Mathematics of Operations Research

Publication details, including instructions for authors and subscription information:
http://pubsonline.informs.org

Efficient Computation of Optimal Auctions via Reduced Forms

Saeed Alaei, Hu Fu, Nima Haghpanah, Jason Hartline, Azarakhsh Malekian

To cite this article:
Saeed Alaei, Hu Fu, Nima Haghpanah, Jason Hartline, Azarakhsh Malekian (2019) Efficient Computation of Optimal Auctions via Reduced Forms. Mathematics of Operations Research 44(3):1058-1086. https://doi.org/10.1287/moor.2018.0958

http://pubsonline.informs.org/journal/moor/

MATHEMATICS OF OPERATIONS RESEARCH
Vol. 44, No. 3, August 2019, pp. 1058–1086
ISSN 0364-765X (print), ISSN 1526-5471 (online)

Efficient Computation of Optimal Auctions via Reduced Forms

Saeed Alaei,[a] Hu Fu,[b] Nima Haghpanah,[c] Jason Hartline,[d] Azarakhsh Malekian[e]

[a] Google Research, Mountain View, California 94043; [b] Department of Computer Science, University of British Columbia, Vancouver, British Columbia V6T 1Z4, Canada; [c] Department of Economics, Pennsylvania State University, State College, Pennsylvania 16802; [d] EECS Department, Northwestern University, Evanston, Illinois 60208; [e] Rotman School of Management, University of Toronto, Toronto, Ontario M5S 3E6, Canada

Contact: saeed.a@gmail.com (SA); fu.hu.thu@gmail.com (HF); haghpanah@psu.edu, http://orcid.org/0000-0001-7025-4282 (NH); hartline@eecs.northwestern.edu (JH); azarakhshm@gmail.com (AM)

Received: March 18, 2016
Revised: September 20, 2017; October 11, 2017; May 1, 2018
Accepted: May 20, 2018
Published Online in Articles in Advance: May 30, 2019

MSC2000 Subject Classification: Primary: 91B26; secondary: 91B32
OR/MS Subject Classification: Primary: game theory, economics, social and behavioral sciences: market models (auctions, bargaining, bidding, selling, etc.); secondary: game theory, economics, social and behavioral sciences: resource and cost allocation

https://doi.org/10.1287/moor.2018.0958

Abstract. We study an optimal auction problem for selecting a subset of agents to receive an item or service, whereby each agent's service can be configured, the agent has multidimensional preferences over configurations, and there is a limit on the number of agents that can be simultaneously served. We give a polynomial time reduction from the multiagent problem to appropriately defined single-agent problems. We further generalize the setting to matroid feasibility constraints and obtain exact and approximately optimal reductions. As applications of this reduction we give polynomial time algorithms for the problem with quasi-linear preferences over configurations or with private budgets. Our approach is to characterize, and in polynomial time optimize and implement feasible *interim allocation rules*. With a single item, we give a new characterization showing that any mechanism has an ex post implementation as a simple token-passing process. These processes can be parameterized and optimized with a quadratic number of linear constraints. With multiple items, we generalize Border's characterization and give algorithms for optimizing interim and implementing ex post allocation rules. These implementations have a simple form; they are randomizations over greedy mechanisms that serve types in a given order.

Funding: S. Alaei is partially supported by the Office of Naval Research Young Investigator Program [Grant N000141110662]; H. Fu is supported by the National Science Foundation (NSF) [Grants CCF-0643934 and AF-0910940] and Natural Sciences and Engineering Research Council of Canada Discovery Grant Accelerator Supplements [RGPAS-2017-507934]. N. Haghpanah is supported by the NSF [Award CCF 0846113]. J. Hartline is supported by the NSF [Awards CCF 0846113 and CCF 0830773]. A. Malekian is partially supported by the NSF [Awards CCF 0846113 and CCF 0830773].

Keywords: auctions • dimensionality reduction • polymatroid constraints • reduced forms

1. Introduction

Bayesian optimal auctions for revenue are analytically complicated outside the standard linear and single-dimensional utility model of Myerson [34]. This paper considers this auction problem from an algorithmic perspective and shows that the space of auctions can be efficiently optimized over and the resulting auction can be efficiently implemented. Our results apply to the canonical single-item environment and, more generally, to multiunit and matriod environments (as defined below). The main challenge that our work resolves is in reducing the computational complexity of these problems from exponential, as it would be in the straightforward mathematical program, to polynomial in the number of agents.

Consider a seller who faces a set of agents who desire service, whereby there may be multiple configurations for each agent's service, possibly with different costs (e.g., when renting a car, you can get a global positioning system or not, you can get various insurance packages, and you will pay a total price). Each agent has preferences over the different possible configurations she can be served. The seller is restricted by a feasibility constraint that bounds the set of agents served (e.g., by the number of cars in the rental shop). When the agents' private preferences are drawn independently from a known prior distribution, the seller would like to design an auction to optimize her objective (e.g., revenue) in expectation over this Bayesian prior distribution, subject to feasibility. A number of applications that fit within this abstract framework are given at the end of this introduction.

One of the main challenges of this optimization problem is its large size: a mechanism must determine for every *profile* of *types* (i.e., preferences) the allocation, that is the subset of agents who receive service, and also the configurations and payments. A Bayesian mechanism can be succinctly described by a profile of *interim mechanisms* (also referred to as the mechanism's *reduced form* in the literature by Border [9]). An agent's interim

mechanism specifies for each of her types a distribution over service configurations and payments she is assigned when types of other agents are drawn at random from their distributions. Importantly, an interim representation of a mechanism removes the exponential dimensionality dependence on the number of agents. Although it is immediate that the profile of interim mechanisms is sufficient for checking each agent's incentive constraints, they are also sufficient for checking the designer's ex post feasibility constraint.

A profile of interim mechanisms is *interim feasible* if it is induced, as above, by an (ex post feasible) mechanism. An *interim allocation rule*, which specifies the probability that an agent is served as a function of her type (but not the configurations or payments), contains all the relevant information from the interim mechanisms for verifying interim feasibility. As an example, consider an allocation problem with a single service and two identical agents. Suppose an agent's type is high or low with probability 1/2 each. Consider two interim allocation rules: rule (a) serves the agent with probability one when her type is high and with probability zero otherwise, and rule (b) serves the agent with probability 1/2 regardless of her type. It is feasible for both agents to have rule (b) or for one agent to have rule (a) and the other to have rule (b); on the other hand, it is infeasible for both agents to have rule (a). This last combination is infeasible because with probability 1/4 both agents have high types, but we cannot simultaneously serve both of them. An important problem in the general theory of auctions is to decide when a profile of interim allocation rules is feasible, and furthermore, when it is feasible to find the ex post description of a mechanism that implements it.

A structural characterization of the necessary and sufficient conditions for the aforementioned interim feasibility is important for the construction of optimal auctions because it effectively allows the auction problem to be decomposed across agents. If we can optimally serve a single agent for a given interim allocation rule and we can check feasibility of a profile of interim allocation rules, then we can optimize over auctions. Effectively, we can reduce the multiagent auction problem to a collection of single-agent auction problems.

Border [9] characterized single-item interim feasibility (that is, when only one agent can be served). A profile of interim allocation rules is feasible if for any subspace of the agent types the expected number of served agents (according to the interim allocation) in this subspace is at most the probability that there is an agent whose type is in this subspace. Returning to our infeasible example above, the expected number of served agents with high types is one (according to the interim allocation), whereas the probability that there is an agent with a high type is 3/4; Border's condition is violated.

The straightforward formulation of interim feasibility via Border's characterization has exponentially many constraints. This issue is addressed in two different lines of work for a single-item feasibility setting. First, in single-parameter settings or with some extensions, and with symmetric agents (whereby the agents' type spaces and distributions are identical), analysis of incentive constraints can be used to simplify feasibility to a polynomial number of constraints. This simplification of the characterization has led to an analytically tractable theory of auctions when agents have budgets (Laffont and Robert [24], Pai and Vohra [35]) or are risk averse (Matthews [29], Maskin and Riley [26]). Second, the combinatorial structure of interim feasibility can be exploited to solve the interim problem with exponentially many constraints. Vohra [43] showed that interim feasible allocation rules form a polymatroid. This implies that the interim feasible allocation rules that optimize any given linear objective (e.g., Myerson's virtual surplus) can be found via the greedy algorithm. Belloni et al. [7] solved a single-item multidimensional problem with configurable quality levels (special case of our model and equivalent to the quasi-linear preferences of Section 3.1) using a cutting plane method for verifying feasibility. However, Belloni et al. [7] did not consider the problem of recovering an ex post description of the mechanism. Our paper contributes to the second line of work by generalizing feasibility from single-item to multiunit feasibility constraints, that is, at most k agents can be served (and more generally with *matroid* feasibility constraints, defined formally later); and by solving and implementing the interim solution in polynomial time in the size of the problem without restriction to any particular preference structure, but instead by assuming access to an algorithmic solution to the single-agent problem.

Our main theorems give computationally tractable (i.e., in polynomial time in the total number of agents' types) methods for optimization and implementation of interim allocation rules. In particular, with single-item feasibility, we show that the exponentially faceted polytope specified by the interim feasibility constraints is a projection of a quadratically faceted polytope in a higher dimension, and an ex post allocation rule implementing a feasible profile of interim allocation rules is given immediately by the latter's preimage in the higher dimensional polytope. These results combine to give a (computationally tractable) reduction from the multiagent auction problem to a collection of single-agent problems. In this reduction, interim feasibility is captured by a linear program with a quadratic number of constraints. Furthermore, our algorithmic procedure characterizes every single-item auction as implementable by a simple token-passing process: the seller issues a token which she initially holds. Agents are ordered arbitrarily, visited one by one in this order, and will be

given the token with a certain probability based on their and the current holder's identity and type. The agent who holds the token at the end of the process is served (no one is served if the holder is the seller). Going back to the example above, when the interim allocation rules are (a) for agent 1 and (b) for agent 2, the transitions are in fact deterministic and are as follows: the high type of agent 1 takes the token deterministically from the seller; any type of agent 2 takes the token deterministically if seller is the holder; all other probabilities are zero (there are also other implementations).

For the more general problem in which the seller faces a multiunit (or matroid) feasibility constraint, we generalize Border's characterization of interim feasibility and use it to design optimal auctions. Our characterization is based on polymatroidal decomposition. We use the decomposition to show that interim feasibility is a polymatroid associated with a submodular function referred to as the "expected rank function," which is a certain expectation over the rank function of the matriod feasibility constraint. First, though the number of constraints from this polymatroidal characterization is exponential in the number of agents, any of the standard algorithms for submodular function minimization can be used as a separation oracle to identify a violated feasibility constraint in any profile of interim allocation rules, as long as the expected rank function can be computed in polynomial time. With such a separation oracle, the expected revenue over the space of interim feasible allocation rules can be efficiently optimized. Second, the vertices of a polymatroid correspond to particularly simple auctions, which can be implemented by simple greedy allocation rules. Because any point in the interior of the polytope of interim feasibility can be implemented as a convex combination of its vertices, the optimal mechanism can be specified as a randomization over these greedy allocation rules. This approach enables us again to reduce the multiagent problem to single-agent problems. When the expected rank function cannot be computed in polynomial time, we present a sampling scheme that yields a fully polynomial time randomized approximation scheme.

The reductions discussed above run in polynomial time in the size of the type space. When the type space is infinite or large but otherwise succinctly described we would prefer reductions that are polynomial time in the size of the succinct representation of the type space. An important example is the case in which agents' preferences are multidimensional but the multidimensional parameters are independently distributed. Although the size of such a type space is exponentially large in the dimension, it can be succinctly described in space linear in the dimension. In Appendix A, we present a $(1+\epsilon)$-approximate reduction for any ϵ; mechanisms from this reduction can be optimized over and implemented in a polynomial, in the number of agents n and $1/\epsilon$, basic computational operations and black box calls to solutions to the single-agent problems. Specifically, there is no dependence on the size or representation of the type space. [Although our results give reductions from multiagent mechanism design to single-agent mechanism design for agents with succinctly represented type spaces, some such single-agent problems are known to be computationally intractable (e.g., Chen et al. [17]). When the single-agent problems are computationally intractable, our reduction does not sidestep this intractability.]

Several revenue-optimization problems are captured with our model and hence can be computationally solved using our methods. Direct applications of our model include selling products with configurable quality levels (Belloni et al. [7]) or delivery time (Dewan and Mendelson [19], Mendelson and Whang [31]), cable TV subscriptions with configurable bundles of channels subject to capacity and network constraints on the set of subscribers (Crawford [18], Bakos and Brynjolfsson [5]), and products that can be accompanied with bundles of information that affect the product's value (such as user data in online advertising).

Another application of our framework is multiattribute procurement (Beil and Wein [6], Parkes and Kalagnanam [36], Ronen and Lehmann [39]). The auctioneer is a firm wishing to acquire a service, for example to outsource a project. The service can be provided with different configurations, for example the duration and quality, the percentage that is off-shored, etc. Each agent has a private preference over payments and each configuration of service (e.g., distinct costs for providing each configuration). The auctioneer wishes to maximize the value for the acquired configuration of service (if any), minus the payment. The mechanism in this scenario determines for each profile of types (parameterizing preferences) the provider of service and configuration, and the payment. Though we have phrased our results in terms of an auctioneer as the provider of a service, the framework applies equally well to solve the problem of multiattribute procurement.

1.1. Related Work

Myerson [34] characterized Bayesian optimal auctions in environments with quasi-linear risk-neutral single-dimensional agent preferences. Bulow and Roberts [11] reinterpreted Myerson's approach as reducing the multiagent single-item auction problem to a related single-agent problem. Alaei [2] relaxes and generalizes this approach to reduce the problem of approximately optimal multi-item auctions for multidimensional

agents to a multidimensional generalization of the single-agent problems in Bulow and Roberts [11]. The single-agent problems considered in Bulow and Roberts [11] and Alaei [2] are given by ex ante constraints (i.e., on the probability that an agent receives a good or service); optimal auctions for multidimensional and nonlinear agents cannot be generally be reduced to these single-agent problems. Inspired by Alaei [2], our work defines the single-agent problems (given by interim constraints, i.e., on the probabilities that each type of an agent receives a good or service) that admit optimal multiagent reductions; and we give such reductions for matroid-constrained auction problems with general agent preferences. The subsequent work of Alaei et al. [3] characterizes the environments for which a Bulow and Roberts [11]–style ex ante reduction is optimal and bounds its loss in revenue more generally when it is not optimal. Importantly, the Bulow and Roberts [11]–style reduction is analytically and computationally much simpler than the interim reductions that we give in this paper.

An important aspect of our approach is that it can be applied to general multidimensional agent preferences. Multidimensional preferences can arise as distinct values for different configurations of the good or service being auctioned, in specifying a private budget and a private value, or in specifying preferences over risk. We briefly review related work for agent preferences with multiple values, budgets, or risk parameters.

Multidimensional valuations are well known to be difficult. For example, Rochet and Chone [38] showed that because *bunching* (i.e., a group of distinct types treated the same way by the mechanism) cannot be ruled out easily, the optimal auctions for multidimensional valuations are dramatically different from those for single-dimensional valuations. Because of this, most results are for cases with special structure (e.g., Armstrong [4], Wilson [44], and McAfee and McMillan [30]) and often, by using such structures, reduce the problems to single-dimensional ones (e.g., Roberts [37], Spence [42], and Mirman and Sibley [33]). Our framework does not need any such structure.

A number of papers consider optimal auctions for agents with budgets; see, for example, Maskin [27], Che and Gale [15], and Pai and Vohra [35]. These papers rely on budgets being public or the agents being symmetric; our technique allows for a nonidentical prior distribution and private budgets. Mechanism design with risk-averse agents was studied by Maskin and Riley [26] and Matthews [28]. Both works assume independent and identically distributed (i.i.d.) prior distributions and have additional assumptions on risk attitudes; our reduction does not require these assumptions.

Characterization and ex post implementation of interim feasible mechanisms play vital roles in this work. For single-item auctions, necessary and sufficient conditions for interim feasibility were developed through a series of works (Border [9, 8], Matthews [29], Mierendorff [32], Maskin and Riley [26]). These characterizations have proved useful for deriving properties of mechanisms; the work of Manelli and Vincent [25] is a recent example. Border [9] characterized symmetric interim feasible auctions for single-item auctions with identically distributed agent preferences. His characterization is based on the definition of "hierarchical auctions." He observes that the space of interim feasible mechanisms is given by a polytope, where vertices of this polytope correspond to hierarchical auctions, and interior points correspond to convex combinations of vertices. Mierendorff [32] generalizes Border's approach and characterization to asymmetric single-item auctions. The characterization via hierarchical auctions differs from our characterization via ordered subset auctions in that hierarchical auctions allow for some types to be relatively unordered with the semantics that these unordered types will be considered in a random order; it is important to allow for this when solving for symmetric auctions. Convex combinations over hierarchical auctions and ordered subset auctions provide the same generality. Our work generalizes the characterization from asymmetric single-item auctions to asymmetric matroid auctions. Independently and concurrently to our work, Che et al. [16] generalize characterization of interim feasibility to auctions with capacity constraints, which includes our matroid feasibility constraints. However, Che et al. [16] are not concerned with computational issues related to optimization and implementation on interim allocation rules, nor applications of the characterization in solving optimal auction problems with general preferences. Che et al. [16] prove the characterization with a network-flow approach, whereas our proof is based on a polymatroidal decomposition. Subsequent to our work, Cai et al. [13, 14] give an algorithmic characterization of interim feasibility for general feasibility constraints as *stochastic virtual welfare maximizers*. Specifically, they show that the vertices of the interim feasible polytope can be implemented by an ex post allocation that optimizes an appropriately defined virtual welfare (and any interior point in the polytope can be implemented as a convex combination of vertices).

Our main result provides computational foundations to the interim feasibility characterizations discussed above. Independently and contemporaneously, Cai et al. [12] provided similar computational foundations for the problem with single-item feasibility constraints. Their approach to the single-item constraints is most comparable to our approach for the matroid constraints, whereby the optimization problem is written as

a convex program that can be solved by the ellipsoid method; although these methods result in strongly polynomial time algorithms, they are not considered practical. In contrast, our single-item approach, when the single-agent problems can be solved by a linear program, gives a single linear program that can be practically solved. Whereas our work gives computationally tractable interim feasibility characterizations with configurable service and matroid constraints, Cai et al. [12] study multi-item auctions with agents with additive preferences (they observe that interim feasibility for multi-item auctions decomposes across the items and, thus, the single-item characterization of interim feasibility of Mierendorff generalizes). The subsequent work of Cai et al. [13, 14] generalizes ours and gives polynomial time algorithms for optimizing over and implementing optimal auctions for general feasibility environments. These algorithms are based on their general characterization of interim feasibility as stochastic virtual welfare optimizers and the ellipsoid method.

The theses of Alaei [1] and Haghpanah [23] summarize and expound on the results of this paper.

1.2. Organization

In Section 2, we describe single- and multiagent mechanism design problems. In Section 3, we give algorithms for solving two kinds of single-agent problems: quasi-linear preferences over configurations and private-value private-budget preferences. In Section 4, we give a high-level description of the multi- to single-agent reduction, which allows for efficiently computing optimal mechanisms for many service-based environments. The key step therein, an efficient algorithm that implements any jointly feasible set of interim allocation rules, is presented in Section 5. This section is divided into three parts, which address single-unit, multiunit, and matroid feasibility constraints, respectively. Conclusions and extensions are discussed in Section 6.

2. Preliminaries

We start by defining single-agent preference structure and related incentive and rationality concepts and then move on to the general multiagent problem definition with interagent feasibility constraints.

