$, $\mathcal{I} = \{A \cup B \mid A \in \mathcal{I}_1, B \in \mathcal{I}_2\}$. Note that if $\mathcal{M}$ is non-separable, then $r_{\mathcal{M}}(E) < |E|$; otherwise $\mathcal{M}$ is the direct sum of singleton matroids. We use the following theorem.

▶ **Theorem 1** (Whitney [21]). *(1) When $\mathcal{M} = (E, \mathcal{I})$ is a non-separable matroid, for every $a, b \in E$, there is a circuit containing both a and b. (2) Any separable matroid $\mathcal{M}$ is a direct sum of two or more non-separable matroids called the* components *of $\mathcal{M}$.*

[2] Note that we deviate slightly from the standard definition in that we do not restrict the ground set.

In most of the remainder of the paper, we focus on winner selection environments $(f, \mathcal{I})$ where $\mathcal{I}$ is the family of independent sets of a matroid $\mathcal{M}$. We therefore also use $(f, \mathcal{M})$ to denote the environment.

2.3 Interim Rules and Border's Theorem

A *(first-order) interim rule* $\boldsymbol{x}$ specifies the winning probability $x_i(t) \in [0,1]$ for all $i \in [n], t \in T_i$ in an environment $(f, \mathcal{I})$. More precisely, we say that a winner-selection rule $\mathcal{A}$ *implements* the interim rule $\boldsymbol{x}$ for a prior f if it satisfies the following: if the type profile $\boldsymbol{t} = (t_1, \ldots, t_n)$ is drawn from the prior distribution $f = f_1 \times \cdots \times f_n$, then $\mathbf{Pr}[i \in \mathcal{A}(\boldsymbol{t}) \mid t_i = t] = x_i(t)$. An interim rule is *feasible* (or *implementable*) within an environment $(f, \mathcal{I})$ if there is a winner-selection rule implementing it that always outputs an independent set of $\mathcal{I}$.

2.3.1 Border's theorem and implications for single-winner environments

The following theorem characterizes the space of feasible interim rules for single-winner settings.

▶ **Theorem 2** (Border [3, 4])**.** *An interim rule $\boldsymbol{x}$ is feasible for a single-winner setting if and only if for all possible type subsets $S_1 \subseteq T_1, S_2 \subseteq T_2, \ldots, S_n \subseteq T_n$,*

$$\sum_{i=1}^{n} \sum_{t \in S_i} f_i(t) x_i(t) \leq 1 - \prod_{i=1}^{n} \left(1 - \sum_{t \in S_i} f_i(t)\right). \tag{1}$$

The following result leverages Theorem 2 to show that efficient algorithms exist for checking the feasibility of an interim rule, and for implementing a feasible interim rule.

▶ **Theorem 3** ([5, 2])**.** *Given explicitly represented priors $f_1, \ldots, f_n$ and an interim rule $\boldsymbol{x}$ in a single-winner setting, the feasibility of $\boldsymbol{x}$ can be checked in time polynomial in the number of candidates and types. Moreover, given a feasible interim rule $\boldsymbol{x}$, an algorithm can find a winner-selection rule implementing $\boldsymbol{x}$ in time polynomial in the number of candidates and types.*

In our efficient construction for single-winner settings, we utilize a structural result which shows that checking only a subset of Border's constraints suffices [3, 16, 5]. This subset of constraints can be identified efficiently.

▶ **Theorem 4** (Theorem 4 of [5])**.** *An interim rule $\boldsymbol{x}$ is feasible for a single-winner setting if and only if for all possible $\alpha \in [0,1]$, the sets $S_i(\alpha) = \{t \in T_i \mid x_i(t) > \alpha\}$ satisfy the following Border's constraint:*

$$\sum_{i=1}^{n} \sum_{t \in S_i(\alpha)} f_i(t) x_i(t) \leq 1 - \prod_{i=1}^{n} \left(1 - \sum_{t \in S_i(\alpha)} f_i(t)\right).$$

When the candidates' type distributions are i.i.d., i.e., T_i and f_i are the same for all candidates i, it is typically sufficient to restrict attention to *symmetric* interim rules. For such rules, $x_i(t)$ is equal for all candidates i. In the i.i.d. setting, we therefore notationally omit the dependence on the candidate and let T refer to the common type set of all candidates, f to the candidates' (common) type distribution, and $x(t)$ to the probability that a particular candidate wins conditioned on having type t. In the i.i.d. setting, only the symmetric constraints from Theorem 2 suffice to characterize feasibility [3]; namely,

$$n \cdot \sum_{t \in S} f(t) x(t) \;\leq\; 1 - \left(1 - \sum_{t \in S} f(t)\right)^{n}, \tag{2}$$

for all $S \subseteq T$. Theorem 4 then implies that it suffices to check Inequality (2) for sets of the form $S(\alpha) = \{t \in T : x(t) > \alpha\}$ with $\alpha \in [0, 1]$.

2.3.2 Border's Theorem for matroid environments

For general settings with matroid constraints, Alaei et al. [2] established the following generalized "Border's Theorem."

▶ **Theorem 5** (Theorem 7 of [2]). *Let ν map each type t to the (unique) candidate i with $t \in T_i$. An interim rule $\boldsymbol{x}$ is feasible within an environment $(f, \mathcal{M})$ if and only if for all possible type subsets $S_1 \subseteq T_1, S_2 \subseteq T_2, \ldots, S_n \subseteq T_n$, $\sum_{i=1}^{n} \sum_{t \in S_i} f_i(t) x_i(t) \leq \mathbb{E}_{\boldsymbol{t} \sim f} \left[r_{\mathcal{M}}(\nu(\boldsymbol{t} \cap S)) \right]$, where $S = \bigcup_{i=1}^{n} S_i$.*

In later sections, we omit the function ν, and for any type set S just write $r_{\mathcal{M}}(S)$ instead of $r_{\mathcal{M}}(\nu(S))$.

2.4 Winner-Selecting Dice

We study winner-selection rules based on *dice*, as a generalization of order sampling to multiple types and general constraints. A *dice-based rule* fixes, for each type $t \in T_i$, a distribution $D_{i,t}$ over real numbers, which we call a *die*. Given as input the type profile $\boldsymbol{t} = (t_1, \ldots, t_n)$, the rule independently draws a score $v_i \sim D_{i,t_i}$ for each candidate i by "rolling his die;" it then selects the feasible set of candidates maximizing the sum of scores as the winner set, breaking ties with a predefined rule. In this paper, we will mainly discuss matroid feasibility constraints, for which a feasible set maximizing the sum of scores can be found by a simple greedy algorithm: candidates are added to the winner set in decreasing order of their scores, breaking ties uniformly at random, as long as the new winner set is still an independent set of the matroid and their scores are positive. When candidates have the same type sets, we call a dice-based rule *symmetric* if $D_{i,t}$ is the same for all i.

Myerson's optimal auction is a dice-based winner-selection rule. In Myerson's nomenclature, $\overline{v}_i$ is candidate i's *virtual value*, and $D_{i,t}$ is a single-sided die with the virtual value.

Let T be the set of all types of all candidates and $\mathcal{D} = (D_t)_{t \in T}$ be a vector of dice, one per type. Given an interim rule $\boldsymbol{x}$, and a winner-selection environment $(f, \mathcal{I})$, we say that $\mathcal{D}$ *implements* $\boldsymbol{x}$, or $\mathcal{D}$ describes *winner-selecting dice* for $\boldsymbol{x}$ in $(f, \mathcal{I})$, if the dice-based rule given by $\mathcal{D}$ implements $\boldsymbol{x}$ within the environment $(f, \mathcal{I})$.

