Simultaneously Achieving Ex-ante and Ex-post Fairness

Haris Aziz[(✉)]

UNSW Sydney and Data61 CSIRO, Sydney, Australia
`haziz@cse.unsw.edu.au`

Abstract. We present a polynomial-time algorithm that computes an ex-ante envy-free lottery over envy-free up to one item (EF1) deterministic allocations. It has the following advantages over a recently proposed algorithm: it does not rely on the linear programming machinery including separation oracles; it is SD-efficient (both ex-ante and ex-post); and the ex-ante outcome is equivalent to the outcome returned by the well-known probabilistic serial rule. As a result, we answer a question raised by Freeman, Shah, and Vaish (2020) whether the outcome of the probabilistic serial rule can be implemented by ex-post EF1 allocations. In the light of a couple of impossibility results that we prove, our algorithm can be viewed as satisfying a maximal set of properties. Under binary utilities, our algorithm is also ex-ante group-strategyproof and ex-ante Pareto optimal. Finally, we also show that checking whether a given random allocation can be implemented by a lottery over EF1 and Pareto optimal allocations is NP-hard.

1 Introduction

Who gets what is a significant and ubiquitous issue. When making any kind of allocation among self-interested agents, fairness is an important concern. Does a fair allocation exist? Is there an efficient algorithm to compute such an allocation? These are important questions that have been studied in fair division for decades. In this paper, we consider the issue of finding probabilistic allocations that are ex-ante and ex-post fair.

Suppose there are two agents who have additive utilities over three items a, b, c. Both agents have the highest value for items a, then b, and then c. From an ex-ante perspective, envy-freeness can be achieved by giving each item to each agent with probability half. However, there are many ways to achieve this expected probability, some perhaps not too fair. For example, the uniform lottery over the following two deterministic allocations: $(\{a, b, c\}, \emptyset)$ and $(\emptyset, \{a, b, c\})$. It may be desirable to achieve both ex-ante envy-freeness and some weaker form of ex-post envy-freeness. For example a uniform lottery over the following allocations is fairer ex-post: $(\{a\}, \{b, c\})$ and $(\{b, c\}, \{a\})$.

As seen from the example above, achieving target fairness properties is easy when we consider fractional outcomes or view outcomes from an ex-ante perspective. Implementing such desirable ex-ante outcomes by randomizing over

desirable deterministic outcomes can pose interesting challenges (see, e.g. [1,11]). This issue was explored by Freeman et al. [15]. They focussed on ex-ante envy-freeness and ex-post envy-freeness up to one item as the target fairness requirements. Both of the properties are known to be individually achievable. An ex-ante envy-free random allocation always exists (for example the outcome of the probabilistic serial rule [8] achieves ex-ante envy-freeness). Similarly, a deterministic envy-free up to one item (EF1) allocation always exists [10]. For example, running the round robin sequential algorithm obtains an EF1 allocation [12]. Freeman et al. [15] explore the question of achieving ex-ante envy-freeness and ex-post EF1 *simultaneously*. They showed that there exists a polynomial-time algorithm to compute a lottery over envy-free up to one item allocations that is also ex-ante envy-free.[1]

The inventive polynomial-time algorithm of Freeman et al. [15] has a couple of possible limitations. Firstly, it requires using the machinery of linear programming separation oracles. It may be desirable to get similar results by simpler combinatorial algorithms. Secondly, the algorithm of Freeman et al. is not ex-post weakly SD (stochastic dominance)-efficient and hence not ex-ante weakly SD-efficient. This is evident from Example 2 of Freeman et al. where they note that their algorithm does not satisfy ordinal efficiency.[2] The fact that an algorithm is not ex-post weakly SD-efficient implies that it can return a deterministic allocation such that there exists another deterministic allocation that gives each agent strictly more utility for all utility functions consistent with the underlying ordinal preferences. Another implication of violating ex-post weak SD-efficiency is that all the agents can trade one of their items for another item to get more utility. Such unamiguous compromise on welfare can be undesirable. For example, the random serial dictatorship rule (which is ex-post SD-efficient) has received criticism that it is not ex-ante SD-efficient [8].

We overcome the two limitations discussed above and show that the algorithmic result of Freeman et al. [15] can be achieved in a relatively simpler and faster way while additionally satisfying SD-efficiency. To the best of our knowledge, our is the first algorithm to simultaneously satisfy weak SD-efficiency, ex-ante EF, and ex-post EF1. The latter two guarantees even hold for all additive utilities consistent with the agents' underlying ordinal preferences. In other words, our algorithm satisfies ex-ante SD-envy-freeness and ex-post SD-EF1. We also show how the algorithm can be further modified by using parametric network flows to additionally achieve both ex-ante and ex-post SD-efficiency. Our results can be viewed as being optimal in the view of the following two impossibility results that we prove. Firstly, ex-ante SD-envy-freeness, ex-post EF1, and

---

[1] Freeman et al. [15] also presented several other results charting the landscape of possibility and impossibility results when considering fairness and efficiency properties ex post and ex-ante. In particular, they study in detail the rule that maximizes ex-ante Nash welfare. However, they show that the rule cannot be implemented by EF1 allocations.

[2] SD-efficiency is also referred to as ordinal efficiency in the literature [8].

ex-post Pareto optimality are incompatible. Secondly, ex-ante Pareto optimality and ex-ante SD-envy-freeness are incompatible.