2.1. Single-Agent Mechanisms

We consider the provisioning of an abstract service. This service may be parameterized by an *attribute* (e.g., quality of service) and may be accompanied by a required payment. We denote the outcome obtained by an agent as $w \in W$. We view this outcome as giving an indicator for whether an agent is served and as describing attributes of the service, such as quality of service and monetary payments. Let $\text{Alloc}(w) \in \{0,1\}$ be an indicator for whether the agent is served; let $\text{Payment}(w) \in \mathbb{R}$ denote any payment the agent is required to make, and let $\text{Cost}(w) \in \mathbb{R}$ be the cost to the seller. In a randomized environment (e.g., randomness from a randomized mechanism or Bayesian environment), the outcome an agent receives is a random variable from a distribution over W. The space of all such distributions is denoted $\Delta(W)$.

The agent has a type t from a finite type space T. This type is drawn from distribution $f \in \Delta(T)$ and we equivalently denote by f the probability mass function. That is, for every $t \in T$, $f(t)$ is the probability that the type is t. The utility function $u : T \times W \to \mathbb{R}$ maps the agent's type and the outcome to real valued utility. The agent is a von Neumann–Morgenstern expected utility maximizer, and we extend u to $\Delta(W)$ linearly, that is, for $w \in \Delta(W)$, $u(t,w)$ is the expectation of u where the outcome is drawn according to w. We do not require the usual assumption of quasi-linearity.

A single-agent mechanism, without loss of generality by the revelation principle, is just an *outcome rule*, a mapping from the agent's type to a distribution over outcomes. We denote an *outcome rule* by $w : T \to \Delta(W)$. We say that an outcome rule w is *incentive compatible* (IC) and *individually rational* (IR) if for all $t, t' \in T$, respectively,

$$u(t, w(t)) \geq u(t, w(t')), \tag{IC}$$

$$u(t, w(t)) \geq 0. \tag{IR}$$

We refer to the indicator of service in the outcome rule as the *allocation rule*. Because the allocation to each agent is a binary random variable, distributions over allocations are fully described by their expected value. Therefore the allocation rule $x : T \to [0,1]$ for a given outcome rule w is $x(t) = \mathbf{E}[\text{Alloc}(w(t))]$.

We give two examples to illustrate the abstract model described above. The first example is the standard single-dimensional linear risk-neutral preference, which is prevalent in auction theory. Here the agent's type space is $T \subset \mathbb{R}_+$, where $t \in T$ represents the agent's valuation for the item. The outcome space is $W = \{0,1\} \times \mathbb{R}_+$, where an outcome w in this space indicates whether the item is sold to the agent, by $\text{Alloc}(w)$, and at what price, by $\text{Payment}(w)$. The agent's quasi-linear utility function is $u(t,w) = t \cdot \text{Alloc}(w) - \text{Payment}(w)$.

The second example is that of an item with m configurations and quasi-linear and risk-neutral preference. Here the type space is $T \subset \mathbb{R}_+^m$, and a type $t \in T$ indicates the agent's valuation for each of the configurations when the agent's value for no service is normalized to zero. An outcome space is $W = \{0, \dots, m\} \times \mathbb{R}_+$. The first coordinate of w specifies which configuration the agent receives or none and $\text{Alloc}(w) = 1$ if it is nonzero; the second coordinate of w specifies the required payment $\text{Payment}(w)$. The agent's utility for w is the value the agent attains for the configuration received less her payment. Beyond these two examples, our framework can easily incorporate more general agent preferences exhibiting, for example, risk aversion or a budget limit (see Section 3.2).

Consider the following single-agent mechanism design problem. A feasibility constraint is given by an upper bound $x(t)$ on the probability that the agent is served as a function of her type t; the distribution on types in T is given by f. The *single-agent problem* is to find the outcome rule w^* that satisfies the allocation constraint of x and maximizes the performance (e.g., revenue). This problem is described by the following program:

$$\begin{aligned} \max_{w}: \quad & \mathbf{E}_{t\sim f, w(t)}\big[\text{Payment}(w(t)) - \text{Cost}(w(t))\big] \\ \text{s.t.} \quad & \mathbf{E}_{w(t)}\big[\text{Alloc}(w(t))\big] \le x(t), \qquad \forall t \in T \\ & w \text{ is IC and IR.} \end{aligned} \tag{SP}$$

We denote the outcome rule w^* that optimizes this program by $\text{Outcome}(x)$ and its revenue by $\text{Rev}(x) = \mathbf{E}_{t\sim f, w^*(t)}\big[\text{Payment}(w^*(t)) - \text{Cost}(w^*(t))\big]$. We note that, although this paper focuses on revenue maximization, the same techniques presented can be applied to maximize (or minimize) general separable objectives, such as social welfare.

2.2. Multiagent Mechanisms

There are n independent agents. Agents need not be identical (i.e., agent i's type space is T_i, the probability mass function for her type is f_i, her outcome space is W_i, and her utility function is u_i). The profile of agent types is denoted by $\mathbf{t} = (t_1, \dots, t_n) \in T_1 \times \cdots \times T_n = \mathbf{T}$, the joint distribution on types is $\mathbf{f} \in \Delta(T_1) \times \cdots \times \Delta(T_n)$, a vector of outcomes is $(w_1, \cdots, w_n) \in \mathbf{W}$, and an allocation is $(x_1, \dots, x_n) \in \{0,1\}^n$. The mechanism has an interagent feasibility constraint that permits serving at most k agents (i.e., $\sum_i x_i \le k$). Furthermore, in Section 5.3, we review the theory of *matroids* and extend our basic results environments with feasibility constraint derived from a matroid set system. A mechanism that obeys this constraint is *feasible*. Importantly, the mechanism has no interagent constraint on attributes or payments.

A mechanism maps type profiles to a (distribution over) outcome vectors via an *ex post outcome rule*, denoted $\hat{\mathbf{w}} : \mathbf{T} \to \Delta(\mathbf{W})$, where $\hat{w}_i(\mathbf{t})$ is the outcome obtained by agent i. We will similarly define $\hat{\mathbf{x}} : \mathbf{T} \to [0,1]^n$ as the *ex post allocation rule* (where $[0,1] \equiv \Delta(\{0,1\})$). The ex post allocation rule $\hat{\mathbf{x}}$ and the probability mass function $\mathbf{f}$ on types induce *interim* outcome and allocation rules. For agent i with type t_i and $\mathbf{t} \sim \mathbf{Dist_t}[\mathbf{t} \,|\, t_i]$ the interim outcome and allocation rules are $w_i(t_i) = \mathbf{Dist_t}\big[\hat{w}_i(\mathbf{t}) \,\big|\, t_i\big]$ and $x_i(t_i) = \mathbf{Dist_t}\big[\hat{x}_i(\mathbf{t}) \,\big|\, t_i\big] \equiv \mathbf{E_t}\big[\hat{x}_i(\mathbf{t}) \,\big|\, t_i\big]$ (we use notation $\mathbf{Dist}[X \,|\, E]$ to denote the distribution of random variable X conditioned on the event E). A profile of interim allocation rules is feasible if it is derived from an ex post allocation rule as described above; the set of all feasible interim allocation rules is denoted by $\mathbb{X}$. A mechanism is Bayesian incentive compatible and interim individually rational if Equations (IC) and (IR), respectively, hold for all i and all t_i.

Consider again the examples described previously of quasi-linear single-dimensional and multi-configuration preferences. For the single-dimensional example, the multiagent mechanism design problem is the standard single-item k-unit auction problem. For the multiple configurations example, the multiagent mechanism design problem is an *attribute auction*. In this problem, there are k units available, and each unit can be configured in one of m ways. Importantly, the designer's feasibility constraint restricts the number of units sold to be k but places no restrictions on how the units can be configured. For example, a restaurant has k tables, but each diner can order any of the m entrees on the menu.

A reduction from multiagent mechanism design to single-agent mechanism design as we have described above would assume that for any type space T_i, any probability mass function f_i, and interim allocation rule x_i, the optimal outcome rule $\text{Outcome}(x_i)$ and its performance $\text{Rev}(x_i)$ can be found efficiently (see Section 3 for examples). The goal then is to construct an optimal multiagent auction from these single-agent mechanisms. Our approach to such a reduction is as follows.

1. Optimize, over all feasible profiles of interim allocation rules $\mathbf{x} = (x_1, \dots, x_n) \in \mathbb{X}$, the sum of performances of the allocation rules $\sum_i \text{Rev}(x_i)$.

2. Implement the profile of interim outcome rules $\mathbf{w}$ given by $w_i = \text{Outcome}(x_i)$ with a feasible ex post outcome rule $\hat{\mathbf{w}}$.

Two issues should be noted. First, step 2 requires an argument that the existence of a feasible ex post outcome rule for a given profile of interim allocation rules implies the existence of one that combines the optimal interim outcome rules from Outcome(·). We address this issue in Section 4. Second, step 1 requires that we optimize over jointly feasible interim allocation rules, and after solving for $\mathbf{x}$, its implementation by an ex post allocation rule is needed to guide step 2. We address this issue in Section 5. For single-unit (i.e., $k = 1$) auctions a characterization of the necessary and sufficient condition for interim feasibility was provided by Kim Border. Roughly speaking, it requires that for any subset of types, the probability of service according to the interim allocations be upper bounded by the probability that a random profile of types includes at least a member of that subset.

Theorem 1 (9). *In a single-item auction environment, interim allocation rules* $\mathbf{x}$ *are feasible (i.e.,* $\mathbf{x} \in \mathbb{X}$*) if and only if the following holds:*

$$\forall S_1 \subseteq T_1, \cdots, \forall S_n \subseteq T_n : \qquad \sum_{i=1}^{n} \mathbf{E}\left[x_i(t_i) \,\middle|\, t_i \in S_i\right] \cdot \mathbf{Pr}\left[t_i \in S_i\right] \le \mathbf{Pr}_{\mathbf{t}\sim\mathbf{f}}\left[\exists i \in [n] : t_i \in S_i\right]. \tag{MRMB}$$

3. The Single-Agent Problem

Given an allocation rule $x(\cdot)$ as a constraint the single-agent problem is to find the (possibly randomized) outcome rule $w(\cdot)$ that allocates no more frequently than $x(\cdot)$, that is, $\forall t \in T$, $\mathbf{E}_{w(t)}\left[\text{Alloc}(w(t))\right] \le x(t)$, with the maximum expected performance. Recall that the optimal such outcome rule is denoted by Outcome(x), and its performance (e.g., revenue) is denoted by Rev(x). We first observe that Rev(·) is concave.

Proposition 1. Rev(·) *is a concave function in* x.

Proof. Consider any two allocation rules x and x', and any $\alpha \in [0,1]$. Define x'' to be $\alpha x + (1-\alpha)x'$. We will show that $\alpha \,\text{Rev}(x) + (1-\alpha)\,\text{Rev}(x') \le \text{Rev}(x'')$, which proves the claim. To see this, let w and w' be Outcome(x) and Outcome(x'), respectively. Define w'' to be the outcome rule that runs w with probability α, and w' with probability $1-\alpha$. The incentive compatibility of outcome rules w and w' imply the incentive compatibility of w'', because for any $t, t' \in T$, we have

$$\begin{aligned}\mathbf{E}\left[u(t, w''(t))\right] &= \alpha\mathbf{E}\left[u(t, w(t))\right] + (1-\alpha)\mathbf{E}\left[u(t, w'(t))\right] \\ &\ge \alpha\mathbf{E}\left[u(t, w(t'))\right] + (1-\alpha)\mathbf{E}\left[u(t, w'(t'))\right] \\ &= \mathbf{E}\left[u(t, w''(t'))\right].\end{aligned}$$

Additionally, w'' is feasible because $\mathbf{E}\left[\text{Alloc}(w''(t))\right] = \alpha\mathbf{E}\left[\text{Alloc}(w(t))\right] + (1-\alpha)\mathbf{E}\left[\text{Alloc}(w'(t))\right] \le x''(t)$ for all $t \in T$. As a result, Rev(x'') is at least the revenue of w'', which is in turn equal to $\alpha \,\text{Rev}(x) + (1-\alpha)\,\text{Rev}(x')$. □

We now give two examples for which the single-agent problem is computationally tractable. Both of these examples are multidimensional. The first example is that of a standard multi-item auction with unit-demand preferences. The second example is that of a single item and a private budget. For both of these problems the single-agent problem can be expressed as a linear program with size polynomial in the cardinality of the agent's type space.

3.1. Quasi-linear Preferences over Configurations

There are m configurations available. For $j \in [m]$, c_j is the cost of configuration j to the seller. There is a finite type space $T \subset \mathbb{R}_+^m$; the outcome space W is the direct product between an assignment to the agent of one of the m configurations, or none, and a required payment. $\Delta(W)$ is the cross-product of a probability distribution over which configuration the agent receives and a probability distribution over payments. Without loss of generality for a quasi-linear agent such a randomized outcome can be represented as $w = (w_1, \ldots, w_m, w_p)$, where for $j \in [m]$, w_j is the probability that the agent receives configuration j and w_p is the agent's required payment.

A single-agent mechanism assigns to each type an outcome as described above. An outcome rule specifies an outcome for any type t of the agent as $w(t) = (w_1(t), \ldots, w_m(t), w_p(t))$. This gives $m+1$ nonnegative real

valued variables for each of $|T|$ types. The following linear program, which is a simple adaptation of one from Briest et al. [10] to include the feasibility constraint given by x, solves for the optimal single-agent mechanism:

$$\begin{aligned} \max: \quad & \textstyle\sum_{t\in T}\Big(w_p(t) - \sum_j w_j(t)c_j\Big) f(t) & \\ \text{s.t.} \quad & \textstyle\sum_j w_j(t) \le x(t) & \forall t \in T \\ & \textstyle\sum_j t_j w_j(t) - w_p(t) \ge \sum_j t_j w_j(t') - w_p(t') & \forall t, t' \in T \\ & \textstyle\sum_j t_j w_j(t) - w_p(t) \ge 0 & \forall t \in T. \end{aligned}$$

The optimal outcome rule from this program is $w^* = \text{Outcome}(x)$ and its performance is $\text{Rev}(x) = \mathbf{E}_{t\sim f}\big[w_p^*(t)\big]$.

Proposition 2. *The single-agent m-configuration quasi-linear problem can be solved in polynomial time in m and $|T|$.*

The model in this section is equivalent to the Belloni et al. [7] problem of different quality levels. Combined with our reduction, our result generalizes the solution from single item to multiple items with matroid feasibility. In addition, we recover an ex post description of the optimal mechanism by implementing feasible interim allocations.

3.2. Private Budget Preferences

There is a single item available with cost c. The agent has a private value for this item and a private budget (i.e., $T \subset \mathbb{R}_+^2$); we will denote by t_v and t_b this value and budget, respectively. The outcome space is $W = \{0,1\} \times \mathbb{R}$, where for $w \in W$ the first coordinate w_x denotes whether the agent receives the item and the second coordinate w_p denotes her payment. The agent's utility is

$$u(t,w) = \begin{cases} t_v w_x - w_p & \text{if } w_p \le t_b, \text{ and} \\ -\infty & \text{otherwise.} \end{cases}$$

Claim 1 below implies that when optimizing over distributions on outcomes we can restrict attention to $[0,1] \times [0,1] \times \mathbb{R}_+ \subset \Delta(W)$, where the first coordinate denotes the probability that the agent receives the item, the second coordinate denotes the probability that the agent makes a nonzero payment, and the third coordinate denotes the nonzero payment made.

Claim 1. *Any incentive compatible and individually rational outcome rule can be converted into an outcome rule above with the same expected revenue that is incentive compatible and individually rational.*

As a sketch of the argument to show this claim, note that if an agent with type t receives randomized outcome w, she is just as happy to receive the item with the same probability and pay her budget with probability equal to her previous expected payment divided by her budget (recall from Section 2 that a randomized outcome rule is individually rational if the utility of the agent is nonnegative *in expectation*; our transformation preserves this property). Such a payment is budget feasible and has the same expectation as before. Furthermore, this transformation only increases the maximum payment that any agent makes, which means that the relevant incentive compatibility constraints are only fewer. Importantly, the only incentive constraints necessary are ones that prevent types with higher budgets from reporting types with lower budgets.

A single-agent mechanism assigns to each type an outcome as described above. We denote the distribution over outcomes for t by $w(t) = (w_x(t), w_\rho(t), t_b)$, where only the first two coordinates are free variables. This gives two nonnegative real valued variables for each of $|T|$ types. The following linear program solves for the optimal single-agent mechanism:

$$\begin{aligned} \max: \quad & \textstyle\sum_{t\in T}\Big(t_b w_\rho(t) - c\Big) f(t) & \\ \text{s.t.} \quad & w_x(t) \le x(t) & \forall t \in T \\ & t_x w_x(t) - t_b w_\rho(t) \ge t_v w_x(t') - t_b' w_\rho(t') & \forall t, t' \in T \text{ with } t_b' \le t_b \\ & t_v w_x(t) - t_b w_\rho(t) \ge 0 & \forall t \in T \\ & w_\rho(t) \le 1 & \forall t \in T. \end{aligned}$$

The optimal outcome rule from this program is $w^* = \text{Outcome}(x)$, and its performance is $\text{Rev}(x) = \mathbf{E}_{t\sim f}\Big[t_b w_\rho^*(t)\Big]$.

Proposition 3. *The single-agent private budget problem can be solved in polynomial time in* $|T|$.

4. Multi- to Single-Agent Reductions

An ex post allocation rule $\hat{\mathbf{x}}$ takes as its input a profile of types $\mathbf{t} = (t_1, \ldots, t_n)$ of the agents, and indicates by $\hat{x}_i(\mathbf{t})$ a set of at most k winners. Agent i's type $t_i \in T_i$ is drawn independently at random from distribution $f_i \in \Delta(T_i)$. An ex post allocation rule implements an interim allocation rule $x_i : T_i \to [0,1]$, for agent i, if the probability of winning for agent i conditioned on her type $t_i \in T_i$ is exactly $x_i(t_i)$, where the probability is taken over the random types of other agents and the random choices of the allocation rule. A profile of interim allocation rules $\mathbf{x} = (x_1, \ldots, x_n)$ is feasible if and only if it can be implemented by some ex post allocation rule. $\mathbb{X}$ denotes the space of all feasible profiles of interim allocation rules.

The optimal performance (e.g., revenue) of the single-agent problem with allocation constraint given by x is denoted by $\mathrm{Rev}(x)$. The outcome rule corresponding to this optimal revenue is $\mathrm{Outcome}(x)$. Given any feasible interim allocation rule $\mathbf{x} \in \mathbb{X}$, say with ex post allocation rule $\hat{\mathbf{x}}$, we would like to construct an auction with revenue $\sum_i \mathrm{Rev}(x_i)$. We need to be careful because $\mathrm{Outcome}(x_i)$, by definition, is only required to have allocation rules *upper bounded* by x_i [see (SP) in Section 2], whereas the ex post allocation rule $\hat{x}_i$ implements x_i exactly, and hence we may need to implement a scaled-down version of $\hat{x}_i$. This is defined formally as follows.

