A solution to the random assignment problem over the complete domain*

Akshay-Kumar Katta† Jay Sethuraman ‡

January 2004

Abstract

We consider the problem of allocating a set of indivisible objects to agents in a fair and efficient manner. In a recent paper, Bogomolnaia and Moulin consider the case in which all agents have *strict* preferences, and propose the *probabilistic serial* (PS) mechanism; they define a new notion of efficiency, called ordinal efficiency, and prove that the probabilistic serial mechanism finds an envy-free ordinally efficient assignment. However, the restrictive assumption of strict preferences is critical to their algorithm. Our main contribution is an analogous algorithm for the complete domain in which agents are allowed to be indifferent between objects. Our algorithm is based on a reinterpretation of the PS mechanism as an iterative algorithm to compute a "flow" in an associated network. In addition we show that on the complete domain it is impossible for any strategyproof mechanism to find a (random) assignment that is both ordinally efficient and envy-free.

*This research was supported by an NSF grant DMI-0093981 and an IBM partnership award. A version of this paper was presented at the Fall 2003 Midwest Theory Conference in Bloomington, Indiana. We gratefully acknowledge the comments of the participants of that conference.

†Department of Industrial Engineering and Operations Research, Columbia University, New York, NY; email: ark2001@columbia.edu

‡Department of Industrial Engineering and Operations Research, Columbia University, New York, NY; email: jay@ieor.columbia.edu

1 Introduction

The problem of allocating a number of indivisible objects to a number of agents, each desiring at most one object, in a *fair* and *efficient* manner is fundamental in many applications. If only one indivisible object must be allocated to $n > 1$ agents, there is really only one *fair* and *efficient* solution: assigning the object to an agent chosen uniformly at random among the n agents. When there are many objects the problem becomes substantially more interesting for a number of reasons: (i) many definitions of fairness and efficiency are possible; (ii) richer mechanisms emerge; and (iii) since preferences of the agents over the objects have to be solicited from the agents, *truthfulness* of the allocation mechanism becomes important.

A natural approach to the problem of allocating multiple objects among many agents is to generalize the simple "lottery" mechanism: order the agents uniformly at random and let them successively choose an available object according to this (random) order; thus the first agent picks his favorite object, the second agent picks his favorite object among the remaining objects, etc. This is the *random priority* mechanism, and has a number of attractive features: it is *ex post* efficient and truthful (or strategyproof: revealing true preferences is a dominant strategy for all the agents); it is fair in the weak sense of *equal treatment of equals* (agents with identical preferences are treated in an identical manner, a priori); however, it is not efficient when agents are endowed with utility functions consistent with their preferences, that is, it is not *ex ante* efficient. Moreover, it is not fair in the stronger sense of envy-freeness (described later).

A second solution to this problem adapts the competitive equilibrium with equal incomes (CEEI) solution for the fair division of unproduced commodities. This solution is envy-free and *ex ante* efficient, but is not strategyproof. Moreover, the computational and informational requirements of implementing this mechanism are prohibitive: it requires the solution of a fixed-point problem, and requires a complete knowledge of the utility functions of the agents. In contrast, the random priority mechanism is very simple to implement, and only requires the agents to report their "preferences" over objects.

Our work is inspired by the remarkable paper of Bogomolnaia and Moulin [3], who proposed a new mechanism (PS) for the random assignment problem. Their mechanism combines the attractive features of the random priority mechanism and the CEEI solution: it requires the agents to report their preferences over objects, not the complete utility functions, and yet computes a random assignment that is *ordinally efficient* and *envy-free*. Ordinal efficiency is stronger than *ex post* efficiency but weaker than *ex ante* efficiency; given the "ordinal" nature of the input to the mechanism (only preference rankings are used, not complete utility functions), this is perhaps the most meaningful notion of efficiency for an ordinal mechanism. Finally, the mechanism proposed by Bogomolnaia and Moulin is not truthful, it satisfies a weaker version of that property.

Unfortunately, the work of Bogomolnaia and Moulin [3] (and most of the existing work so far) assumes that all preferences are strict, which is a fairly restrictive requirement in many practical settings. Toward the end of their paper, they pose the problem of proposing

a similar mechanism for the complete domain, in which agents are allowed to be indifferent between objects, as an intersting and challenging problem, worthy of further study. Our main contribution is a solution to this problem that can be viewed as a natural extension of their PS mechanism. Our proposed solution is ordinally efficient and envy-free. Furthermore we show that for this richer preference domain, it is not possible for any mechanism to find an envy-free, ordinally efficient assignment and still satisfy the weaker version of strategyproofness satisfied by the PS mechanism in the strict preference domain. This observation indeed reveals that the complete domain is subtler and fundamentally different from the strict preference domain for the problems studied here. Our techniques rely on standard tools from network flow theory: the mechanism we propose can be viewed as an iterative application of an algorithm to compute the maximum flow in a parametric network. We first consider an extreme special case in which each agent has a single indifference class: she finds a subset of objects to be acceptable, but is indifferent between all of these objects. Our main algorithm is an iterative application of the algorithm for this special case.

The remainder of this paper is organized as follows. In §2 we describe the problem, define various notions of fairness and efficiency, and describe the notation. A description of relevant past work appears in §3 followed by the impossibility result in §4. We consider the special case of a single indifference class in §5, and show that the parametric max flow algorithm is ordinally efficient, envy-free, and strategyproof. In fact, we show stronger forms of efficiency and strategyproofness for this special case, which may be of independent interest. This algorithm serves as the basis for the main result, described in §6, which also contains a characterization of ordinally efficient assignments as well as a method to compute them.

2 Problem Description and Definitions

Problem Description. We consider the problem of allocating indivisible objects to agents so that each agent receives at most one object. Agents have preferences over the objects, and the allocation mechanism takes this profile of preferences as input. The preference ordering of each agent is assumed to be transitive and connected (every pair of objects is comparable). Thus, given a pair of objects, an agent strictly prefers one to the other or is indifferent between them; both the indifference relation and the strict preference relation are transitive. The preference ordering of agent i is denoted by $\geq_i$, with $>_i$ and $=_i$ denoting the strict preference relation and the indifference relation of agent i respectively. The set of all preference orderings is called the preference domain, and denoted by $\mathcal{A}$. Throughout the paper we let N be the set of agents and A be the set of objects. For convenience, we assume that $|N| = |A|$, and we denote this common cardinality by n; we also assume that agents' preference lists are *complete*, meaning no agent prefers being unassigned to receiving an object. These assumptions can be relaxed easily by introducing dummy objects or dummy agents and appropriately modifying the preferences of the exising agents or by introducing arbitrary preferences for the dummy agents.

A *deterministic assignment* is an allocation in which each agent receives exactly one object and each object is allocated to exactly one agent. More formally, it is a one-to-one corre-

spondence between the set of agents, N, and the set of objects, A. Often, it is convenient to think of a deterministic assignment as a $0-1$ matrix, with rows indexed by agents and columns indexed by objects: a $0-1$ matrix represents a deterministic assignment if and only if it contains exactly one 1 in each row and each column. (Such matrices are called *permutation* matrices.) We let $\mathcal{D}$ be the set of all deterministic assignments.

