Fair and Efficient Allocations of Chores under Bivalued Preferences

Jugal Garg,[1] **Aniket Murhekar,** [1] **John Qin** [1]

[1] University of Illinois - Urbana Champaign
jugal@illinois.edu, aniket2@illinois.edu, johnqin2@illinois.edu

Abstract

We study the problem of fair and efficient allocation of a set of indivisible *chores* to agents with additive cost functions. We consider the popular fairness notion of envy-freeness up to one good (EF1) with the efficiency notion of Pareto-optimality (PO). While it is known that an EF1+PO allocation exists and can be computed in pseudo-polynomial time in the case of goods, the same problem is open for chores.

Our first result is a strongly polynomial-time algorithm for computing an EF1+PO allocation for *bivalued instances*, where agents have (at most) two disutility values for the chores. To the best of our knowledge, this is the first non-trivial class of indivisible chores to admit an EF1+PO allocation and an efficient algorithm for its computation.

We also study the problem of computing an envy-free (EF) and PO allocation for the case of divisible chores. While the existence of EF+PO allocation is known via competitive equilibrium with equal incomes, its efficient computation is open. Our second result shows that for bivalued instances, an EF+PO allocation can be computed in strongly polynomial-time.

1 Introduction

The problem of fair division is concerned with allocating items to agents in a *fair* and *efficient* manner. Formally introduced by (Steinhaus 1949), fair division is an active area of research studied across fields like computer science and economics. Most work has focused on the fair division of *goods*: items which provide non-negative *value* (or utility) to the agents to whom they are allocated. However, several practical scenarios involve *chores* (or bads). Chores are items which impose a *cost* (or disutility) to the agent to whom they are allocated. For instance, household chores such as cleaning and cooking often need to be fairly distributed among members of the household. Likewise, teachers have to divide teaching load, stakeholders have to divide liabilities upon dissolution of a firm, etc. These examples highlight the importance of allocating *chores* in a fair and efficient manner. Agencies responsible for designing such allocations must take into account the differences in preferences of agents in order for the allocation to be acceptable to all those involved.

Arguably, the most popular notion of fairness is envy-freeness (EF) (Foley 1967; Varian 1974), which requires that every agent weakly prefers the bundle of items allocated to her over the bundle of any other agent. When items are *divisible*, i.e., can be shared among agents, EF allocations are known to exist. However, in the case of *indivisible* items, EF allocations need not exist. For instance, while dividing one chore between two agents, the agent who is assigned the chore will envy the other. Since the fair division of indivisible items remains an important problem, several relaxations of envy-freeness have been defined, first in the context of goods, and later adapted to chores.

A widely studied relaxation of envy-freeness is *envy-freeness up to one item* (EF1), defined by (Budish 2011) in the context of goods. For chores, an allocation is said to be EF1 if for every agent, the envy disappears after removing one chore assigned to her. It is known that an EF1 allocation of chores exists and can be efficiently computed (Lipton et al. 2004; Bhaskar, Sricharan, and Vaish 2020). However, an EF1 allocation may be highly inefficient. Consider for example two agents A_1 and A_2 and 2 chores j_1 and j_2 where A_i has almost zero cost for j_i and high cost for the other chore. The allocation in which j_1 is assigned to A_2 and j_2 is assigned to A_1 is clearly EF1. However both agents incur high cost, which is highly inefficient. The allocation in which j_i is assigned to A_i is more desirable since it is both fair as well as efficient.

The standard notion of economic efficiency is Pareto optimality (PO). An allocation is said to be PO if no other allocation makes an agent better off without making someone else worse off. Fractional Pareto optimality (fPO) is a stronger notion requiring that no other fractional allocation makes an agent better off without making someone else worse off. Every fPO allocation is therefore PO, but not vice-versa.

An important question is whether the fairness and efficiency notions of EF1 and PO (or fPO) can be achieved in conjunction, and if yes, can they be computed in polynomial-time. For the case of goods, (Barman, Krishnamurthy, and Vaish 2018a) showed that an EF1+PO allocation exists and can be computed in pseudopolynomial-time. Improving this result, (Murhekar and Garg 2021) showed that an EF1+fPO allocation can be computed in pseudopolynomial-time. For the case of chores, it is unclear whether EF1+PO allocations even exist, except for

simple cases like identical valuations. Settling the existence of EF1+PO allocations (and developing algorithms for computing them) has turned out to be a challenging open problem.

In this paper, we present the first non-trivial results on the EF1+PO problem for chores. We study the class of *bivalued* instances, where there are only two costs, i.e., for every agent a chore costs either a or b, for two positive numbers[1] a and b. Bivalued instances are a well-studied class in the fair division literature, we list several works in Section 1.1. In particular, (Amanatidis et al. 2020) showed that allocations that are envy-free up to *any* good (EFX), which is a strengthening of EF1, can be efficiently computed for bivalued goods. Recently, (Garg and Murhekar 2021) showed that EFX+fPO allocations can be computed in polynomial-time for bivalued goods. Showing positive results for bivalued chores, our first result is:

Result 1. *For bivalued instances with n agents and m indivisible chores, EF1+fPO allocations exist and can be computed in* poly(n, m)-*time.*