2.5 Second-order Interim Rules

A (first-order) interim rule, as defined in Section 2.3, specifies, for each candidate i, the conditional type distribution of i in the event that i is chosen as the winner. We define a *second-order interim rule*[3] which maintains strictly more information, as needed for describing the incentive constraints of Bayesian persuasion. Such a rule specifies, for each pair of candidates i and i' (where i' may or may not be equal to i), the conditional type distribution of i' in the event that i is chosen as the winner. Formally, a second-order interim

[3] Our notion of second-order interim rules is different from the notion defined in [6]. Because Cai et al. [6] consider correlation in types, their notion of second-order interim rules is aimed at capturing the allocation dependencies arising through such type correlation, rather than solely through the mechanism's choice.

rule $\boldsymbol{X}$ specifies $x_{i,i',t} \in [0,1]$ for each pair of candidates $i, i' \in [n]$, and type $t \in T_{i'}$. We say that a winner-selection rule $\mathcal{A}$ *implements* $\boldsymbol{X}$ for a prior f if it satisfies the following: if the type profile $\boldsymbol{t} = (t_1, \ldots, t_n)$ is drawn from the prior distribution $f = f_1 \times \cdots \times f_n$, then $\mathbf{Pr}[i \in \mathcal{A}(\boldsymbol{t}) \mid t_{i'} = t] = x_{i,i',t}$. A second-order interim rule is *feasible* if there is a winner selection rule implementing it.

3 Existence of Dice Implementation for Matroids

In this section, we outline the proof of our first theorem:

▶ **Theorem 6.** *Let $(f, \mathcal{M})$ be a matroid winner selection-environment with a total of m types, and let $\boldsymbol{x}$ be an interim rule that is feasible within $(f, \mathcal{M})$. There exist winner-selecting dice $\mathcal{D}$, each of which has at most $m+1$ faces, which implement $\boldsymbol{x}$.*

The proof consists of two parts. First, we generalize the result of [1], which showed the existence of continuous winner-selecting dice for feasible interim rules for a k-uniform matroid with fixed types, to general matroids and multiple types. Second, we convert the continuous dice to dice with at most $m+1$ faces each, while keeping the interim probabilities unchanged.

3.1 Continuous Winner-Selecting Dice

▶ **Theorem 7.** *Let $\mathcal{M}$ be a matroid, and $\boldsymbol{x}$ a feasible interim rule within the winner-selection environment $(f, \mathcal{M})$. There exist winner-selecting dice $\mathcal{D}$ over $\mathbb{R}$ that implement $\boldsymbol{x}$ in $(f, \mathcal{M})$.*

We assume without loss of generality that the candidates' type sets are disjoint, and use $T = \biguplus_{i=1}^{n} T_i$ to denote the set of all types. We use $f(t)$ and $x(t)$ as shorthand for $f_i(t)$ and $x_i(t)$, where $i = \nu(t)$ is the candidate for whom $t \in T_i$. Moreover, given a set of types $S \subseteq T$, we write $S_i = S \cap T_i$. Recall the Border constraints

$$\sum_{i=1}^{n} \sum_{t \in S_i} f_i(t) x_i(t) \leq R(S), \tag{3}$$

where $R(S) = \mathbb{E}_{\boldsymbol{t} \sim f}[r_{\mathcal{M}}(\boldsymbol{t} \cap S)]$ is the expected rank of types in S which show up, a submodular function over the type set T. The Border constraints can therefore be interpreted as follows: An interim rule $\boldsymbol{x} : T \to [0,1]$ is feasible for f and $\mathcal{M}$ if and only if $\widetilde{\boldsymbol{x}}$ is in the polymatroid given by $R(S)$, where $\widetilde{x}(t) := f(t)x(t)$. Equivalently, $\boldsymbol{x}$ is feasible if and only if the submodular *slack function* $\sigma(S) = R(S) - \sum_{t \in S} f(t)x(t)$ is non-negative everywhere.

When $\boldsymbol{x}$ is feasible, we call a set $S \subseteq T$ *tight* for $(f, \boldsymbol{x}, \mathcal{M})$ if the Border constraint (3) corresponding to S is tight at $\boldsymbol{x}$, i.e., $\sum_{i=1}^{n} \sum_{t \in S_i} f(t)x(t) = R(S)$. By definition, $S = \emptyset$ is always tight. The family of tight sets, being the family of minimizers of the submodular slack function, forms a lattice: the intersection and the union of two tight sets is a tight set.

▶ Remark. The tightness of a set S means that the expected number of winners from S equals the expected rank of types in S which show up. In other words, S is tight if and only if a maximum independent subset of $\boldsymbol{t} \cap S$ is always selected as winners.

By the preceding remark, the types in minimal non-empty tight sets need to be treated preferentially, i.e., assigned higher faces on their dice, compared to types outside them. Because they play such an important role, we define them as *barrier sets*. Formally, we define the set of *active* types $T^+ = \{t \in T : f(t)x(t) > 0\}$ to be the types who win with positive probability. Barrier sets are subsets of T^+. If there is at least one non-empty tight set of active types, we define the barrier sets as the (inclusion-wise) minimal such sets. Otherwise, we designate the entire set T^+ of active types as the unique barrier set.

To prove Theorem 7, we first assume that the matroid $\mathcal{M}$ is non-separable. Separable matroids will be handled in the proof of Theorem 7 by combining the dice of their non-separable components. Because barrier sets get precedence, we first show how to construct dice for barrier sets with Lemma 8. Once we have dice for barrier sets, we can repeatedly "peel off" the tight sets and combine their dice, which is captured in Lemma 9. We start with the existence of dice for barrier sets:

▶ **Lemma 8.** *Let $\mathcal{M}$ be a non-separable matroid, and $\boldsymbol{x} > \mathbf{0}$ a feasible interim rule within the winner-selection environment $(f, \mathcal{M})$. Let S be a barrier set for $(f, \boldsymbol{x}, \mathcal{M})$, and define $\boldsymbol{x}_S$ to be $(x(t))_{t\in S}$. There exists a vector of distributions $\mathcal{D} = (D_t)_{t\in S}$ over $\mathbb{R}$, such that $\mathcal{D}$ implements $\boldsymbol{x}_S$ in $(f, \mathcal{M}|S)$.*

The proof is quite technically involved, and due to space constraints, it is entirely deferred to the full version. The high-level idea is to base the dice upon any full-support continuous distribution for the dice. This distribution is scaled by different parameters θ_i for different types i, and shifted by a constant τ. Matching the prescribed interim winning probabilities $x(i)$ imposes a system of non-linear equations on the θ_i. We establish that the winning probabilities of different types satisfy certain key monotonicity properties in the parameters θ_i; these monotonicity properties are proved using the fact that the feasible sets form a matroid. Then, we apply the Intermediate Value Theorem inductively for all types i to non-constructively prove the existence of the desired θ_i.

To generalize Lemma 8 to arbitrary sets of types, we will need a construction that allows us to "scale" the faces of some dice such that they will always be above/below the faces of another set of newly introduced dice; such a construction will allow us to give dice for types in barrier sets higher faces than other dice. For the types of full-support distributions over $[0, \infty)$ we have been using so far, this would be impossible. There is a simple mapping that guarantees our desired properties: we map faces from $(0, \infty)$ to the set $(1, 2)$ by mapping all positive $s \mapsto 2 - \frac{1}{1+s}$, and mapping all negative s to -1. Notice that the new dice implement the same interim rule as the old ones: in matroid environments, the set maximizing the sum of die rolls is determined by the greedy algorithm, and hence, only the relative order between die faces matters. With the help of this mapping, we prove the following lemma, similar to Lemma 8. The proof is again only given in the full version.