Definition 1. An optimal auction $\hat{\mathbf{w}}^*$ for feasible interim allocation rule $\mathbf{x}$ (with corresponding ex post allocation rule $\hat{\mathbf{x}}$) is defined as follows on $\mathbf{t}$. For agent i:

1. Let $w_i^* = \mathrm{Outcome}(x_i)$ be the optimal outcome rule for allocation constraint x_i.
2. Let $x_i^* = \mathbf{E}\big[\mathrm{Alloc}(w_i^*)\big]$ be the allocation rule corresponding to outcome rule w_i^*.
3. If $\hat{x}_i(\mathbf{t}) = 1$, output

$$\hat{w}_i^*(\mathbf{t}) \sim \begin{cases} \mathbf{Dist}\left[w_i^*(t_i) \,\middle|\, \mathrm{Alloc}(w_i^*(t_i)) = 1\right] & \text{w.p. } x_i^*(t_i)/x_i(t_i), \text{ and} \\ \mathbf{Dist}\left[w_i^*(t_i) \,\middle|\, \mathrm{Alloc}(w_i^*(t_i)) = 0\right] & \text{otherwise.} \end{cases}$$

4. Otherwise (when $\hat{x}_i(\mathbf{t}) = 0$), output $\hat{w}_i^*(\mathbf{t}) \sim \mathbf{Dist}\left[w_i^*(t_i) \,\middle|\, \mathrm{Alloc}(w_i^*(t_i)) = 0\right]$.

Proposition 4. *For any feasible interim allocation rule* $\mathbf{x} \in \mathbb{X}$*, the optimal auction for this rule has expected revenue* $\sum_i \mathrm{Rev}(x_i)$.

Proof. The ex post outcome rule $\hat{\mathbf{w}}^*$ of the auction, by construction, induces interim outcome rule $\mathbf{w}^*$ for which the revenue is as desired. □

The optimal multiagent auction is the solution to optimizing the cumulative revenue of individual single-agent problems subject to the joint interim feasibility constraint given by $\mathbf{x} \in \mathbb{X}$.

Proposition 5. *The optimal revenue is given by the convex program*

$$\max_{\mathbf{x} \in \mathbb{X}} : \sum\nolimits_i \mathrm{Rev}_i(x_i). \tag{CP}$$

Proof. This is a convex program because $\mathrm{Rev}(\cdot)$ is concave and $\mathbb{X}$ is convex (convex combinations of feasible interim allocation rules are feasible). By Proposition 4 this revenue is attainable; therefore, it is optimal. □

5. Optimization and Implementation of Interim Allocation Rules

In this section, we address the computational issues pertaining to (1) solving optimization problems over the space of feasible interim allocation rules and (2) ex post implementation of such a feasible interim allocation rule. We present computationally tractable methods for both problems.

Normalized Interim Allocation Rules

It will be useful to "flatten" the interim allocation rule $\mathbf{x}$ for which $\mathbf{x}_i(t_i)$ denotes the probability that i with type t_i is served (randomizing over the mechanism and the draws of other agent types); we do so as follows. Without loss of generality, we assume that the type spaces of different agents are disjoint. This can be achieved by labeling all types of each agent with the name of that agent; that is, for each $i \in [n]$ we can replace T_i with $T_i' = \{(i,t) | t \in T_i\}$ so that $T_1', \cdots, T_n'$ are disjoint. Denoting the set of all types by $T_N = \bigcup_i T_i$, the interim allocation rule can be flattened as a vector in $[0,1]^{T_N}$.

Definition 2. The *normalized interim allocation rule* $\overline{x} \in [0,1]^{T_N}$ corresponding to interim allocation rule $\mathbf{x}$ under distribution $\mathbf{f}$ is defined as

$$\overline{x}(t_i) = x_i(t_i)\, f_i(t_i) \qquad \forall t_i \in T_N.$$

For the rest of this section, we refer to interim allocation rules via $\overline{x}$ instead of $\mathbf{x}$. Note that there is a one-to-one correspondence between $\overline{x}$ and $\mathbf{x}$ as specified by the above linear equation; so any linear or convex optimization problem involving $\mathbf{x}$ can be written in terms of $\overline{x}$ without affecting its linearity or convexity. Because $\mathbb{X}$ denotes the space of feasible interim allocation rules $\mathbf{x}$, we will use $\overline{\mathbb{X}}$ to denote the space of feasible normalized interim allocation rules.

In the remainder of this section, we characterize interim feasibility and show that normalized interim allocation rules can be optimized over and implemented in polynomial time.

5.2. Single-Unit Feasibility Constraints

In this section, we consider environments in which at most one agent can be allocated to. For such environments, we characterize interim feasibility as implementability via a particular, simple *stochastic sequential allocation* mechanism. Importantly, the parameters of this mechanism are easy to optimize efficiently.

A stochastic sequential allocation mechanism is parameterized by a stochastic transition table. Such a table specifies the probability by which an agent with a given type can steal a token from a preceding agent with a given type. For simplicity in describing the process we will assume the token starts under the possession of a "dummy agent" indexed by 0; the agents are then considered in the arbitrary order from 1 to n; and the agent with the token at the end of the process is the one that is allocated (or none are allocated if the dummy agent retains the token).

Definition 3 (Stochastic Sequential Allocation Mechanism). Parameterized by a stochastic transition table π, the *stochastic sequential allocation mechanism* (SSA) computes the allocations for a type profile $\mathbf{t} \in \mathbf{T}$ as follows:

1. Give the token to the dummy agent 0 with dummy type t_0.
2. For each agent i: (in order of 1 to n)
   if agent i' has the token, transfer the token to agent i with probability $\pi(t_{i'}, t_i)$.
3. Allocate to the agent who has the token (or none if the dummy agent has it).

We provide some examples of SSA in Appendix B. In particular, we present two examples of feasible ex post allocation rules and show that SSA can implement only the first one. We then show that even though the second ex post allocation rule cannot be implemented via SSA, SSA still can implement the corresponding interim allocation rule (with a different ex post allocation). This phenomenon is general. Later in this section we show that any feasible interim allocation rule is implementable by an SSA.

We next present a dynamic program, in the form of a collection of linear equations, for calculating the interim allocation rule implemented by SSA for a given π. Let $y(t_{i'}, i)$ denote the ex ante probability of the event that agent i' has type $t_{i'}$ and is holding the token at the end of iteration i. Let $z(t_{i'}, t_i)$ denote the ex ante probability in iteration i of SSA that agent i has type t_i and takes the token from agent i', who has type $t_{i'}$.

The following additional notation will be useful in this section. For any subset of agents $N' \subseteq N = \{1, \dots, n\}$, we define $T_{N'} = \bigcup_{i \in N'} T_i$. (Recall that without loss of generality agent type spaces are assumed to be disjoint.) The shorthand notation $t_i \in S$ for $S \subseteq T_N$ will be used to quantify over all types in S and their corresponding agents (i.e., $\forall t_i \in S$ is equivalent to $\forall i \in N, \forall t_i \in S \cap T_i$).

The normalized interim allocation rule $\overline{x}$ resulting from the SSA is exactly given by the dynamic program specified by the following linear equations.

$$y(t_0, 0) = 1, \tag{𝕊.1}$$

$$y(t_i, i) = \sum_{t_{i'} \in T_{\{0,\dots,i-1\}}} z(t_{i'}, t_i), \qquad \forall t_i \in T_{\{1,\dots,n\}} \tag{𝕊.2}$$

$$y(t_{i'}, i) = y(t_{i'}, i-1) - \sum_{t_i \in T_i} z(t_{i'}, t_i), \qquad \forall i \in \{1, \dots, n\}, \forall t_{i'} \in T_{\{0,\dots,i-1\}} \tag{𝕊.3}$$

$$z(t_{i'}, t_i) = y(t_{i'}, i-1)\pi(t_{i'}, t_i)\, f_i(t_i), \qquad \forall t_i \in T_{\{1,\dots,n\}}, \forall t_{i'} \in T_{\{0,\dots,i-1\}} \tag{π}$$

$$\overline{x}(t_i) = y(t_i, n), \qquad \forall t_i \in T_{\{1,\dots,n\}}.$$

Note that π is the only adjustable parameter in the SSA algorithm, so by relaxing the equation (π) and replacing it with the following inequality we can specify all possible dynamics of the SSA algorithm.

$$0 \le z(t_{i'}, t_i) \le y(t_{i'}, i-1)\, f_i(t_i), \qquad \forall t_i \in T_{\{1,\dots,n\}}, \forall t_{i'} \in T_{\{0,\dots,i-1\}}. \tag{𝕊.4}$$

Let $\mathbb{S}$ denote the convex polytope captured by the four sets of linear constraints ($\mathbb{S}$.1) through ($\mathbb{S}$.4) above, that is, $(y,z) \in \mathbb{S}$ if and only if y and z satisfy the aforementioned constraints. Note that every $(y,z) \in \mathbb{S}$ corresponds to some stochastic transition table π by solving equation (π) for $\pi(t_i, t_{i'})$. We show that $\mathbb{S}$ captures all feasible normalized interim allocation rules; that is, the projection of $\mathbb{S}$ on $\overline{x}(\cdot) = y(\cdot, n)$ is exactly $\overline{\mathbb{X}}$, as formally stated by the following theorem.

Theorem 2. *A normalized interim allocation rule $\overline{x}$ is feasible if and only if it can be implemented by the SSA algorithm for some choice of stochastic transition table π. In other words, $\overline{x} \in \overline{\mathbb{X}}$ if and only if there exists $(y,z) \in \mathbb{S}$ such that $\overline{x}(t_i) = y(t_i, n)$ for all $t_i \in T_N$.*

Corollary 1. *Given a blackbox for each agent i that solves for the optimal expected revenue $\mathrm{Rev}_i(x_i)$ for any feasible interim allocation rule $\mathbf{x}$, the optimal interim allocation rule can be computed by the following convex program, which is of quadratic size in the total number of types.*

$$\begin{aligned} \text{maximize} \quad & \sum_{i=1}^{n} \mathrm{Rev}_i(x_i) \\ \text{subject to} \quad & y(t_i, n) = \overline{x}(t_i) = x_i(t_i)\, f_i(t_i), \qquad \forall t_i \in T_N \\ & (y,z) \in \mathbb{S}. \end{aligned}$$

Furthermore, given an optimal assignment for this program, the computed interim allocation rule can be implemented by SSA using the stochastic transition table defined by

$$\pi(t_{i'}, t_i) = \frac{z(t_{i'}, t_i)}{y(t_{i'}, i-1)\, f_i(t_i)}, \qquad \forall t_i \in T_{\{1,\dots,n\}}, \forall t_{i'} \in T_{\{0,\dots,i-1\}}.$$

If $y(t_i, i'-1) = 0$, $\pi(t_i, t_{i'})$ can be set to an arbitrary value in $[0,1]$.

Next, we present a few definitions and lemmas that are used in the proof of Theorem 2. Two transition tables π and π' are considered *equivalent* if their induced normalized interim allocation rules for SSA are equal. Type t_i is called *degenerate* for π if in the execution of SSA the token is sometimes passed to type t_i but it is always taken away from t_i later, that is, if $y(t_i, i) > 0$ but $y(t_i, n) = 0$. The stochastic transition table π is degenerate if there is a degenerate type. For π, type t_i is *augmentable* if there exists a π' (with a corresponding y') that is *equivalent* to π for all types except t_i and has $y(t_i, n) < y'(t_i, n)$ (define t_0 to be augmentable unless the dummy agent never retains the token, in which case all agents are nonaugmentable, and for technical reasons we declare the dummy agent to be nonaugmentable as well).

Lemma 1. *For any stochastic transition table π there exists an equivalent π' that is nondegenerate.*

Lemma 2. *For any nondegenerate stochastic transition table π, any nonaugmentable type t_i always wins against any augmentable type $t_{i'}$. That is,*

- *if $i' < i$ and $t_{i'}$ has nonzero probability of holding the token, then $\pi(t_{i'}, t_i) = 1$, that is, t_i always takes the token away from $t_{i'}$; and*
- *if $i < i'$ and t_i has nonzero probability of holding the token, then $\pi(t_i, t_{i'}) = 0$, that is, $t_{i'}$ never takes the token away from t_i.*

It is possible to view the token passing in stochastic sequential allocation as a network flow. From this perspective, the augmentable and nonaugmentable types form a minimum-cut and Lemma 2 states that the token must eventually flow from the augmentable to nonaugmentable types. We defer the proof of this lemma to Appendix B where the main difficulty in its proof is that the edges in the relevant flow problem have dynamic (nonconstant) capacities.

Proof of Theorem 2. Any normalized interim allocation rule that can be implemented by the SSA algorithm is feasible because at most one real agent will hold the token at the end, so we only need to prove the opposite direction. The proof is by contradiction; that is, given a normalized interim allocation rule $\overline{x}$, we show that if there is no $(y,z) \in \mathbb{S}$ such that $\overline{x}(\cdot) = y(\cdot, n)$, then $\overline{x}$ must be infeasible. Consider the following linear program for a given $\overline{x}$ (i.e., $\overline{x}$ is constant).

$$\begin{aligned} \text{maximize} \quad & \sum_{t_i \in T_{\{1,\dots,n\}}} y(t_i, n) \\ \text{subject to} \quad & y(t_i, n) \le \overline{x}(t_i), \qquad \forall t_i \in T_{\{1,\dots,n\}} \\ & (y,z) \in \mathbb{S}. \end{aligned}$$

Let (y, z) be an optimal assignment of this linear programming (LP). If the first set of inequalities are all tight (i.e., $\overline{x}(\cdot) = y(\cdot, n)$), then $\overline{x}$ can be implemented by the SSA, so by contradiction there must exists a type $\tau^* \in T_N$ for which the inequality is not tight. Note that τ^* cannot be augmentable—otherwise, by the definition of augmentability, the objective of the LP could be improved. Partition T_N to augmentable types T_N^+ and nonaugmentable types T_N^-. Note that T_N^- is nonempty because $\tau^* \in T_N^-$. Without loss of generality, by Lemma 1 we may assume that (y, z) is nondegenerate, because there exists a nondegenerate assignment with the same objective value.

An agent wins if she holds the token at the end of the SSA algorithm. The ex ante probability that some agent with nonaugmentable type wins is $\sum_{t_i \in T_N^-} y(t_i, n)$. On the other hand, Lemma 2 implies that the first (in the order agents are considered by SSA) agent with nonaugmentable type will take the token from her predecessors and, although she may lose the token to another nonaugmentable type, the token will not be relinquished to any augmentable type. Therefore, the probability that an agent with a nonaugmentable type is the winner is exactly equal to the probability that at least one such agent exists, therefore

$$\mathbf{Pr}_{\mathbf{t}\sim\mathbf{f}}\left[\exists i : t_i \in T_N^-\right] = \sum_{t_i \in T_N^-} y(t_i, n) < \sum_{t_i \in T_N^-} \overline{x}(t_i).$$

The second inequality follows from the assumption above that τ^* satisfies $y(\tau^*, n) < \overline{x}(\tau^*)$. We conclude that $\overline{x}$ requires an agent with nonaugmentable type to win more frequently than such an agent exists, which is a contradiction to interim feasibility of $\overline{x}$. □

The contradiction that we derived in the proof of Theorem 2 yields a necessary and sufficient condition, as formally stated in the following corollary, for feasibility of any given normalized interim allocation rule.

Corollary 2. *A normalized interim allocation rule $\overline{x}$ is feasible if and only if*

$$\sum_{\tau \in S} \overline{x}(\tau) \le \mathbf{Pr}_{\mathbf{t}\sim\mathbf{f}}\left[\exists i : t_i \in S\right], \qquad \forall S \subseteq T_N. \tag{MRMB}$$

The necessity of condition (MRMB) is trivial because the left-hand side denotes the probability of some type in S happening and also being allocated to, whereas the right-hand side denotes the probability of at least one type in S happening. Its sufficiency was previously proved by Border [9]. This condition implies that the space of all feasible normalized interim allocation rules, $\overline{\mathbb{X}}$, can be specified by 2^D linear constraints on D-dimensional vectors $\overline{x}$. An important consequence of Theorem 2 is that $\overline{\mathbb{X}}$ can equivalently be formulated by only $O(D^2)$ variables and $O(D^2)$ linear constraints as a projection of $\mathbb{S}$, therefore any optimization problem over $\overline{\mathbb{X}}$ can equivalently be solved over $\mathbb{S}$.

5.3. k-Unit Feasibility Constraints

In this section, we consider environments in which at most k agents can be simultaneously allocated to. First, we generalize Border's characterization of interim feasibility to environments with k-unit feasibility constraint. Our generalization implies that the space of feasible normalized interim allocation rules is a polymatroid (Theorem 3). Second, we show that the normalized interim allocation rules corresponding to the vertices of this polymatroid are implemented by simple deterministic ordered-subset-based allocation mechanisms (Theorem 4). Third, we observe that optimization problems can be efficiently solved over polymatroids; this allows us to optimize over feasible interim allocation rules (Lemma 4). Furthermore, for any point in this polymatroid, the corresponding normalized interim allocation rule can be implemented by (1) expressing it as a convex combination of the vertices of the polymatroid, (2) sampling from this convex combination, and (3) using the ordered subset mechanism corresponding to the sampled vertex. We present an efficient randomized rounding routine for rounding a point in a polymatroid to a vertex, which combines the steps (1) and (2) (Theorem 6). These approaches together yield efficient algorithms for optimizing and implementing interim allocation rules.

5.3.1. Polymatroid Preliminaries. This subsection defines polymatroids and their related concepts. The main construct is a standard characterization of vertices of a polymatroid using ordered subsets of its ground set in Proposition 6 (see Schrijver [40] for a comprehensive treatment of polymatroids).

Consider an arbitrary set function $\mathcal{F} : 2^U \to \mathbb{R}_+$ defined over an arbitrary finite set U; let $P(\mathcal{F})$ denote the polytope associated with $\mathcal{F}$ defined as

$$P(\mathcal{F}) = \left\{ y \in \mathbb{R}_+^U \middle| \forall S \subseteq U : y(S) \le \mathcal{F}(S) \right\},$$

where $y(S)$ denotes $\sum_{s \in S} y(s)$. The convex polytope $P(\mathcal{F})$ is called a *polymatroid* if $\mathcal{F}$ is a submodular function. Even though a polymatroid is defined by an exponential number of linear inequalities, the separation problem

for any given $y \in \mathbb{R}_+^U$ can be solved in polynomial time as follows: find $S^* = \arg\min_S \mathcal{F}(S) - y(S)$; if y is infeasible, the inequality $y(S^*) \le \mathcal{F}(S^*)$ must be violated, and that yields a separating hyperplane for y. Note that $\mathcal{F}(S) - y(S)$ is itself submodular in S, so it can be minimized in strong polynomial time. Consequently, convex optimization problems can be solved over polymatroids in polynomial time. Next, we describe a characterization of the vertices of a polymatroid. This characterization plays an important role in our proofs and also in our ex post implementation of interim allocation rules.

Definition 4. For an arbitrary finite set U, an *ordered subset* $\pi \subseteq U$ is given by an ordering on elements $\pi = (\pi_1, \dots, \pi_{|\pi|})$, where shorthand notation $\pi_r \in \pi$ denotes the rth element in π.

The following characterization of vertices of polymatroid via ordered subsets was shown by Edmonds [21] and Shapley [41].

Proposition 6. *Let $\mathcal{F} : 2^U \to \mathbb{R}_+$ be an arbitrary nondecreasing submodular function with $\mathcal{F}(\emptyset) = 0$ and let $P(\mathcal{F})$ be the associated polymatroid with the set of vertices* $\textsc{Vertex}(P(\mathcal{F}))$. *Every ordered subset π of U (see Definition 4) corresponds to a vertex of $(P(\mathcal{F}))$, denoted by* $\textsc{Vertex}(P(\mathcal{F}), \pi)$, *which is computed as follows.*

$$\forall s \in U: \qquad y(s) = \begin{cases} \mathcal{F}(\{\pi_1, \dots, \pi_r\}) - \mathcal{F}(\{\pi_1, \dots, \pi_{r-1}\}) & \text{if } s = \pi_r \in \pi \\ 0 & \text{if } s \notin \pi. \end{cases}$$

Furthermore, for every $y \in \textsc{Vertex}(P(\mathcal{F}))$ there exists a corresponding π.