Next, we study the problem of computing an EF+PO allocation of divisible chores. For goods, it is known that an EF+PO allocation always exists (Varian 1974) and is in fact polynomial-time computable via the Eisenberg-Gale convex program (Nisan et al. 2007). This is done by computing the competitive equilibrium with equal incomes (CEEI). Here, the idea is to provide each agent with the same amount of fictitious money and then find prices and an allocation of items such that all items are completely bought and each agent buys her most preferred bundle subject to their budget constraint. This is an example of a *market* where demand (of agents) equals supply (of items), and is known as the Fisher market. For goods, there are polynomial-time algorithms for computing the competitive equilibrium (CE) (Devanur et al. 2008; Orlin 2010). For chores, the problem is harder: (Bogomolnaia et al. 2017) showed that the CE rule can be non-convex, multi-valued and disconnected. Algorithms with exponential run-times are known for computing CE for chores (Brânzei and Sandomirskiy 2019; Chaudhury et al. 2020), but designing a polynomial-time algorithm is an open problem. Working towards this goal, our second result shows:

Result 2. *For bivalued instances with n agents and m divisible chores, an EF+PO allocation can be computed in* poly(n, m)-*time.*

1.1 Further related work

(Barman, Krishnamurthy, and Vaish 2018a) showed that for n agents and m goods, an EF1+PO allocation can be computed in time poly(n, m, V), where V is the maximum utility value. Their algorithm first perturbs the values to a desirable form, and then computes an EF1+fPO allocation for the perturbed instance, which for a small-enough perturbation is EF1+PO for the original instance. Their approach is via *integral market equilibria*, which guarantees fPO at every step, and the concept of price-envy-freeness up to one good (pEF1) which is a strengthening of EF1. Using similar tools, (Murhekar and Garg 2021) showed that an EF1+fPO allocation can be computed in poly(n, m, V)-time. They also showed that an EF1+fPO allocation can be computed in poly(n, m)-time for k-ary instances (agents have at most k values for the goods) where k is a constant. It may seem a natural idea to try and use these approaches for chores, however they do not extend easily. While our algorithm also uses integral market equilibria to obtain the fPO property and pEF1 for chores to argue about EF1, our algorithm and its analysis are much more involved and significantly different from previous works.

Bivalued preferences are a well-studied class in literature. (Aziz et al. 2019) showed PO is efficiently verifiable for bivalued instances and coNP-hard for 3-valued instances; (Aziz 2020), and (Vazirani and Yannakakis 2021) studied the Hylland-Zeckhauser scheme for probabilistic assignment of goods in bivalued instances; and (Bogomolnaia and Moulin 2004) studied matching problems with bivalued (dichotomous) preferences. More generally, instances with few values have also been studied: (Barman, Krishnamurthy, and Vaish 2018b) showed that EF1+PO allocations can be computed for binary valuations; (Babaioff, Ezra, and Feige 2021) studied truthful mechanisms for dichotomous valuations; (Golovin 2005) presented approximation algorithms and hardness results for computing max-min fair allocations in 3-valued instances; (Bliem, Bredereck, and Niedermeier 2016) studied fixed-parameter tractability for computing EF+PO allocations with parameter $n + z$, where z is the number of values; and (Garg et al. 2009) studied leximin assignments of papers ranked by reviewers on a small scale, in particular they present an efficient algorithm for 2 ranks, i.e., "high or low interest" and show NP-hardness for 3 ranks. Such instances have also been studied in resource allocation contexts, including makespan minimization with 2 or 3 job sizes (Woeginger 1997; Chakrabarty, Khanna, and Li 2015).

The fairness notion of equitability requires that each agent get the same amount of utility or disutility. Similar to EF1 and EFX, equitability up to one (resp. any) item (EQ1 (resp. EQX)) are relaxations of equitability. Using approaches inspired by (Barman, Krishnamurthy, and Vaish 2018a), pseudopolynomial time algorithms for computing EQ1+PO allocations were developed for both goods (Freeman et al. 2019) and chores (Freeman et al. 2020). (Garg and Murhekar 2021) showed that for bivalued goods an EQX+PO allocation is polynomial time computable.

2 Preliminaries

Problem instance. A *fair division instance* (of chores) is a tuple (N, M, C), where $N = [n]$ is a set of $n \in \mathbb{N}$ agents, $M = [m]$ is a set of $m \in \mathbb{N}$ indivisible chores, and $C = \{c_1, \ldots, c_n\}$ is a set of cost or disutility functions, one for each agent $i \in N$. Each cost function $c_i : M \to \mathbb{R}_{\geq 0}$ is specified by m numbers $c_{ij} \in \mathbb{R}_{\geq 0}$, one for each chore

---

[1]We can assume the two values are positive since one of them being zero implies the setting is *binary*, in which case computing an EF1+PO allocation is trivial by first assigning chores to agents which have 0 cost for them, and then allocating almost equal number of chores of non-zero cost to everyone.

$j \in M$, which denotes the cost agent i has for performing (receiving) chore j. We assume that the cost functions are additive, that is, for every agent $i \in N$, and for $S \subseteq M$, $c_i(S) = \sum_{j \in S} c_{ij}$. For notational ease, we write $c_i(S \setminus j)$ instead of $c(S \setminus \{j\})$).

We call a fair division instance (N, M, C) a *bivalued instance* if there exist $a, b \in \mathbb{R}^+$, with $a \geq b$, such that for all $i \in N$ and $j \in M$, $c_{ij} \in \{a, b\}$. That is, the cost of any chore to any agent is one of at most two given positive numbers. We can assume w.l.o.g. for bivalued instances that all costs $c_{ij} \in \{1, k\}$, where $k = a/b \geq 1$, by re-scaling the costs. Given such an instance, we partition the set of chores into sets of low-cost chores M_{low} and high-cost chores M_{high}:

- $M_{low} = \{j \in M : \exists i \in N \text{ s.t. } c_{ij} = 1\}$, and
- $M_{high} = \{j \in M : \forall i \in N, c_{ij} = k\}$.