Our algorithm calls the probabilistic serial algorithm as well as the Birkhoff's decomposition algorithm as subroutines. Freeman et al. raised the question whether the outcome of the probabilistic serial algorithm can be implemented using ex-post EF1 randomized allocations: "*we were not able to determine whether the fractional allocation produced by probabilistic serial can always be implemented using an ex-post EF1 randomized allocation.*" We answer the question in the affirmative: our algorithm's outcome is ex-ante equivalent to the outcome of the probabilistic serial rule. In particular, it can be viewed as a desirable way to instantiate the probabilistic serial outcome. Under binary utilities, our algorithm is group-strategyproof, ex-ante efficient, ex-ante envy-free, and ex-post EF1. Finally, we also show that checking whether a given random allocation can be represented over a lottery over EF1 and Pareto optimal allocations is NP-hard.

2 Preliminaries

An allocation problem is a triple (N, O, u) such that $N = \{1, \ldots, n\}$ is the set of agents, $O = \{o_1, \ldots, o_m\}$ is the set of objects, and u specifies an additive utility function $u_i : O \to \mathbb{R}^+$. The utility function profile u induces the preference profile $\succsim = (\succsim_1, \ldots, \succsim_n)$ which specifies for each agent i his preferences $\succsim_i$ over objects in O such that $o \succsim_i o'$ if and only if $u_i(o) \geq u_i(o')$. We use $\succ_i$ for the strict part of $\succsim_i$, i.e., $o \succ_i o'$ iff $o \succsim_i o'$ but not $o' \succsim_i o$. A random allocation p is a $(n \times m)$ matrix $[p_{i,o_j}]$ such that $p_{i,o_j} \in [0, 1]$ for all $i \in N$, and $o_j \in O$; and $\sum_{i \in N} p_{i,o_j} = 1$ for all $o_j \in O$. For a given set $S \subset N$, we will refer by $\succsim_S$ the preference profile restricted to agents in S.

The value p_{i,o_j} represents the probability of object o_j being allocated to agent i. Each row $p_i = (p_{i,o_1}, \ldots, p_{i,o_m})$ represents the allocation of agent i. The set of columns correspond to the objects $o_1, \ldots, o_m$. A feasible random allocation is *deterministic* if $p_{i,o} \in \{0, 1\}$ for all $i \in N$ and $o \in O$. When we say 'an allocation', we will mean random allocation unless we specially specify it is deterministic.

For any agent $i, j \in N$ and an allocation p, the utility of agent i for a bundle p_j is $u_i(p_j) = \sum_{o \in O} p_{j,o} u_i(o)$. Given two random allocations p and q, $p_i \succsim_i^{SD} q_i$ that is, an agent i *SD prefers* allocation p_i to allocation q_i if $\sum_{o_j \in \{o_k : o_k \succsim_i o\}} p_{i,o_j} \geq \sum_{o_j \in \{o_k : o_k \succsim_i o\}} q_{i,o_j}$ for all $o \in O$. We write $p_i \succ_i^{SD} q_i$ if $p_i \succsim_i^{SD} q_i$ and not $q_i \succsim_i^{SD} p_i$.

Fairness Properties. A random allocation p is *SD-envy-free* if for all $i, j \in N$, $p_i \succsim_i^{SD} p_j$. An random allocation p is *envy-free (EF)* if $u_i(p_i) \geq u_i(p_j)$ for all $i, j \in N$. For an agent's allocation p_j, we will denote by p_j^{-o} the allocation p_j in which $p_{j,o}$ is set to 0. For an agent's allocation p_j and $S \subseteq O$, we will denote by p_j^{-S} the allocation p_j in which $p_{j,o}$ is set to 0 for all $o \in S$. A random allocation p is *SD-EF1* if for all $i, j \in N$, either $p_i \succsim_i^{SD} p_j$ or $p_i \succsim_i^{SD} p_j^{-o}$ for some o. o. A random allocation p is *envy-free up to k items (EFk)* if there exist some $S \subset O$

such that $|S| \leq k$ such that $u_i(p_i^{-S}) \geq u_i(p_j^{-S})$. Note that SD-envy-freeness implies envy-freeness which implies EFk. And SD-EF implies SD-EFk.

A given random allocation can be implemented by a lottery over deterministic allocations.[3] We call the latter an implementation of the given random allocation. We say that random allocation p satisfies a property X *ex-ante* if the fractional allocation representing p satisfies property X. When we discuss the ex post properties of a random allocation p, we will also need to consider the lottery implementation over deterministic allocations which achieves the random allocation p. In that case we say that random assignment with a lottery implementation deterministic allocations over $M_1, \ldots, M_K$ satisfies property X *ex-post* if $M_1, \ldots, M_K$ satisfy property X. Therefore for any given property for allocations, we consider it ex-ante as well as ex-post. Figure 1 shows the key fairness concepts that are appropriate from ex-ante and ex-post perspectives. Note that we do not focus ex-post envy-freeness since a deterministic envy-free allocation is not guaranteed to exist. Furthermore, checking whether a deterministic envy-free allocation exists is NP-complete even for 1-0 utilities [2].
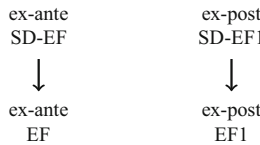
$$
\begin{array}{ccc}
\text{ex-ante} & & \text{ex-post} \\
\text{SD-EF} & & \text{SD-EF1} \\
\downarrow & & \downarrow \\
\text{ex-ante} & & \text{ex-post} \\
\text{EF} & & \text{EF1}
\end{array}
$$

Fig. 1. Logical relations between fairness concepts.

Example 1. Consider the example in which $N = \{1, 2\}$, $O = \{a, b, c, d\}$ and the agents have the following utilities over four items.