A *random assignment* is a probability distribution over deterministic assignments; the corresponding convex combination of permutation matrices is a stochastic matrix, whose $(i,j)^{\text{th}}$ entry represents the probability with which agent i receives object j. For our purposes, different random assignments that give rise to the same stochastic matrix are equivalent, so we do not distinguish a random assignment from its associated matrix. Given a random assignment matrix P, we let P_i be the i^{th} row, which represents the *allocation* for agent i in this random assignment. (An *allocation* is simply a probability distribution over the set of objects A.) We let $\mathcal{R}$ be the set of all random assignments.

A *random assignment mechanism* is simply a mapping from $\mathcal{A}^n$ to $\mathcal{R}$ Our objective is to find a random assignment mechanism satisfying some desirable efficiency and fairness properties. To describe these properties formally, we extend the agents' preferences over objects to preferences over random assignments. Given two random assignments P and Q, an agent i prefers P to Q (denoted $P \succeq_i Q$) if and only if the allocation P_i stochastically dominates the allocation Q_i, where the stochastic dominance is with respect to $\geq_i$, agent i's preference ordering over sure objects. Formally,

$$P \succeq_i Q \iff \sum_{k:k\geq_i j} p_{ik} \geq \sum_{k:k\geq_i j} q_{ik},\ \forall j \in A. \tag{1}$$

If, in addition, $\sum_{k:k\geq_i j} p_{ik} > \sum_{k:k\geq_i j} q_{ik}$ for some $j \in A$, then agent i *strictly prefers* P to Q, and this is denoted by $P \succ_i Q$; on the other hand, if $\sum_{k:k\geq_i j} p_{ik} = \sum_{k:k\geq_i j} q_{ik}$ for all $j \in A$, then agent i is indifferent between P and Q, and this is denoted by $P \sim_i Q$. We emphasize that this relation $\succeq_i$ is only a *partial order* on the set of all random assignments, and so not every pair of random assignments is comparable. Finally, a random assignment P *dominates* a random assignment Q (denoted $P \succeq Q$), if every agent prefers P to Q, that is, if $P \succeq_i Q$ for all $i \in N$. Even if P and Q are comparable for every agent, it may be the case that $P \not\succ Q$ and $Q \not\succ P$; this is the case if some agent prefers P to Q and another prefers Q to P. We are now ready to describe the notions of efficiency and fairness adopted here.

Efficiency. A random assignment P is *ordinally efficient* if it is not dominated by any other random assignment Q. If we apply this definition to a deterministic assignment—a random assignment in which the entries are 0 or 1—then we recover the familiar definition of Pareto efficiency. Thus, ordinal efficiency can be viewed as a natural extension of Pareto efficiency to the random assignment setting. Other extensions are possible and have been considered in the literature; we briefly discuss two such extensions, one weaker and one stronger. A random assignment is *ex post efficient* if it can be represented as a probability distribution over Pareto efficient deterministic assignments. A random assignment is *ex ante efficient* if for any profile

of utility functions consistent with the preference profile of the agents, the resulting expected utility vector is Pareto efficient: for any random assignment in which the expected utility of some agent is strictly greater, there must be another agent whose expected utility is strictly lower. It is easy to show that ex ante efficiency implies ordinal efficiency, which implies ex post efficiency. The attractive feature of ordinal efficiency is that its informational requirements are low: it relies only on agents' preferences over objects, not on their preferences over random allocations; yet, it is stronger than ex post efficiency (which also only requires preferences over objects). For this reason we shall restrict ourselves to ordinally efficient assignments in the rest of this paper.

Fairness. A random assignment P is *envy-free* if each agent prefers her allocation to that of any other agent. Formally, P is envy-free if $P_i \succeq_i P_{i'}$ for all $i, i' \in N$. A weaker version of envy-freeness can also be defined: P is *weakly envy-free* if no agent strictly prefers someone else's allocation to hers, that is, if $P_{i'} \not\succ_i P_i$ for all $i, i' \in N$. A still weaker definition is possible too: A random assignment P satisfies *equal treatement of equals* if agents with identical preferences get identical allocations ($P_i = P_{i'}$ if $>_i = >_{i'}$). Again, it is easy to show that envy-freeness implies weak envy-freeness and equal treatment of equals. (It is easy to see that weak envy-freeness and equal treatment of equals are not comparable: it is easy to construct assignments that satisfy one property but not the other.) In most of this paper, we shall restrict ourselves to envy-free assignments.

Incentives. A random assignment mechanism is said to be *strategyproof* if revealing the true preference ordering is a dominant strategy for each agent. Again a weaker notion of strategyproofness can be defined: A mechanism is weakly strategyproof if by falsifying her preference list an agent cannot obtain an allocation that she strictly prefers to her true allocation. Specifically, consider the two preference profiles $(>_1, >_2, \ldots, >_n)$ and $(>_1^*, >_2, \ldots, >_n)$ in which the preferences of all the agents except agent 1 is fixed; let P and P^* be the respective random assignments computed by the mechanism. The mechanism is *strategyproof* if $P \succeq_1 P^*$. The mechanism is *weakly strategyproof* if $P^* \not\succ_1 P$.

3 Related Work

Several solutions have been proposed to the random assignment problem. The earliest work on this problem is due to Hylland and Zeckhauser [11], who adapt the competitive equilibrium with equal incomes (CEEI) solution. The mechanism is "expensive" both in terms of its informational requirements (needs utility functions of all individuals) and in terms of its computational requirements (requires the solution of a fixed-point problem). the resulting solution is envy-free and Pareto efficient with respect to the utility functions, but the mechanism is not strategyproof. Gale [7] conjectured that this is the best possible, specifically, that no strategyproof mechanism that elicits utility functions and achieves Pareto efficiency with respect to

these utility functions can find a "fair" solution, even in the weaker sense of equal treatment of equals. This conjecture was proved by Zhou [19].

The prohibitive cost of CEEI motivated research on finding simpler mechanisms. A natural candidate was the *random priority* (RP) mechanism: order the agents uniformly at random, and let them successively choose an object in that order. This was analyzed by Abdulkadiroglu and Sonmez [1] who showed that RP is equivalent to the unique core allocation of the Shapley-Scarf housing market [18] in which each agent is endowed with an object chosen uniformly at random; Roth and Postlewaite [17] had earlier shown that any Shapley-Scarf housing market with strict preferences has a unique core, and Roth [16] proved that the core (from random endowments) is *stategyproof*, *anonymous* (does not depend on the labels of the agents), and *ex-post Pareto efficient*. The equivalence of RP with the core from random endowments implies that RP itself is strategyproof, anonymous, and always an ex-post Pareto efficient solution; in fact, Abdulkadiroglu and Sonmez showed [1] that the only Pareto efficient matching mechanisms are *serial dictatorships*, which are like *RP*, except that the initial ordering of the agents is chosen in a deterministic fashion. Zhou [19] showed that the solution computed by RP may not be efficient if agents are endowed with utility functions consistent with their preferences.