We can additionally assume for bivalued instances that for every agent i, there is at least one chore j s.t. $c_{ij} = 1$. This is w.l.o.g., since if $c_{ij} = k$ for all $j \in M$, then we can re-scale costs to set $c_{ij} = 1$ for all $j \in M$.

Allocation. An *allocation* $\mathbf{x}$ of chores to agents is a n-partition of the chores $\mathbf{x}_1, \ldots, \mathbf{x}_n$, where agent i is allotted $\mathbf{x}_i \subseteq M$, and gets a total cost of $c_i(\mathbf{x}_i)$. A *fractional allocation* $\mathbf{x} \in [0, 1]^{n \times m}$ is a fractional assignment such that for each chore $j \in M$, $\sum_{i \in N} x_{ij} = 1$. Here, $x_{ij} \in [0, 1]$ denotes the fraction of chore j allotted to agent i.

Fairness notions. An allocation $\mathbf{x}$ is said to be *envy-free up to one chore* (EF1) if for all $i, h \in N$, where $\mathbf{x}_i \neq \varnothing$, there exists a chore $j \in \mathbf{x}_i$ such that $c_i(\mathbf{x}_i \setminus j) \leq c_i(\mathbf{x}_h)$. We say that an agent i *EF1-envies* an agent h if for all $j \in \mathbf{x}_i$, $c_i(\mathbf{x}_i \setminus j) > c_i(\mathbf{x}_h)$, i.e., the EF1 condition between i and h is violated.

A (fractional) allocation $\mathbf{x}$ is said to be *envy-free* if for all $i, h \in N$, $c_i(\mathbf{x}_i) \leq c_i(\mathbf{x}_h)$. We say that an agent i *envies* an agent h if $c_i(\mathbf{x}_i) > c_i(\mathbf{x}_h)$, i.e., the EF condition between i and h is violated.

Pareto-optimality. An allocation $\mathbf{y}$ dominates an allocation $\mathbf{x}$ if $c_i(\mathbf{y}_i) \leq c_i(\mathbf{x}_i), \forall i$ and there exists h s.t. $c_h(\mathbf{y}_h) < c_h(\mathbf{x}_h)$. An allocation is said to be *Pareto optimal* (PO) if no allocation dominates it. Further, an allocation is said to be fractionally PO (fPO) if no fractional allocation dominates it. Thus, a fPO allocation is PO, but not vice-versa.

Fisher markets. A Fisher market or a *market instance* is a tuple (N, M, C, e), where the first three terms are interpreted as before, and $e = \{e_1, \ldots, e_n\}$ is the set of agents' *minimum payments*, where $e_i \geq 0$, for each $i \in N$. In this model, chores can be allocated fractionally. Given a payment vector, also called a *price*[2] vector, $\mathbf{p} = (p_1, \ldots, p_m)$, each chore j pays p_j per unit of chore. Agents perform chores in exchange for payment. Given chore payments, each agent i aims to obtain the set of chores that minimizes her total cost subject to her payment constraint, i.e., receiving a total payment of at least e_i.

----
[2] We refer to payments as prices for sake of similarity with the Fisher market model in the goods case.

A *market outcome* is a (fractional) allocation $\mathbf{x}$ of the chores to the agents and a set of prices $\mathbf{p}$ of the chores. The *spending* of an agent i under the market outcome $(\mathbf{x}, \mathbf{p})$ is given by $\mathbf{p}(\mathbf{x}_i) = \sum_{j \in M} p_j x_{ij}$. For an agent i, we define the *bang-per-buck* ratio α_{ij} of good j as c_{ij}/p_j, and the *minimum bang-per-buck* (mBB) ratio $\alpha_i = \min_j \alpha_{ij}$. We define $\text{mBB}_i = \{j \in M : c_{ij}/p_j = \alpha_i\}$, called the *mBB-set*, to be the set of chores that give mBB to agent i at prices $\mathbf{p}$. A market outcome $(\mathbf{x}, \mathbf{p})$ is said to be *'on mBB'* if for all agents i and chores j, $x_{ij} > 0 \Rightarrow j \in \text{mBB}_i$. For integral $\mathbf{x}$, this means that $\mathbf{x}_i \subseteq \text{mBB}_i$ for all $i \in N$.

A market outcome $(\mathbf{x}, \mathbf{p})$ is said to be a *market equilibrium* if (i) the market clears, i.e., all chores are fully allocated. Thus, for all j, $\sum_{i \in N} x_{ij} = 1$, (ii) each agent receives their minimum payment, for all $i \in N$, $\sum_{j \in M} x_{ij} p_j = e_i$, and, (iii) agents only receive chores that give them minimum bang-per-buck, i.e., $(\mathbf{x}, \mathbf{p})$ is on mBB.