We next show that for any vertex of a polymatroid a corresponding ordered subset of elements can be computed efficiently.

Proposition 7. *For any $y \in \textsc{Vertex}(P(\mathcal{F}))$ the corresponding π can be computed efficiently via a greedy algorithm.*

Proof. The greedy algorithm works as follows. For each i from 1 to $|U|$, find $s \in U \setminus \{\pi_1, \dots, \pi_{i-1}\}$ such that $y(s) = \mathcal{F}(\{\pi_1, \dots, \pi_{i-1}, s\}) - \mathcal{F}(\{\pi_1, \dots, \pi_{i-1}\})$, breaking ties arbitrarily, and set $\pi_i \leftarrow s$. We show the correctness of the the greedy algorithm via contradiction. Suppose i is the first iteration for which we cannot find such s. Let π^* denote the ordered subset corresponding to y. Let i^* be the smallest index such that $\pi^*_{i^*} \notin \{\pi_1, \dots, \pi_{i-1}\}$. Both $\{\pi_1, \dots, \pi_{i-1}\}$ and $\{\pi^*_1, \dots, \pi^*_{i^*}\}$ are tight sets with respect to y, so by submodularity of $\mathcal{F}$ their union, which is $\{\pi_1, \dots, \pi_{i-1}\} \cup \{\pi^*_{i^*}\}$, is also a tight set, hence we could have picked $s = \pi^*_{i^*}$ in iteration i, contradicting the assumption that we could not find such s. □

5.3.2. Characterization and Implementation of Interim Feasibility. This subsection characterizes interim feasible allocation rules as an appropriately defined polymatroid. It also defines *ordered subset allocation mechanisms* and uses Proposition 6 to show that they are simple ex post implementations of interim feasible allocations.

Characterization of Interim Feasibility. We next characterize interim feasible mechanisms with k units, generalizing Border's result. Border's characterization of interim feasibility for $k = 1$ unit auctions states that the probability of serving a type in a subspace of type space is no more than the probability that a type in that subspace shows up. This upper bound is equivalent to the expected minimum of one and the number of types from the subspace that show up; furthermore, this equivalent phrasing of the upper bound extends to characterize interim feasibility in k-unit auctions.

We express an ex post allocation for type profile $\mathbf{t}$ by $\hat{x}^{\mathbf{t}} \in \{0,1\}^{T_N}$ as follows. For all $t'_i \in T_N$, $\hat{x}^{\mathbf{t}}(t'_i) = 1$ if player i is served and $t_i = t'_i$ and 0 otherwise. This definition of ex post allocations is convenient because the normalized interim allocation rule is calculated by taking its expectation, that is, $\overline{x}(t'_i) = \mathbf{E}_{\mathbf{t}}[\hat{x}^{\mathbf{t}}(t'_i)]$. Ex post feasibility requires that

$$\hat{x}^{\mathbf{t}}(S) \le \min(|\mathbf{t} \cap S|, k), \qquad \forall \mathbf{t} \in \mathbf{T}, \forall S \subseteq T_N, \tag{1}$$

where $\mathbf{t} \cap S$ denotes $\{t_1, \dots, t_n\} \cap S$. In other words, for any profile of types $\mathbf{t}$, the number of types in S that are served by $\hat{x}^{\mathbf{t}}$ must be at most the number of types in S that showed up in $\mathbf{t}$ and the upper bound k. Taking expectations of both sides of this equation with respect to $\mathbf{t}$ motivates the following definition and theorem.

Definition 5. The *expected rank function* for distribution $\mathbf{f}$ and subspace $S \subset T_N$ is

$$g_k(S) = \mathbf{E}_{\mathbf{t} \sim \mathbf{f}}\left[\min\left(|\mathbf{t} \cap S|, k\right)\right]. \tag{gk}$$

Theorem 3. *For supply constraint k and distribution $\mathbf{f}$, the space of all feasible normalized interim allocation rules, $\overline{\mathbb{X}}$, is the polymatroid associated with g_k, that is, $\overline{\mathbb{X}} = P(g_k)$. In particular, for all $\overline{x} \in \overline{\mathbb{X}}$,*

$$\overline{x}(S) \le \mathbf{E}_{\mathbf{t}}\left[\min(|\mathbf{t} \cap S|, k)\right] = g_k(S), \qquad \forall S \subseteq T_N. \tag{2}$$

The proof of this theorem will be deferred to the next section, where we will derive a more general theorem. A key step in the proof will be relating the statement of the theorem to the polymatroid theory described already. To show that the constraint of the theorem is a polymatroid, we observe that the expected rank function is submodular.

Lemma 3. *The expected rank function g_k is submodular.*

Proof. Observe that for any fixed $\mathbf{t}$, $\min(\mathbf{t} \cap S, k)$ is a submodular function in S, and therefore g_k is a convex combination (taking the expectation is the same as taking a convex combination) of submodular functions, so g_k is submodular. □

Implementation of Interim Allocation Rules. We now relate vertices of the polymatroid to ordered subset allocation mechanisms, defined below.

Definition 6 (Ordered Subset Allocation Mechanism). Parameterized by an ordered subset π of T_N (see Definition 4), the *ordered subset mechanism*, on profile of types $\mathbf{t} \in \mathbf{T}$, orders the agents on the basis of their types according to π and allocates to the agents greedily (e.g., with k units available the k first-ordered agents received a unit). An agent i with type $t_i \notin \pi$ is never served.

Remark 1. The virtual valuation-maximizing mechanisms from the classic literature on revenue-maximizing auctions are ordered subset mechanisms; see, for example, Myerson [34], an observation made previously by Edith [20]. The difference between these ordered subset mechanisms and the classic virtual valuation maximization mechanisms is that our ordered subset will come from solving an optimization on the whole auction problem, whereas Myerson's virtual values come directly from single-agent optimizations.

Theorem 4. *For supply constraint k, for an arbitrary vertex $\overline{x} = \textsc{Vertex}(P(g_k), \pi) \in \overline{\mathbb{X}}$ of the polymatroid $P(g_k)$, the unique ex post implementation is the ordered subset mechanism induced by π (Definition 6).*

Proof. Let $\overline{x} = \textsc{Vertex}(P(g_k), \pi)$ be an arbitrary vertex of $P(g_k)$ with a corresponding ordered subset π; by Proposition 6, such a π exists for every vertex of a polymatroid. For every integer $r \le |\pi|$, define $S^r = \{\pi_1, \ldots, \pi_r\}$ as the r-element prefix of the ordering. By Proposition 6, inequality (2) must be tight for every S^r, which implies that inequality (1) must also be tight for every S^r and every $\mathbf{t} \in \mathbf{T}$. Observe that inequality (1) being tight for a subset S of types implies that any ex post allocation mechanism implementing $\overline{x}$ must allocate as much as possible to types in S. By definition, an ordered subset mechanism allocates to as many types as possible (up to k) from each S^r; this is the unique outcome given that inequality (1) is tight for every S^r. □

5.3.3. Computationally Efficient Optimization and Implementation. The characterization of interim feasibility as a polymatroid constraint immediately enables efficient solving of optimization problems over the feasible interim allocation rules, as long as we can compute g_k efficiently (see Schrijver [40] for optimization over polymatroids). The following lemma states that g_k can be computed efficiently.

Lemma 4. *For independent agents (i.e., if $\mathbf{f}$ is a product distribution), $g_k(S)$ can be exactly computed in time $O((n + |S|) \cdot k)$ for any $S \in T_N$ using dynamic programming.*

Ex Post Implementation of Feasible Interim Allocation Rules. We now address the task of finding an ex post implementation corresponding to any $\overline{x} \in \overline{\mathbb{X}}$. By Theorem 7, if $\overline{x}$ is a vertex of $\overline{\mathbb{X}}$, it can be implemented by an ordered subset allocation mechanism as explained in Definition 6. Because any point in the polymatroid (or any convex polytope) can be specified as a convex combination of its vertices, to implement the corresponding interim allocation rule it is enough to show that this convex combination can be efficiently sampled from. An ex post implementation can then be obtained by sampling a vertex and using the ordered subset mechanism corresponding to that vertex. Instead of explicitly computing this convex combination, we present a general randomized rounding routine $\textsc{RandRound}(\cdot)$, which takes a point in a polymatroid and returns a vertex of the polymatroid such that the expected value of every coordinate of the returned vertex is the same as the original point. This approach is formally described next.

Definition 7 (Randomized Ordered Subset Allocation Mechanism). Parameterized by a normalized interim allocation rule $\overline{x} \in \overline{\mathbb{X}}$, a *randomized ordered subset allocation mechanism* (RRA) computes the allocation for a profile of types $\mathbf{t} \in \mathbf{T}$ as follows.

1. Let $(\overline{x}^*, \pi^*) \leftarrow \textsc{RandRound}(\overline{x})$.
2. Run the ordered subset mechanism (Definition 6) with ordered subset π^*.

Theorem 5. *Any normalized interim allocation rule $\overline{x} \in \overline{\mathbb{X}}$ can be implemented by the randomized ordered subset allocation mechanism (Definition 7) as a distribution over deterministic ordered subset allocation mechanisms.*

Proof. The proof follows from linearity of expectation. □

Randomized Rounding for Polymatroids. We describe $\textsc{RandRound}(\cdot)$ for general polymatroids. First we present a few definitions and give an overview of the rounding operator. Consider an arbitrary finite set U and a polymatroid $P(\mathcal{F})$ associated with a nondecreasing submodular function $\mathcal{F}: 2^U \to \mathbb{R}_+$ with $\mathcal{F}(\emptyset) = 0$. A set $S \subseteq U$ is called *tight* with respect to a $y \in P(\mathcal{F})$ if and only if $y(S) = \mathcal{F}(S)$. A set $\mathbb{S} = \{S^0, \dots, S^m\}$ of subsets of U is called a *nested family of tight sets* with respect to $y \in P(\mathcal{F})$ if and only if the elements of $\mathbb{S}$ can be ordered and indexed such that $\emptyset = S^0 \subset \cdots \subset S^m \subseteq U$, and such that S^r is tight with respect to y (for every $r \in [m]$).

$\textsc{RandRound}(y)$ takes an arbitrary $y \in P(\mathcal{F})$ and iteratively makes small changes to it until a vertex is reached. At each iteration ℓ it computes $y^\ell \in P(\mathcal{F})$ and a nested family of tight sets $\mathbb{S}^\ell$ (with respect to y^ℓ) such that

- $\mathbf{E}[y^\ell | y^{\ell-1}] = y^{\ell-1}$, and
- y^ℓ is closer to a vertex (compared with $y^{\ell-1}$) in the sense that either the number of nonzero coordinates has decreased by one or the number of tight sets has increased by one.

Observe that the above process must stop after at most $2|U|$ iterations (in fact we will show that it stops after at most $|U|$ iterations). At each iteration ℓ of the rounding process, a vector $\hat{y} \in \mathbb{R}^U$ and $\delta, \delta' \in \mathbb{R}_+$ are computed such that both $y^{\ell-1} + \delta \cdot \hat{y}$ and $y^{\ell-1} - \delta' \cdot \hat{y}$ are still in $P(\mathcal{F})$, but closer to a vertex. The algorithm then chooses a random $\delta'' \in \{\delta, -\delta'\}$ such that $\mathbf{E}[\delta''] = 0$, and sets $y^\ell \leftarrow y^{\ell-1} + \delta'' \cdot \hat{y}$.

Definition 8 ($\textsc{RandRound}(y)$). This operator takes as its input a point $y \in P(\mathcal{F})$ and returns as its output a pair (y^*, π^*), where y^* is a random vertex of $P(\mathcal{F})$ and π^* is its associated ordered subset (see Proposition 6) and such that $\mathbf{E}[y^*] = y$.

The algorithm modifies y iteratively until a vertex is reached. It also maintains a nested family of tight sets $\mathbb{S}$ with respect to y. As we modify $\mathbb{S}$, we always maintain an ordered labeling of its elements, that is, if $\mathbb{S} = \{S^0, \dots, S^m\}$, we assume that $\emptyset = S^0 \subset \cdots \subset S^m \subseteq U$; in particular, the indices are updated whenever a new tight set is added. For each $s \in U$, define $\mathbf{1}_s \in [0,1]^U$ as a vector whose value is 1 at coordinate s and 0 everywhere else.

1. Initialize $\mathbb{S} \leftarrow \{\emptyset\}$.
2. As long as any of the following steps is applicable, apply the step (in any order):
   - If there exist distinct $s, s' \in S^r \setminus S^{r-1}$ for some $r \in [m]$:

(a) Set $\hat{y} \leftarrow \mathbf{1}_s - \mathbf{1}_{s'}$, and compute $\delta, \delta' \in \mathbb{R}_+$ such that $y + \delta \cdot \hat{y}$ has a new tight set S and $y - \delta' \cdot \hat{y}$ has a new tight set S'; that is,

—set $S \leftarrow \underset{S^{r-1}+s \subseteq S \subseteq S^r - s'}{\arg\min} \mathcal{F}(S) - y(S)$, and $\delta \leftarrow \mathcal{F}(S) - y(S)$;
—set $S' \leftarrow \underset{S^{r-1}+s' \subseteq S' \subseteq S^r - s}{\arg\min} \mathcal{F}(S') - y(S')$, and $\delta' \leftarrow \mathcal{F}(S') - y(S')$.

(b) $\begin{cases} \text{with prob. } \frac{\delta}{\delta+\delta'}: & \text{set } y \leftarrow y + \delta \cdot \hat{y}, \text{and add } S \text{ to } \mathbb{S}. \\ \text{with prob. } \frac{\delta'}{\delta+\delta'}: & \text{set } y \leftarrow y - \delta' \cdot \hat{y}, \text{and add } S' \text{ to } \mathbb{S}. \end{cases}$

   - If there exists $s \in U \setminus S^m$ for which $y(s) > 0$:

(a) Set $\hat{y} \leftarrow \mathbf{1}_s$, and compute $\delta, \delta' \in \mathbb{R}_+$ such that $y + \delta \cdot \hat{y}$ has a new tight set S and $y - \delta' \cdot \hat{y}$ has a zero at coordinate s; that is,

—set $S \leftarrow \underset{S \supseteq S^m + s}{\arg\min} \mathcal{F}(S) - y(S)$, and $\delta \leftarrow \mathcal{F}(S) - y(S)$;
—set $\delta' \leftarrow y(s)$.

(b) $\begin{cases} \text{with prob. } \frac{\delta}{\delta+\delta'}: & \text{set } y \leftarrow y + \delta \cdot \hat{y}, \text{and add } S \text{ to } \mathbb{S}. \\ \text{with prob. } \frac{\delta'}{\delta+\delta'}: & \text{set } y \leftarrow y - \delta' \cdot \hat{y} \end{cases}$

3. Set $y^* \leftarrow y$ and define the ordered subset $\pi^*: S^m \to [m]$ according to $\mathbb{S}$; that is, for each $r \in [m]$ and $s \in S^r \setminus S^{r-1}$, define $\pi^*(s) = r$.
4. Return (y^*, π^*).

Theorem 6. *For any nondecreasing submodular function $\mathcal{F}: 2^U \to \mathbb{R}_+$ and any $y \in P(\mathcal{F})$, the operator* RandRound(y) *returns a random (y^*, π^*) such that $y^* \in$ Vertex$(P(\mathcal{F}))$, and π^* is the ordered subset corresponding to y^* (see Proposition 6), and such that $\mathbf{E}[y^*] = y$. Furthermore, the algorithm runs in strong polynomial time. In particular, it runs for $O(|U|)$ iterations, whereby each iteration involves solving two submodular minimizations.*

5.4. Matroid Feasibility Constraints

In this section, we consider environments where the feasibility constraints are encoded by a matroid $\mathcal{M} = (T_N, \mathcal{I})$. For every type profile $\mathbf{t} \in \mathbf{T}$, a subset $S \subseteq \{t_1, \ldots, t_n\}$ can be simultaneously allocated to if and only if $S \in \mathcal{I}$. We show that the results of subsection 5.3 can be generalized to environments with matroid feasibility constraints.

5.4.1. Matroid Preliminaries. A matroid $\mathcal{M} = (U, \mathcal{I})$ consists of a ground set U and a family of independent sets $\mathcal{I} \subseteq 2^U$ with the following two properties.

- For every I, I', if $I' \subset I \in \mathcal{I}$, then $I' \in \mathcal{I}$.
- For every $I, I' \in \mathcal{I}$, if $|I'| < |I|$, there exists $s \in I \setminus I'$ such that $I' \cup \{s\} \in \mathcal{I}$.

For every matroid $\mathcal{M}$, the rank function $r_{\mathcal{M}}: 2^U \to \mathbb{N} \cup \{0\}$ is defined as follows: for each $S \subseteq U$, $r_{\mathcal{M}}(S)$ is the size of the maximum independent subset of S. A matroid can be uniquely characterized by its rank function; that is, a set $I \subseteq U$ is an independent set if and only if $r_{\mathcal{M}}(I) = |I|$. A matroid rank function has the following two properties:

- $r_{\mathcal{M}}(\cdot)$ is a nonnegative nondecreasing integral submodular function.
- $r_{\mathcal{M}}(S) \le |S|$, for all $S \subseteq U$.

Furthermore, every function with the above properties defines a matroid.

Any set $S \subseteq U$ can be equivalently represented by its incidence vector $\chi^S \in \{0,1\}^U$, which has a 1 at every coordinate $s \in S$ and 0 everywhere else.

Proposition 8. *Consider an arbitrary finite matroid $\mathcal{M} = (U, \mathcal{I})$ with rank function $r_{\mathcal{M}}(\cdot)$. Let $P(r_{\mathcal{M}})$ denote the polymatroid associated with $r_{\mathcal{M}}(\cdot)$ (see Section 5.3); the vertices of $P(r_{\mathcal{M}})$ are exactly the incidence vectors of the independent sets of $\mathcal{M}$.*

See Schrijver [40] for a comprehensive treatment of matroids.

5.4.2. Characterization of Interim Feasibility. We now generalize the characterization of interim feasibility as the polymatroid given by the expected rank of the matroid. From this generalization the computational results of the preceding section can be extended from k-unit environments to matroids.

Let b denote the random bits used by an ex post allocation rule, and let $\hat{x}^{\mathbf{t},b} \in \{0,1\}^{T_N}$ denote the ex post allocation rule (i.e., the incidence vector of the subset of types that get allocated to) for type profile $\mathbf{t} \in \mathbf{T}$ and random bits b. By definition of polymatroidal feasibility constraint, $\hat{x}^{\mathbf{t},b}$ is a feasible ex post allocation if and only if it satisfies the following class of inequalities.

$$\hat{x}^{\mathbf{t},b}(S) \le r_{\mathcal{M}}(\mathbf{t} \cap S), \qquad \forall \mathbf{t} \in \mathbf{T}, \forall S \subseteq T_N. \tag{3}$$

The above inequality states that the subset of types that get allocated to must be an independent set of the restriction of matroid $\mathcal{M}$ to $\{t_1, \ldots, t_n\}$. The expectation of the left-hand side is exactly the normalized interim allocation rule; that is, for any $t_i' \in T_N$, $\overline{x}(t_i') = \mathbf{E}_{\mathbf{t},b}\left[\hat{x}^{\mathbf{t},b}(t_i')\right]$. Taking expectations of both sides of (3) then motivates the following definition and theorem that characterize interim feasibility.