▶ **Lemma 9.** *Let $\boldsymbol{x} > \mathbf{0}$ be a feasible interim rule within a winner-selection environment $(f, \mathcal{M})$, where $\mathcal{M}$ is a non-separable matroid. Fix a tight set S, and let $\widehat{S}$ be a minimal tight set that includes S as a proper subset, if such a set exists; otherwise let $\widehat{S} = T$. Given dice $\mathcal{D} = (D_t)_{t\in S}$ that implement $\boldsymbol{x}_S$ in $(f, \mathcal{M}|S)$, there are dice $\mathcal{D}' = (D'_t)_{t\in\widehat{S}}$ which implement $\boldsymbol{x}_{\widehat{S}}$ in $(f, \mathcal{M}|\widehat{S})$.*

Proof of Theorem 7. First consider the case when $\mathcal{M}$ is non-separable. Let T be the type set with m types. We define the dice system as follows: First, for all types t with $x(t) = 1$, assign them a point distribution (single-sided die) at 2, so they always win. Next, for all types t with $x(t) = 0$, assign them a point distribution at -1, so they never win. Next, we create dice for barrier sets S according to Lemma 8. Then, starting from the barrier sets, we repeatedly apply Lemma 9 to construct dice for larger tight sets $\widehat{S} \supsetneq S$ (or all of T) implementing $\boldsymbol{x}_{\widehat{S}}$.

If $\mathcal{M}$ is separable, let $\mathcal{M}_1, \ldots, \mathcal{M}_k$ be the components of $\mathcal{M}$. Using the construction from the previous paragraph for each $\mathcal{M}_j$, let $\mathcal{D}_j$ be the dice set constructed for $\mathcal{M}_j$. Since there is no circuit containing two candidates from different components, the winner set of one component has no effect on the winner set of any other component. Thus, the union $\bigcup_{j=1}^{k} \mathcal{D}_j$ of dice implements the desired interim rule. ◀

3.2 Winner-Selecting Dice with polynomially many faces

The proof is based on the following generalization of the fundamental theorem of linear programming to uncountable dimensions.

▶ **Theorem 10** (Theorem B.11 from [13]). *Let $f_1, \ldots, f_m : \mathbb{X} \to \mathbb{R}$ be Borel measurable on a measurable space $\mathbb{X}$, and let μ be a probability measure on $\mathbb{X}$ such that f_i is integrable with respect to μ for each $i = 1, \ldots, m$. Then, there exists a probability measure φ on $\mathbb{X}$ with finite support, such that $\int_{\mathbb{X}} f_i d\varphi = \int_{\mathbb{X}} f_i d\mu$ for all i. Moreover, there is such a φ whose support consists of at most $m+1$ points.*

Proof of Theorem 6. Theorem 7 establishes that there is a vector of probability measures $\mathcal{D} = (D_t)_{t \in T}$ over $V = (1, 2) \cup \{-1\}$, satisfying the following for all i and $t_i \in T_i$:

$$\int_V \sum_{\boldsymbol{t} \in T_{-i}} \prod_{j \neq i} f(t_j) \int \cdots \int_{V^{n-1}} w_{t_i}(s_1, \ldots, s_n) dD_{t_1}(s_1) \cdots dD_{t_n}(s_n) dD_{t_i}(s_i) = x(t_i). \quad (4)$$

where $w_{t_i}(s_1, \ldots, s_n)$ equals 1 if type t_i is a winning type with dice rolls $s_1, \ldots, s_n$, and equals 0 otherwise.

For a fixed type $t^* \in T_i$ and an equation corresponding to (j, t'_j), we can change the order of integration to make dD_{t^*} the outermost integral, and isolate the terms that do not involve dD_{t^*}. Specifically, we define

$$q_{j,t'_j}(s_i) = \sum_{\substack{\boldsymbol{t} \in T \\ t_j = t'_j, t_i = t^*}} \prod_{k \neq j} f(t_k) \int \cdots \int_{V^{n-1}} w_{t_j}(s_1, \ldots, s_n) dD_{t_1}(s_1) \cdots dD_{t_n}(s_n)$$

to be the inner integral over distributions of $t \neq t^*$, as a function of s_i, and

$$c_{j,t'_j} = \begin{cases} 0, \text{ for } t'_j = t^* \\ \displaystyle\int_V \sum_{\boldsymbol{t} \in T_{-j}, t_i \neq t^*} \prod_{k \neq j} f(t_k) \int \cdots \int_{V^{n-1}} w_{t_j}(s_1, \ldots, s_n) \\ \qquad\qquad dD_{t_1}(s_1) \cdots dD_{t_n}(s_n) dD_{t_j}(s_j), \text{ for } t'_j \neq t^* \end{cases}$$

as the component of the integral that does not have dD_{t^*} involved. Thus, Equation (4) can be rewritten as $\int_V q_{j,t'_j}(s_i) dD_{t^*}(s_i) + c_{j,t'_j} = x(t'_j)$. By Theorem 10, D_{t^*} can be changed to a measure D'_{t^*} with support of size $m+1$, such that for all j and $t_j \in T_j$, $\int_V q_{j,t_j}(s_i) dD_{t^*}(s_i) = \int_V q_{j,t_j}(s_i) dD'_{t^*}(s_i)$. In other words, we can replace D_{t^*} with D'_{t^*}, which has at most $m+1$ faces. Applying the same procedure to each type t^* in turn, all dice can be replaced by dice with at most $m+1$ faces. ◀

4 Efficient Construction of Dice for Single-Winner Environments

In the preceding section, we proved the existence of winner-selecting dice by a "construction." However, the construction involves repeated appeals to the Intermediate Value Theorem, and is thus inherently non-computational. It is certainly not clear how to implement it efficiently. In this section, we show that when the matroid $\mathcal{M}$ is 1-uniform, i.e., at most one winner can be selected, winner-selecting dice can be computed efficiently. We prove the following.

▶ **Theorem 11.** *Consider a winner-selection environment with n candidates, where each candidate i's type is drawn independently from a prior f_i supported on T_i, and at most one winner can be selected. If $\boldsymbol{x}$ is a feasible interim rule for $f = f_1 \times \cdots \times f_n$, an explicit representation of the associated dice can be computed in time polynomial in n and $m = \sum_i |T_i|$.*

ALGORITHM 1: FIND BARRIER SET$(f, \boldsymbol{x})$.

$T^+ \leftarrow \{t \in T : f(t)x(t) > 0\}$.
Define $g(S) = f(S) - \sum_{t \in S} f(t)x(t)$ for $S \subseteq T^+$.
if $\min\left\{g(S) : \emptyset \subsetneq S \subseteq T^+\right\} \neq 0$ **then**
    return T^+.
else
    $T^* \leftarrow T^+$.
    while there is a type $t \in T^*$ such that $\min\{g(S) : \emptyset \subsetneq S \subseteq T^* \setminus \{t\}\} = 0$ **do** $T^* \leftarrow T^* \setminus \{t\}$.
    return T^*.

We use the same notation $f(t)$ and $x(t)$ as in Section 3. In the single-winner setting, the function $R(S) = \mathbb{E}_{\boldsymbol{t} \sim f}\left[r_{\mathcal{M}}(\boldsymbol{t} \cap S)\right]$ can be treated as a natural extension of f to subsets of T: given $S \subseteq T$, we let $R(S) = f(S) = 1 - \prod_{i=1}^{n}(1 - \sum_{t \in S_i} f(t))$, which is the probability that at least one type in S shows up. Border's constraints can then be written as follows: $\sum_{t \in S} f(t)x(t) \leq f(S)$ for all $S \subseteq T$. The slack function becomes $\sigma_{f,\boldsymbol{x}}(S) = f(S) - \sum_{t \in S} f(t)x(t)$ and is nonnegative everywhere when $\boldsymbol{x}$ is feasible.