Given a market outcome $(\mathbf{x}, \mathbf{p})$ with $\mathbf{x}$ integral, we say it is *price envy-free up to one chore* (pEF1) if for all $i, h \in N$ there is a chore $j \in \mathbf{x}_i$ such that $\mathbf{p}(\mathbf{x}_i \setminus j) \leq \mathbf{p}(\mathbf{x}_h)$. We say that an agent i *pEF1-envies* an agent h, if for all $j \in \mathbf{x}_i$, $\mathbf{p}(\mathbf{x}_i \setminus j) > \mathbf{p}(\mathbf{x}_h)$, i.e., the pEF1 condition between i and h is violated. For integral market outcomes on mBB, the pEF1 condition implies the EF1 condition.

Lemma 1. *Let* $(\mathbf{x}, \mathbf{p})$ *be an integral market outcome on mBB. If* $(\mathbf{x}, \mathbf{p})$ *is pEF1 then* $\mathbf{x}$ *is EF1 and fPO.*

Proof. We first show that $(\mathbf{x}, \mathbf{p})$ forms a market equilibrium for the Fisher market instance (N, M, C, e), where for every $i \in N$, $e_i = \mathbf{p}(\mathbf{x}_i)$. It is easy to see that the market clears and each agent receives their minimum payment. Further $\mathbf{x}$ is on mBB as per our assumption. Now the fact that $\mathbf{x}$ is fPO follows from the First Welfare Theorem (Mas-Colell, Whinston, and Green 1995), which shows that for any market equilibrium $(\mathbf{x}, \mathbf{p})$, the allocation $\mathbf{x}$ is fPO.

Since $(\mathbf{x}, \mathbf{p})$ is pEF1, for all pairs of agents $i, h \in N$, there is some chore $j \in \mathbf{x}_i$ s.t. $\mathbf{p}(\mathbf{x}_i \setminus j) \leq \mathbf{p}(\mathbf{x}_h)$. Since $(\mathbf{x}, \mathbf{p})$ is on mBB, $\mathbf{x}_i \subseteq \text{mBB}_i$. Let α_i be the mBB-ratio of i at the prices $\mathbf{p}$. By definition of mBB, $c_i(\mathbf{x}_i \setminus j) = \alpha_i \mathbf{p}(\mathbf{x}_i \setminus j)$, and $c_i(\mathbf{x}_h) \geq \alpha_i \mathbf{p}(\mathbf{x}_h)$. Combining these implies $\mathbf{x}$ is EF1. $\square$

We now define least spenders as agents with minimum spending, and big spenders as agents with maximum spending after the removal of their highest-priced chore.

Definition 1 (Least and big spenders)**.** An agent $\ell \in \arg\min_{i \in N} \mathbf{p}(\mathbf{x}_i)$ is referred to as a *least spender* (LS). An agent $b \in \arg\max_{i \in N} \min_{j \in \mathbf{x}_i} \mathbf{p}(\mathbf{x}_i \setminus j)$ is referred to as a *big spender* (BS).

We break ties arbitrarily to decide a unique LS and BS. Together with Lemma 1, the following lemma shows that in order to obtain an EF1 allocation, it is sufficient to focus on the pEF1-envy the big spender has towards the least spender.

Lemma 2. *Let* $(\mathbf{x}, \mathbf{p})$ *be an integral market outcome on mBB. If* $\mathbf{x}$ *is not EF1, then the big spender* b *pEF1-envies the least spender* ℓ.

Proof. If $\mathbf{x}$ is not EF1, then Lemma 1 implies that $\mathbf{x}$ is not pEF1. Hence there is a pair of agents i, h s.t. for every chore

$j \in \mathbf{x}_i$, $\mathbf{p}(\mathbf{x}_i \setminus j) > \mathbf{p}(\mathbf{x}_h)$. By definition of big spender, we know $\mathbf{p}(\mathbf{x}_b \setminus j') \geq \mathbf{p}(\mathbf{x}_i \setminus j)$, for some $j' \in \mathbf{x}_b$. By definition of least spender, $\mathbf{p}(\mathbf{x}_h) \geq \mathbf{p}(\mathbf{x}_\ell)$. Putting these together we get $\mathbf{p}(\mathbf{x}_b \setminus j') > \mathbf{p}(\mathbf{x}_\ell)$, implying that b pEF1-envies ℓ. □

Given a price vector $\mathbf{p}$, we define the mBB graph to be a bipartite graph $G = (N, M, E)$ where for an agent i and good j, $(i, j) \in E$ iff $j \in \mathrm{mBB}_i$. Such edges are called *mBB* edges. Given an allocation $\mathbf{x}$, we supplement G to include *allocation edges*, i.e., an edge between i and j if $j \in \mathbf{x}_i$.

For agents $i_0, \ldots, i_\ell$ and goods $j_1, \ldots, j_\ell$, consider a path $P = (i_0, j_1, i_1, j_2, \ldots, j_\ell, i_\ell)$ in the supplemented mBB graph, where for all $1 \leq \ell' \leq \ell$, $j_{\ell'} \in \mathbf{x}_{i_{\ell'-1}} \cap \mathrm{mBB}_{i_{\ell'}}$. Define the *level* of an agent h to be half the length of the shortest such path from the BS to h, and to be n if no such path exists. Define *alternating paths* to be such paths where the edges are between agents at a lower level to agents at a strictly higher level. The edges in an alternating path alternate between allocation edges and mBB edges.

Definition 2 (Component C_i of a big spender i). For a big spender i, define C_i^ℓ to be the set of all chores and agents which lie on alternating paths of length ℓ. Call $C_i = \bigcup_\ell C_i^\ell$ the *component* of i, the set of all chores and agents reachable from the big spender i through alternating paths.