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

Then, the following is one possible random allocation.

$$
p = \begin{array}{c} 1 \\ 2 \end{array} \begin{pmatrix} \begin{array}{cccc} a & b & c & d \end{array} \\ 1/2 \ \ 1 \ \ 0 \ \ 1/2 \\ 1/2 \ \ 0 \ \ 1 \ \ 1/2 \end{pmatrix}
$$

In the allocation, $u_1(p_1) = \frac{1}{2}(4) + 1(3) + \frac{1}{2}(1) = 5.5$ and $u_1(p_2) = \frac{1}{2}(4) + 1(2) + \frac{1}{2}(1) = 4.5$. Hence agent 1 is not envious of agent 2.

Allocation p can be implemented by the following uniform lottery over two deterministic allocations as follows.

---

[3] The statement follows from the well-known Carathéodory's Theorem.

$$p \;=\; \tfrac{1}{2} \;\begin{matrix}1\\2\end{matrix}\begin{pmatrix} \overset{a}{1} & \overset{b}{1} & \overset{c}{0} & \overset{d}{0} \\ 0 & 0 & 1 & 1 \end{pmatrix} + \tfrac{1}{2}\;\begin{matrix}1\\2\end{matrix}\begin{pmatrix} \overset{a}{0} & \overset{b}{1} & \overset{c}{0} & \overset{d}{1} \\ 1 & 0 & 1 & 0 \end{pmatrix}$$

We say that a deterministic allocation q is *consistent* with a random allocation p if for each $q_{i,o} = 1$, we have that $p_{i,o} > 0$. For $n = m$, a deterministic allocation can be represented by a permutation matrix in which an entry of one denotes the row agent getting the column object. A *decomposition* of a random allocation p is a sum $\sum_{i=1}^{k} \lambda_i P_i$ such that $\lambda_i \in (0,1]$ for $i \in \{1, \ldots, k\}$, $\sum_{i=1}^{k} \lambda_i = 1$, and each P_i is a permutation matrix (consistent with p).

3   The PS-Lottery Algorithm

In this section, we present our main algorithm that we refer to as the PS-Lottery Algorithm. Before we proceed, we summarize two well-known algorithms that we will use as building blocks for our algorithm to simultaneously achieve ex-ante EF and ex-post EF1.

Probabilistic Serial (PS) Algorithm. The PS rule [8] takes as input the strict ordinal preferences of agents over items as well as the available amounts of each of the items. Agents start eating their most preferred item at unit speed until the item is consumed. They continue eating their most preferred items until all the items are consumed. The outcome is a random allocation in which each agent's probability of getting an item is the fraction of the item that she ate. Initially, only presented for the case of single-unit demands, the rule extends seamlessly for the case where agents want to get multiple items [21]. Although described as a continuous rule where agents eat infinitesimal amounts, the PS outcome can be computed by a discrete algorithm in polynomial time $O(nm)$ (see the appendix).

Birkhoff's Algorithm. Consider any random allocation with n agents and n items in which each agent gets one unit of items. Birkhoff's algorithm can decompose such a random allocation (which can be represented by a bistochastic matrix) into a convex combination of at most $n^2 - n + 1$ deterministic allocations (represented by permutation matrices) [7,22]. The following is a description of Birkhoff's algorithm. We initialize i to 1. For a bistochastic matrix M, a permutation matrix P_i consistent with M is guaranteed to exist. Such a permutation matrix corresponds to a perfect matching in a bipartite graph $(N \cup O, E)$ where $(i, o) \in E$ iff $M_{i,o} > 0$. Such a perfect matching and hence the permutation matrix can be computed via the Hopcroft-Karp-Karzanov algorithm which takes time $O(n^{2.5})$ [18,19]. We initialize index i to 1. M is set to $M - \lambda_i P_i$ where $\lambda_i \in (0,1]$ is the smallest non-zero entry in P_i. Index i is incremented by one. The updated M is again bistochastic. The process is repeated (say $k-1$ times) until M is the zero matrix. Then $M = \sum_{i=1}^{k} \lambda_i P_i$.

Now that we have defined the two algorithms, we are in a position to present Algorithm 1. The high-level description of the algorithm is as follows. We first add some dummy items to ensure that there are nc items. The expanded set of items is called O'. We then simulate PS. We track information about how much of each item has been eaten at time steps $1, \ldots, c$. We use this information to form an allocation q' of items in O' to agents in $N' = \{i_1, \ldots, i_c : i \in N\}$. The agents $i_1, \ldots, i_c$ are called the representative agents of each agent i. An agent i_j's allocation is what agent i ate in time interval $[j-1, j]$. Allocation q' can be represented by a bistochastic matrix. We decompose q' into a convex combination of permutation matrices via Birkhoff's algorithm. The permutation matrices are suitably modified to remove the dummy items and also give the allocation of all representatives to the agent they represent. The convex combination over the modified permutation matrices gives us the desired solution, which is both ex-ante EF and ex-post EF1.

Algorithm 1. PS-Lottery Algorithm

Input: $I = (N, O, \succsim)$ where $|N| = n$, $|O| = m$ and $c = \lceil m/n \rceil$.

Output: EF fractional allocation $q = \sum_{j=1}^{K} \lambda_j P_i$ where each P_j represents a deterministic EF1 allocation and $K \leq (cn)^2 - 2cn + 2$.