Bogomolnaia and Moulin [3] considered mechanisms that combined the "best" of CEEI and RP: the CEEI solution has strong efficiency and fairness properties, but is not strategyproof; the RP solution is strategyproof but has weaker efficiency and fairness properties. The key contribution of Bogomolnaia and Moulin [3] is the definition of an intermediate notion of efficiency—*ordinal efficiency*–and natural mechanisms to find all ordinally efficient solutions. They also showed that one such mechanism—the *probabilistic serial* (PS) mechanism—is weakly strategyproof, and achieves an envy-free, ordinally efficient solution. In a result parallel to Zhou's impossibility theorem, they show that no strategyproof mechanism can achieve both ordinal efficiency and fairness, even in the weak sense of equal treatment of equals. All of their results were derived in the setting of a strict preference domain; the extension to the "full domain" (that allows for "subjective indifferences") was left as a challenging open problem at the end of their work, and is addressed here.

The PS mechanism was introduced by Cres and Moulin [6] in the context of a scheduling problem with opting out. In that model, every agent has a job that needs unit time on a machine by a certain deadline; thus the objects are really possible time-slots to which jobs can be assigned. Agents have strictly decreasing, positive, utilities for the time-slots to which their jobs can be feasibly assigned, and a utility of zero for not being assigned a time-slot before their deadline. (Thus agents' preference orderings are identical; only their deadlines and utilities may vary.) Cres and Moulin [6] show that in their model the PS solution stochastically domainates the RP solution. Bogomolnaia and Moulin [4] introduced the concept of ordinal efficiency in that model and provided two different characterizations of the PS mechanism: it is the only mechanism that is ordinally efficient, strategyproof, and satisfies equal treatment of equals; and it is the only mechanism that is ordinally efficient and envy-free.

Simultaneous Eating Algorithm. For our purposes, it will be useful to review the work of Bogomolnaia and Moulin [3], who propose a class of algorithms, called *simultaneous eating* algorithms, that generate all ordinally efficient assignments. The preference domain is the set of all *strict* preference orderings, that is, the set of all permutations of the set A. Each algorithm in this class is specified by a set of n *eating speed* functions w_i, for $i \in N$. We can view $w_i(t)$ as the speed at which agent i is allowed to consume objects at time t. We assume that $w_i(t) \geq 0$ and

$$\int_{t=0}^{t=1} w_i(t)dt \;=\; 1.$$

For convenience, we also assume that $w_i(t) < \infty$, although this is not necessary for the results to hold. In the simultaneous eating algorithm for the profile of eating speeds $(w_i)_{i\in N}$, each agent consumes her best available object at rate $w_i(t)$ at time t; each object can be viewed as having size 1, and once a unit amount of an object has been consumed, it becomes unavailable for further consumption. The probabilistic serial (PS) mechanism is obtained when all the eating rate functions are identically 1.

This is best illustrated by an example from Bogomolnaia and Moulin [3]. Let $N = 1, 2, 3, 4$, $A = a, b, c, d$, and consider the preference profile in which agents 1 and 2 rank the objects $a > b > c > d$ and agents 3 and 4 rank the objects $b > a > d > c$. Suppose all the eating rate functions are identically 1. Initially, agents 1 and 2 eat object a, whereas agents 3 and 4 eat object b; at $t = 1/2$, unit amounts of a and b are consumed, so a and b are no longer available; from this point on, agents 1 and 2 eat object c (their best available object) and agents 3 and 4 eat object d. At $t = 1$, all of the objects are consumed, and each agent has consumed a unit amount. The resulting random assignment is

	a	b	c	d
1	1/2	0	1/2	0
2	1/2	0	1/2	0
3	0	1/2	0	1/2
4	0	1/2	0	1/2

Consider the PS mechanism. As described, it isn't clear how this mechanism can be readily extended to handle indifferences. The difficulty is that if an agent is indifferent between a set of objects at some point, it is not at all clear precisely which objects he should eat. This ambiguity does not arise when preferences are strict; and resolving this issue is critical to achieving ordinal efficiency. Later in the paper, we shall arrive at the PS mechanism from a slightly different point of view, which will readily extend to the complete preference domain.

4 An Impossibility Result

Our first result establishes that on the complete preference domain, ordinal efficiency and envy-freeness are incompatible with strategyproofness even in the weaker sense. This is in contrast to the result of Bogomolnaia and Moulin [3], who showed that on the strict preference

domain, the PS mechanism is weakly strategyproof and achieves an envy-free, ordinally efficient solution.

Example 1. Let $N = \{1, 2, 3\}$, $A = \{a, b, c\}$, and consider the following preference profile:

$$\begin{array}{cccc} 1 & \{a,b\} & & c \\ 2 & a & b & c \\ 3 & a & c & b \end{array}$$

Agent 1 is indifferent between a and b, but prefers both of these to c; agent 2 prefers a to b to c, and agent 3 prefers a to c to b. It is easy to verify that

$$\begin{array}{c|ccc} & a & b & c \\ \hline 1 & 0 & 3/4 & 1/4 \\ 2 & 1/2 & 1/4 & 1/4 \\ 3 & 1/2 & 0 & 1/2 \end{array} \tag{2}$$

is ordinally efficient and envy-free. We first argue that this is the only assignment that is both ordinally efficient and envy-free. To that end, we make a few observations. For agents 2 and 3 to not envy each other, we must have

$$p_{2a} \;=\; p_{3a};$$

similarly, for agents 1 and 2 to not envy each other, we must have

$$p_{1a} + p_{1b} \;=\; p_{2a} + p_{2b},$$

which also implies

$$p_{1c} \;=\; p_{2c}.$$

Also, ordinal efficiency implies that $p_{1a} = 0$ and $p_{3b} = 0$: if $p_{1a} > 0$, then agent 1 can give up her share of a in exchange for an equal amount of b from agents 2 or 3; if $p_{3b} > 0$, agent 3 can give up her share of b in exchange for an equal amount of c from agents 1 or 2. In either case, the resulting assignment dominates the original one. Taking all of these observations together, we see that $p_{2a} = p_{3a} = 1/2$; as $p_{3b} = 0$, $p_{3c} = 1/2$ (because agent 3's allocation should sum to 1). Since $p_{1c} = p_{2c}$, we must have $p_{1c} = p_{2c} = 1/4$. This means that $p_{1b} = 3/4$ and $p_{2b} = 1/4$. Thus, the only envy-free, ordinally efficient assignment for this instance is the one shown in (2).

Now suppose agent 1 reports that she prefers a to b to c. Then, envy-freeness implies $p_{1a} = p_{2a} = p_{3a} = 1/3$, and that agents 1 and 2 must receive identical allocations. As described earlier, p_{3b} must be 0, otherwise the assignment cannot be ordinally efficient. Thus, $p_{3c} = 2/3$, which implies $p_{1c} = p_{2c} = 1/6$, which implies $p_{1b} = p_{2b} = 1/2$. Thus, agent 1's allocation is $(1/3, 1/2, 1/6)$ if she submits a preference list $a > b > c$; whereas her allocation is $(0, 3/4, 1/4)$ if she submits her true preference list $\{a, b\} > c$. Both these allocations are uniquely determined

by ordinal efficiency and envy-freeness, and the former stochastically dominates the latter. So it is impossible to find an envy-free, ordinally efficient allocation even if the mechanism is only required to be weakly strategyproof.