Tight sets and barrier sets are defined as the special case of the definition for general matroids in Section 3. The algorithm FIND BARRIER SET, given as Algorithm 1, simply implements the definition of barrier sets as minimal non-empty tight sets, and therefore correctly computes a barrier set.

The following lemma characterizes the key useful structure of barrier sets for the single-winner setting.

▶ **Lemma 12.** *Let f and $\boldsymbol{x}$ be such that $\boldsymbol{x}$ is feasible for f. If there are multiple barrier sets for $(f, \boldsymbol{x})$, then there is a candidate i^* such that each barrier set is a singleton $\{t\}$ with $t \in T_{i^*}$.*

Proof. Let A, B be any two barrier sets. Because A and B are both tight, the lattice property of tight sets implies that $A \cap B = \emptyset$.

We first show that there is a candidate i^* with $A \subseteq T_{i^*}$ and $B \subseteq T_{i^*}$. Suppose not for contradiction; then, there exist $i \neq j$ and types $t_i \in A \cap T_i$ and $t_j \in B \cap T_j$. With non-zero probability, the types t_i and t_j show up at the same time. However, according to the definition of a tight set, when a type in A shows up, the winner must be a candidate with type in A, and the same must hold for B. Then, the winner's type would have to be in $A \cap B$ with non-zero probability. This contradicts the disjointness of A and B.

It remains to show that all barrier sets are singletons. Since A is tight and all types in A belong to the same candidate, $\sum_{t \in A} f(t)x(t) = f(A) = \sum_{t \in A} f(t)$. Hence, $x(t) = 1$ for all $t \in A$, and because barrier sets are minimally tight, A must be a singleton. ◀

4.1 Description of the Algorithm

Given a prior f and a feasible interim rule $\boldsymbol{x}$, the recursive procedure CONTRUCT DICE, shown in Algorithm 2, returns a family of dice $\mathcal{D}$ implementing $\boldsymbol{x}$ for f. It operates as follows. There are two simple base cases: when no candidate ever wins, and when a single type of a single candidate always wins. In the recursive case, the algorithm carefully selects a type t^* and awards its die the highest-valued face M'. It assigns this new face a probability as large as possible, subject to still permitting implementation of $\boldsymbol{x}$. We choose t^* as a member of a barrier set; this is important in order to guarantee that the algorithm makes significant progress.

ALGORITHM 2: CONTRUCT DICE $(f, \boldsymbol{x})$.

Input : PDFs $f_1, \dots, f_n$ supported on disjoint type sets $T_1, \dots, T_n$.
Input : Interim rule $\boldsymbol{x}$ feasible for f.
Output : Vector of dice $(D_t)_{t \in \biguplus_{i=1}^n T_i}$.

Let $T = \biguplus_i T_i$.
Let $T_i^+ = \{t \in T_i : f_i(t)x_i(t) > 0\}$, and let $T^+ = \biguplus_{i=1}^n T_i^+$.
if $T^+ = \emptyset$ **then**
    for all types $t \in T$, let D_t be a single-sided die with a -1 face.
else if *there is a type* $t^* \in T^+$ *with* $f(t^*)x(t^*) = 1$ **then**
    Let D_{t^*} be a single-sided die with a $+1$ face.
    for all other types $t \in T \setminus \{t^*\}$, let D_t be a single-sided die with a -1 face.
else
    Let $T^* = $ FIND BARRIER SET$(f, \boldsymbol{x})$.
    Let $t^* \in T^*$ be a type chosen arbitrarily.
    Let $(f', \boldsymbol{x'}) = $ DECREMENT$(f, \boldsymbol{x}, t^*, q^*)$, for the largest value of $q^* \in [0, f(t^*)x(t^*)]$ such that $\boldsymbol{x'}$ is feasible for f'. `/* Note that` $f(t^*)x(t^*) < 1$. `*/`
    Let $(D'_t)_{t \in T} \leftarrow $ CONTRUCT DICE$(f', \boldsymbol{x'})$.
    Let M be the maximum possible face of any die D'_t, and $M' := \max(M, 0) + 1$.
    Let $D_t = D'_t$ for all types $t \neq t^*$.
    Let D_{t^*} be the die which rolls M' with probability $\frac{q^*}{f(t^*)}$, and D'_{t^*} with probability $1 - \frac{q^*}{f(t^*)}$.
return $(D_t)_{t \in T}$.

ALGORITHM 3: DECREMENT$(f, \boldsymbol{x}, t^*, q)$.

`/*` $q \geq 0$ `is the probability allocated to the highest face. Because it is a contribution to the` *`unconditional`* `winning probability` $f(t^*)x(t^*)$ `of type` t^*`, and we separated out the case that a single type has unconditional winning probability 1,` q `satisfies` $q \leq f(t^*)x(t^*) < 1$. `*/`

if $q = f(t^*)$, **then** let $f'(t^*) \leftarrow 0$ and $x'(t^*) \leftarrow 0$
else let $f'(t^*) \leftarrow \frac{f(t^*)-q}{1-q}$ and $x'(t^*) \leftarrow \frac{f(t^*)x(t^*)-q}{f(t^*)-q}$.
Let i^* be such that $t^* \in T_{i^*}$.
for all $t \in T_{i^*}, t \neq t^*$, let $f'(t) \leftarrow \frac{f(t)}{1-q}$ and $x'(t) \leftarrow x(t)$.
for all $t \in T \setminus T_{i^*}$, let $f'(t) \leftarrow f(t)$ and $x'(t) \leftarrow \frac{x(t)}{1-q}$.
return $(f', \boldsymbol{x'})$.

The subroutine DECREMENT, shown as Algorithm 3, essentially conditions both f and $\boldsymbol{x}$ on the face M' *not* winning. Specifically, DECREMENT computes the conditional type distribution f', and an interim rule $\boldsymbol{x'}$, such that if there were a dice implementation of $\boldsymbol{x'}$ for f', then adding M' to the die of t^* would yield a set of dice implementing $\boldsymbol{x}$ for f.

We now provide a formal analysis of our algorithm. Theorem 11 follows from Lemmas 13–15.

▶ **Lemma 13.** *If* CONTRUCT DICE *terminates, it outputs dice implementing* $\boldsymbol{x}$ *for* f.

▶ **Lemma 14.** CONTRUCT DICE *terminates after at most* m^2 *recursive calls. (Recall that* $m = \sum_i |T_i|$*.)*

▶ **Lemma 15.** *Excluding the recursive call, each invocation of* CONTRUCT DICE *can be implemented in time polynomial in* n *and* m.

4.2 Proof of Lemma 13 (Correctness)

We prove the lemma by induction over the algorithm's calls. Correctness is obvious for the two base cases: when $T^+ = \emptyset$ (no type should win), and when there exists a type t^* with $f(t^*)x(t^*) = 1$ (t^* always shows up and should always win). For the inductive step, suppose that the recursive call in step 12 returns dice $\mathcal{D}' = (D'_t)_{t\in T}$, correctly implementing $\boldsymbol{x'}$ for f', and let $\mathcal{D} = (D_t)_{t\in T}$ be the new dice defined in steps 14 and 15.

We analyze the interim winning probability of each type when using the dice-based winner selection rule given by $\mathcal{D}$. For each type t, let $\overline{v}_t \sim D_t$ be a roll of the die for type t, and for each i, let $t_i \sim f_i$ be a draw of a type; all $\overline{v}_t$ and t_i are mutually independent. In other words, we may assume that the die of *every* type is rolled (including types that do not show up), then the type profile is drawn independently. The winning type is then the type t_i with largest positive $\overline{v}_{t_i}$; if all $\overline{v}_{t_i}$ are negative, then no type wins.