1: If m is a multiple of n, $D = \emptyset$. Else, $D = \{d_1, \ldots, d_{nc-m}\}$.
2: $O' \leftarrow O \cup D$ so that $|O'| = cn$.
3: $N' = \{i_1, \ldots, i_c : i \in N\}$. The agents $i_1, \ldots i_c$ are termed as the representatives of agent i.
4: Set preference profile $\succsim'$ of agents in $N' \cup N$ as follows: for all $o, o' \in O$ and for all i_j for $j \in \{1, \ldots, c\}$, $o \succsim'_{i_j} o'$ iff $o \succsim_i o'$. For all $o \in O$ and $d \in D$, $o \succ'_{i_j} d$. All the ties in $\succsim'$ are broken lexicographically.
5: Run PS on instance $(N, O', \succsim'_N)$ to get a random outcome r.
6: For each bundle r_i, let agent i re-eat her bundle at unit-speed according to preferences of her representative agents $\succsim'_{i_k}$ with each representative agent i_j eating on behalf of agent i in time interval $[j-1, j]$. Let the result of this eating be allocation q' which is an allocation of items O' to agent representatives in N'.
7: For the (bistochastic) matrix corresponding to q', compute a Birkhoff decomposition $q' = \sum_{j=1}^{K} \lambda_j P_j'$ where $K \leq (cn)^2 - 2cn + 2$.
8: Convert $q' = \sum_{j=1}^{K} \lambda_j P_j'$ into $q = \sum_{j=1}^{K} \lambda_j P_j$ where all the dummy items are ignored and each agent gets the allocation of its representatives.
9: **return** Allocation q for instance I and its decomposition $\sum_{j=1}^{K} \lambda_j P_j$.

Before we prove the main properties of the PS-Lottery Algorithm, we recall a class of deterministic allocation algorithms. The *sequential allocation* algorithm takes as input a sequence π of turns of the agents and returns a deterministic allocation which is a result of agents picking a most preferred unallocated item

in their turn. A sequence of turns is called *recursively balanced (RB)* if at each prefix, all agents have the same number of turns, or differ by one [5]. An RB sequence can be viewed as agents coming in c rounds. Note that $cn \leq (m+n)$. In each round except the last one, each agent gets exactly one turn. Since each agent weakly prefers her picked item over all items picked in later rounds, it can easily be proved that the outcome of sequential allocation with an RB sequence is EF1 [3].[4] Since sequential allocation with an RB sequence only uses ordinal preferences of the agents, it is EF1 with respect to all positive utilities consistent with the ordinal preferences [3] and hence SD-EF. An allocation is called an *RB-allocation* if it is an outcome of sequential allocation with respect to some RB-sequence. We will use the perspective of RB-allocations to establish that our algorithm returns a lottery over EF1 allocations.

Theorem 1. *Let $c = \lceil m/n \rceil$. Algorithm 1 is polynomial-time algorithm that takes time $O((cn)^4)$ that computes a lottery over at most $(cn)^2$ deterministic EF1 allocations that is equivalent to the outcome of the probabilistic serial algorithm.*

Proof. Algorithm 1 works as follows. If $m < n$, we set $D = \{d_1, \ldots, d_{n-m}\}$. If $m > n$, we set $D = \{d_1, \ldots, d_{cn-m}\}$. We are now in a position to fix a new allocation instance $I' = (N', O', \succeq')$ that only uses ordinal preferences. The item set O' is $O \cup D$ where $|O'| = cn$. The 'representative' set N' is $\{i_1, \ldots, i_c : i \in N\}$. Note that the number of representatives $|N'|$ is equal to the number of items $|O'|$. The preferences are consistent with the underlying preference profile. The preferences $\succeq'$ of the representatives are set as follows: for all $o, o' \in O$ and for all i_j for $j \in \{1, \ldots, c\}$ $o \succeq'_{i_j} o$ iff $o \succeq_i o$. For all $o \in O$ and $d \in D$, $o \succ'_{i_j} d$. All the ties in $\succeq'$ are broken lexicographically.

Note that for the modified allocation problem instance I', an allocation has a corresponding allocation in the original instance I: an agent i gets all the allocations of its representatives $i_1, \ldots i_c$. The allocation of dummy items is ignored.

Let q' be the allocation as a result of applying PS with agent set N and item set O', but for each $j = 0$ to $c - 1$, we change the name of each agent i to i_{j+1} in time interval $[j, j+1]$. Note that computing r and q' takes time $(cn)^2$. The allocation has a corresponding bistochastic matrix in which the rows correspond to the representatives and the columns correspond to the items. Each entry in the matrix represents the amount of the corresponding item eaten by the corresponding representative.

Note that since q' is bistochastic, a permutation matrix P'_k consistent with q' exists by Birkhoff's theorem. We want to show that any such matrix P'_k must correspond to an RB-allocation of items in O' to agents in N. The RB-allocation is viewed as proceeding in rounds. In each round, each of the representatives representing the n agents pick a most preferred available item. In the j-th round, the representatives involved are $1_j, \ldots, n_j$. In any P'_k, each item is allocated to an agent representative and each agent representative gets one item. In order to

---

[4] In fact an RB allocation satisfies a stronger properly called strong EF1. Stronger EF1 requires that upon removing the same item from agent i's bundle, no other agent j envies i, for all i and j. The property was proposed by Conitzer et al. [13].

establish that P'_k is an RB-allocation of N, it is sufficient to prove two claims:
(1) no representative agent strictly prefers any item picked in a later round; and
(2) within each round, the items allocated to the representative agents are as a
result of sequential allocation.

Claim (1) follows from the fact that no representative i_j strictly prefers any
item allocated in a later round. The reason is that when it stopped eating in its
turn, it was always eating an item at least as preferred as in later rounds.

Next, we prove Claim (2). Consider any round in which each representative
receives one item. We claim that no set of representatives want to reallocate
the items given in that round to get an improvement for all representatives
in the set. Suppose for contradiction there is a trading cycle in which every
agent in the cycle improves: $o_1, 1, o_2, 2, \ldots, o_j, j$. Representative 1 prefers item
o_2 over o_1 which means that it started eating o_1 after o_2 was finished. Since 1
ate a strictly positive fraction of o_1, it implies that o_1 finishes strictly after o_2.
By a similar argument each $i \in \{1, \ldots j - 1\}$ wants to get o_{i+1} which means
that it started eating o_i after o_{i+1} was finished. Agent j prefers item o_1 over
o_j which means that it started eating o_j after o_1 was finished which means
that o_j finishes strictly after o_1. But then the order of the items according to
the finishing times is: $o_1, o_j, o_{j-1}, \ldots, o_3, o_2, o_1$. We have shown that o_1 has two
different finishing times which is a contradiction. Since there exists no trading
cycle for representatives in the same round, we know that the items in the round
can be allocated as a result of sequential allocation.