3 EF1+fPO allocation of indivisible chores

In this section, we present our main result:

Theorem 1. *Given a bivalued fair division instance (N, M, C) of indivisible chores with all $c_{ij} \in \{a, b\}$ for some $a, b \in \mathbb{Z}^+$, an EF1+fPO allocation can be computed in strongly polynomial-time.*

We prove Theorem 1 by showing that our Algorithm 2 computes an EF1+fPO allocation in polynomial-time. For ease of presentation and space constraints, we defer proofs of this section to Appendix A.

3.1 Obtaining Initial Groups

Recall that we can scale the costs so that they are in $\{1, k\}$. The first step of Algorithm 2 is to obtain a partition of the set N of agents into *groups* $N_1, \ldots, N_R$ with desirable properties. For this, we use Algorithm 1 (called MakeInitGroups).

Algorithm 1 starts with a cost-minimizing market outcome $(\mathbf{x}, \mathbf{p})$ where each chore j is assigned to an agent who has minimum cost for j. This ensures the allocation is fPO. The chore prices are set as follows. Each low-cost chore j is assigned to an agent i s.t. $c_{ij} = 1$. If an agent values all chores at k, then we can re-scale all values to 1. Each low-cost chore is priced at 1, and each high-cost chore is priced at k. This pricing ensures that the mBB ratio of every agent is 1. The algorithm then eliminates pEF1-envy from the component of the big spender b by identifying an agent i in C_b that is pEF1-envied by b, and transferring an additional chore j_ℓ to i from an agent $h_{\ell-1}$ who lies along a shortest alternating path from b to i (Lines 7 & 8). Once the component of the big spender b is pEF1, the same process is applied to the next big spender outside the previously made components. Repeated application of this process leaves us with disjoint partial components $H_1, \ldots, H_R$ of agent sets

---

Algorithm 1: MakeInitGroups

Input: Fair division instance (N, M, C) with $c_{ij} \in \{1, k\}$
Output: Integral alloc. $\mathbf{x}$, prices $\mathbf{p}$, agent groups $\{N_r\}_{r \in [R]}$

1: $(\mathbf{x}, \mathbf{p}) \leftarrow$ initial cost minimizing integral market allocation, where $p_j = c_{ij}$ for $j \in \mathbf{x}_i$.
2: $R \leftarrow 1, N' \leftarrow N$
3: **while** $N' \neq \emptyset$ **do**
4:    $b \leftarrow \mathrm{argmax}_{i \in N'} \min_{j \in \mathbf{x}_i} \mathbf{p}(\mathbf{x}_i \setminus j)$ ▷ Big Spender
5:    $C_b \leftarrow$ Component of b ▷ See Definition 2
6:    **while** $\exists$ agent $i \in C_b$ s.t. $\forall j \in \mathbf{x}_b, \mathbf{p}(\mathbf{x}_b \setminus j) > \mathbf{p}(\mathbf{x}_i)$
7:      Let $(b, j_1, h_1, j_2, \ldots, h_{\ell-1}, j_\ell, i)$ be the shortest alternating path from b to i
8:      $\mathbf{x}_{h_{\ell-1}} \leftarrow \mathbf{x}_{h_{\ell-1}} \setminus \{j_\ell\}$ ▷ Chore transfer
9:      $\mathbf{x}_i \leftarrow \mathbf{x}_i \cup \{j_\ell\}$
10:     $b \leftarrow \mathrm{argmax}_{i \in N'} \min_{j \in \mathbf{x}_i} \mathbf{p}(\mathbf{x}_i \setminus j)$
11:   $H_R \leftarrow C_b \cap (N' \cup \mathbf{x}_{N'})$ ▷ Partial component
12:   $N_R \leftarrow H_R \cap N$ ▷ Agent group
13:   $N' \leftarrow N' \setminus N_R, R \leftarrow R + 1$
14: **return** $(\mathbf{x}, \mathbf{p}, \{N_r\}_{r \in [R]})$

$N_1, \ldots, N_R$, where $R \leq n$, all of which are pEF1. We refer to $N_1, \ldots, N_R$ as agent *groups*, and $H_1, \ldots, H_R$ as initial (partial) components. Note also that the spending (up to the removal of the biggest chore) of the big spender h_r of H_r is weakly decreasing with r. We now record several properties of the output of Algorithm 1 .

Lemma 3. *Algorithm 1 returns in $\mathrm{poly}(n, m)$-time a market outcome $(\mathbf{x}, \mathbf{p})$ with agents partitioned into groups $N_1, \ldots, N_R$, with the following properties:*

(i) For all low-cost chores $j \in M$, $p_j = 1$, and for all high-cost chores $j \in M$, $p_j = k$.
(ii) The mBB ratio α_i of every agent i is 1.
(iii) Let H_r be the collection of agents N_r and chores allocated to them in $(\mathbf{x}, \mathbf{p})$. Then each H_r is a *partial component* of some agent. That is, for each $r \in [R]$, there is an agent $h_r \in H_r$ s.t. H_r comprises of all agents and chores not in $\bigcup_{r' < r} H_{r'}$ reachable through alternating paths from h_r. Further, h_r is the big spender among agents not in $\bigcup_{r' < r} H_{r'}$:

$$h_r \in \mathrm{argmax}_{i \notin (\bigcup_{r' < r} H_{r'})} \min_{j \in \mathbf{x}_i} \mathbf{p}(\mathbf{x}_i \setminus j)$$

(iv) The spending (up to removal of the largest chore) $f(r)$ of the big spender in H_r weakly decreases with r. Here $f(r) = \max_{i \in H_r} \min_{j \in \mathbf{x}_i} \mathbf{p}(\mathbf{x}_i \setminus j)$.
(v) Each *group is pEF1*, i.e., an agent does not pEF1-envy other agents in the same group.
(vi) For every $i \in H_r$ and $j \in H_{r'}$ with $r' < r$, $c_{ij} = k$.
(vii) All high-cost chores belong to H_R.