Definition 9. The *expected rank* for distribution $\mathbf{f}$, subspace $S \subset T_N$, and matroid $\mathcal{M}$ with rank function $r_{\mathcal{M}}$ is

$$g_{\mathcal{M}}(S) = \mathbf{E}_{\mathbf{t}\sim\mathbf{f}}\left[r_{\mathcal{M}}(\mathbf{t} \cap S)\right], \tag{$g\mathcal{M}$}$$

where $\mathbf{t} \cap S$ denotes $\{t_1, \ldots, t_n\} \cap S$.

Theorem 7. *For matroid $\mathcal{M}$ and distribution $\mathbf{f}$, the space of all feasible normalized interim allocation rules, $\overline{\mathbb{X}}$, is the polymatroid associated with g_k, that is, $\overline{\mathbb{X}} = P(g_{\mathcal{M}})$, that is, for all $\overline{x} \in \overline{\mathbb{X}}$,*

$$\overline{x}(S) \le \mathbf{E}_{\mathbf{t}}\left[r_{\mathcal{M}}(\mathbf{t} \cap S)\right] = g_{\mathcal{M}}(S), \qquad \forall S \subseteq T_N. \tag{4}$$

Theorem 8. *For matroid $\mathcal{M}$, if $\overline{x} \in \overline{\mathbb{X}}$ is the vertex* VERTEX$(P(g_{\mathcal{M}}), \pi)$ *of the polymatroid $P(g_{\mathcal{M}})$ the unique ex post implementation is the ordered subset mechanism induced by π (Definition 6).*

To prove the above theorems, we use the following decomposition lemma, which applies to general polymatroids.

Lemma 5 (Polymatroidal Decomposition). *Let U be an arbitrary finite set, $\mathcal{F}^1, \ldots, \mathcal{F}^m : 2^U \to \mathbb{R}_+$ be arbitrary non-decreasing submodular functions, and $\mathcal{F}^* = \sum_{j=1}^m \lambda^j \mathcal{F}^j$ be an arbitrary convex combination of them. For every y^* the following holds: $y^* \in P(\mathcal{F}^*)$ if and only if it can be decomposed as $y^* = \sum_{j=1}^m \lambda^j y^j$ such that $y^j \in P(\mathcal{F}^j)$ (for each $j \in [m]$). Furthermore, if y^* is a vertex of $P(\mathcal{F}^*)$, this decomposition is unique. More precisely, if $y^* = \textsc{Vertex}(P(\mathcal{F}^*), \pi)$ for some ordered subset π, then $y^j = \textsc{Vertex}(P(\mathcal{F}^j), \pi)$ (for each $j \in [m]$).*

Proof. First, observe that the only-if part is true; that is, if $y^j \in P(\mathcal{F}^j)$ (for each $j \in [m]$), we can write

$$y^j(S) \le \mathcal{F}^j(S) \qquad \forall S \subseteq U, \tag{5}$$

multiplying both sides by λ^j and summing over all $j \in [m]$ we obtain

$$y^*(S) = \sum_{i=1}^m \lambda^j y^j(S) \le \sum_{j=1}^m \lambda^j \mathcal{F}^j(S) = \mathcal{F}^*(S) \qquad \forall S \subseteq U, \tag{6}$$

which implies that $y^* \in P(\mathcal{F}^*)$.

Next, we prove that for every $y^* \in P(\mathcal{F}^*)$ such a decomposition exists. Note that a polymatroid is a convex polytope, so any $y^* \in P(\mathcal{F}^*)$ can be written as a convex combination of vertices as $y^* = \sum_\ell \alpha^\ell y^{*\ell}$, where each $y^{*\ell}$ is a vertex of $P(\mathcal{F}^*)$; consequently, if we prove the claim for the vertices of $P(\mathcal{F}^*)$, that is, that $y^{*\ell} = \sum_{j=1}^m \lambda^j y^{\ell,j}$ for some $y^{\ell,j} \in P(\mathcal{F}^j)$, then a decomposition of $y^* = \sum_{j=1}^m \lambda^j y^j$ can be obtained by setting $y^j = \sum_\ell \alpha^\ell y^{\ell,j}$.

Next, we prove the second part of the theorem, which also implies that a decomposition exists for every vertex of $P(\mathcal{F}^*)$. Let $y^* = \textsc{Vertex}(P(\mathcal{F}^*), \pi)$ be an arbitrary vertex of $P(\mathcal{F}^*)$ with a corresponding ordered subset π; by Proposition 6, such a π exists for every vertex of a polymatroid. For every integer $r \le |\pi|$, define $S^r = \{\pi_1, \ldots, \pi_r\}$ as the r-element prefix of the ordering. By Proposition 6, inequality (6) is tight for every S^r, which implies that inequality (5) must also be tight for each S^r and for every $j \in [m]$. Consequently, for each $s \in \pi$ with r being the index for which $s = \pi_r$, and each $j \in [m]$, by taking the difference of the inequality (6) for S^r and S^{r-1}, given that they are tight, we obtain

$$y^j(s) = \mathcal{F}^j(S^r) - \mathcal{F}^j(S^{r-1}).$$

Furthermore, for each $s \notin \pi$, $y^*(s) = 0$, which implies that $y^j(s) = 0$ for every $j \in [m]$. Observe that we have obtained a unique y^j for each $j \in [m]$, which is exactly the vertex of $P(\mathcal{F}^j)$ corresponding to π as described in Proposition 6, and we have

$$\sum_{j=1}^m \lambda^j y^j(s) = \sum_{j=1}^m \lambda_j (\mathcal{F}^j(S^r) - \mathcal{F}^j(S^{r-1})) = y^*(S^r) - y^*(S^{r-1}) = y^*(s). \qquad \square$$

Proof of Theorem 7. The inequality in Equation (3) states that the subset of types that get allocated to must be an independent set of the restriction of matroid $\mathcal{M}$ to $\{t_1, \ldots, t_n\}$. Define $r^{\mathbf{t}}_{\mathcal{M}}(S) = r_{\mathcal{M}}(\mathbf{t} \cap S)$ (for all $(S \subseteq T_N)$. Notice that $r^{\mathbf{t}}_{\mathcal{M}}$ is a submodular function. The above inequality implies that $\hat{x}^{\mathbf{t},b} \in P(r^{\mathbf{t}}_{\mathcal{M}})$. Define $\hat{x}^{\mathbf{t}} = \mathbf{E}_b[\hat{x}^{\mathbf{t},b}]$. Observe that $\overline{x} = \mathbf{E}_{\mathbf{t}}[\hat{x}^{\mathbf{t}}]$, so $\overline{x}$ is a feasible normalized interim allocation rule if and only if it can be decomposed as $\overline{x} = \sum_{\mathbf{t} \in \mathbf{T}} f(\mathbf{t}) \hat{x}^{\mathbf{t}}$, where $\hat{x}^{\mathbf{t}} \in P(r^{\mathbf{t}}_{\mathcal{M}})$ for every $\mathbf{t} \in \mathbf{T}$; by Lemma 5, this is equivalent to $\overline{x} \in P(g_{\mathcal{M}})$, where $g_{\mathcal{M}}(S) = \sum_{\mathbf{t} \in \mathbf{T}} f(\mathbf{t}) r^{\mathbf{t}}_{\mathcal{M}}(S) = \mathbf{E}_{\mathbf{t}}[r^{\mathbf{t}}_{\mathcal{M}}(S)]$ (for all $S \subseteq T_N$), as defined in Definition 9. $\square$

Proof of Theorem 8. Suppose $\overline{x} = \textsc{Vertex}(P(g_{\mathcal{M}}), \pi)$ for some ordered subset π. By Lemma 5, the decomposition of $\overline{x}$ is unique and is given by $\hat{x}^{\mathbf{t}} = \textsc{Vertex}(P(r^{\mathbf{t}}_{\mathcal{M}}), \pi)$. Notice that this is the same allocation obtained by the deterministic rank-based allocation mechanism, which ranks according to π (see Definition 6). $\square$

5.4.3. Computationally Efficient Optimization and Implementation. As in Section 5.3, the characterization of interim feasibility as a polymatroid constraint immediately enables efficient solving of optimization problems over the feasible normalized interim allocation rules as long as we can compute $g_{\mathcal{M}}$ efficiently (see Schrijver [40] for optimization over polymatroids). Depending on the specific matroid, it might be possible to compute $g_{\mathcal{M}}$ exactly in polynomial time (e.g., as in Lemma 4); otherwise, an approximate expected rank function $\hat{g}_{\mathcal{M}}$ can be computed in polynomial time such that $\hat{g}_{\mathcal{M}}$ is a $[1-\delta, 1+\delta]$-approximation of $g_{\mathcal{M}}$ everywhere with high probability as formally stated in Theorem 9; we then compute the optimal interim allocation rules with respect

to the polymatroid of $(1-\delta)\hat{g}_{\mathcal{M}}$. Note that with high probability the polymatroid of $(1-\delta)\hat{g}_{\mathcal{M}}$ is contained in the polymatroid of $g_{\mathcal{M}}$; conversely, with high probability the polymatroid of $(1-2\delta)g_{\mathcal{M}}$ is contained in $(1-\delta)\hat{g}_{\mathcal{M}}$; therefore with high probability the optimal interim allocation rule with respect to $(1-\delta)\hat{g}_{\mathcal{M}}$ is feasible with respect to $g_{\mathcal{M}}$, and the optimal mechanism with respect to $(1-\delta)\hat{g}_{\mathcal{M}}$ is a $1-2\delta$ approximation of the optimal mechanism with respect to $g_{\mathcal{M}}$. Subsequent to our work, Gopalan et al. [22] showed that exact calculation of optimal auctions is computationally hard in this setting (we will revisit this discussion in Section 6).

Theorem 9. *Let $g_{\mathcal{M}}(S) = \mathbf{E}_{\mathbf{t}\sim\mathbf{f}}[r_{\mathcal{M}}(\mathbf{t}\cap S)]$ be an expected rank function, and let $D = \sum_i |T_i|$ be the total number of agent types. For any $\epsilon, \delta \in (0,1)$, an approximate expected rank function $\hat{g}_{\mathcal{M}}(S)$ can be constructed by using $O\left(\frac{n^4 D^3(\ln(1/\epsilon)+D)}{\delta^4}\right)$ randomly sampled type profiles such that, with probability $1-\epsilon$, $\hat{g}_{\mathcal{M}}$ is a $[1-\delta, 1+\delta]$-approximation of $g_{\mathcal{M}}$.*

Proof. The basic idea is to randomly sample a set of type profiles in advance and then use them to estimate $g_{\mathcal{M}}(S)$ on any given S by taking the average of $r_{\mathcal{M}}(\mathbf{t}\cap S)$ over all the samples. The problem with this approach is that the number of samples necessary to obtain the desired multiplicative approximation for $g_{\mathcal{M}}(S)$ on any given S is proportional to $1/g_{\mathcal{M}}(S)$, which could be arbitrarily large. Note that even though $r_{\mathcal{M}}(\mathbf{t}\cap S)$ is integral, $g_{\mathcal{M}}(S) = \mathbf{E}_{\mathbf{t}\sim\mathbf{f}}[r_{\mathcal{M}}(\mathbf{t}\cap S)]$ could still be arbitrarily small because the types in S could happen with arbitrarily low probability. We show that the problem can be avoided by a nonuniform sampling scheme in which we only sample type profiles with at most one rare type by fixing one agent and one rare type at a time and randomly sampling the nonrare types of the rest of the agents as formally described next.

A type $\tau \in T_i$ is called *rare* if and only if $\mathbf{Pr}[\mathbf{t}_i = \tau] \le \theta$ for some choice of θ to be specified later. Let $\mathcal{R}$ be the random variable denoting the number of rare types in $\mathbf{t}$. Let $\mathcal{R}_i$ be the indicator random variable for $\mathbf{t}_i$ being rare, and let $\mathcal{R}_{-i}$ be the random variable for the number of rare types in $\mathbf{t}_{-i}$. Furthermore, let $R_i \subseteq T_i$ denote the set of all rare types of agent i.

Given an event $\mathcal{E}$, let $g^{\mathcal{E}}_{\mathcal{M}}(S)$ denote the expected rank function conditioned on $\mathcal{E}$ defined as $g^{\mathcal{E}}_{\mathcal{M}}(S) = \mathbf{E}_{\mathbf{t}}[r_{\mathcal{M}}(\mathbf{t}\cap S)\ \mathbf{1}(\mathbf{t}\in\mathcal{E})]$.

Sampling Scheme. Consider the set of disjoint events

$$H = \left\{\{\mathcal{R}=0\}\right\} \cup \left\{\{\mathbf{t}_i = \tau, \mathcal{R}_{-i} = 0\}\right\}_{i\in[n],\tau\in R_i}.$$

For each $\mathcal{E}\in H$, let $\{\mathbf{t}^{\mathcal{E},j}\}_{j\in[N]}$ be N type profiles sampled independently at random with replacement conditioned on $\mathcal{E}$. We compute the estimated expected rank function as

$$\hat{g}_{\mathcal{M}}(S) = \sum_{\mathcal{E}\in H} \frac{1}{N}\sum_{j=1}^{N} r_{\mathcal{M}}(\mathbf{t}^{\mathcal{E},j}\cap S)\ \mathbf{Pr}[\mathcal{E}]. \tag{7}$$

Observe that $\mathbf{E}[\hat{g}_{\mathcal{M}}(S)] = \sum_{\mathcal{E}\in H} g^{\mathcal{E}}_{\mathcal{M}}(S) = g^{\mathcal{R}\le 1}_{\mathcal{M}}(S)$ because $\mathbf{E}[r_{\mathcal{M}}(\mathbf{t}^{\mathcal{E},j}\cap S)\,|\,\mathcal{E}]\mathbf{Pr}[\mathcal{E}] = g^{\mathcal{E}}_{\mathcal{M}}(S)$ and because H is a partition of the event $\{\mathcal{R}\le 1\}$. Therefore the following two steps are enough to complete the proof:

(I) We prove that $g^{\mathcal{R}\le 1}_{\mathcal{M}}$ is a $[1-\delta/2, 1]$-approximation of $g_{\mathcal{M}}$.

(II) We prove that, with probability at least $1-\epsilon$, $\hat{g}_{\mathcal{M}}$ is within $[1-\delta/2, 1+\delta/2]$ factor of its mean, which is $g^{\mathcal{R}\le 1}_{\mathcal{M}}$.

***Step* 1**. We prove that setting θ to $\delta/(2nD)$ is enough to guarantee $g^{\mathcal{R}\le 1}_{\mathcal{M}}$ to be a $[1-\delta/2, 1]$-approximation of $g_{\mathcal{M}}$. Let $\mu = \mathbf{Pr}[\mathcal{R}\ge 1]$ be the probability of at least one agent having a rare type.

First, we establish a lower bound on $g^{\mathcal{R}\le 1}_{\mathcal{M}}(S)$ in terms of $g(\cdot)$ and μ:

$$\begin{aligned}
g^{\mathcal{R}\le 1}_{\mathcal{M}}(S) &= \mathbf{E}_{\mathbf{t}}\left[r_{\mathcal{M}}(\mathbf{t}\cap S)\ \mathbf{1}(\mathcal{R}\le 1)\right] \\
&\ge \mathbf{E}_{\mathbf{t}}\left[\max_i r_{\mathcal{M}}(\mathbf{t}_i\cap S_i)\ \mathbf{1}(\mathcal{R}\le 1)\right] \\
&\ge \max_i \mathbf{E}_{\mathbf{t}}\left[r_{\mathcal{M}}(\mathbf{t}_i\cap S_i)\ \mathbf{1}(\mathcal{R}\le 1)\right] \\
&\ge \max_i\ \mathbf{E}_{\mathbf{t}}\left[r_{\mathcal{M}}(\mathbf{t}_i\cap S_i)\ \mathbf{1}(\mathcal{R}_{-i}=0)\right] \\
&= \max_i\left(\mathbf{E}_{\mathbf{t}_i}\left[r_{\mathcal{M}}(\mathbf{t}_i\cap S_i)\right]\ \mathbf{Pr}_{\mathbf{t}_{-i}}\left[\mathcal{R}_{-i}=0\right]\right) \qquad \text{by independence of } \mathbf{t}_i \text{ and } \mathcal{R}_{-i} \\
&\ge \max_i g_{\mathcal{M}}(S_i)\ (1-\mu).
\end{aligned} \tag{8}$$

Second, we establish an upper bound on $g_{\mathcal{M}}^{\mathcal{R}\geq 2}(S)$ in terms of $g(\cdot)$ and μ:

$$\begin{aligned}
g_{\mathcal{M}}^{\mathcal{R}\geq 2}(S) &= \mathbf{E}_{\mathbf{t}}\left[r_{\mathcal{M}}(\mathbf{t}\cap S)\ \mathbf{1}(\mathcal{R}\geq 2)\right] \\
&\leq \mathbf{E}_{\mathbf{t}}\left[\sum_i r_{\mathcal{M}}(\mathbf{t}_i\cap S_i)\ \mathbf{1}(\mathcal{R}\geq 2)\right] && \text{by submodularity of } r_{\mathcal{M}} \\
&= \sum_i \mathbf{E}_{\mathbf{t}}\left[r_{\mathcal{M}}(\mathbf{t}_i\cap S_i)\ \mathbf{1}(\mathcal{R}\geq 2)\right] \\
&\leq \sum_i \mathbf{E}_{\mathbf{t}}\left[r_{\mathcal{M}}(\mathbf{t}_i\cap S_i)\ \mathbf{1}(\mathcal{R}_{-i}\geq 1)\right] \\
&= \sum_i \mathbf{E}_{\mathbf{t}_i}\left[r_{\mathcal{M}}(\mathbf{t}_i\cap S_i)\right]\ \mathbf{Pr}_{\mathbf{t}_{-i}}\left[\mathcal{R}_{-i}\geq 1\right] && \text{by independence of } \mathbf{t}_i \text{ and } \mathcal{R}_{-i} \\
&\leq n\max_i g_{\mathcal{M}}(S_i)\ \mu. && (9)
\end{aligned}$$

By combining (8) and (9) we get

$$g_{\mathcal{M}}^{\mathcal{R}\geq 2}(S)\leq \frac{n\mu}{1-\mu}\ g_{\mathcal{M}}^{\mathcal{R}\leq 1}(S),$$

which together with the fact that $g_{\mathcal{M}}(S)=g_{\mathcal{M}}^{\mathcal{R}\leq 1}(S)+g_{\mathcal{M}}^{\mathcal{R}\geq 2}(S)$ implies

$$g_{\mathcal{M}}^{\mathcal{R}\leq 1}(S)\geq \frac{1-\mu}{1+(n-1)\mu}g_{\mathcal{M}}(S).$$

To obtain the desired approximation factor we need to satisfy $\frac{1-\mu}{1+(n-1)\mu}\geq 1-\delta/2$, for which it is enough to ensure $\mu\leq\delta/(2n)$. Given that $\mu\leq D\theta$, where $D=\sum_i|T_i|$ is the total number of agent types, it is enough to choose θ such that

$$\theta\leq\delta/(2nD). \tag{10}$$

***Step* 2.** We derive a lower bound on N which guarantees, with probability at least $1-\epsilon$, $\hat{g}_{\mathcal{M}}(S)$ is within $[1-\delta/2,1+\delta/2]$ factor of its mean, which is $g_{\mathcal{M}}^{\mathcal{R}\leq 1}(S)$, for all S. Note that there are 2^D choices of S, so by applying union bound it is enough to show that the approximation holds for each S with probability at least $1-\epsilon/2^D$.