From the two claims above, the allocation P'_k is an RB-allocation for agents in
N if each agent gets the allocations of its representatives. Since any permutation
matrix consistent with q' also corresponds to an RB-allocation, we can use P'_k
as one of the permutation matrices in which q' is decomposed during Birkhoff's
decomposition. We can continue decomposing q' into permutation matrices until
we can represent q' by a convex combination of at most $K \leq (cn)^2$ permuta-
tion matrices $P'_1, \ldots, P'_K$. Each permutation matrix in the decomposition can be
computed by computing a perfect matching in a corresponding bipartite graph
via the Hopcroft-Karp-Karzanov algorithm which takes time $O((cn)^{2.5})$.

Finally, note that we can convert allocations $(q', P'_1, \ldots, P'_K)$ for instance I'
into the corresponding allocations $(q, P_1, \ldots, P_K)$ for instance I. We do so by
removing the dummy items and for each $i \in N$, giving the allocations of all the
representatives $i_1, \ldots, i_c$ to agent i. Note that q is the outcome of running PS
on instance I. Also, $P_1, \ldots, P_K$ are RB-allocations for instance I and hence EF1
for instance I.   □

Remark 1. Algorithm 1 is combinatorial algorithm that computes a lottery over
at most $(cn)^2 \leq (m + n)^2$ deterministic allocations. By Carathéodory's Theo-
rem, any $n \times m$ random allocation that is represented by a convex combination
of a given K deterministic allocations, can be represented by at most $nm + 1$
deterministic allocations among the K deterministic allocations. We can reduce
the support of the lottery returned by Algorithm 1 to one involving at most
$nm+1$ deterministic EF1 and SD-efficient allocations as follows. By using Gaus-
sian elimination, we compute the subset of the set of matrices $\{P_1, \ldots, P_k\}$ that

forms the basis of $P_1, \ldots, P_k$. We can compute a convex combination of the matrices in the basis to achieve the same outcome q.

We note that whereas our algorithm provides a way to implement PS by EF1 allocations, not every implementation of the PS outcome may satisfy ex-post EF1. For example, consider the case of two agents with identical preferences over two items. In that case, tossing a coin and then giving both items to one agent is ex-ante equivalent to the PS outcome. However, it is not EF1 if agents have strictly positive utilities for both items.

Algorithm 1 bears similarities to the exponential-time Algorithm 1 (Recursive PS) of Freeman et al. [15]. Just like their algorithm, we make agents successively eat one unit of items. Unlike the algorithm of Freeman et al., we derive the lottery decomposition only after the PS outcome has been computed. In contrast, Freeman et al. probabilistically generate a partial deterministic allocation after each unit time. Their algorithm "branches out into a polynomial number of subinstances" a polynomial number of times which makes it an exponential-time algorithm. In order to ensure polynomial-time computability, they resort to a result about convex polytopes and separation oracles [16].

4   Additionally Achieving Efficiency

In this section, we consider the additional issue of efficiency. Before, we proceed, we present some definitions.

Efficiency Properties. A random allocation p is *fractional Pareto optimal (fPO)* if there exists no other random allocation q such that $u_i(q_i) \geq u_i(p_i)$ for all $i \in N$ and $u_i(q_i) > u_i(p_i)$ for some $i \in N$. A deterministic allocation p is *Pareto optimal (PO)* if there exists no other deterministic allocation q such that $u_i(q_i) \geq u_i(p_i)$ for all $i \in N$ and $u_i(q_i) > u_i(p_i)$ for some $i \in N$. A random allocation p is *SD-efficient* is there exists no random allocation q such that $q_i \succsim_i^{SD} p_i$ for all $i \in N$ and $q_i \succ_i^{SD} p_i$ for some $i \in N$. An allocation p is *weakly SD-efficient* is there exists no allocation q such that $q_i \succ_i^{SD} p_i$ for all $i \in N$. Note that fPO implies PO which implies SD-efficiency which in turn implies weak SD-efficiency. Just as in the case of fairness, we will consider efficiency of both the ex-ante random allocation as well as efficiency properties of the ex-post deterministic allocations that are involved in the lottery.

We note that the random allocation maximizing the Nash social welfare is well-known to be equivalent to the competitive equilibrium with equal incomes solution (see e.g., [14,23]) and satisfies fPO as well as ex-ante envy-freeness. However, due to Theorem 3 of Freeman et al. [15], a rule that is fPO and ex-ante envy-free cannot be ex-post EF1.

Since the outcome returned by Algorithm 1 is a lottery implementation of the PS rule outcome, our algorithm also inherits all the desirable ex-ante properties that the PS rule and its outcome are known to satisfy. Note that Algorithm 1 first breaks ties in the ordinal preferences before running the PS algorithm.