We have that:

Lemma 4. *Algorithm 1 terminates in time $\mathrm{poly}(n, m)$.*

3.2 Overview of Algorithm 2

Our main algorithm (Algorithm 2) begins by calling Algorithm 1, which returns a market outcome $(\mathbf{x}, \mathbf{p})$ and a set of

Algorithm 2: Computing an EF1+fPO allocation

Input: Fair division instance (N, M, C) with $c_{ij} \in \{1, k\}$
Output: An integral allocation $\mathbf{x}$

1: $(\mathbf{x}, \mathbf{p}, \{N_r\}_{r \in [R]}) \leftarrow \mathsf{MakeInitGroups}(\mathsf{N}, \mathsf{M}, \mathsf{C})$
2: $U \leftarrow [R]$ ▷ Unraised groups
3: $\mathbf{x}^0 \leftarrow \mathbf{x}$ ▷ Copy of initial allocation, used in Line 21
4: $b \leftarrow \arg\max_{i \in N} \min_{j \in \mathbf{x}_i} \mathbf{p}(\mathbf{x}_i \setminus j)$ ▷ Big Spender
5: $\ell \leftarrow \arg\min_{i \in N} \mathbf{p}(\mathbf{x}_i)$ ▷ Least Spender
6: Maintain (r, s) s.t. $b \in H_r, \ell \in H_s$
7: **while** $(\mathbf{x}, \mathbf{p})$ is not pEF1 **and** $s \in U$ **do**
8:     **if** $r \in U$ **then**
9:         Raise prices of chores in H_r by a factor of k
10:         $U \leftarrow U \setminus \{r\}$
11:     **else**
12:         Transfer a chore from b to ℓ along an mBB edge
13:     $b \leftarrow \arg\max_{i \in N} \min_{j \in \mathbf{x}_i} \mathbf{p}(\mathbf{x}_i \setminus j)$
14:     $\ell \leftarrow \arg\min_{i \in N} \mathbf{p}(\mathbf{x}_i)$
15: **while** $(\mathbf{x}, \mathbf{p})$ is not pEF1 **do**
16:     **if** $s > r$ **then**
17:         Transfer a chore from b to ℓ along an mBB edge
18:     **else if** $s < r$ **then**
19:         $\exists i \in N_{r'}$ s.t. $r' \in U$ and $\exists j \in \mathbf{x}_i$ s.t. $j \in \mathbf{x}_\ell^0$
20:         Transfer j from i to ℓ
21:         Transfer a chore from b to i
22:     $b \leftarrow \arg\max_{i \in N} \min_{j \in \mathbf{x}_i} \mathbf{p}(\mathbf{x}_i \setminus j)$
23:     $\ell \leftarrow \arg\min_{i \in N} \mathbf{p}(\mathbf{x}_i)$
24: **return** $(\mathbf{x}, \mathbf{p})$

---

agent groups $\{N_r\}_{r \in [R]}$ (with associated partial components $\{H_r\}_{r \in [R]}$) satisfying properties in Lemma 3. In the subsequent discussion, we refer to $(\mathbf{x}, \mathbf{p})$ as the *initial allocation*. Also in the subsequent discussion, all mentions of an agent receiving or losing chores are relative to this initial allocation. The following is an important invariant of Algorithm 2.

Lemma 5. *The spending of the least spender does not decrease in the run of Algorithm 2.*

We say that a group N_r is above (resp. below) group N_s if $r < s$ (resp. $r > s$). Lemma 3 shows that each group N_r is initially pEF1. Hence if the initial allocation $(\mathbf{x}, \mathbf{p})$ is not pEF1, then the big spender b and the least spender ℓ must be in different components. Since $b \in H_1$, it must be the case that $\ell \in H_s$ for some $s > 1$. Since we want to obtain an fPO allocation, we can only transfer along mBB edges. Hence we raise the prices of all chores in H_1. We show that doing so creates an mBB edge from all agents $i \notin H_1$ to all $j \in H_1$ (Lemma 9 below). In particular, there is an mBB edge from ℓ to a chore assigned to b. Hence we transfer a chore *directly* from b to ℓ, thus reducing the pEF1-envy of b. This may change the identity of the big and least spenders. If the allocation is not yet pEF1, we must continue this process.

Let t be an arbitrary time-step in the run of the algorithm, and let b and ℓ be the big and least spenders at t. If the allocation is not pEF1, then b pEF1-envies ℓ (Lemma 2). We consider cases based on the relative positions of b and ℓ. First we argue that b and ℓ cannot lie in the same group, by

showing that:

Lemma 6. *Throughout the run of Algorithm 2, each group N_r remains pEF1.*

Hence b and ℓ must lie in different groups. Once again, since we want to transfer chores away from b to reduce the pEF1-envy, and we want to obtain an fPO allocation, we only transfer chores along mBB edges. Doing so may require us to raise the prices of all chores belonging to certain agents in order to create new mBB edges to facilitate chore transfer. In our algorithm, *all* agents in a group undergo price-rise together. We call a group N_r a *raised group* if its agents have undergone price-rise, else it is called an *unraised group*. The set U (Line 2) records the set of unraised components.