Let $t^* \in T_{i^*}$ be as defined in step 10. Let $\mathcal{E}$ be the event that i^* has type t^* and that $\overline{v}_{t^*} = M'$, and let $\overline{\mathcal{E}}$ be its complement. By independence of the random choices, the probability of $\mathcal{E}$ is $f(t^*) \cdot \frac{q^*}{f(t^*)} = q^*$. Type t^* always wins under the event $\mathcal{E}$. Conditioned on $\overline{\mathcal{E}}$, each $\overline{v}_t$ (including $\overline{v}_{t^*}$) is distributed as a draw from D'_t, the type vector $\boldsymbol{t}$ is distributed as a draw from $f'_1 \times \cdots \times f'_n$, and the $\overline{v}_t$'s and $\boldsymbol{t}$ are mutually independent. By the inductive hypothesis, conditioned on $\overline{\mathcal{E}}$, each type t wins with probability $f'(t)x'(t)$. Using the definition of f' and $\boldsymbol{x'}$ from the DECREMENT subroutine, the total winning probability for t^* is $q^* \cdot 1 + (1-q^*) \cdot f'(t^*)x'(t^*) = q^* + (1-q^*) \cdot \frac{f(t^*)x(t^*)-q^*}{1-q^*} = f(t^*)x(t^*)$. For $t \neq t^*$, the total winning probability is $q^* \cdot 0 + (1-q^*)f'(t)x'(t) = (1-q^*) \cdot \frac{f(t)x(t)}{1-q^*} = f(t)x(t)$. Therefore, the interim winning probability for each type t is $x(t)$, and the dice $\mathcal{D}$ implement $\boldsymbol{x}$ for f.

4.3 Proof of Lemma 14 (Number of Recursive Calls)

The following lemma is essential in that it shows that invoking DECREMENT maintains feasibility and tightness of sets.

▶ **Lemma 16.** *Let f, $\boldsymbol{x}$, t^* and q be valid inputs for* DECREMENT*, and f', $\boldsymbol{x'}$ the output of the call to* DECREMENT$(f, \boldsymbol{x}, t^*, q)$*. Let S be any set of types with $t^* \in S$. Then,*

1. *The Border constraint for S is satisfied for $(f', \boldsymbol{x'})$ if and only if it is satisfied for $(f, \boldsymbol{x})$.*
2. *The Border constraint for S is tight for $(f', \boldsymbol{x'})$ if and only if it is tight for $(f, \boldsymbol{x})$.*

Proof. We will show that the slack for every $S \ni t^*$ satisfies $\sigma_{f',\boldsymbol{x'}}(S) = \frac{\sigma_{f,\boldsymbol{x}}(S)}{1-q}$, which implies both claims. Let i^* be such that $t^* \in T_{i^*}$. Using the definitions of f' and $\boldsymbol{x'}$,

$$\begin{aligned}
\sigma_{f',\boldsymbol{x'}}(S) &= f'(S) - \sum_{t\in S} f'(t)x'(t) \\
&= 1 - \left(1 - \frac{f(S_{i^*}) - q}{1-q}\right) \cdot \prod_{i\neq i^*}(1 - f(S_i)) - \frac{(\sum_{t\in S} f(t)x(t)) - q}{1-q} \\
&= \frac{1}{1-q} \cdot \left(1 - (1 - f(S_{i^*})) \cdot \prod_{i\neq i^*}(1 - f(S_i)) - \sum_{t\in S} f(t)x(t)\right) = \frac{\sigma_{f,\boldsymbol{x}}(S)}{1-q}.
\end{aligned}$$

◀

We are now ready to prove Lemma 14. We will show that with each recursive invocation of CONTRUCT DICE (step 12), at least one of the following happens: (1) The number of active types $|T^+|$ decreases; (2) The size of the barrier set $|T^*|$ decreases.

Notice that the number of active types or the size of a barrier set never *increase*. Because the size of the barrier set can only decrease at most m times, the number of active types must decrease at least every m recursive invocations. It, too, can decrease at most m times, implying the claim of the lemma.

Let T^*, t^*, and $(q^*, f', \boldsymbol{x'})$ be as chosen in steps 9, 10, and 11, respectively. Let candidate i^* be such that $t^* \in T_{i^*}$. If $q^* = f(t^*)x(t^*)$, then the type t^* will be inactive in $(f', \boldsymbol{x'})$, and there will be one fewer active type in the subsequent invocation of CONTRUCT DICE (step 12). We distinguish the cases $|T^*| = 1$ and $|T^*| > 1$.

If $|T^*| = 1$, then $T^* = \{t^*\}$. By definition of a barrier set, T^* is tight, implying (for a singleton set) that $x(t^*) = 1$. We claim that q^* is set to $f(t^*) = f(t^*)x(t^*)$ in step 11, implying that the number of active types decreases. To prove that $q^* = f(t^*)$, we will show that this choice of q^* is feasible in the invocation of $\textsc{Decrement}(f, \boldsymbol{x}, t^*, f(t^*))$. Consider the $\widehat{f}, \widehat{\boldsymbol{x}}$ resulting from such an invocation of DECREMENT. Lemma 16 implies that the feasibility of each Border constraint corresponding to a set $S \ni t^*$ is preserved for $(\widehat{f}, \widehat{\boldsymbol{x}})$. For type sets S excluding t^*,

$$\begin{aligned}
\sigma_{\widehat{f},\widehat{\boldsymbol{x}}}(S) = \widehat{f}(S) - \sum_{t\in S} \widehat{f}(t)\widehat{x}(t) \;&=\; 1 - \left(1 - \frac{f(S_{i^*})}{1 - f(t^*)}\right) \cdot \prod_{i\neq i^*} (1 - f(S_i)) - \frac{\sum_{t\in S} f(t)x(t)}{1 - f(t^*)} \\
&= \frac{1}{1 - f(t^*)} \cdot \left(1 - (1 - f(S_{i^*}) + f(t^*)) \cdot \prod_{i\neq i^*} (1 - f(S_i)) \right. \\
&\qquad\qquad \left. - \left(\sum_{t\in S} f(t)x(t) + f(t^*)x(t^*)\right)\right) \\
&= \frac{\sigma_{f,\boldsymbol{x}}(S \cup \{t^*\})}{1 - f(t^*)},
\end{aligned}$$

which is nonnegative because $\boldsymbol{x}$ is feasible for f. Therefore, step 11 indeed chooses $q^* = f(t^*)$.

Next, we consider the case $|T^*| > 1$, and assume that $q^* < f(t^*)x(t^*)$ (since otherwise, we are done). Then, the set of active types T^+ is the same for both $(f, \boldsymbol{x})$ and $(f', \boldsymbol{x'})$.

If the instance $(f', \boldsymbol{x'})$ for the recursive call has multiple barrier sets, then by Lemma 12, they are all singletons, and indeed the size of the barrier set in the next recursive call (which is 1) is strictly smaller than $|T^*|$. So we assume that $(f', \boldsymbol{x'})$ has a unique barrier set T'.

Because $|T^*| > 1$, Lemma 12 implies that T^* is the unique barrier set for $(f, \boldsymbol{x})$. Therefore, by the definition of barrier sets, T^* (and hence also t^*) is contained in every tight set of active types for $(f, \boldsymbol{x})$ (if any). Because t^* is contained in all tight sets, Lemma 16 implies that for every $q \in [0, f(t^*)x(t^*)]$, the result of $\textsc{Decrement}(f, \boldsymbol{x}, t^*, q)$ does not violate any constraints which are already tight for $(f, \boldsymbol{x})$, and in fact preserves their tightness.