First, we establish an upper bound on the probability of $\hat{g}_{\mathcal{M}}(S)$ not being within $[1-\delta/2,1+\delta/2]$ factor of its means $g_{\mathcal{M}}^{\mathcal{R}\leq 1}(S)$, which we formally denote by $\eta(S)=\mathbf{Pr}[|\hat{g}_{\mathcal{M}}(S)-g_{\mathcal{M}}^{\mathcal{R}\leq 1}(S)|>\frac{\delta}{2}g_{\mathcal{M}}^{\mathcal{R}\leq 1}(S)]$. According to (7), $\hat{g}_{\mathcal{M}}$ is defined as a sum in which each term $\frac{1}{N}r_{\mathcal{M}}(\mathbf{t}^{\mathcal{E},j}\cap S)\ \mathbf{Pr}[\mathcal{E}]$ is either an independent random variable in the range $[0,n\ \mathbf{Pr}[\mathcal{E}]/N]$ or is trivially always 0 for the given combination of S and $\mathcal{E}$. Let $H(S)$ be the subset of events in H for which $r_{\mathcal{M}}(\mathbf{t}^{\mathcal{E},j}\cap S)$ is nonzero with a nonzero probability.

Applying Hoeffding's theorem yields the following bound on $\eta(S)$:

$$\begin{aligned}
\eta(S) &\leq 2\exp\left(\frac{-2\left(\frac{\delta}{2}\sum_{\mathcal{E}\in H(S)}\frac{1}{N}\sum_{j=1}^{N}g_{\mathcal{M}}^{\mathcal{E}}(S)\right)^2}{\sum_{\mathcal{E}\in H(S)}\sum_{j=1}^{N}(n\mathbf{Pr}[\mathcal{E}]/N)^2}\right) \\
&= 2\exp\left(\frac{-N\delta^2\left(\sum_{\mathcal{E}\in H(S)}g_{\mathcal{M}}^{\mathcal{E}}(S)\right)^2}{2n^2\sum_{\mathcal{E}\in H(S)}\mathbf{Pr}[\mathcal{E}]^2}\right) \\
&\leq 2\exp\left(\frac{-N\delta^2\left(\sum_{\mathcal{E}\in H(S)}\theta\mathbf{Pr}[\mathcal{E}]\right)^2}{2n^2\sum_{\mathcal{E}\in H(S)}\mathbf{Pr}[\mathcal{E}]^2}\right) && \text{by } g_{\mathcal{M}}^{\mathcal{E}}(S)\geq\theta\,\mathbf{Pr}[\mathcal{E}] \text{ to be proven} \\
&\leq 2\exp\left(\frac{-N\delta^2\theta^2}{2n^2}\right).
\end{aligned}$$

For the second to last inequality to hold we prove $g_{\mathcal{M}}^{\mathcal{E}}(S)=\mathbf{E}[r_{\mathcal{M}}(\mathbf{t}\cap S)\,|\,\mathcal{E}]\mathbf{Pr}[\mathcal{E}]\geq\theta\mathbf{Pr}[\mathcal{E}]$ as follows: given that $\mathcal{E}$ is in $H(S)$, there must exist a type $\tau\in S_i$ for some agent i with $r_{\mathcal{M}}(\{\tau\})=1$ such that either τ is a nonrare type

occurring with probability at least θ under $\mathscr{E}$, or τ is a rare type occurring with probability 1 under $\mathscr{E}$, with $\mathscr{E}$ being the event $\{\mathbf{t}_i = \tau, \mathcal{R}_{-i} = 0\}$; in either case it follows that $\mathbf{E}[r_{\mathcal{M}}(\mathbf{t} \cap S) \mid \mathscr{E}] \geq \mathbf{E}[r_{\mathcal{M}}(\mathbf{t} \cap \{\tau\}) \mid \mathscr{E}] = \mathbf{Pr}[\mathbf{t}_i = \tau \mid \mathscr{E}] \geq \theta$, which proves $g^{\mathscr{E}}_{\mathcal{M}}(S) \geq \theta \, \mathbf{Pr}[\mathscr{E}]$.

Next, to guarantee the desired bound on the overall probability of error it is enough to choose N such that $\eta(S) \leq 2\exp\left(\frac{-N\delta^2\theta^2}{2n^2}\right) \leq \epsilon/2^D$, which implies

$$\begin{aligned} N &\geq \frac{2n^2(\ln(1/\epsilon) + (D+1)\ \ln 2)}{\delta^2\theta^2} \\ &= \frac{8n^4D^2(\ln(1/\epsilon) + (D+1)\ \ln 2)}{\delta^4} \qquad \text{by substituting } \theta \text{ from (10).} \end{aligned} \tag{11}$$

The total number of samples is at most DN, which completes the proof. □

Ex Post Implementation of Feasible Interim Allocation Rules. An ex post implementation for any $\overline{x} \in \overline{\mathbb{X}}$ can be obtained exactly as in Section 5.3.

Corollary 3. *Any normalized interim allocation rule $\overline{x} \in \overline{\mathbb{X}}$ can be implemented by the randomized rank-based allocation mechanism (Definition 7) as a distribution over deterministic rank-based allocation mechanisms (Definition 6).*

6. Conclusions and Extensions

In this paper, we have focused on binary allocation problems in which an agent is either served or not served. For these binary allocation problems distributions over allocations are given by a single number (i.e., the probability that the agent is served). Our results can be extended to environments with multiunit demand when the agent's utility is linear in the expected number of units the agent receives.

In Section 5, we described algorithms for optimizing over feasible interim allocation rules and for (ex post) implementation of the resulting rules. Neither these algorithms nor the generalization of Border's condition require the types of the agents to be independently distributed. However, our formulation of incentive compatibility for interim allocation rules does require independence. For correlated distributions the interim allocation rule is a function of the actual type of the agent (which conditions the types of the other agents) and the reported type of the agent. Therefore, this generalization of our theorem to correlated environments has little relevance for mechanism design.

The algorithms in Section 5 do not require the feasibility constraint to be known in advance. A simple example in which this generalization is interesting is a multiunit auction in which the supply k is stochastically drawn from a known distribution. Our result shows that the optimal auction in such an environment can be described by picking the random ordering on types and allocating greedily by this ordering while supplies last. We do not know of many examples other than this in which this generalization is interesting.

Our techniques can also be used in conjunction with the approach of Cai et al. [12] for solving multi-item auction problems for agents with additive values.

In symmetric environments, that is, when the agents' type distribution and the designer's feasibility constraint are symmetric (e.g., for i.i.d. multiunit auctions), the optimal interim allocation constraint imposed by feasibility is symmetric; furthermore, the constraint is given by a simple formula. Therefore, symmetric multiagent problems reduce to solving the single-agent problem for a very specific constraint on the interim allocation rule. In comparison with the computational task of optimizing over feasible interim allocation rules and solving for ex post implementations (e.g., from Section 5), this multiagent reduction for symmetric environments is computationally trivial.

Our characterization of interim feasibility allowed us to efficiently compute optimal auctions with k items, and approximately optimal auctions more generally with matroid feasibility. The challenge with the matroid feasibility setting is to calculate the expected rank function (Theorem 9). Subsequent work of Gopalan et al. [22] showed, among other results, that the optimal auction problem with matroid feasibility is #P hard (the hardness is shown for graphical matroids).

Acknowledgments

The authors thank anonymous referees and the associate editor for useful suggestions.

Appendix A. Large and Continuous Type Spaces

This section contains approximate reductions when the type space is infinite or large but has a succinct representation. If we relax our objective of finding an optimal mechanism to finding a $(1+\epsilon)$-approximation for any given ϵ (i.e., a polynomial

time approximation scheme, a.k.a., a PTAS), then we can significantly reduce the complexity of the interface between the multi- and single-agent problems in the reduction. Although the interface via the type space described in the preceding sections gave a direct approach to the reduction, the interface we present in this section is more intuitive.

Recall from Definition 2 that for allocation rule x the normalized interim allocation rule specifies the probability that an agent with a specific type is allocated, that is, $\bar{x}(t) = f(t)x(t)$. Here the probability is taken over the type of the agent and randomization in the allocation rule. It is useful to view this normalized interim probability as the area of a rectangle with width $f(t)$ and height $x(t)$. Imagine arranging these rectangles by decreasing height.

The heights and widths in this decreasing order of height induce a monotone decreasing function on the interval $[0,1]$ (note: the widths correspond to probabilities that sum to one). We refer to the horizontal axis as *quantile* $q \in [0,1]$ and the function as the *quantile allocation rule* $x(q)$ for all q (the use of x will be clear from the context, because we denote the allocation rule $x(t)$ and the quantile allocation rule $x(q)$). The allocation rule can be interpreted as inducing an ordering on agents whereby stronger agents are more likely to be allocated and weaker agents are less likely to be allocated. The quantile of a type is then the probability that a random type from the distribution is stronger. For discrete distributions and type spaces this normalized allocation rule is piece-wise constant; it can be similarly defined for continuous distributions and type spaces.

We will refer to the integral of the quantile allocation rule as the cumulative allocation rule, notated X. Because the quantile allocation rule is monotone decreasing the cumulative allocation rule is concave.

Although we considered the single-agent problem with an allocation constraint in type-space, we could equivalently have expressed it in quantile space. Recall definitions $\bar{x}(S) = \sum_{t\in S} \bar{x}(t)$ and $f(S) = \sum_{t\in S} f(t)$. By monotonicity of x we must have

$$\forall S \subset T,\ \bar{x}(S) \le X(f(S)).$$

The composition a concave function X with the linear set function $f(\cdot)$ is submodular, therefore, the feasibility constraint is polymatroid. This fact is useful for solving single-agent problems; however, we will not be doing so here.

Intuitively now the goal for reducing multiagent problems to single-agent problems is to optimize the sum of the single-agent revenues as a function of feasible normalized interim allocation constraints. For discrete type spaces the normalized interim allocation constraints are piece-wise constant, with a number of pieces equal to the number of types. The main task to be addressed in this section is to show that this interface between the multi- and single-agent problems can be approximated with piece-wise constant normalized interim allocation constraints with a very small number of pieces. We will give $(1+\epsilon)$ approximations with $O(\frac{1}{\epsilon}\log\frac{n}{\epsilon})$ pieces. These can be plugged into the theorems of the preceding section to give effective type space T_N of size $O(\frac{n}{\epsilon}\log\frac{n}{\epsilon})$.

Our approach is in two steps. First, given any feasible quantile allocation constraint, there is a feasible quantile allocation constraint that for all agents i is constant on $[0,\frac{\epsilon}{n}]$ and is a $(1+\epsilon)$-approximation to the original revenue for each agent. Second, given any feasible quantile allocation constraint that is constant on $[0,\frac{\epsilon}{n}]$ for each agent, there is a feasible quantile allocation constraint that is piece-wise constant with $O(\frac{1}{\epsilon}\log\frac{n}{\epsilon})$ pieces with widths of the form $\frac{\epsilon}{n}(1+\epsilon)^j$ for integers j and is a $(1+\epsilon)$-approximation to the original revenue for each agent. This construction shows that the optimal mechanism constrained to piece-wise constant normalized interim allocation rules (of the above form) is within an $(1+\epsilon)^2$-approximation of the optimal unconstrained mechanism. Of course, $(1+\epsilon)^2 \approx (1+2\epsilon)$, so a change of variables yields the desired approximation.

Lemma A.1. *Given any feasible quantile allocation constraint x, there is a feasible quantile allocation constraint x' that for all agents i is constant on $[0,\frac{\epsilon}{n}]$ and is a $(1+\epsilon)$-approximation to the original revenue for each agent.*

Proof. Let $\delta = \frac{\epsilon}{n} \ll 1$. We make the following transformation to each x_i. First define $x_i(q) = 1$ for $q \in [-\delta, 0]$. Now define x_i'' as the right-shift of x_i by δ, that is, $x_i''(q) = x_i(q-\delta)$. Notice that x_i'' is a more permissive constraint than x_i, and therefore the optimal single-agent revenue is only (weakly) improved. Now we define x_i' as x_i'' scaled down by a factor of $(1+\epsilon)$, that is

$$x_i'(q) = \frac{x_i''(q)}{1+\epsilon}.$$

This can be viewed as a convex combination of the allocation rule with one that never allocates, and therefore the revenue loss from this transformation is exactly a $(1+\epsilon)$-factor.

We will show that feasibility of x implies feasibility of x'. Feasibility of x and Theorem 7 state that

$$\forall \mathbf{q} \in [0,1]^n,\ \textstyle\sum_i X_i(q_i) \le g(\mathbf{q}). \tag{A.1}$$

We would like to show that

$$\forall \mathbf{q} \in [0,1]^n,\ \textstyle\sum_i X_i'(q_i) \le g(\mathbf{q}).$$

Note that by definition for all $\mathbf{q} \in [\delta, 1]^n$,

$$\begin{aligned}\textstyle\sum_i X_i'(q_i) &= \frac{1}{1+\epsilon}\textstyle\sum_i X_i''(q_i)\\ &= \frac{1}{1+\epsilon}\textstyle\sum_i \delta + X_i(q_i-\delta)\\ &= \frac{1}{1+\epsilon}(n\delta + \textstyle\sum_i X_i(q_i-\delta))\\ &\le \frac{1}{1+\epsilon}(n\delta + g(\mathbf{q}-\delta)),\end{aligned}$$

where the last inequality followed by feasibility of X_i (A.1), and short-hand notation $\mathbf{q}-\delta$ denotes the vector with coordinates given by $q_i - \delta$, for all i. It is therefore sufficient to show that

$$\frac{1}{1+\epsilon}(n\delta + g(\mathbf{q}-\delta)) \le g(\mathbf{q}). \tag{A.2}$$

We will next show that inequality (A.2) is satisfied when $\epsilon = \delta n$. Notice that this condition depends only on $g(\cdot)$ and not on either allocation rule. We are left only to show that this inequality holds for the ϵ and δ satisfying $\epsilon = \delta n$ as specified in the statement of the lemma.

Observe that $g(\mathbf{q}_{-i}, q_i)$ for any fixed $\mathbf{q}_{-i}$ is linear in q_i. In fact, this value of g is exactly $g(\mathbf{q}_{-i}, 0)$ plus q_i times the probability that i is independent from the set of agents j with quantile exceeding their bound q_j; however, this specific formula will not be important for the proof of the lemma. This linearity implies that both sides of (A.2) are linear and therefore the difference between the left- and right-hand sides with $\mathbf{q} \in [\delta, 1]^n$ is minimized at $\mathbf{q} \in \{\delta, 1\}^n$. In terms of δ, we will get bounds on ϵ that are sufficient for this minimum to be nonnegative, which is sufficient for implementability via (A.2). Rearranging (A.2), we need to show that

$$\forall \mathbf{q} \in \{\delta, 1\}^n,\ \epsilon \ge \frac{n\delta + g(\mathbf{q}-\delta)}{g(\mathbf{q})} - 1. \tag{A.3}$$

We break this into two cases, the first in which there exists an i with $q_i = 1$ and the second in which $q_i = \delta$ for all i. In the first case, note that $q_i = 1$ for some i implies that $g(\mathbf{q}) \ge 1$. In addition, monotonicity of g implies that $g(\mathbf{q}-\delta)/g(\mathbf{q}) \le 1$. As a result,

$$\delta n \ge \frac{n\delta + g(\mathbf{q}-\delta)}{g(\mathbf{q})} - 1.$$

Therefore, by setting $\epsilon = \delta n$, the inequality (A.3) is satisfied.

Now consider the second case, in which $\mathbf{q} = \delta$ is the all-δ vector. First note that $g(\mathbf{q}-\delta) = 0$. Note also that the expected rank of the number of types that show up is lower bounded by the probability any type shows up, that is, $g(\delta) \ge 1-(1-\delta)^n$ (for the $k = 1$ unit auction environment, this bound on g is in fact the definition of g). Now we have

$$g(\delta) \ge 1-(1-\delta)^n \ge 1-e^{-n\delta} \ge 1-\frac{1}{n\delta+1} = \frac{n\delta}{n\delta+1},$$

where second inequality follows from $1-\delta \le e^{-\delta}$ and the third inequality follows from $\log(1+z) \le z$. Combining the above inequality with the fact that $g(\mathbf{q}-\delta) = 0$ implies that the right-hand side of inequality (A.3) is at most $\delta n = \epsilon$. □

Lemma A.2. *Given any feasible normalized interim allocation constraint x which is constant for all agent i on $q_i \in [0, \delta]$, there is a feasible normalized interim allocation constraint x' which is piece-wise constant for all agent i on intervals of the form $[\delta(1+\epsilon)^j, \delta(1+\epsilon)^{j+1}]$ for all integer j and is a $(1+\epsilon)$-approximation to the original revenue for each agent.*

Proof. Define the cumulative allocation constraint X' as the piece-wise linear function that interpolates between the points $(q, X(q))$ for $q = (1+\epsilon)^j$ for integer j or zero. The optimal revenue given cumulative allocation constraint $X/(1+\epsilon)$ is exactly a $(1+\epsilon)$ factor from that of X; we will show that this rule is more restrictive than X', therefore the optimal revenue from constraint X' is at least a $(1+\epsilon)$-approximation.

We will show that for all q, $X_i'(q) \ge X_i(q)/(1+\epsilon)$. Let $a = \delta(1+\epsilon)^j$ and $b = a(1+\epsilon)$ and consider the interval $[a, b]$. We show that the maximum of $X_i(q)/X_i'(q)$ over this interval is at most $1+\epsilon$. Let $\gamma = X_i(a)/a$ be the slope of the line from the origin through point $(a, X_i(a))$. Concavity of $X_i(\cdot)$ implies that the line γq upperbounds $X_i(q)$ on $[a, b]$. Therefore, an upper bound on $X_i(q)$ for q in this interval is γb. Because X_i' is monotone and $X_i'(a) = X_i(a)$, a lower bound on $X_i'(q)$ is $X_i(a) = \gamma a$. Therefore the maximum ratio of the former to the latter $X_i(q)/X_i'(q)$ over this interval is at most $\gamma b/(\gamma a) = (1+\epsilon)$. This argument addresses all intervals except for $[0, \delta]$; of course, $X_i(q) = X_i'(q)$ on this interval, and so the same bound holds. □

Appendix B. Examples and Proofs from Section 5.2

SSA Examples

Consider a setting with a single item and two agents, with types $\{H_1, L_1\}$ for agent 1 and $\{H_2, L_2\}$ for agent 2, where for each agent, each type happens with probability 1/2. Consider the ex post allocation $\hat{\mathbf{x}}^1(\cdot,\cdot)$ presented in Table B.1.

Table B.1. SSA can implement ex post allocation $\hat{\mathbf{x}}^1$.

	H_2	L_2
H_1	$(1,0)$	$(1,0)$
L_1	$(0,1)$	$(0,0)$

SSA can implement $\hat{\mathbf{x}}^1$ with transition table π presented in Figure B.1. The mechanism visits agent 1 first. The token is passed to agent 1 if and only if the type is H_1. The mechanism then visits agent 2. The token is passed to agent 2 if and only if its type is H_2 and the current holder of the token is the dummy type.

Now consider the post allocation $\hat{\mathbf{x}}^2$ presented in Table B.2. We show that this allocation rule cannot be implemented by any stochastic sequential allocation mechanism. Consider two cases. If the mechanism visits agent 1 first, then as shown in the top of Figure B.2, $\hat{\mathbf{x}}^2(H_1,H_2)=(1,0)$ requires that type H_1 must take the token from the dummy type; but then if the type of agent 2 is L_2, because the token must either remain with H_1 or be passed to L_2 (or a randomization over these two decisions), the requirement that $\hat{\mathbf{x}}^2(H_1,L_2)=(0,0)$ would be violated. Similarly, consider the other case, in which the mechanism visits agent 2 first. As presented in the bottom of Figure B.2, the requirement that $\hat{\mathbf{x}}^2(L_1,L_2)=(0,1)$ requires that L_2 must take the token from the dummy type, but then $\hat{\mathbf{x}}^2(H_1,L_2)=(0,0)$ would be violated because the token should either remain with L_2 or be passed to H_1.