This also formally verfies our intuitive feeling that the strict preference domain is indeed special, and allowing for indifferences changes the character of the problem.

5 Random Assignment: A Single Indifference Class

We first consider the special case of the random assignment problem over the domain in which every agent has a single indifference class. Specifically, each agent declares a subset of the objects as acceptable, but is indifferent between acceptable objects. Two reasons underlie our focus on this special case: first, our solution to the overall problem is essentially an iterative application of the ideas that help solve this special case; and second, stronger results can be proved for this special case, which may be of independent interest. In particular, the impossibility result of §4 does not hold, and in fact, it is possible to find a strategyproof mechanism that achieves both ordinal efficiency and envy-freeness. We note that it is possible to adapt *random priority* to this special case; we do not describe this adaptation because (a) the results for the natural adaptation of the random priority mechanism are weaker; and (b) our goal is to use this mechanism to eventually find an ordinally efficient assignment on the complete domain, and we know from Bogomolnaia and Moulin [3] that the random priority mechanism does not find an ordinally efficient assignment.

Model. Consider a scheduling problem in which jobs require unit processing times on a machine. Each job has a set of acceptable time slots in which it can be processed, and the utility associated with processing a job in any one of those time slots is the same. Thus, one can view the jobs as agents; the time-slots as objects; and the acceptable time-slots for a particular job as the set of acceptable objects for the corresponding agent. The assumption on utilities means that every agent is indifferent between her acceptable objects. Our objective is to find a strategyproof mechanism that is envy-free and ordinally efficient. For convenience, we relax the assumption that the number of objects is the same as the number of agents in this section alone; thus, each agent indicates only the subset of objects that she finds acceptable, each of which gives unit utility. We assume that each agent finds at least one object acceptable; agents who do not meet this assumption are irrelevant to the problem as they will necessarily have zero utility in all solutions.

Let $L_i \subseteq A$ be the set of all *acceptable* objects for agent i; recall that she is indifferent between any two objects in L_i. Thus, for a random assignment P, the utility (or welfare-level) u_i of agent i is simply $\sum_{j:j\in L_i} p_{ij}$, the probability that she is assigned an acceptable object.

Algorithm and Analysis. We now propose a mechanism that computes a random assignment for any given preference profile. We shall later show that the mechanism is strategyproof and that the computed random assignment is both envy-free and ordinally efficient. In fact,

we shall prove some stronger properties of both the mechanism and the computed random assignment. The algorithm can be viewed as finding a specific type of *flow* in a suitably defined *network*. (For background on network flow problems and techniques, we recommend [2].)

Suppose the preference profile in a given instance of the random assignment problem is such that some set of three agents have only two acceptable objects among them. Then, it is obvious that the combined utilities of these three agents cannot exceed 2. The general algorithm for solving the problem builds on this trivial observation: it consists of locating such a "bottleneck" subset of agents, and allocating their acceptable objects amongst them in a "fair" way, eliminating these agents and their acceptable objects, and recursively applying the idea on the remaining set of agents and objects. To formalize this, let $\Gamma(Y)$ denote the set of objects acceptable to at least one agent in Y, for any $Y \subseteq N$, and let

$$v \;=\; \min_{Y \subseteq N} \frac{|\Gamma(Y)|}{|Y|},$$

with X denoting the largest cardinality set $Y \subseteq N$ with $|\Gamma(Y)|/|Y| = v$. If $v \geq 1$, there are more objects than agents competing for it, so each agent can be assigned a distinct object, and there is nothing more to do. If $v < 1$, the algorithm allocates the objects in $\Gamma(X)$ among the agents in X such that the utility of any agent $i \in X$ is exactly v; such an assignment would completely use up the objects in $\Gamma(X)$, so none of these objects can even be fractionally assigned to the other agents; moreover, the agents in X cannot be assigned any other object. Thus, the agents in X and the objects in $\Gamma(X)$ can be removed from the problem as they play no further role. The same calculations are carried out in the residual problem with agents $N \setminus X$, objects $A \setminus \Gamma(X)$, and the preference profiles of the agents restricted to the these "currently available" objects.

The only remaining details are the identification of the "bottleneck" subset X of agents, and the precise allocation of the "bottleneck" set of objects, $\Gamma(X)$, to the agents in X. It is an elementary exercise to show that this in fact follows from the *max-flow min-cut* theorem [2]. We provide the complete argument for the sake of completeness.

Consider the bipartite graph with nodes corresponding to N and A, and edges (i, j) for $i \in N, j \in L_i$. Let these edges have infinite capacity. Augment the network by adding a source node s, a sink node t; edges with capacity $\lambda \geq 0$ going from s to each node in N, and edges with unit capacity going from each node in A to t. A cut in this graph is a collection of nodes that contain s but not t; the capacity of a cut is the sum of the capacities of the edges from the "s-component" to the "t-component." A fundamental theorem in network flow theory is the *max-flow min-cut* theorem, which states that the size of the maximum flow from s to t in a given directed network is equal to the minimum capacity cut separating s from t.

We view $\lambda \geq 0$ as a *parameter*, and study the minimum-capacity $s - t$ cuts (or simply minimum cuts) in the network as λ is varied. For the application of interest, it will not be necessary to consider $\lambda > 1$, so we may assume that $\lambda \in [0, 1]$. Fix any such λ. Any cut separating s from t must be of the form $s \cup X \cup W$ for some $X \subseteq N$ and $W \subseteq A$. Furthermore since the "original" edges (i.e. those from the agents to their acceptable objects) have infinite

capacity, $W \supseteq \Gamma(X)$ in any minimum cut. The capacity of such a cut is then

$$\lambda(|N| - |X|) + |W|,$$

which shows that $W = \Gamma(X)$ in any minimum cut. Thus, any minimum cut is completely determined by the set X and is of the form $s \cup X \cup \Gamma(X)$ with capacity

$$\lambda(|N| - |X|) + |\Gamma(X)|.$$

When several minimum cuts exist, we pick the minimum-cut with the largest $|X|$. (This is unique.)