Because all other constraints have slack, the optimal q^* is strictly positive. By assumption, we also have that $q^* < f(t^*)x(t^*)$; therefore, a non-empty set S' which was not tight for $(f, \boldsymbol{x})$ must have become tight for $(f', \boldsymbol{x'}) = \textsc{Decrement}(f, \boldsymbol{x}, t^*, q^*)$. By Lemma 16, this set S' does not include t^*. Because discarding inactive types preserves tightness, we may assume without loss of generality that $S' \subseteq T^+$.

We have shown the existence of a non-empty tight set $S' \subseteq T^+ \setminus \{t^*\}$ for $(f', \boldsymbol{x'})$. By definition, the barrier set T' is the (unique, in our case) minimal tight set for $(f', \boldsymbol{x'})$, so $T' \subseteq S'$. We distinguish two cases, based on the possible definitions of barrier sets:

- If $T^* = T^+$, then because $S' \subsetneq T^+ = T^*$, the barrier set T' is strictly smaller than T^*.
- If T^* is tight for $(f, \boldsymbol{x})$, then it is also tight for $(f', \boldsymbol{x'})$ by Lemma 16; since both S' and T^* are tight for $(f', \boldsymbol{x'})$, we get that $T' \subseteq T^* \cap S' \subseteq T^* \setminus \{t^*\}$ is strictly smaller than T^*.

4.4 Proof of Lemma 15 (Runtime per Call)

There are only two steps for which polynomial runtime is not immediate: the computation of q^* in step (11) of Contruct Dice, and finding a non-empty minimizer of a submodular function in steps (3) and (7) of Find Barrier Set. We prove polynomial-time implementability of both steps in the following lemmas.

▶ **Lemma 17.** *In step 11 of* Contruct Dice*, q^* can be computed in* poly(m) *time.*

Proof. Let f, $\boldsymbol{x}$, and t^* be as in step 11, and let the candidate i^* be such that $t^* \in T_{i^*}$. For each $q \in [0, f(t^*)x(t^*)] \subseteq [0,1)$, let $(f_q, \boldsymbol{x}_q) = \textsc{Decrement}(f, \boldsymbol{x}, t^*, q)$ be the result of running $\textsc{Decrement}(f, \boldsymbol{x}, t^*, q)$ with parameter q. Lemma 16 implies that all Border constraints for $S \ni t^*$ remain feasible for $(f_q, \boldsymbol{x}_q)$. For type sets $S \not\ni t^*$, we can write the slack in the corresponding Border constraint as a function of q as follows:

$$\begin{aligned}
\sigma_{f_q,\boldsymbol{x}_q}(S) &= f_q(S) - \sum_{t\in S} f_q(t)x_q(t) \\
&= 1 - \left(1 - \frac{f(S_{i^*})}{1-q}\right) \cdot \prod_{i\neq i^*} (1 - f(S_i)) - \frac{\sum_{t\in S} f(t)x(t)}{1-q} \\
&= \frac{1}{1-q} \cdot \left(1 - (1 - f(S_{i^*}) - q) \cdot \prod_{i\neq i^*} (1 - f(S_i)) - \sum_{t\in S} f(t)x(t) - q\right) \\
&= \frac{1}{1-q} \left(\sigma_{f,\boldsymbol{x}}(S) - q f(S \setminus S_{i^*})\right).
\end{aligned}$$

The preceding expression is nonnegative if and only if $q \le h(S) := \frac{\sigma_{f,\boldsymbol{x}}(S)}{f(S\setminus S_{i^*})}$. Therefore, q^* is the minimum of $f(t^*)x(t^*)$ and $\min_{S\subseteq T\setminus\{t^*\}} h(S)$. The function $h(S)$ does not appear to be submodular, and hence efficient minimization is not immediate. We utilize Theorem 4 to reduce the search space and compute q^* efficiently.

For an interim rule $\boldsymbol{x} : T \to [0,1]$, we call $S \subseteq T$ a *level set* of $\boldsymbol{x}$ if there exists an $\alpha \in [0,1]$ such that $S = \{t \in T : x(t) > \alpha\}$. If $q^* = \min_{S\not\ni t^*} h(S)$, then at least one Border constraint just becomes tight at $q = q^*$. Theorem 4 implies that at least one level set of $\boldsymbol{x}_{q^*}$ corresponds to one of these newly tightened constraints, and h is minimized by such a level set. It follows that, in order to compute q^*, it suffices to minimize h over all those sets $S \not\ni t^*$ which could possibly arise as level sets of some $\boldsymbol{x}_q$ for $q \in [0, f(t^*)x(t^*)]$.

Let $t_1, \ldots, t_K$ be the types in $T_{i^*} \setminus \{t^*\}$, ordered by non-increasing $x(t)$; similarly, let $t'_1, \ldots, t'_L$ be the types in $T \setminus T_{i^*}$, ordered by non-increasing $x(t)$. The relative order of types in $T_{i^*} \setminus \{t^*\}$ is the same under $x_q(t)$ as under $x(t)$, because $x_q(t) = x(t)$; similarly, the relative order of types in $T \setminus T_{i^*}$ is the same under $x_q(t)$ as under $x(t)$, because $x_q(t) = \frac{x(t)}{1-q}$. Therefore, the family $\{\{t_1, \ldots, t_k, t'_1, \ldots, t'_\ell\} : k \le K, \ell \le L\}$ includes all level sets of every $\boldsymbol{x}_q$ excluding t^*. There are at most m^2 type sets in this family, and those sets can be enumerated efficiently to minimize h. ◀

▶ **Lemma 18.** *There is an algorithm for computing a non-empty minimizer of a submodular function in the value oracle model, with runtime polynomial in the size of the ground set.*

Proof. For a submodular function $g : 2^T \to \mathbb{R}$, let $g_t(S) = g(S \cup \{t\})$, for $t \in T$. g_t is also submodular, and can be minimized in time polynomial in $|T|$ [10]. Let S_t be a minimizer of g_t and $t^* \in \arg\min_{t\in T} g_t(S_t)$. $S_{t^*} \cup \{t^*\}$ is a non-empty minimizer of g. ◀

5 From Winner-Selecting Dice to Interim Rules for Single-Winner Settings

Having shown how to compute winner-selecting dice implementing a given interim rule, we next show the easier converse direction: how to compute the interim rule given winner-selecting dice $\{D_{i,t}\}$ in single-winner environments. As before, we denote the type set of candidate i by T_i, and assume without loss of generality that the type sets of different candidates are disjoint. For simplicity of exposition, we assume that each die has a given finite support[4], and we write $U_i := \bigcup_{t \in T_i} \text{supp}(D_{i,t})$ for the combined support of candidate i's dice. We also assume that we can evaluate the probability $\mathbf{Pr}[D_{i,t} = u]$ of the face labeled with u, for all candidates i, types t, and faces $u \in U_i$.

▶ **Theorem 19.** *Consider a single-winner selection environment with n candidates and independent priors $f_1, \dots, f_n$, where f_i is supported on T_i. Given dice $\{D_{i,t} \mid i \in [n], t \in T_i\}$ represented explicitly, the interim rule of the corresponding dice-based winner selection rule can be computed in time polynomial in n, $m = \sum_i |T_i|$, and the total support size $|\bigcup_i U_i|$ of all the dice.*

Proof. First, we can compute the probability mass function of each candidate i's (random) score $\overline{v}_i$. Given $u \in U_i$, we have $\mathbf{Pr}[\overline{v}_i = u] = \sum_{t \in T_i} f_i(t) \cdot \mathbf{Pr}[D_{i,t} = u]$.