Note that in this example, $x(H_1)=1/2$, $x(L_1)=0$, $x(H_2)=0$, and $x(L_2)=1/2$, which can be implemented via SSA presented in Figure B.3. This mechanism visits agent 1 first. If agent 1 has type H_1, the token will be passed to him with probability 1/2. Otherwise, the token will stay with dummy. The mechanism then visits agent 2. If the current holder of the token is dummy type (which happens with probability 1/2) and the type of agent 2 is L_2, then he will take the token from the dummy.

A Network Flow Formulation of $\mathbb{S}$

We first describe another interpretation of $\mathbb{S}$ as a type of network flow with dynamic edge capacities, which will be used to prove Lemma 1 and Lemma 2.

We construct a network with dynamic capacities in which every feasible flow corresponds to some $(y,z)\in\mathbb{S}$. The network (see Figure B.4) has a source node $\langle\textsc{Src}\rangle$, a sink node $\langle\textsc{Snk}\rangle$, and $n-i+1$ nodes for every $t_i\in T_N$ labeled as $\langle t_i,i\rangle,\cdots,\langle t_i,n\rangle$, where each node $\langle t_{i'},i\rangle$ corresponds to the type $t_{i'}$ at the time SSA algorithm is visiting agent i.

For each $t_{i'}\in T_N$ and for each $i\in\{i',\dots,n-1\}$ there is an edge $(\langle t_{i'},i\rangle,\langle t_{i'},i+1\rangle)$ with infinite capacity whose flow is equal to $y(t_{i'},i)$; we refer to these edges as *horizontal edges*. The flow on horizontal edge $(\langle t_{i'},i\rangle,\langle t_{i'},i+1\rangle)$ represent the probability that $t_{i'}$ has the token at the beginning of time i and the token is not passed to t_i by the end of time i, which was denoted by $y(t_{i'},i)$. For every $t_{i'}$ and every t_i where $i'<i$, there is an edge $(\langle t_{i'},i\rangle,\langle t_i,i\rangle)$ whose flow is equal to $z(t_{i'},t_i)$ and whose capacity is equal to the total amount of flow that enters $\langle t_{i'},i\rangle$ multiplied by $f_i(t_i)$, that is, it has a dynamic capacity which is equal to $y(t_{i'},i-1)\,f_i(t_i)$; we refer to these edges as *diagonal edges*. The flow on these edges represents the probability that $t_{i'}$ has the token at the beginning of time i and will pass it to t_i by the end of time i. The dynamic capacity constraint on each diagonal edge $(\langle t_{i'},i\rangle,\langle t_i,i\rangle)$ ensures that (π) also holds. Moreover, because $(y,z)\in\mathbb{S}$, $(\mathbb{S}.2)$ and $(\mathbb{S}.3)$ holds if and only if we have the conservation of flow.

There is an edge $(\langle\textsc{Src}\rangle,t_0)$ through which the source node pushes exactly one unit of flow. Finally, for every $t_i\in T_N$, there is an edge $(\langle t_i,n\rangle,\langle\textsc{Snk}\rangle)$ with unlimited capacity whose flow is equal to $y(t_i,n)$. Note that a flow (y,z) is feasible if and only if it satisfies both conservation of flow and also the capacity constraints, which is equivalent to $(y,z)\in\mathbb{S}$. To simplify the proofs we sometimes use $\langle t_0,0\rangle$ as an alias for the source node $\langle\textsc{Src}\rangle$ and $\langle t_i,n+1\rangle$ as aliases for the sink node $\textsc{Snk}$. The network always has a feasible flow because all the flow can be routed along the path $\langle\textsc{Src}\rangle,\langle t_0,1\rangle,\dots,\langle t_0,n\rangle,\langle\textsc{Snk}\rangle$.

We define the *residual capacity* between two types $t_{i'},t_i\in T_N$ with respect to a given $(y,z)\in\mathbb{S}$ as follows.

$$\textsc{ResCap}_{y,z}(t_{i'},t_i)=\begin{cases}y(t_{i'},i-1)f_i(t_i)-z(t_{i'},t_i) & i>i'\\ z(t_i,t_{i'}) & i<i'\\ 0 & \text{otherwise.}\end{cases}\qquad(\textsc{ResCap})$$

Because of dynamic capacity constraints, it is not possible to augment a flow along a path with positive residual capacity by simply changing the amount of the flow along the edges of the path, because reducing the total flow entering a node also decreases the capacity of the diagonal edges leaving that node, which could potentially violate their capacity constraints. Therefore, we introduce an operator $\textsc{Reroute}(t_{i'},t_i,\rho)$ (Algorithm B.1 and Figure B.5), which modifies an

Figure B.1. SSA mechanism that implements $\hat{\mathbf{x}}^1$.

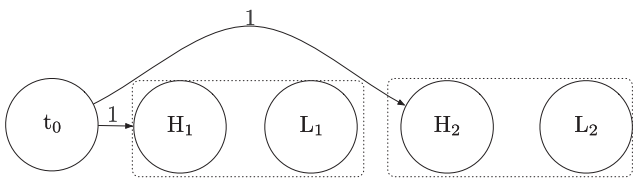


Table B.2. SSA cannot implement ex post allocation $\hat{\mathbf{x}}^2$.

	H_2	L_2
H_1	$(1,0)$	$(0,0)$
L_1	$(0,1)$	$(0,0)$

existing $(y,z) \in \mathbb{S}$, while maintaining its feasibility, to transfer a ρ-fraction of $y(t_i,n)$ to $y(t_{i'},n)$. The algorithm $\textsc{Reroute}(t_{i'},t_i,\rho)$ takes out ρ-fraction of the flow on the horizontal edge to $\langle t_{i'},\max(i',i)\rangle$ and reroutes this flow through $\langle t_i,\max(i',i)\rangle$, while keeping the flow feasible.

Without loss of generality assume $i' < i$. Algorithm $\textsc{Reroute}(t_{i'},t_i,\rho)$ keeps only $(1-\rho)$ fraction of the flow on each edge of the subtree consisting of the path $\langle t_{i'},i\rangle,\dots,\langle t_{i'},n\rangle,\langle \textsc{Snk}\rangle$ and all the diagonal edges leaving this path, and reassigns the remaining ρ fraction of the flow to the subtree rooted at $\langle t_i,i\rangle$. The reassignment first sends the subtracted ρ fraction of incoming flow via the horizontal edge of $\langle t_{i'},i\rangle$ toward $\langle t_i,i\rangle$ via the diagonal edge. Then, each $\langle t_i,t\rangle$ redistributes this extra fraction of flow on the diagonal edge toward $\langle t_{i''},t\rangle$, and the amount of this additional flow is exactly equal to the subtracted amount on the diagonal edge from $\langle t_{i'},t\rangle$ toward $\langle t_{i''},t\rangle$ (see Figure B.5).

Algorithm B.1 ($\textsc{Reroute}(t_{i'},t_i,\rho)$)

Input: An existing $(y,z) \in \mathbb{S}$ given implicitly, a source type $t_{i'} \in T_N$, a destination type $t_i \in T_N$ where $i' \neq i$, and a fraction $\rho \in [0,1]$.

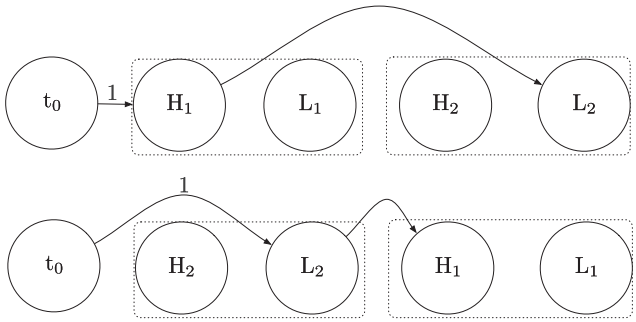
Output: Modify (y,z) to transfer a ρ-fraction of $y(t_{i'},n)$ to $y(t_i,n)$ while ensuring that the modified assignment is still in $\mathbb{S}$.

1: **if** $i' < i$ **then**
2: Increase $z(t_{i'},t_i)$ by $\rho \cdot y(t_{i'},i)$.
3: **else**
4: Decrease $z(t_i,t_{i'})$ by $\rho \cdot y(t_{i'},i')$.
5: **end if**
6: **for** $i'' = \max(i',i)$ to n **do**
7: Increase $y(t_i,i'')$ by $\rho \cdot y(t_{i'},i'')$.
8: Decrease $y(t_{i'},i'')$ by $\rho \cdot y(t_{i'},i'')$.
9: **end for**
10: **for** $t_{i''} \in T_{\{\max(i',i)+1,\dots,n\}}$ **do**
11: Increase $z(t_i,t_{i''})$ by $\rho \cdot z(t_{i'},t_{i''})$.
12: Decrease $z(t_{i'},t_{i''})$ by $\rho \cdot z(t_{i'},t_{i''})$.
13: **end for**

We next show that by applying $\textsc{Reroute}(t_{i'},t_i,\rho)$ to a feasible flow $(y,z) \in \mathbb{S}$, the modified flow is still feasible. Without loss of generality assume $i < i'$ and let (Y,Z) denote the obtained flow from applying $\textsc{Reroute}(t_{i'},t_i,\rho)$ on (y,z). Following the steps of the algorithm, we obtain

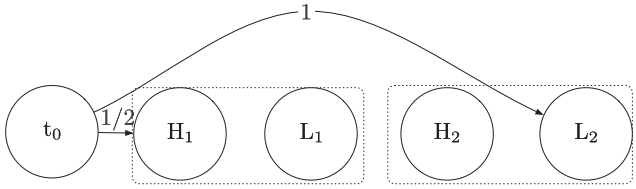
$$\begin{aligned}
Z(t_{i'},t_i) &= \rho y(t_{i'},i) + z(t_{i'},t_i),\\
Y(t_{i'},i'') &= (1-\rho)y(t_{i'},i''), \text{ for } i'' \geq i,\\
Z(t_{i'},t_{i''}) &= (1-\rho)z(t_{i'},t_{i''}), \text{ for } i'' > i,\\
Y(t_i,i'') &= y(t_i,i'') + \rho y(t_{i'},i''), \text{ for } i'' \geq i,\\
Z(t_i,t_{i''}) &= z(t_i,t_{i''}) + \rho z(t_{i'},t_{i''}), \text{ for } i'' > i.
\end{aligned}$$

Figure B.2. SSA cannot implement $\hat{\mathbf{x}}^2$.



Note. Either agent 1 goes first (top), which implies that with probability 1 the token moves from t_0 to H_1, contradicting $\hat{\mathbf{x}}^2(H_1,L_2) = (0,0)$; or agent 2 goes first (bottom), which implies that with probability 1 the token should be passed to L_2 by dummy, contradicting $\hat{\mathbf{x}}^2(H_1,L_2) = (0,0)$.

Figure B.3. Even though SSA cannot implement $\hat{\mathbf{x}}^2$, SSA can implement the corresponding interim allocation.



We next show that $(Y, Z) \in \mathbb{S}$. We first verify ($\mathbb{S}$.2). For any $j < i$, the incoming and outgoing edges from $\langle t_j, j\rangle$ stay unchanged. For $j > i$ we have

$$\begin{aligned} Y(t_j, j) = y(t_j, j) &= \sum_{t_{j'} \in \{0,\cdots,j-1\}} z(t_{j'}, t_j), \\ &= \sum_{t_{j'} \in \{0,\cdots,j-1\}, j' \neq \{i,i'\}} Z(t_{j'}, t_j) + z(t_{i'}, t_j) + z(t_i, t_j) = \sum_{t_{j'} \in \{0,\cdots,j-1\}} Z(t_{j'}, t_j). \end{aligned}$$

Similarly for i (or i') we have

$$Y(t_i, i) = y(t_i, i) + \rho y(t_{j'}, i) = \sum_{t_{j'} \in \{0,\cdots,i-1\}} z(t_{j'}, t_i) + \rho y(t_{j'}, i) = \sum_{t_{j'} \in \{0,\cdots,i-1\}} Z(t_{j'}, t_i).$$

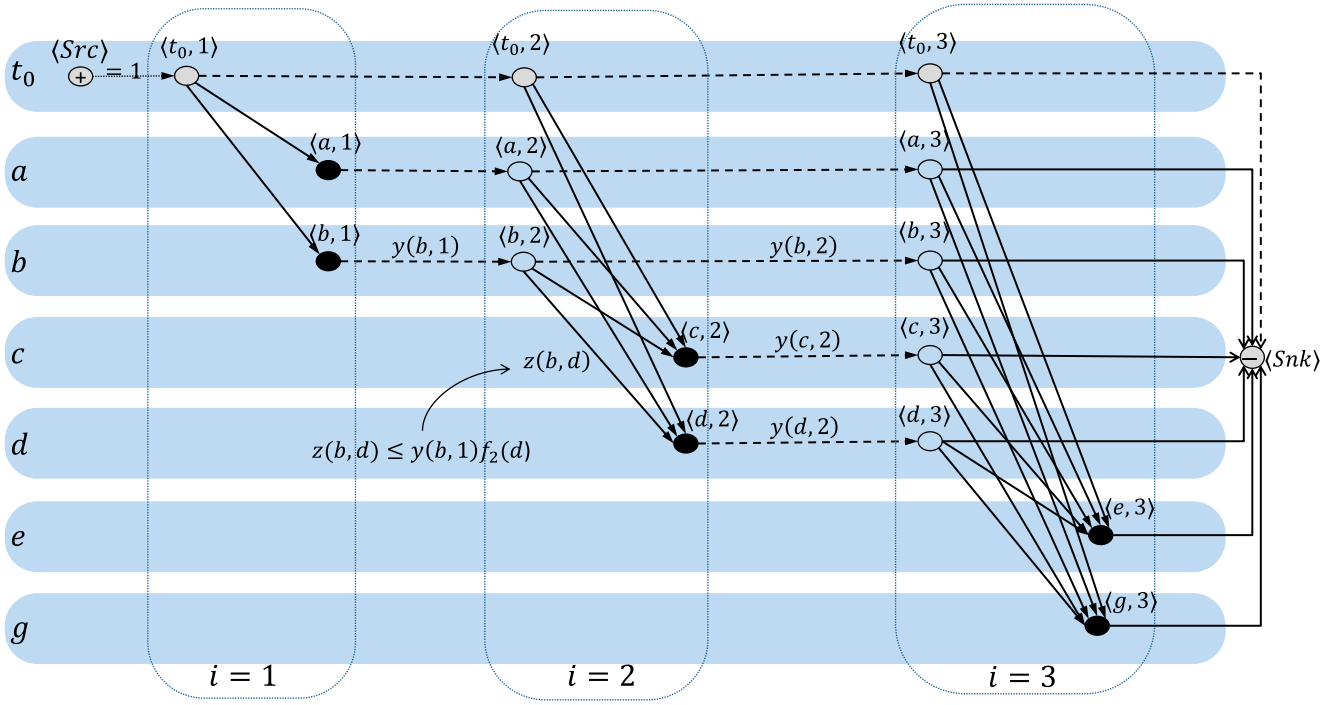
We next show ($\mathbb{S}$.3), that is,

$$Y(t_{j'}, j) = Y(t_{j'}, j-1) - \sum_{t_j \in t_j} Z(t_{j'}, t_j).$$

If $j' \neq i, i'$ then the equation trivially holds. For $j' = i$ and $i < j - 1$, we have

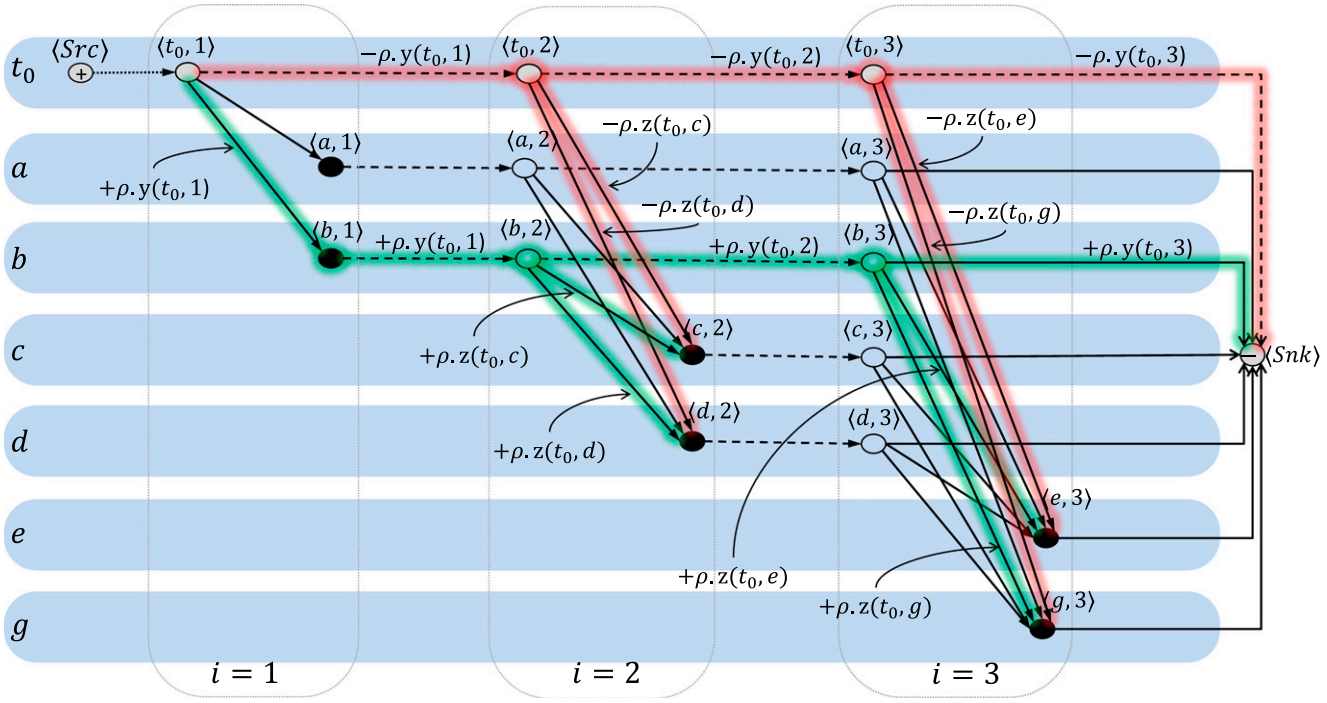
$$\begin{aligned} Y(t_i, j) = y(t_i, j) + \rho y(t_{i'}, j) &= y(t_i, j-1) - \sum_{t_j \in t_j} z(t_i, t_j) + \rho y(t_{i'}, j) \\ &= Y(t_i, j-1) - \rho y(t_{i'}, j-1) - \sum_{t_j \in t_j} \big(Z(t_i, t_j) - \rho z(t_{i'}, t_j)\big) + \rho y(t_{i'}, j) \\ &= Y(t_i, j-1) - \sum_{t_j \in t_j} Z(t_i, t_j). \end{aligned}$$

Figure B.4. (Color online) The flow network corresponding to the SSA Definition 3.



Notes. In this instance, there are three agents with type spaces $T_1 = \{a, b\}$, $T_2 = \{c, d\}$, and $T_3 = \{e, g\}$. All nodes in the same row correspond to the same type. The diagonal edges have dynamic capacity constraints, whereas all other edges have no capacity constraints. The flow going from $\langle t_{i'}, i\rangle$ to $\langle t_i, i\rangle$ corresponds to the ex ante probability of t_i taking the token away from $t_{i'}$. The flow going from $\langle t_{i'}, i\rangle$ to $\langle t_{i'}, i+1\rangle$ corresponds to the ex ante probability of $t_{i'}$ still holding the token after agent i is visited.