For λ sufficiently close to 0, it is clear that $X = \emptyset$ gives the minimum cut, whose capacity grows linearly in λ. We consider two possibilities, depending on whether or not $X = \emptyset$ is a minimum cut for $\lambda = 1$. If it is, it will be clear from the following argument that each agent can be assigned an acceptable object with probability 1, and there is nothing more to do. If $X = \emptyset$ is *not* a minimum cut for $\lambda = 1$, let $\lambda^* \in (0, 1)$ be the smallest value of λ for which $X = \emptyset$ is not *the* minimum cut, that is, some set X with $|X| > 0$ is also a minimum cut. (By our convention, $X = \emptyset$ could be *a* minimum cut, but is not the only one.) Let X^* be the minimum cut for $\lambda = \lambda^*$. By definition, for any $\lambda < \lambda^*$, $X = \emptyset$ was the minimum-cut, with capacity $\lambda|N|$. It is clear (by continuity) that the capacity of the minimum-cut for $\lambda = \lambda^*$ must be exactly $\lambda^*|N|$; since X^* is a minimum-cut for $\lambda = \lambda^*$ and has capacity $\lambda^*(|N| - |X^*|) + |\Gamma(X^*)|$, we have

$$\lambda^*|N| \;=\; \lambda^*(|N| - |X^*|) + |\Gamma(X^*)|,$$

from which we get

$$\lambda^* \;=\; \frac{|\Gamma(X^*)|}{|X^*|}. \tag{3}$$

Let $\lambda = \lambda^*$, and consider any $Y \subseteq N$; the capacity of the cut $s \cup Y \cup \Gamma(Y)$ is simply $\lambda^*(|N| - |Y|) + |\Gamma(Y)|$, which must be at least as large as the minimum-cut capacity $\lambda^*|N|$. This, combined with Eq. (3), establishes

$$\lambda^* \;=\; \min_{Y \subseteq N} \frac{|\Gamma(Y)|}{|Y|}; \quad X^* \;=\; \arg\min_{Y \subseteq N} \frac{|\Gamma(Y)|}{|Y|}.$$

By our convention, X^* is the largest cardinality subset of agents with this property.

Consider the network with $\lambda = \lambda^*$. By the max-flow min-cut theorem, the maximum flow from s to t in this network matches the capacity of the minimum cut, which equals $\lambda^*|N|$. However, every edge from s has capacity exactly λ^* (and there are exactly $|N|$ such edges), so all of these edges carry a flow of λ^* in a maximum flow; in particular, λ^* units reach the sink from each $i \in X^*$, which gives us the required random assignment.

The quantity λ^* is the (smallest) *breakpoint* of the min-cut capacity function of the parametric network. The breakpoints of the min-cut capacity function of a parametric network are well-understood [8]. Efficient algorithms to compute these breakpoints have been discovered by several researchers, see [2]; the fastest of these methods, discovered by Gallo, Grigoriadis

and Tarjan [8], is based on the preflow-push algorithm for the maximum flow problem due to Goldberg and Tarjan [9]. Gallo et al. [8] propose algorithms to find maximum flows in an n-node, m-arc network for $O(n)$ values of the parameter in $O(nm \log(n^2/m))$ time. In addition, they study the min-cut capacity as a function of a parameter λ and propose algorithms to compute its smallest (or largest) breakpoint in $O(nm \log(n^2/m))$ time; a more complicated algorithm but with the same running time in fact finds *all* the breakpoints of the min-cut capacity function. We note that variants of this problem have been studied in the operations research literature as "flow sharing" problems in the context of network transmission [5, 12, 13, 14, 15] and network vulnerability [10]. For this reason, we do not describe the algorithms to compute the breakpoints and the associated bottleneck sets, but rather appeal to these results to conclude that these can be found efficiently.

Properties. We now proceed to formally prove some attractive properties of the random assignment found by the parametric max-flow algorithm. Let $\lambda_1, \lambda_2, \ldots, \lambda_r$ be the successive "breakpoints" found by the algorithm, with $X_1, X_2, \ldots, X_r$ being the associated sets of agents. Let $P = (p_{ij})$ be the random assignment found by the parametric max-flow algorithm. (Thus, for agent i, $\sum_{j \in L_i} p_{ij} = \lambda_i$.)

Theorem 1 *The random assignment computed by the parametric max-flow algorithm lexicographically maximizes the n-vector, whose i^{th} component is the i^{th} smallest utility. In particular, the random assignment is ordinally efficient.*

Proof. From the description of the algorithm, it is clear that $0 < \lambda_1 < \lambda_2 < \ldots < \lambda_r \leq 1$: When $\lambda = \lambda_1$, the agents in X_1 collectively use up all the objects in $\Gamma(X_1)$, but the remaining agents are able to still send (more than) λ_1 units of flow to the sink. Consider the agents in X_1. It is clear that these agents can collectively achieve a utility of at most $\lambda_1 |X_1|$, so no agent in X_1 can achieve a utility greater than λ_1 without some other agent achieving a utility less than λ_1. Thus, the proposed algorithm is *maximin fair*: it maximizes the welfare of the agent who is worst-off. Inductively, it follows that for any $r' \leq r$, among all assignments that guarantee a utility of λ_i for agents in X_i, $i = 1, 2 \ldots, r' - 1$, the agents in $X_{r'}$ achieve the maximum possible utility. This proves that the utility vector– arranged in nondecreasing order– associated with the random assignment found by the algorithm is lexicographically optimal. Ordinal efficiency of the is immediate: for otherwise, there is some other assignment in which no agent is worse-off and at least one agent is strictly better off; such an assignment would have a utility vector that is lexicographically larger, which is not possible. ■

We note that a version of this result has been observed in the network-flow literature [14, 15, 8].

Theorem 2 *The random assignment P computed by the parametric max-flow algorithm is envy-free.*

Proof. An agent $k \in X_i$ cannot envy an agent $k' \in X_1 \cup X_2 \cup \ldots \cup X_{i-1}$ because

$$\sum_{j \in L_k} p_{k'j} \ \leq \ \sum_{j \in A} p_{k'j} < \lambda_i,$$

so k strictly prefers her allocation to that of any agent $k' \in X_1 \cup X_2 \cup \ldots \cup X_{i-1}$. Moreover, k cannot envy an agent $k' \in X_{i+1} \cup X_{i+2} \cup \ldots \cup X_r$ because none of these agents can be assigned to any object that k finds acceptable. (When k is assigned an object, any acceptable object that has not been allocated to agents in $X_1, X_2, \ldots, X_{i-1}$ will be completely allocated to agents in X_i and will not be available at the next stage.) ■

Let $S \neq \emptyset$ be any coalition of agents, with L_i and L'_i being the true and stated preferences of agent $i \in S$. We now consider the outcome of the algorithm for the preference profiles $(L_i, i \in S; L_k, k \in N \setminus S)$ and $(L'_i, i \in S; L_k, k \in N \setminus S)$. We view the first as the "true" preference profile and the second as the "stated" preference profile. A mechanism is said to be *group strategyproof* if it is not possible for any coalition S to improve its allocation (meaning, each agent in S is at least as well off, and one of them is strictly better off) by stating false preferences.

We next show that the parametric max-flow algorithm is group strategyproof.

Theorem 3 *The parametric max-flow mechanism is group strategyproof.*

Proof. Consider the outcome of the parametric max-flow algorithm on the profiles $(L_i, i \in S; L_k, k \in N \setminus S)$ and $(L'_i, i \in S; L_k, k \in N \setminus S)$ for some $S \subseteq N$. Let $\lambda_1, \lambda_2, \ldots, \lambda_r$ be the successive breakpoints found by the parametric max-flow algorithm on the true preference profile, with $X_1, X_2, \ldots, X_r$ being the associated sets. We first claim that no agent in X_1 can strictly benefit by being a part of S.