From this probability mass function, we easily compute $\mathbf{Pr}[\overline{v}_i \le u]$ for each $u \in U_i$ by the appropriate summation. When all dice faces are distinct, this is all we need; since $\mathbf{Pr}[\overline{v}_{i'} < u] = \mathbf{Pr}[\overline{v}_{i'} \le u]$ for $i' \neq i$ and $u \in U_i$, the interim rule is given by the following simple equation:

$$x_i(t) = \sum_{u \in \text{supp}(D_{i,t})} \mathbf{Pr}[D_{i,t} = u] \cdot \prod_{i' \neq i} \mathbf{Pr}[\overline{v}_{i'} \le u].$$

When the dice's faces are not distinct, recall that we break ties uniformly at random. To account for the contribution of this tie-breaking rule, we need the distribution of the number of other candidates that tie candidate i's score of u; this is a Poisson Binomial distribution. More, precisely, we need the Poisson Binomial distribution with the $n-1$ parameters $\left(\frac{\mathbf{Pr}[\overline{v}_{i'}=u]}{\mathbf{Pr}[\overline{v}_{i'} \le u]}\right)_{i' \neq i}$; we denote its probability mass function by $B_{i,u}$. It is well known, and easy to verify, that a simple dynamic program computes the probability mass function of a Poisson Binomial distribution in time polynomial in its number of parameters. Therefore, we can compute $B_{i,u}(k)$ for each $i \in [n], u \in U_i$ and $k \in \{1, \dots, n-1\}$. The interim rule is then given by the following equation:

$$x_i(t) = \sum_{u \in \text{supp}(D_{i,t})} \mathbf{Pr}[D_{i,t} = u] \cdot \left(\prod_{i' \neq i} \mathbf{Pr}[\overline{v}_{i'} \le u] \right) \cdot \sum_{k=0}^{n-1} \frac{B_{i,u}(k)}{k+1}.$$

It is easy to verify that all the above computations satisfy the claimed runtime. ◀

[4] Our approach extends easily to the case of continuously supported dice, so long as we can perform integration with respect to the distributions of the various dice.

6 Winner-Selecting Dice and Persuasion

In this section, we investigate the existence of winner-selecting dice for instances of Bayesian persuasion. Our main result (Theorem 20) is to exhibit an instance with independent non-identical actions for which there is no optimal signaling scheme that can be implemented using winner-selecting dice. This result is contrasted with Theorem 21, which shows that when the actions' types are not just independent, but identically distributed as well, a dice-based implementation always *does* exist.

▶ **Theorem 20.** *There is an instance of Bayesian persuasion (given in Table 1) with independent actions which does not admit a dice-based implementation of any optimal signaling scheme. Consequently, there exists a second-order interim rule which does not admit a dice-based implementation.*

▶ **Theorem 21.** *Every Bayesian persuasion instance with i.i.d. actions admits an optimal dice-based signaling scheme. Moreover, when the prior type distribution is given explicitly, the corresponding dice can be computed in time polynomial in the number of actions and types.*

The negative result of Theorem 20 has interesting implications. Since second-order interim rules summarize all the attributes of a winner selection rule relevant to persuasion, second-order interim rules, unlike their first-order brethren, can in general not be implemented by dice. Most importantly, this result draws a sharp contrast between persuasion and single-item auctions, despite their superficial similarity: it rules out a Myerson-like virtual-value characterization of optimal persuasion schemes, and it joins the #P-hardness result of [8] as evidence of the intractability of optimal persuasion.

6.1 Basics of Bayesian Persuasion

In *Bayesian persuasion*, the n candidates are *actions* which a *receiver* can take. Each action i has a type t_i, drawn *independently*[5] from the set T_i, according to a commonly known distribution f_i. Each type t_i has associated payoffs $s(i, t_i)$ and $r(i, t_i)$ for the sender and receiver, respectively. The *sender* (or *principal*) also has access to the *actual* draws $\boldsymbol{t} = (t_1, \ldots, t_n)$ of the types, and would like to use this leverage to persuade the receiver to take an action favorable to him[6].

Thereto, the sender can commit to a (typically randomized) policy $\mathcal{A}$– called a *signaling scheme* – of revealing some of this information to the receiver. It was shown by Kamenica and Gentzkow [12] that the sender can restrict attention, without loss, to *direct schemes*: randomized functions $\mathcal{A}$ mapping type profiles to recommended actions. Naturally, the function must be *persuasive*: if action i is recommended, the receiver's posterior expected utility from action i must be no less than her posterior expected utility from any other action i'. In this sense, direct schemes can be viewed as winner selection rules in which the actions are the candidates, and persuasiveness constraints must be obeyed.

6.2 Proof of Theorems

Proof of Theorem 20. The persuasion instance, shown in Table 1, features three actions $\{A, B, C\}$, each of which has two types $\{1, 2\}$. The types of the different actions are distributed independently. In the instance, the sender's utility from any particular action is a constant, independent of the action's type.

[5] The draws are independent in this paper. In more general Bayesian persuasion models, they can be correlated.

[6] To avoid ambiguities, we always use male pronouns for the sender and female ones for the receiver.

Table 1 A Persuasion instance with no dice-based implementation. The notation $p \times (s, r)$ denotes that the type (s, r) (in which the sender and receiver payoffs are s and r, respectively) has probability p.

action \ type	1	2
A	$0.5 \times (100, 2)$	$0.5 \times (100, -\infty)$
B	$0.99 \times (1, 3)$	$0.01 \times (1, -\infty)$
C	$0.5 \times (0, 0)$	$0.5 \times (0, 6)$

One (optimal, as we will show implicitly) signaling scheme is the following. (In writing a type vector, here and below, we use $*$ to denote that the type of an action is irrelevant.)

- If the type vector is $(1, *, 1)$, then recommend action A.
- If the type vector is $(1, *, 2)$, then recommend each of A, C with equal probability $\frac{1}{2}$.
- If the type vector is $(2, 1, *)$, then recommend action B.
- If the type vector is $(2, 2, *)$, then recommend action C.

While this is not the unique optimal scheme, we next prove that none of the optimal persuasion schemes admit a dice-based implementation.

The given signaling scheme recommends action A with probability $3/8$ overall, action B with probability $\frac{99}{200}$ overall, and action C with the remaining probability. No persuasive signaling scheme can recommend A with probability strictly more than $3/8$, because conditioned on receiving the recommendation A, action C must be at least twice as likely to be of type 1 as of type 2, in addition to action A being of type 1 with probability 1. Similarly, no persuasive scheme can recommend B with probability strictly more than $\frac{99}{200}$, because action C must be at least as likely to be of type 1 as of type 2 when C is recommended, in addition to action B being of type 1 with probability 1. Hence, any optimal signaling scheme must recommend A with probability $3/8$ and B with probability $\frac{99}{200}$, and the given scheme is in fact optimal.

Suppose for a contradiction that there exist dice $(D_{i,j})_{i \in \{A,B,C\}, j \in \{1,2\}}$ implementing an optimal signaling scheme. We gradually derive properties of these optimal signaling schemes, eventually leading to a contradiction.

1. Since action A can never be recommended when it has type 2 (the receiver would never follow the recommendation), it must be recommended with probability $\frac{3}{4}$ conditioned on having type 1.
2. In particular, whenever the type profile is $(1, *, 1)$, action A must be recommended, regardless of the type of action B. This is because action C must be at least twice as likely of type 1 as of type 2 for a recommendation of A to be persuasive.
3. Therefore, all faces on $D_{A,1}$ must be larger than all faces on $D_{B,1}$ and on $D_{B,2}$.
4. Because of this, action B can never be recommended when the type profile is $(1, *, 2)$.
5. Thus, when the type profile is $(1, *, 2)$, the signaling scheme has to recommend each of A and C with probability $\frac{1}{2}$. (The recommendation could of course follow different distributions based on the type of B; such a correlation is immaterial for our argument.)
6. Given that action B cannot be recommended when action A has type 1, or when action B has type 2, it must *always* be recommended for type vectors $(2, 1, *)$.
7. This implies that all faces of $D_{B,1}$ must be larger than all faces on $D_{C,1}$ and on $D_{C,2}$.
8. This is a contradiction to Step 5, which states that with positive probability, $D_{C,2}$ beats $D_{B,1}$.