We will use the terms *time-step* or *iteration* interchangeably to denote either a chore transfer or a price-rise step. We say 'at time-step t', to refer to the state of the algorithm *just before* the event at t happens. We denote by $(\mathbf{x}^t, \mathbf{p}^t)$ the allocation and price vector at time-step t.

Let T be the first time-step that the current LS enters a raised group. Note that such an event may or may not happen. Our algorithm performs instructions in Lines 7-14 before T, and Lines 15-23 after T, as we describe below.

3.3 Algorithm prior to T (Lines 7-14)

We first record some properties of the algorithm prior to T. These observations directly follow from the algorithm.

Lemma 7. *Prior to T, the following hold:*

1. *Any transfer of chores only takes place directly from the big spender b to the least spender ℓ. Thus, an agent receives a chore only if she is a least spender, and an agent loses a chore only if she is a big spender.*

2. *An agent ceases to be a least spender only if she receives a chore. An agent ceases to be a big spender only if she loses a chore.*

3. *A group undergoes price-rise at t only if the group contains the big spender at t.*

We now show that:

Lemma 8. *If at any point in the run of Algorithm 2 prior to T, the big spender lies in a group which contains a former least spender, then the allocation is pEF1.*

This allows us to show:

Lemma 9. *Prior to T, the following hold:*

(i) *Let r be the number of price-rise steps until time-step t, where $t < T$. Then the raised groups are exactly $N_1, \ldots, N_r$. Further they underwent price-rise exactly once and in that order.*

(ii) *For any chore j allocated to an agent in a raised group N_r and any agent i in an unraised group $N_{r'}$, where $r' > r$, $j \in mBB_i$.*

(iii) *For each $r' \in [r]$, at the time of price-rise of $N_{r'}$, no agent in $N_{r'}$ has either received or lost a chore since the initial allocation.*

Proof. We prove (i), (ii) and (iii) by induction. For $r = 0$, they are trivially true since there are no raised groups. Assume that at some time-step t, (i) groups $N_1, \ldots, N_r$ have undergone price-rise, once and in that order, for some $r \geq 1$, and (ii) and (iii) hold.

Note that our algorithm only raises the prices of chores owned by a group if the group contains the big spender at the time (Lemma 7). If the current BS is in a raised group, then the induction hypothesis ensures that there is an mBB edge from the LS (who is in an unraised group prior to T) to chores owned by the BS. The algorithm therefore continues to do direct chore transfers and no price-rise is necessary.

If eventually the BS enters an unraised group, then a price-rise step is potentially necessary. Suppose b is an agent who has received a chore prior to t. If this happens then b must have been a former LS by Lemma 7. Then Lemma 8 shows that the allocation must already be pEF1.

Hence we assume b has not received a new chore. Further since b pEF1-envies the LS ℓ, it must be the case that $\ell \in N_s$ where $s \geq r + 2$. Lemma 3 shows that there is no mBB edge from ℓ to chores owned by b, hence a direct chore transfer is not possible and it is necessary for N_{r+1} to undergo price-rise. This shows (i).

Now if an agent $i \in N_{r+1}$ had previously received a chore, then i is a former LS at t. Lemma 8 shows that b is in a group containing a former LS, implying that the allocation must be pEF1. Similarly Lemma 7 shows that no agent in N_{r+1} can have lost a chore. This is because only BS agents lose chores. Prior to t, no agent of N_{r+1} can be BS. Hence no agent in N_{r+1} has received or lost a chore since the initial allocation at the time N_{r+1} undergoes price-rise, thus showing (iii).

The algorithm next raises the prices of all chores owned by N_{r+1} by a factor of k, and N_{r+1} becomes a raised group. Consider an agent $i \in N_{r'}$ for $r' \geq r + 2$ and a chore allocated to an agent in N_{r+1}. Since the mBB ratio only changes upon a price-rise, the mBB ratio of i is 1 since $N_{r'}$ does not undergo a price-rise before N_{r+1}.

Observe that since $i \notin N_{r+1}$, there is no alternating path from agents in N_{r+1} to i. Hence $j \notin \text{mBB}_i$ before the price-rise. Thus $c_{ij}/p_j^t > 1$, showing $c_{ij} = k$ and $p_j^t = 1$. After the price-rise, we have that $p_j^{t+1} = k$, and $\alpha_i = c_{ij}/p_j^{t+1}$. Thus, $j \in \text{mBB}_i$ after the price-rise, which shows (ii). □

To summarize the behavior of the Algorithm prior to T, we have argued in the above proof that if the allocation is not pEF1, we can always (i) transfer a chore directly from b to ℓ, or (ii) perform a price-rise on the group of b and then transfer a chore from b to ℓ. Further, we argue that the Algorithm makes progress towards getting a pEF1 allocation.