This results in the outcome satisfying weak SD-efficiency rather than SD-efficiency if there are indeed ties in the original preferences. If we care about SD-efficiency, then we do not artificially break any ties and can run the extended probabilistic serial (EPS) algorithm [20]. The EPS algorithm makes coordinated choices for agents to eat one of their most preferred items and uses parametric network flows to compute the outcome. For number of items $m \geq n$, the algorithm takes time $O(m^3 \log m)$.[5]

The exact specification of our *EPS-Lottery algorithm* is to take the PS-Lottery algorithm and replace Step 5 with the following step: Run EPS on instance $(N, O', \succsim_N'')$ to get a random outcome r. Here, the preference profile $\succsim_N''$ is the same as $\succsim'$ except that only ties within D are broken lexicographically and ties are within O are not broken. Therefore the returned outcome r and hence q' is SD-efficient rather than just weak SD-efficient. The argument of implementing the outcome with EF1 deterministic allocations remains unchanged. The running time is unchanged as well as the bottleneck step is to compute a Birkhoff decomposition which takes time $O((cn)^4)$.

Note that if a random allocation q is SD-efficient, then in any decomposition of q, each of the deterministic allocations is SD-efficient as well. The reason is that if one of the deterministic allocations is not SD-efficient, then q is not SD-efficient. Hence, our algorithm additionally achieves SD-efficiency both ex-ante and ex-post.

Theorem 2. *Let $c = \lceil m/n \rceil$. The EPS-Lottery Algorithm runs takes time $O((cn)^4)$ and computes a lottery over at most $(cn)^2 \leq (m+n)^2$ deterministic EF1 allocations that is equivalent to the outcome of the extended probabilistic serial algorithm (which is SD-envy-free and SD-efficient).*

We note that our algorithm does not achieve ex-post Pareto optimality. One approach to achieving ex-post PO and ex-post EF1 is to check certain random allocations for these properties. Next, we show for an arbitrary random allocation, checking whether it is ex-post EF1 and ex-post Pareto optimal is NP-hard.

Theorem 3. *For n agents and n items, checking whether a given random allocation can be implemented by a lottery over EF1 and Pareto optimal allocations is NP-hard. For n agents and n items, checking whether a given random allocation can be implemented by a lottery over SD-EF1 and Pareto optimal allocations is NP-hard.*

Proof. It was proved that for n agents and n items, checking whether a given random allocation can be implemented by a lottery over balanced Pareto optimal allocations is NP-hard [4]. Their setting assumed ordinal preferences but it works as well for any cardinal preferences consistent with the ordinal preferences.

---

[5] The original EPS algorithm [20] is presented for the case of single-unit demands. However, it can easily be extended to the case of multiple items (see e.g., the Controlled Cake Eating Algorithm (CCEA) algorithm [6]). CCEA is described in the context of cake cutting with piecewise constant valuations. It also applies to allocation of items: each cake segment can be treated as a separate item.

We consider utility functions u_i consistent with ordinal preference $\succ_i$ and assume that $u_i(o) > 0$ for all $o \in O$. Since $u_i(o) > 0$ for all $o \in O$, we know that in any unbalanced deterministic allocation one agent $i \in N$ gets zero items and another agent j gets at least two items. Even if one of j's items is removed, i will be envious of j. Hence, an unbalanced allocation is not EF1. In the other direction, a balanced allocation gives one item to each agent. Even if an agent $i \in N$ is envious of agent j, agent i will not be envious if j's item is removed. We have established that for n agents and n items, the set of deterministic EF1 allocations is equal to the set of deterministic balanced allocations. Therefore, the set of deterministic EF1 and Pareto optimal allocations is equivalent to the set of deterministic balanced and Pareto optimal allocations. It follows that checking whether a given random allocation can be implemented by a lottery over EF1 and Pareto optimal allocation is NP-hard.

The same argument also works for the problem of checking whether a given random allocation can be implemented by a lottery over *SD-EF1* and Pareto optimal allocations. □

5   Impossibility Results

We first recall that Freeman et al. [15] proved that even for two agents, ex-ante fPO, ex-ante envy-freeness, and ex-post EF1 are incompatible. In this section, we present a couple of more impossibility results. The results are logically incomparable to the main impossibility result of Freeman et al. [15]. Our first impossibility is the following one.

Theorem 4. *Ex-ante SD-EF, ex-post EF1, and ex-post PO are incompatible even for 2 agents.*

Proof. Consider the example in which $N = \{1, 2\}$, $O = \{a, b_1, b_2, b_3\}$ and the agents have the following utilities over four items.

	a	b_1	b_2	b_3
1	7	1	1	1
2	4	2	2	2

The three items b_1, b_2, b_3 are identical items that we refer to as b items. Ex-ante SD-EF implies that each agent in expectation gets $1/2$ of a and 1.5 units of type b items. Our first claim is that in any lottery implementing such an ex-ante SD-EF allocation, there is at least one ex-post allocation in which agent 2 must get item a. This follows from the fact that agent 2 gets $1/2$ of a in expectation.

Our second claim is that in any deterministic ex-post EF1 and ex-post PO allocation, agent 2 cannot get item a. Suppose for contradiction that agent 2 gets a. Then, EF1 requires that agent 1 gets at least 2 items of type b. But then, agent 1 can exchange these two items for a to obtain a Pareto improvement.

From the two claims above, it follows that for the problem instance, there exists no lottery over ex-post EF1 and ex-post PO outcomes that implements the SD-EF random outcome. □

Next, we point out that ex-ante fPO and ex-ante SD-EF are incompatible even for 2 agents. The theorem follows directly from Theorem 5 [6] but we re-prove it in our context for the sake of completeness.