Thus, we have proved that there is no dice-based implementation of any optimal signaling scheme for the given instance. ◀

Proof of Theorem 21. When the actions' (or more generally: candidates') type distributions are i.i.d., i.e., T_i and f_i are the same for all candidates i, Dughmi and Xu [8] have shown that there is an optimal *symmetric* signaling scheme, or more generally a symmetric second-order interim rule $\boldsymbol{X}$. We show that any symmetric second-order interim rule $\boldsymbol{X}$ is uniquely determined by its first-order component, a fact implicit in [8].

For symmetric rules, $x_{i,i',t}$ depends only on whether $i = i'$ or $i \neq i'$, but not on the identities of the candidates i and i'. Therefore, $\boldsymbol{X}$ can be equivalently described by two type-indexed vectors $\boldsymbol{y}$ and $\boldsymbol{z}$, where $y_t = x_{i,i,t}$ for all candidates i, and $z_t = x_{i,i',t}$ for all candidates i and i' with $i \neq i'$. The vector $\boldsymbol{y}$ is a first-order interim rule, and we refer to it as the *first-order component* of $\boldsymbol{X}$. If $\boldsymbol{X}$ is feasible and implemented by $\mathcal{A}$, then $y_t = x_{i,i,t} = \mathbf{Pr}[\mathcal{A}(\boldsymbol{t}) = i \mid t_i = t]$ for all candidates i, so $\boldsymbol{y}$ is the first-order interim rule implemented by $\mathcal{A}$. For every candidate i and type t, we have

$$1 \;=\; \sum_{i'=1}^{n} \mathbf{Pr}[\mathcal{A}(\boldsymbol{t}) = i' \mid t_i = t] \;=\; (n-1)z_t + y_t.$$

Therefore, $\boldsymbol{z} = \frac{\mathbf{1}-\boldsymbol{y}}{n-1}$, and the first-order component of a symmetric second-order interim rule suffices to fully describe it. The second-order rule is also, by the preceding argument, efficiently computable from its first-order component, and is feasible if and only if its first-order component is a feasible symmetric interim rule. Moreover, by [5], feasibility of symmetric second-order interim rules can be checked in time polynomial in the number of types and candidates, and given a feasible symmetric second-order interim rule $\boldsymbol{X}$, a winner selection rule implementing $\boldsymbol{X}$ can be evaluated in polynomial time. ◀

7 Directions for Future Work

We have begun an investigation of dice-based winner selection rules, in which each of several candidates independently draws a "score" from a distribution (rolling a "die"), and a candidate set is selected to maximize the sum of scores, subject to a feasibility constraint. We have shown that dice-based winner selection rules can implement all first-order interim rules with matroid constraints, but not all second-order interim rules; in particular, there are instances of Bayesian persuasion in which no optimal signaling scheme can be implemented using dice.

A natural direction for future work is to understand the limits of dice-based winner selection rules. While our existence proof uses matroid properties, matroid constraints are not the limit of implementability by dice: in the full version, we show an example in which the feasible sets do not form a matroid, yet every feasible interim rule within the environment is implementable with dice. This rules out a characterization of the form "a feasibility constraint $\mathcal{I}$ has all feasible $(\boldsymbol{x}, f)$ implementable by dice if and only if $\mathcal{I}$ is a matroid." In fact, we do not know of any feasibility constraint and corresponding first-order feasible interim rule for which a dice-based implementation can be ruled out, though we strongly suspect that such examples exist. A difficulty in verifying our conjecture is that we are not aware of a useful general technique for proving the *non-existence* of a dice-based implementation for a given interim rule.

Another direction is to find an efficient algorithm for matroid environments. To derive an efficient algorithm from our existential proof, the functions g_t and h_t would have to be evaluated efficiently. Furthermore, even if a set of continuous dice is given, it is still unclear how to convert them to dice with finitely many faces in polynomial time.

References

1 Nibia Aires, Johan Jonasson, and Olle Nerman. Order sampling design with prescribed inclusion probabilities. *Scandinavian journal of statistics*, 29(1):183–187, 2002.

2 Saeed Alaei, Hu Fu, Nima Haghpanah, Jason D. Hartline, and Azarakhsh Malekian. Bayesian optimal auctions via multi-to single-agent reduction. In *Proc. 13th ACM Conf. on Electronic Commerce*, page 17, 2012.

3 Kim C. Border. Implementation of Reduced Form Auctions: A Geometric Approach. *Econometrica*, 59(4):1175–1187, 1991.

4 Kim C. Border. Reduced form auctions revisited. *Economic Theory*, 31(1):167–181, 2007.

5 Yang Cai, Constantinos Daskalakis, and S. Matthew Weinberg. An algorithmic characterization of multi-dimensional mechanisms. In *Proc. 44th ACM Symp. on Theory of Computing*, pages 459–478, 2012.

6 Yang Cai, Constantinos Daskalakis, and S. Matthew Weinberg. Optimal multi-dimensional mechanism design: Reducing revenue to welfare maximization. In *Proc. 53rd IEEE Symp. on Foundations of Computer Science*, pages 130–139, 2012.

7 Shaddin Dughmi. Algorithmic information structure design: a survey. *ACM SIGecom Exchanges*, 15(2):2–24, 2017.

8 Shaddin Dughmi and Haifeng Xu. Algorithmic Bayesian Persuasion. In *Proc. 48th ACM Symp. on Theory of Computing*, pages 412–425, 2016.

9 Parikshit Gopalan, Noam Nisan, and Tim Roughgarden. Public projects, Boolean functions and the borders of Border's Theorem. In *Proc. 16th ACM Conf. on Economics and Computation*, page 395, 2015.

10 Martin Grötschel, László Lovász, and Alexander Schrijver. *Geometric algorithms and combinatorial optimization*, volume 2. Springer Science & Business Media, 2012.

11 Jason D. Hartline. *Mechanism design and approximation*. Now Publishers, 2013.

12 Emir Kamenica and Matthew Gentzkow. Bayesian persuasion. *American Economic Review*, 101(6):2590–2615, 2011.

13 Jean-Bernard Lasserre. *Moments, positive polynomials and their applications*, volume 1. World Scientific, 2010.

14 Eric Maskin and John Riley. Optimal auctions with risk averse buyers. *Econometrica*, 52(6):1473–1518, 1984.

15 Steven A. Matthews. On the implementability of reduced form auctions. *Econometrica*, 52(6):1519–1522, 1984.

16 Konrad Mierendorff. Asymmetric reduced form auctions. *Economics Letters*, 110(1):41–44, 2011.

17 Roger B. Myerson. Optimal Auction Design. *Mathematics of Operations Research*, 6(1):58–73, 1981.

18 James G Oxley. *Matroid theory*, volume 3. Oxford University Press, USA, 2006.

19 Mallesh M. Pai and Rakesh Vohra. Optimal auctions with financially constrained buyers. *Journal of Economic Theory*, 150:383–425, 2014.

20 Bengt Rosén. Asymptotic theory for order sampling. *Journal of Statistical Planning and Inference*, 62(2):135–158, 1997.

21 Hassler Whitney. On the abstract properties of linear dependence. *American Journal of Mathematics*, 57(3):509–533, 1935.