Figure B.5. (Color online) Changes made by applying reroute(t_0, b, ρ).



Notes. A ρ-fraction of the subtree rooted at $\langle t_0, 2\rangle$ (red online) is taken out and reassigned to the subtree rooted at $\langle b, 1\rangle$ (green online). The exact amount of change is indicated for each green and each red edge. The flow along all other edges stays intact. The operator has the effect of reassigning ρ-fraction of ex ante probability of allocation for type t_0 to type b.

The proof for the case $j' = i'$ is also similar. Finally, we show that (𝕊.4) also holds, that is, $Z(t_{j'}, t_j) \le Y(t_{j'}, j-1) f_j(t_j)$. If $j' \neq i, i'$, then the inequality trivially holds. For $j' = i$, $j > i-1$ we have

$$Z(t_i, t_j) = z(t_i, t_j) + \rho z(t_{i'}, t_j) \le y(t_i, j-1) f_j(t_j) + \rho y(t_{i'}, j-1) f_j(t_j) = Y(t_i, j-1) f_j(t_j).$$

For $j' = i'$, we have

$$Z(t_{i'}, t_j) = (1-\rho) z(t_{i'}, t_j) \le (1-\rho) y(t_{i'}, j-1) f_j(t_j) = Y(t_{i'}, j-1) f_j(t_j),$$

completing the proof. □

Proof of Lemma 1. For any given $(y, z) \in \mathbb{S}$ we show that it is always possible to modify y and z to obtain a nondegenerate feasible assignment with the same induced interim allocation probabilities (i.e., the same $y(\cdot, n)$). Let d denote the number of degenerate types with respect to (y, z), that is, define

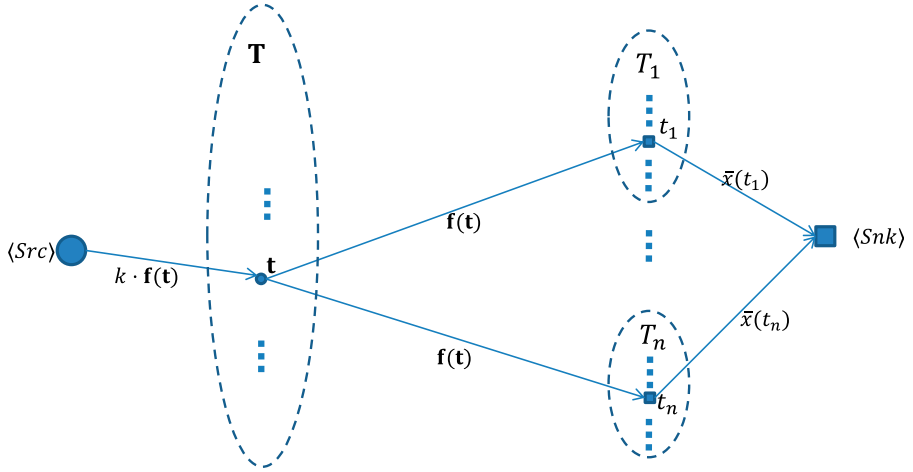
$$d = \#\left\{ t_i \in T_{\{1,\dots,n\}} \middle| y(t_i, n) = 0, y(t_i, i) > 0 \right\}.$$

The proof is by induction on d. The base case is $d = 0$, which is trivial. We prove the claim for $d > 0$ by modifying y and z, reducing the number of degenerate types to $d-1$ and then applying the induction hypothesis. Let t_i be a degenerate type. For each $t_{i'} \in T_{\{0,\dots,i-1\}}$, we apply the operator $\textsc{Reroute}(t_i, t_{i'}, \frac{z(t_{i'}, t_i)}{y(t_i, i)})$ unless $y(t_i, i)$ has already reached 0. Applying this operator to each type $t_{i'}$ eliminates the flow from $\langle t_{i'}, i\rangle$ to $\langle t_i, i\rangle$, so eventually $y(t_i, i)$ reaches 0 and t_i is no longer degenerate. Note that after applying these operations no new degenerate type is introduced, therefore the number of degenerate types is reduced to $d-1$. Furthermore, for all $t_{i'} \in T_N$, $y(t_{i'}, n)$ stays unchanged after applying these operations because $y(t_i, n) = 0$, which completes the proof. □

Proof of Lemma 2. To prove the lemma it is enough to show that for any augmentable type $t_{i'}$ and any nonaugmentable type t_i, $\textsc{ResCap}_{y,z}(t_{i'}, t_i) = 0$, which is equivalent to the statement of the lemma (the equivalence follows from the definition of $\textsc{ResCap}$ and equation (π)). The proof is by contradiction. Suppose $t_{i'}$ is augmentable and $\textsc{ResCap}_{y,z}(t_{i'}, t_i) = \delta$ for some positive δ; we show that t_i is also augmentable. Because $t_{i'}$ is augmentable, there exists a $(y', z') \in \mathbb{S}$ such that $y'(\tau, n) = y(\tau, n)$ for all $\tau \in T_N \setminus \{t_0, t_{i'}\}$ and $y'(t_{i'}, n) - y(t_{i'}, n) = \epsilon > 0$. Define

$$(y'', z'') = (1-\alpha) \cdot (y, z) + \alpha \cdot (y', z'),$$

Figure C.1. (Color online) The bipartite graph used in the max-flow/min-cut argument of the Proof of Theorem 3.



Note. The capacities are indicated on the edges.

where $\alpha \in [0,1]$ is a parameter that we specify later. Note that in (y'', z''), $t_{i'}$ is augmented by $\alpha\epsilon$, and $\textsc{ResCap}_{y'',z''}(t_{i'}, t_i) \geq (1-\alpha)\delta$, and $(y'', z'') \in \mathbb{S}$ because it is a convex combination of (y, z) and (y', z'). Consider applying $\textsc{Reroute}(t_{i'}, t_i, \rho)$ to (y'', z'') for some parameter $\rho \in [0,1]$. The idea is to choose α and ρ such that the exact amount, by which $t_{i'}$ was augmented, gets reassigned to t_i, by applying $\textsc{Reroute}(t_{i'}, t_i, \rho)$; so that eventually t_i is augmented, whereas every other type (except t_0) has the same allocation probabilities as they originally had in (y, z). In fact, by choosing

$$\alpha = \frac{y(t_{i'}, n)\delta}{2}$$

and

$$\rho = \frac{\epsilon\delta}{2 + \epsilon\delta},$$

we get a feasible assignment in which the allocation probability of t_i is augmented by $\alpha\epsilon$, whereas every other type (except t_0) has the same allocation probabilities as in (y, z). We still need to show that $\alpha > 0$. The proof is again by contradiction. Suppose $\alpha = 0$, so it must be $y(t_{i'}, n) = 0$, which would imply that $t_{i'}$ is a degenerate type because $y(t_{i'}, i') > 0$ (because $\textsc{ResCap}_{y,z}(t_{i'}, t_i) > 0$); however, (y, z) is a nondegenerate assignment by the hypothesis of the lemma, which is a contradiction. That completes the proof. □

Appendix C. Proofs from Section 5.3

Rest of the Proof of Theorem 3. We give a proof of $P(g_k) \subseteq \overline{\mathbb{X}}$ based on the min-cut/max-flow theorem. We start by constructing a directed bipartite graph as illustrated in Figure C.1. On one side we put a node $\langle \mathbf{t} \rangle$, for each type profile $\mathbf{t} \in \mathbf{T}$. On the other side we put a node $\langle t_i \rangle$, for each type $t_i \in T_{\{1,\dots,n\}}$. We also add a source node $\langle \textsc{Src} \rangle$ and a sink node $\langle \textsc{Snk} \rangle$. We add a directed edge from $\langle \textsc{Src} \rangle$ to the node $\langle \mathbf{t} \rangle$ for each $\mathbf{t} \in \mathbf{T}$ and set the capacity of this edge to $k \cdot \mathbf{f}(\mathbf{t})$. We also add n outgoing edges for every node $\langle \mathbf{t} \rangle$, each one going to one of the nodes $\langle t_1 \rangle, \dots, \langle t_n \rangle$ and with a capacity of $\mathbf{f}(\mathbf{t})$. Finally we add a directed edge from the node $\langle t_i \rangle$, for each $t_i \in T_{\{1,\dots,n\}}$, to $\langle \textsc{Snk} \rangle$ with capacity of $\bar{x}(t_i)$. Consider a maximum flow from $\langle \textsc{Src} \rangle$ to $\langle \textsc{Snk} \rangle$. For an interim allocation $\bar{x}$, there exists a feasible ex post implementation if and only if all the edges to the sink node $\langle \textsc{Snk} \rangle$ are saturated. In particular, if $\rho(\mathbf{t}, t_i)$ denotes the amount of flow from $\langle \mathbf{t} \rangle$ to $\langle t_i \rangle$, a feasible ex post implementation can be obtained by allocating to each type t_i with probability $\rho(\mathbf{t}, t_i)/\mathbf{f}(\mathbf{t})$ when the type profile $\mathbf{t}$ is reported by the agents.

We show that if a feasible ex post implementation does not exist, then $\bar{x} \notin P(g_k)$. Observe that if a feasible ex post implementation does not exist, then some of the incoming edges of $\langle \textsc{Snk} \rangle$ are not saturated by the max-flow. Let (A, B) be a minimum cut such that $\langle \textsc{Src} \rangle \in A$ and $\langle \textsc{Snk} \rangle \in B$. Let $B' = B \cap T_N$. We show that the polymatroid inequality

$$\bar{x}(B') \leq g_k(B') \tag{C.1}$$

must have been violated. The size of the cut between A and B is given by the following equation:

$$\textsc{Cut}(A, B) = \sum_{\mathbf{t} \in \mathbf{T} \cap A} \#\{i | t_i \in B\} \mathbf{f}(\mathbf{t}) + \sum_{\mathbf{t} \in \mathbf{T} \cap B} k \cdot \mathbf{f}(\mathbf{t}) + \sum_{\tau \in T_N \cap A} \bar{x}(\tau).$$

Observe that for each $\mathbf{t} \in \mathbf{T} \cap A$, it must be that $\#\{i|t_i \in B\} \le k$, otherwise moving $\langle \mathbf{t} \rangle$ to B would decrease the size of the cut. So the size of the minimum cut can be in simply written as

$$\textsc{Cut}(A, B) = \sum_{\mathbf{t} \in \mathbf{T}} \min\left(\#\{i|t_i \in B\}, k\right) \mathbf{f}(\mathbf{t}) + \sum_{\tau \in T_N \cap A} \overline{x}(\tau).$$

On the other hand, because some of the incoming edges of $\langle \textsc{Snk} \rangle$ are not saturated by the max-flow, it must be that

$$\sum_{\tau \in T_N} \overline{x}(\tau) = \textsc{Cut}(A \cup B - \langle \textsc{Snk} \rangle, \langle \textsc{Snk} \rangle) > \textsc{Cut}(A, B),$$

so

$$\sum_{\tau \in T_N \cap B} \overline{x}(\tau) > \sum_{\mathbf{t} \in \mathbf{T}} \min(\#\{i|t_i \in B\}, k)\ \mathbf{f}(\mathbf{t}).$$

The right-hand side of the above inequality is the same as $\mathbf{E}_{\mathbf{t} \sim \mathbf{f}}[\min(\#\{i|t_i \in B\}, k)]$, which shows that polymatroid inequality (C.1) of $P(g_k)$ is violated so $\overline{x} \notin P(g_k)$. That completes the proof. □

Proof of Lemma 4. Assuming that agents are independent (i.e., assuming $\mathbf{f}(\cdot)$ is a product distribution), $g_k(S)$ can be computed in time $O((n + |S|) \cdot k)$ using the following dynamic program in which G^i_j denotes the probability of the event that $\min(|\mathbf{t} \cap S \cap T_{\{1,\dots,i\}}|, k) = j$.

$$g_k(S) = \sum_{j=1}^{k} j \cdot G^n_j$$

$$G^i_j = \begin{cases} G^{i-1}_k + \left(\sum_{t_i \in S \cap T_i} f_i(t_i)\right) \cdot G^{i-1}_{k-1} & 1 \le i \le n, j = k \\ G^{i-1}_j + \left(\sum_{t_i \in S \cap T_i} f_i(t_i)\right) \cdot \left(G^{i-1}_{j-1} - G^{i-1}_j\right) & 1 \le i \le n, 0 \le j < k \\ 1 & i = 0, j = 0 \\ 0 & \text{otherwise.} \end{cases}$$ □

References

[1] Alaei S (2012) Mechanism design with general utilities. PhD thesis, University of Maryland, College Park.
[2] Alaei S (2014) Bayesian combinatorial auctions: Expanding single buyer mechanisms to many buyers. *SIAM J. Comput.* 43(2):930–972.
[3] Alaei S, Fu H, Haghpanah N, Hartline J (2013) The simple economics of approximately optimal auctions. *Proc. 2013 IEEE 54th Annual Sympos. Foundations Comput. Sci.* (IEEE, New York), 628–637.
[4] Armstrong M (1996) Multiproduct nonlinear pricing. *Econometrica* 64(1):51–75.
[5] Bakos Y, Brynjolfsson E (1999) Bundling information goods: Pricing, profits, and efficiency. *Management Sci.* 45(12):1613–1630.
[6] Beil DR, Wein LM (2003) An inverse-optimization-based auction mechanism to support a multiattribute RFQ process. *Management Sci.* 49(11):1529–1545.
[7] Belloni A, Lopomo G, Wang S (2010) Multidimensional mechanism design: Finite-dimensional approximations and efficient computation. *Oper. Res.* 58(4-part-2):1079–1089.
[8] Border K (2007) Reduced form auctions revisited. *Econom. Theory* 31(1):167–181.
[9] Border KC (1991) Implementation of reduced form auctions: A geometric approach. *Econometrica* 59(4):1175–1187.
[10] Briest P, Chawla S, Kleinberg R, Weinberg SM (2010) Pricing randomized allocations. *Proc. ACM-SIAM Sympos. Discrete Algorithms* (SIAM, Phildelphia), 585–597.
[11] Bulow J, Roberts J (1989) The simple economics of optimal auctions. *J. Political Econom.* 97(5):1060–1090.
[12] Cai Y, Daskalakis C, Weinberg SM (2012) An algorithmic characterization of multi-dimensional mechanisms. *Proc. 44th Sympos. Theory Comput. Conf.* (ACM, New York), 459–478.
[13] Cai Y, Daskalakis C, Weinberg SM (2012) Optimal multi-dimensional mechanism design: Reducing revenue to welfare maximization. *Proc. 53rd Annual IEEE Sympos. Foundations Comput. Sci., FOCS 2012* (IEEE, New York), 130–139.
[14] Cai Y, Daskalakis C, Weinberg SM (2013) Understanding incentives: Mechanism design becomes algorithm design. *Proc. 54th Annual IEEE Sympos. Foundations Comput. Sci.* (IEEE, New York), 618–627.
[15] Che Y, Gale I (2000) The optimal mechanism for selling to budget-constrained consumers. *J. Econom. Theory* 92(2):198–233.
[16] Che YK, Kim J, Mierendorff K (2013) Generalized reduced-form auctions: A network-flow approach. *Econometrica* 81(6):2487–2520.
[17] Chen X, Diakonikolas I, Orfanou A, Paparas D, Sun X, Yannakakis M (2015) On the complexity of optimal lottery pricing and randomized mechanisms. *Proc. 2015 IEEE 56th Annual Sympos. Foundations Comput. Sci.* (IEEE, New York), 1464–1479.
[18] Crawford GS (2008) The discriminatory incentives to bundle in the cable television industry. *Quant. Marketing Econom.* 6(1):41–78.
[19] Dewan S, Mendelson H (1990) User delay costs and internal pricing for a service facility. *Management Sci.* 36(12):1502–1517.
[20] Edith E (2007) Designing and learning optimal finite support auctions. *Proc. 18th Annual ACM-SIAM Sympos. Discrete Algorithms* (SIAM, Philadelphia), 736–745.
[21] Edmonds J (1970) Submodular functions, matroids and certain polyhedra. Guy R, ed. *Combinatorial Structures and Their Applications* (Gordon and Breach, New York), 69–87.
[22] Gopalan P, Nisan N, Roughgarden T (2015) Public projects, boolean functions and the borders of borders theorem. *Proc. 16th ACM Conf. Electronic Commerce*, vol. 395 (ACM, New York), 395.

[23] Haghpanah N (2014) Optimal multi-parameter auction design. PhD thesis, Northwestern University, Chicago.
[24] Laffont JJ, Robert J (1996) Optimal auction with financially constrained buyers. *Econom. Lett.* 52(2):181–186.
[25] Manelli AM, Vincent DR (2010) Bayesian and dominant-strategy implementation in the independent private-values model. *Econometrica* 78(6):1905–1938.
[26] Maskin E, Riley J (1984) Optimal auctions with risk averse buyers. *Econometrica* 52(6):1473–1518.
[27] Maskin ES (2000) Auctions, development, and privatization: Efficient auctions with liquidity-constrained buyers. *Eur. Econom. Rev.* 44(4–6): 667–681.
[28] Matthews S (1983) Selling to risk averse buyers with unobservable tastes. *J. Econom. Theory* 30(2):370–400.
[29] Matthews SA (1984) On the implementability of reduced form auctions. *Econometrica* 52(6):1519–1522.
[30] McAfee RP, McMillan J (1988) Multidimensional incentive compatibility and mechanism design. *J. Econom. Theory* 46(2):335–354.
[31] Mendelson H, Whang S (1990) Optimal incentive-compatible priority pricing for the M/M/1 queue. *Oper. Res.* 38(5):870–883.
[32] Mierendorff K (2011) Asymmetric reduced form auctions. *Econom. Lett.* 110(1):41–44.
[33] Mirman LJ, Sibley D (1980) Optimal nonlinear prices for multiproduct monopolies. *Bell J. Econom.* 11(2):659–670.
[34] Myerson RB (1981) Optimal auction design. *Math. Oper. Res.* 6(1):58–73.
[35] Pai MM, Vohra R (2008) Optimal auctions with financially constrained bidders. Discussion papers, Center for Mathematical Studies in Economics and Management Science, Northwestern University, Chicago.
[36] Parkes DC, Kalagnanam J (2005) Models for iterative multiattribute procurement auctions. *Management Sci.* 51(3):435–451.
[37] Roberts KWS (1979) Welfare considerations of nonlinear pricing. *Econom. J.* 89(353):66–83.
[38] Rochet JC, Chone P (1998) Ironing, sweeping, and multidimensional screening. *Econometrica* 66(4):783–826.
[39] Ronen A, Lehmann D (2005) Nearly optimal multi attribute auctions. *Proc. 6th ACM Conf. Electronic Commerce* (ACM, New York), 279–285.
[40] Schrijver A (2003) *Combinatorial Optimization: Polyhedra and Efficiency*, *Algorithms and Combinatorics* (Springer, New York).
[41] Shapley LS (1971) Cores of convex games. *Internat. J. Game Theory* 1(1):11–26.
[42] Spence AM (1980) Multi-product quantity-dependent prices and profitability constraints. *Rev. Econom. Stud.* 47(5):821–41.
[43] Vohra RV (2011) *Mechanism Design: A Linear Programming Approach*, vol. 47 (Cambridge University Press, New York).
[44] Wilson R (1994) Nonlinear pricing. *J. Political Econom.* 102(6):1288–1291.