Theorem 5. *Ex-ante fPO and ex-ante SD-EF are incompatible even for 2 agents.*

Proof. Consider the following two-agent profile.

$$
\begin{array}{c|cc}
 & a & b \\
\hline
1 & u_1(a) & u_1(b) \\
2 & u_2(a) & u_2(b)
\end{array}
$$

Consider an SD-EF and ex-ante PO allocation p. Suppose $u_1(a), v_b^1,$ $u_2(a), u_2(b) > 0$ in such a way that $u_1(a) > u_1(b)$ and $u_2(a) > u_2(b)$ and $\frac{u_1(a)}{u_1(b)} > \frac{u_2(a)}{u_2(b)}$. Due to SD-EF, the outcome should be

$$
p = \begin{array}{c}
1 \\
2
\end{array}
\begin{pmatrix}
\overset{a}{1/2} & \overset{b}{1/2} \\
1/2 & 1/2
\end{pmatrix}
$$

On the other hand, in order for the mechanism to be ex-ante fPO, $p_{1,b} = 0$ or $p_{2,a} = 1$. □

6   Binary Utilities

We assumed that the agents have additive utilities. If we consider the case in which agents have 1-0 utilities, we can achieve stronger results. We show that our EPS-lottery algorithm satisfies very strong properties when agents have 1-0 utilities. In order to ensure ex-ante efficiency of the EPS-lottery algorithm under 1-0 utilities, we can assume that agents do not consume zero utility items and leave them for the consumption by other agents as is done by the Controlled Cake Eating Algorithm (CCEA) algorithm [6]. In case this leads to unbalanced allocations, we can make the allocation balanced by adding appropriate number of *extra* dummy items so that we can implement our lottery decomposition algorithm for a balanced allocation.

Before we proceed, let us recall the definition of leximin optimality. For an allocation π we denote by $\boldsymbol{u}(\pi) \in \mathbb{R}^n$ the vector of the utilities in π sorted in increasing order. For two vectors $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^k$, we say that $\boldsymbol{u}$ leximin-dominates $\boldsymbol{v}$, written $\boldsymbol{u} \succ_{lex} \boldsymbol{v}$, if there exists an $i \leq k$ such that $\boldsymbol{u}_j = \boldsymbol{v}_j, \forall j < i$, and $\boldsymbol{u}_i > \boldsymbol{v}_i$. Finally, π is leximin-optimal if there is no π' such that $\boldsymbol{u}(\pi') \succ_{lex} \boldsymbol{u}(\pi)$.

Under 1-0 utilities, it is known that the following rules are equivalent and polynomial-time computable: (1) leximin rule (2) maximum Nash welfare (MNW) rule (3) competitive equilibrium with equal incomes (CEEI) [23] and (4) Controlled Cake Eating Algorithm (CCEA) rule [6] (which can be viewed as an extension for EPS for multi-unit demands that is also careful about zero

utilities). For example, CEEI and MNW are well-known to be equivalent even for general additive utilities. Under binary utilities, leximin, CEEI, and CCEA are equivalent [6]. CCEA satisfies envy-freeness. The conclusion about envy-freeness is also derived from the fact that CEEI outcomes are envy-free (see, e.g. [24]). It is well-known that under additive utilities, the utility profile of the agents is unique (see, e.g., [24]).

For 1-0 utilities, the rules above are known to be ex-ante group-strategyproof (no group of agents can misreport their preferences so that all agents get at least as much utility and at least one agent gets strictly more utility). This fact has been known before as well (see, e.g., [9,20] and [6]). Since the rules are equivalent to the leximin rule, the outcome is by definition leximin optimal and hence ex-ante fPO.

We have already shown that an outcome of the EPS rule can be implemented by a lottery over EF1 allocations. Also, every deterministic allocation consistent with the SD-efficient random outcome is SD-efficient (follows from Lemma 2 [20]) and hence ex-post Pareto optimal for binary utilities. Therefore, we achieve ex-post EF1 and ex-post Pareto optimality.

Theorem 6. *For binary utilities, the EPS-Lottery Algorithm is group-strategyproof, ex-ante fPO, ex-post fPO, ex-ante envy-free, and ex-post EF1. Its outcome is ex-ante equivalent to the leximin random allocation as well as the maximum Nash welfare allocation.*

The theorem above recovers some results that have been proved by Halpern et al. [17] including their Theorem 4 and Corollary 1.

7   Conclusion

We studied the problem of simultaneously achieving desirable fairness properties ex-post and ex-ante. Our main contribution is an algorithm to find a lottery over EF1 allocations that is ex-ante equivalent to the outcome of the (E)PS rule. We noted that we actually compute a lottery over RB-allocations that satisfy strong EF1.

Figure 2 depicts the logical relations between various properties. It also shows some sets of properties that are possible or not possible to satisfy simultaneously. We noted that under 1-0 utilities, all meaningful ex-ante and ex-post fairness and efficiency properties are simultaneously satisfied. Coming back to general additive utilities, we recall that our algorithm achieves ex-ante SD-efficiency and ex-ante SD-EF. If we wish to replace ex-ante SD-efficiency with ex-ante fPO, then such an algorithm does not exist in view of Theorem 5. Again, note that our algorithm achieves ex-post SD-efficiency, ex-ante SD-EF, and ex-post SD-EF1. Even if we weaken ex-post SD-EF1 to ex-post EF1 but strengthen ex-post SD-efficiency to ex-post PO, we again get an impossibility (Theorem 4).
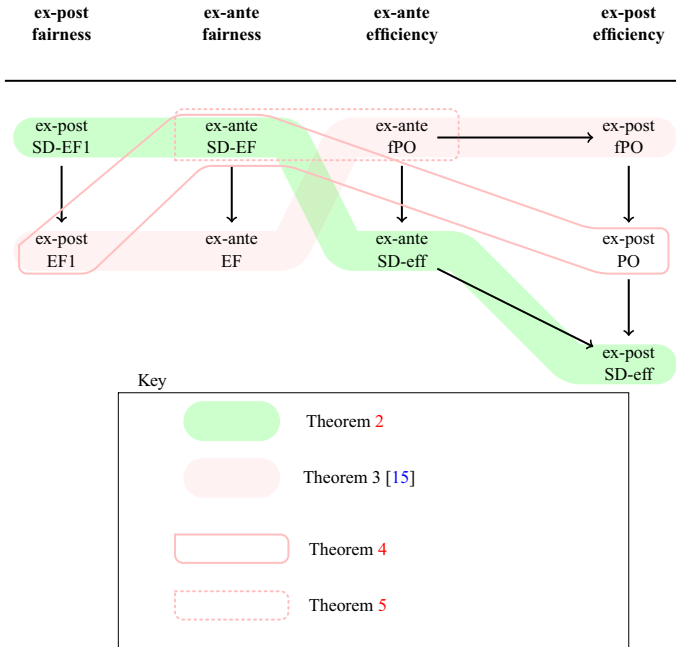
Fig. 2. Logical relations between fairness and efficiency concepts. An arrow from (A) to (B) denotes that (A) implies (B). The properties in green are simultaneously satisfied by our algorithm. The combined properties in the pink shapes (dotted, dashed, or shaded) are impossible to simultaneously satisfy. (Color figure online)