Suppose otherwise. Since the collective true utilities of the agents in X_1 is at most $\lambda_1 |X_1|$, if one of the agents in X_1 achieves a strictly larger utility, then some agent in X_1 must achieve a strictly smaller utility. Such an agent must be truthful, by the definition of group strategyproofness. Let W_1 be the set of (truthful) agents whose utility is strictly less than λ_1. Since every other member of X_1 achieves a utility of at least λ_1, the parametric max-flow algorithm will process all of the agents in W_1 before processing any agent in $X_1 \setminus W_1$; in particular, no member of $X_1 \setminus W_1$ can be assigned an object in $\Gamma(W_1)$. So the collective utility achievable by the agents in $X_1 \setminus W_1$ is at most $|\Gamma(X_1)| - |\Gamma(W_1)|$. Because each agent in $X_1 \setminus W_1$ has a utility of at least λ_1 with one of them having a strictly better utility, we must have

$$|\Gamma(X_1)| - |\Gamma(W_1)| \ > \ (|X_1| - |W_1|)\lambda_1 \ = \ (|X_1| - |W_1|)\frac{|\Gamma(X_1)|}{|X_1|}; \tag{4}$$

this implies

$$\frac{|\Gamma(W_1)|}{|W_1|} < \frac{|\Gamma(X_1)|}{|X_1|}, \tag{5}$$

which contradicts the fact that X_1 is the bottleneck subset of agents. So no agent in X_1 can strictly benefit by lying.

We now use induction: suppose that no agent in $X_1, X_2, \ldots, X_k$ can strictly benefit by lying. Then, the utility of any agent in X_i is exactly λ_i, for $i = 1, 2, \ldots, k$. Thus, none of the objects in $\Gamma(X_1) \cup \Gamma(X_2) \cup \ldots \cup \Gamma(X_k)$ can be allocated to any agent outside of $X_1 \cup X_2 \cup \ldots \cup X_k$. Therefore, the agents in X_{k+1} together can collectively achieve a utility of at most $\lambda_{k+1}|X_{k+1}|$; for an agent in X_{k+1} to achieve a strictly larger utility, some other (truthful) agent must achieve a strictly smaller utility, which means W_{k+1} must be nonempty. Using W_{k+1} and X_{k+1} instead of W_1 and X_1 in Eqs. (4) and (5), we observe that this is not possible. ■

6 Random Assignment: The complete domain

We now extend the parametric max-flow algorithm to compute an envy-free, ordinally efficient assignment on the complete domain. (In view of the impossibility result of §4, the algorithm cannot be strategyproof.) It will be instructive to first consider the strict preference domain and rederive the PS mechanism as an iterative application of the parametric max-flow algorithm.

Another view of PS. Bogomolnaia and Moulin [3] proposed the PS algorithm for the special case in which each agent has strict preferences over the objects. In the PS algorithm, each object is assumed to have a size of 1 and is initially available for consumption; each agent consumes her best available object at a unit rate; and an object becomes unavailable for consumption, once a unit amount of that object has been eaten. This algorithm can be viewed as an iterative application of parametric maximum flow algorithm that terminates in (at most) n phases. The network structure remains the same as in Section 5, with the obvious modification that each agent now has only one outgoing arc—to her best available object; the capacities of the arcs from the source to the agents and the set of available objects vary from phase to phase. In phase k, the capacity of the arc from the source to agent $i \in N$ is $c(i,k) + \lambda$, where $c(i,k)$ is a constant determined at the end of the $(k-1)^{\text{st}}$ phase, and λ is the parameter. Let λ_k^* be the smallest breakpoint in the associated parametric maximum flow problem. Note that λ_k^* is the smallest value of λ for which some subset B of the available objects is completely consumed; this set B can be viewed as the bottleneck set of objects. If i's best object in this phase belongs to this set B, then then we give her $c(i,k) + \lambda_k^*$ units of this object, and set $c_{i,k+1} = 0$ (this is because i has no share over the other remaining objects as she has not yet started "eating" them); if not we set $c(i,k+1) = c(i,k) + \lambda_k^*$ (this is because i has already "eaten" this amount of her best available object at the end of this phase, and this object will still be available for consumption, possibly by other agents as well; in a sense, this is like pledging an amount of $c(i,k) + \lambda_k^*$ to agent i). The objects in B are removed from the set of available objects, and we move to the $(k+1)^{\text{st}}$ phase. Since $|B| \geq 1$, the procedure ends in at most n phases. We set $c(i,1) = 0$ for all $i \in N$, as initially none of the agents has started eating. We record these observations formally in Figure 1; its equivalence with the PS mechanism of Bogomolnaia and Moulin [3] is evident. Let $H(i, A')$ denote agent i's most preferred object in A' and let P be the assignment matrix.

PS

1. Initialize $A' = A$; $P \equiv 0$; $c(i,1) = 0$ for all $i \in N$; and $k = 1$.
2. Construct G with each agent $i \in N$ pointing to the object $H(i, A')$; every object in A' connected to a common sink through an arc of capacity 1 ; and every agent $i \in N$ connected from the source through an arc of capacity $c(i,k) + \lambda$.
3. Solve the corresponding parametric max-flow problem. Let λ_k^* be the smallest break point and B be the bottleneck set of objects.
4. For an agent i, if $H(i, A') \in B$ then update $P_{i,H(i,A')} = c(i,k) + \lambda_k^*$ and $c(i, k+1) = 0$; if not update $c(i, k+1) = c(i,k) + \lambda_k^*$.
5. Update $A' = A' - B$ and $k = k + 1$. If $A' \neq \phi$ then go to step 2. Else, terminate the algorithm.

Figure 1: An alternative view of the PS algorithm

Algorithm for the complete domain and its analysis. The main difficulty in extending the PS algorithm (as presented by Bogomolnaia and Moulin [3]) to the complete domain is as follows: if an agent is indifferent between many objects, it is not at all clear which of these objects she should consume, and at what rate. Consider an agent who is indifferent between two objects a and b, and suppose there are no other indifferences in the problem. This agent has three options: to eat a alone, to eat b alone, and to eat both of these objects. It is easy to construct examples in which each of these options is the unique solution. Also, note that in the last case, there are actually infinitely many possibilities because the rate at which the agent eats the two objects could be anything as long as they sum to 1. Thus, the "combinatorial" approach to this problem seems to present inherent difficulties. However, the alternative view of the PS mechanism—a minor change in our point of view—avoids these difficulties: the crucial step in the algorithm is solving the parametric max flow problem of an associated network, which can be done easily even when each agent points to (has arcs to) several objects. This algorithm, called EPS for "extended" PS, is simply a natural combination of the ideas in §5 and the alternative view of PS, and closely follows the algorithm described in Figure 1: the only change from this algorithm is in Step 2, where each agent now points to her most preferred *set* of objects. A formal description appears in Figure 2; as before, we let $H(i, A')$ be agent i's most preferred *set* of objects in A' and let P be the assignment matrix.