References

1. Aziz, H.: A probabilistic approach to voting, allocation, matching, and coalition formation. In: Laslier, J.-F., Moulin, H., Sanver, M.R., Zwicker, W.S. (eds.) The Future of Economic Design. SED, pp. 45–50. Springer, Cham (2019). https://doi.org/10.1007/978-3-030-18050-8_8
2. Aziz, H., Gaspers, S., Mackenzie, S., Walsh, T.: Fair assignment of indivisible objects under ordinal preferences. Artif. Intell. **227**, 71–92 (2015)
3. Aziz, H., Huang, X., Mattei, N., Segal-Halevi, E.: The constrained round robin algorithm for fair and efficient allocation. CoRR abs/1908.00161 (2019), http://arxiv.org/abs/1908.00161
4. Aziz, H., Mackenzie, S., Xia, L., Ye, C.: Ex post efficiency of random assignments. In: Proceedings of 14th AAMAS Conference, pp. 1639–1640 (2015)
5. Aziz, H., Walsh, T., Xia, L.: Possible and necessary allocations via sequential mechanisms. In: Proceedings of 24th IJCAI, pp. 468–474 (2015)

6. Aziz, H., Ye, C.: Cake cutting algorithms for piecewise constant and piecewise uniform valuations. In: Liu, T.-Y., Qi, Q., Ye, Y. (eds.) WINE 2014. LNCS, vol. 8877, pp. 1–14. Springer, Cham (2014). https://doi.org/10.1007/978-3-319-13129-0_1

7. Birkhoff, G.: Three observations on linear algebra. Univ. Nac. Tacuman Rev. Ser. A **5**, 147–151 (1946)

8. Bogomolnaia, A., Moulin, H.: A new solution to the random assignment problem. J. Econ. Theory **100**(2), 295–328 (2001)

9. Bogomolnaia, A., Moulin, H.: Random matching under dichotomous preferences. Econometrica **72**(1), 257–279 (2004)

10. Budish, E.: The combinatorial assignment problem: approximate competitive equilibrium from equal incomes. J. Polit. Econ. **119**(6), 1061–1103 (2011)

11. Budish, E., Che, Y.K., Kojima, F., Milgrom, P.: Designing random allocation mechanisms: theory and applications. Am. Econ. Rev. **103**(2), 585–623 (2013)

12. Caragiannis, I., Kurokawa, D., Moulin, H., Procaccia, A.D., Shah, N., Wang, J.: The unreasonable fairness of maximum Nash welfare. In: Proceedings of 17th ACM-EC Conference, pp. 305–322 (2016)

13. Conitzer, V., Freeman, R., Shah, N., Vaughan, J.W.: Group fairness for the allocation of indivisible goods. In: Proceedings of 33rd AAAI Conference (2019)

14. Devanur, N., Papadimitriou, C.H., Saberi, A., Vazirani, V.: Market equilibrium via a primal-dual algorithm for a convex program. J. ACM **55**(5), 1–8 (2008)

15. Freeman, R., Shah, N., Vaish, R.: Best of both worlds: ex-ante and ex-post fairness in resource allocation. Technical report (2020)

16. Grötschel, M., Lovász, L., Schrijver, A.: The ellipsoid method and its consequences in combinatorial optimization. Combinatorica **1**(2), 169–197 (1981). https://doi.org/10.1007/BF02579273

17. Halpern, D., Procaccia, A.D., Psomas, A., Shah, N.: Fair division with binary valuations: one rule to rule them all. Technical report (2020)

18. Hopcroft, J.E., Karp, R.M.: An $n^{5/2}$ algorithm for maximum matchings in bipartite graphs. SIAM J. Comput. **2**(4), 225–231 (1973)

19. Karzanov, A.V.: An exact estimate of an algorithm for finding a maximum flow, applied to the problem on representatives. Probl. Cybern. **5**, 66–70 (1973)

20. Katta, A.K., Sethuraman, J.: A solution to the random assignment problem on the full preference domain. J. Econ. Theory **131**(1), 231–250 (2006)

21. Kojima, F.: Random assignment of multiple indivisible objects. Math. Soc. Sci. **57**(1), 134–142 (2009)

22. Lovász, L., Plummer, M.D.: Matching Theory. AMS Chelsea Publishing, Providence (2009)

23. Varian, H.R.: Equity, envy, and efficiency. J. Econ. Theory **9**, 63–91 (1974)

24. Vazirani, V.V.: Combinatorial algorithms for market equilibria. In: Nisan, N., Roughgarden, T., Tardos, É., Vazirani, V. (eds.) Algorithmic Game Theory, chap. 5, pp. 103–134. Cambridge University Press, Cambridge (2007)