Note that in this algorithm, the most expensive computation is solving the parametric maximum flow problem. As noted before, the complexity of parametric maximum flow algorithm is $O(nm \ \log(n^2/m))$, where 'm' is the number of arcs in the network. Therefore, the overall complexity of the algorithm EPS is $O(\sum_{k=1}^{r} n^2 m_k \ \log(n^2/m_k))$, where m_k is the number of

EPS

1. Initialize $A' = A$; $P \equiv 0$; $c(i,1) = 0$ for all $i \in N$; and $k = 1$.
2. Construct G with each agent $i \in N$ pointing to the objects $H(i, A')$; every object in A' connected to a common sink through an arc of capacity 1 ; and every agent $i \in N$ connected from the source through an arc of capacity $c(i,k) + \lambda$.
3. Solve the corresponding parametric max-flow problem. Let λ_k^* be the smallest break point and B be the bottleneck set of objects.
4. For an agent i, if $H(i, A') \subseteq B$ then update $c(i, k+1) = 0$ and give her an amount $c(i,k)+\lambda_k^*$ from the set $H(a, A')$; if not update $c(i, k+1) = c(i,k)+\lambda_k^*$
5. Update $A' = A' - B$ and $k = k + 1$. If $A' \neq \phi$ then go to step 2. Else, terminate the algorithm.

Figure 2: The extended PS algorithm for the complete domain

arcs in the k^{th} phase and r is the total number of phases.

Properties. We now proceed to prove that the random assignment P generated by the algorithm EPS is ordinally efficient and envy-free. Let P be a random assignment matrix over the preference profile $\geq$. Given this, we define a binary relation $\tau(P, \geq)$ over the set of objects, A, as follows:

$$\forall a, b \in A : \ a\tau(P,\geq)b \iff \{ \exists\ i \in N : a \geq_i b \ \text{ and } \ p_{ib} > 0\} \tag{6}$$

The binary relation is *strict* if in the above expression $a >_i b$, that is, an agent i strictly prefers a to b and $p_{ib} > 0$. To save notation, we denote the above relation in short by $a\tau b$ when the random assignment matrix is clear from the context. The relation $\tau(P, \geq)$ is *cyclic* if there exists a cycle of relations $a^1\tau a^2, ..., a^{(r-1)}\tau a^r, ..., a^R\tau a^1$ with at least one of these binary relations being *strict*. We next characterize ordinally efficient assignments. (This result parallels the characterization of ordinaly efficiency in the strict preference domain due to Bogomolnaia and Moulin [3].)

Lemma 4 *The random assignment P is ordinally efficient w.r.t. the profile $\geq$ if and only if the relation $\tau(P, \geq)$ is acyclic.*

Proof. *Statement only if* :Suppose that the relation τ has a cycle: $a_1\tau a_2; a_2\tau a_3; ...; a_{R-1}\tau a_R; a_R\tau a_1$. We assume, without loss of generality, that the first relation is a strict one. By definition of τ, we can construct a sequence $i_1, ..., i_R$ in N such that: $p_{i_1,a_2} > 0$ and $a_1 >_{i_1} a_2; p_{i_2,a_3} > 0$ and $a_2 \geq_{i_2} a_3; ...; p_{i_R,a_1} > 0$ and $a_R \geq_{i_R} a_1$ (note that the agents $i_k, k = 1, ..., R$, may not

be all different). Choose $\delta > 0$ such that $\delta \leq p_{i_k a_{k+1}}$ for $k = 1, ..., R$ with the understanding $a_{R+1} = a_1$. Then, we define the matrix Q as follows: $Q = P + \Delta$ where $\delta_{i_k a_k} = \delta$; $\delta_{i_k a_{k+1}} = -\delta$ for $k = 1, ..., K-1$ and $\delta_{ia} = 0$ otherwise. By construction, Q is a bistochastic matrix. We go from P_{i_k} to Q_{i_k} by shifting some probability from object a_{k+1} to an object a_k, which is at least as good as a_{k+1} according to i_k. Moreover, this preference is strict for $k = 1$. Therefore, Q stochastically dominates P in a strict sense (if the same agent appears more than once, we use the transitivity of stochastic dominance). Hence, P cannot be ordinally efficient.

*Statement if :*Suppose Q stochastically dominates P in a strict sense wrt $\geq$. Let i_1 be an agent such that $Q_{i_1} \neq P_{i_1}$ (we can always do this because of definition of strict stochastic dominance). By definition of stochastic dominance, there exist two objects a_1, a_2 such that

$$a_1 >_{i_1} a_2; \;\; q_{i_1,a_1} > p_{i_1,a_1} \text{ and } q_{i_1,a_2} < p_{i_1,a_2} \tag{7}$$

In other words, $a_1 \tau(P, \geq) a_2$ in a *strict* sense. Next by feasibility of Q, there exists an agent i_2 such that $q_{i_2,a_2} > p_{i_2 a_2}$. Repeating the same argument, we find an a_3 such that $a_2 \geq_{i_2} a_3$ and $p_{i_2,a_3} > q_{i_2,a_3}$; and hence $a_2 \tau a_3$. Due to finiteness of A and N, we will eventually find a *cycle* of the relation τ. ■

Theorem 5 *The algorithm proposed is ordinally efficient.*

Proof. Suppose not. Then, by Lemma 4, we can find a cycle in the relation τ over the assignment matrix Q: $a^1 \tau a^2, .., a^r \tau a^{r+1}, .., a^R \tau a^1$. Without loss of generality, assume that the last binary relation is strict. Let i_r be an agent such that $a^r \geq_{i_r} a^{r+1}$ and $p_{i_r, a^{r+1}} > 0$ ($r \in 1, .., R$, with the convention $a^{R+1} = a^1$). Let s^r be the phase of the EPS algorithm during which a^r was in the bottleneck set. Since a^r is at least as good as a^{r+1} for the agent i_r, and i_r gets a positive amount of a^{r+1} , a^{r+1} could not have been in the bottleneck set before a^r. Therefore, $s^r \leq s^{r+1}$. Moreover, if a^r is strictly preferred over a^{r+1} , then $s^r < s^{r+1}$. Note that $s^{R+1} = s^1$.Therefore, $s^1 \leq s^2 \leq .... \leq s^R < s^1$. But, this is a contradiction because $s^0 = s^{R+1}$. Therefore, the algorithm is ordinally efficient. ■

Next, we prove the that assignment P found by algorithm EPS is envy-free.

Theorem 6 *The matrix P given by the algorithm EPS is envy-free.*

Proof. Let the preference profile be $\geq$. Consider any two agents $i, j \in N$ and an object $a \in A$. Define the set $M(i, a) = \{b : b \geq_i a\}$. Let k be the first phase of the algorithm by the end of which all the objects in $M(i, a)$ are used up. Note that in the first k phases, i will be allocated objects only from the set $M(i, a)$. Therefore, the number of units allocated to i from the objects in $M(i, a)$ is $\sum_{l \in M(i,a)} p_{i,l} = \lambda_1^* + ... + \lambda_k^* \geq \sum_{l \in M(i,a)} p_{j,l}$ (since j could have used some other objects outside the set $M(i, a)$). Therefore, i cannot envy j and hence EPS always finds an envy-free random assignment. ■

Illustrative example. Let $N = \{1, 2, 3\}$, $A = \{a, b, c\}$, and consider the following preference profile:

$$\begin{array}{cccc} 1 & \{a,b\} & & c \\ 2 & a & b & c \\ 3 & a & c & b \end{array}$$

Agent 1 is indifferent between a and b, but prefers both of these to c; agent 2 prefers a to b to c, and agent 3 prefers a to c to b. At the beginning of first phase, agent 1 is connected to both a and b; agents 2 and 3 are connected only to a. The smallest breakpoint in the parametric maximum flow problem is $\lambda_1^* = \frac{1}{2}$. The corresponding minimum cut is $s \cup \{2, 3\} \cup \{a\}$ (following the notation of section 5). We update $p_{2,a} = \frac{1}{2}$; $p_{3,a} = \frac{1}{2}$; $c(1, 2) = \frac{1}{2}$; $c(2, 2) = 0$; $c(3, 2) = 0$; $A' = \{b, c\}$. At this stage, a is fully consumed. In the second phase agents 1 and 2 are connected to b and agent 3 is connected to c. The smallest breakpoint is $\lambda_2^* = \frac{1}{4}$ and the corresponding minimum cut is $s \cup \{1, 2\} \cup \{b\}$. We update $p_{1,b} = \frac{3}{4}$; $p_{2,b} = \frac{1}{4}$; $c(1, 3) = 0$; $c(2, 3) = 0$; $c(3, 3) = \frac{1}{4}$;$A' = \{c\}$. At this stage, both a and b are totally consumed. In the third phase all the agents are connected to c. The smallest breakpoint is $\lambda_3^* = \frac{1}{4}$ and the corresponding minimum cut is $s \cup \{1, 2, 3\} \cup \{c\}$. We update $p_{1,c} = \frac{1}{4}$; $p_{2,c} = \frac{1}{4}$; $p(3, c) = \frac{1}{2}$. At this stage, all the objects are fully consumed and the algorithm ends. The resulting random assignment is

$$\begin{array}{c|ccc} & a & b & c \\ \hline 1 & 0 & 3/4 & 1/4 \\ 2 & 1/2 & 1/4 & 1/4 \\ 3 & 1/2 & 0 & 1/2 \end{array} \tag{8}$$

Variable eating rates. In the EPS algorithm, all the agents have same eating rates. Suppose that each agent $i \in N$ has a different eating rate function $w_i(\cdot)$, where $\int_{t=0}^{t=1} w_i(t)dt = 1$. The EPS algorithm can be modified in a simple way to accommodate for these different eating rate functions. The only modification required is that the parametric maximum flow problem needs to be redefined appropriately, which we do here. We refer to this algorithm as EER. Let t_k be the starting time of the k^{th} phase. The capacity of the arc from the source to agent $i \in N$ is $c(i, k) + \int_{t_k}^{t} w_i(u)du$, where $c(i, k)$ is a constant determined at the end of the $(k-1)^{st}$ phase, and each agent points to her most preferred set of objects. In this problem the parameter is $t \geq t_k$. The smallest breakpoint, in this context, is the value of t after which the maximum flow stops increasing linearly with time. A method for doing this was described in Gallo, Grigoriadis and Tarjan [8]. We denote the smallest breakpoint by t_{k+1} and the corresponding bottleneck by B. If i's preferred set of objects is a subset of B, then we give her $c(i, k) + \int_{t_k}^{t_{k+1}} w_i(u)du$ amount from these objects (this can be done, as before, based on the maximum flow solution at the breakpoint t_{k+1}) and also update $c(i, k + 1) = 0$; if not we update $c(i, k + 1) = c(i, k) + \int_{t_k}^{t_{k+1}} w_i(u)du$. Since $|B| \geq 1$, this procedure also has to end in at most n phases. As before, $c(i, 1) = 0$ and $t_1 = 0$.

Let P_{EER} denote the random assignment obtained from the EER algorithm. Then P_{EER} is ordinally efficient. This can be seen from the proof of Theorem 5 . Notice that the proof

relies only on the phase at which an object is in the bottleneck set; and not upon the rate at which it is consumed. So, essentially the same argument goes through. In fact, more is true: every ordinal efficient can be found in this way; we omit the straightforward proof. (This result parallels a result of Bogomolnaia and Moulin [3] for the strict preference domain.)

Theorem 7 *Fix a preference profile $\geq$. Then, a random assignment matrix P is ordinally efficient if and only if there exists a profile of eating speeds $w = (w_i)_{i \in N}$ such that $P = P_{EER}$.*

References

[1] A.Abdulkadiroglu and T.Sonmez (1998), "Random serial dictatorship and the core from random endowments in house allocation problems", *Econometrica*, 66, 689–701.

[2] Ahuja, Magnanti and Orlin ," Network Flows: Theory, Algorithms, and Applications", Prentice Hall, 1993.

[3] A.Bogomolnaia and H. Moulin (2001), "A new solution to the Random Assignment problem", *Journal of Economic Theory*, 100, 295–328.

[4] A. Bogomolnaia and H. Moulin (2002), "A Simple Random Assignment Problem with a Unique Solution", *Economic Theory*, 19, 623–636.

[5] J.R.Brown (1979),"The sharing problem", *Oper. Res.* , 27, 324–340.

[6] H. Cres and H. Moulin (2001), "Scheduling with Opting Out: Improving upon Random Priority", *Operations Research*, 49(4):565–576.

[7] D. Gale (1987), "College course assignments and optimal lotteries", mimeo, University of California, Berkeley, 1987.

[8] Giorgio Gallo, Michael D Grigoriadis and Robert E Tarjan (1989),"A Fast Parametric Maximum Flow Algorithm and Applications", *SIAM J. Comput.* , 18, 30–55.

[9] A.V.Goldberg and R.E.Tarjan , "A new approach to the maximum flow problem", Proc. 18h Annual ACM Symposium on Theory of Computing, 1986, 136–146; *J. Assoc. Comput. Mach.*, 35 (1988).

[10] D.Gusfield (1987),"Computing the strength of a network in $O(|V|^3|E|)$ time", Tech. report CSE-87-2, Department of Electrical and Computer Engineering, University of California, Davis, CA.

[11] A.Hylland and R.Zeckhauser (1979), "The efficient allocation of individuals to positions", *J. Polit. Econ.* , 91, 293–313.

[12] T.Ichimori, H.Ishii and T.Nishida (1982),"Optimal sharing", *Math. Programming*, 23, 341–348.

[13] A.Itai and M.Rodeh (1985),"Scheduling transmissions in a network", *J. Algorithms*, 6, 409–429.

[14] N. Megiddo (1974),"Optimal flows in networks with multiple sources and sinks", *Math. Programming*, 7, 97–107.

[15] N. Megiddo (1979),"A good algorithm for lexicographically optimal flows in multi-terminal networks", *Bull. Amer. Math. Soc.*, 83, 407–409.

[16] A. E. Roth (1982), "Incentive compatibility in a market with indivisibilities," *Economics Letters*, 9, 127–132.

[17] A. E. Roth and A. Postlewaite (1977), "Weak versus strong domination in a market with indivisible goods," *Journal of Mathematical Economics*, 4, 131–137.

[18] L. Shapley and H. Scarf (1974), "On cores and indivisibility," *Journal of Mathematical Economics*, 1, 23–28.

[19] L. Zhou (1990), "On a conjecture by Gale about one-sided matching problems," *Journal of Economic Theory*, 52, 123–135.