Lemma 10. *Algorithm 2 performs at most* $\text{poly}(n, m)$ *steps prior to T.*

Proof. Prior to T, the LS always remains in an unraised group. Chores are transferred away from agents who become big spenders in raised groups. Once an agent undergoes price-rise, she cannot gain any additional chores, since doing so would mean she is LS in a raised group, which cannot happen prior to T. When the BS is in an unraised

group, the group undergoes a price-rise. Thus effectively, either agents in raised components only lose chores, or the BS 'climbs-down' in the group list $N_1, \ldots, N_R$, while the LS remains below the BS. Since there are $R \leq n$ groups, and at most m chores allocated to raised groups, after $\text{poly}(n, m)$ steps either of two events happen: (i) the LS and BS both belong to the same group, or (ii) the LS enters a raised group. In the former case, the allocation is pEF1 due to Lemma 6, and the algorithm terminates in $\text{poly}(n, m)$ steps. We discuss the latter case in the next section. Thus, there are at most $\text{poly}(n, m)$ steps prior to T. □

3.4 Algorithm after T (Lines 15-23)

We now describe the algorithm after T, i.e., once the LS enters a raised component (Lines 15-23). We show that subsequent to T, as long as the allocation is not pEF1, we can either (i) transfer a chore directly from b to ℓ, or (ii) transfer a chore via an alternating path containing 3 agents. We do not perform any price-rises subsequent to T.

From Lemma 9, we know that at T, groups $N_1, \ldots, N_r$ have undergone price-rise, for some $r \in [R]$. Let $N_{<r} = \bigcup_{r' < r} N_{r'}$, and $N_{>r} = \bigcup_{r' > r} N_{r'}$. The allocation at T need not be pEF1, but we argue that it is already very close to being pEF1. Specifically, we show:

Lemma 11. *At T, agents in $N_{<r}$ are pEF1 towards others.*

Lemma 12. *At T, agents in $N_{>r}$ are pEF1 towards others.*

The above two lemmas imply that if the BS is not in N_r, then the allocation is pEF1. Let us assume that the allocation is not pEF1 at T. Let b, the BS at T be in N_r, and ℓ, the LS at T be in $N_{<r}$.

Suppose ℓ has never lost a chore. Let $\ell \in N_{r'}$, where $r' < r$, and t' be the time when $N_{r'}$ underwent price-rise. Let b' the BS at t'. Since the spending of the BS (up to removal of one chore) just after price-rises does not increase, we have:

$$\mathbf{p}^T(\mathbf{x}_b^T \setminus j) \leq \mathbf{p}^{t'}(\mathbf{x}_{b'}^{t'} \setminus j') \leq \mathbf{p}^{t'}(\mathbf{x}_\ell^{t'}) = \mathbf{p}^T(\mathbf{x}_\ell^T),$$

for some chores $j \in \mathbf{x}_b^T, j' \in \mathbf{x}_{b'}^{t'}$. The intermediate transition follows from the property that $N_{r'}$ is pEF1. This shows that the allocation is pEF1.

On the other hand, suppose ℓ has lost at least one chore j prior to T. At T, j must be assigned to some unraised agent i. Further, there is a chore $j' \in \mathbf{x}_b^T$ s.t. $j' \in \text{mBB}_i$. Thus, $b \to j' \to i \to j \to \ell$ is an alternating path and we now transfer chores along this path.

Note that as long as ℓ does not own a chore that she initially owned, such a path is available, and such a transfer is possible. If not, then it is as if ℓ has never lost a chore, and in that case the previous argument shows that the allocation must be pEF1.

Once again, doing such transfers may change the identities of the BS and LS, and we must continue the algorithm. We show that:

Lemma 13. *After T, the following are invariant:*

(i) Agents in $N_{<r}$ do not pEF1-envy any other agent.
(ii) Agents in $N_{>r}$ do not pEF1-envy any other agent.
(iii) Each group is pEF1.

Just after T, the BS is in N_r and the LS is in $N_{<r}$. After a chore transfer, the identity of the LS or BS can change. If the BS enters either $N_{<r}$ or $N_{>r}$, then using Lemma 13 the allocation would be pEF1. While the BS is in N_r: (i) if the LS is in N_r, the allocation would be pEF1, (ii) if the LS is in $N_{>r}$, then we can transfer from BS to LS directly along an mBB edge (which exists due to Lemma 9), (iii) if the LS is in $N_{<r}$, then we can transfer from the BS to LS via an alternating path with three agents as described above.

Finally we argue termination in polynomial-time:

Lemma 14. *Algorithm 2 performs at most* $\mathsf{poly}(n, m)$ *steps after T and terminates with a pEF1 allocation.*

Proof. After T, there are no price-rises, hence the spending gap weakly decreases. As long as the allocation is not pEF1, the BS must be in N_r. Based on whether the LS is in $N_{<r}$ or $N_{>r}$ we can perform chore transfers which weakly decrease the spending gap. While the allocation is not pEF1, such a transfer is always possible. Further, each such transfer reduces the number of chores owned by agents in N_r, and such agents do not receive any chores again. Hence there can only be $\mathsf{poly}(n, m)$ steps after T eventually terminating in a pEF1 allocation. $\square$

3.5 Summarizing our EF1+fPO algorithm

We summarize Algorithm 2.

1. Algorithm 2 first calls Algorithm 1 to partition agents into groups $N_1, \ldots, N_R$ with properties as in Lemma 3 and obtain an initial allocation. Lemma 4 shows this takes $\mathsf{poly}(n, m)$ steps.

2. When the current allocation $(\mathbf{x}, \mathbf{p})$ is not pEF1, the BS b pEF1-envies the LS ℓ. While there is an mBB edge from ℓ to a chore owned by b, we transfer a chore directly from ℓ to b. If not, in order to transfer along mBB edges, we may have to raise the prices of chores belonging to the group of b, creating *raised* groups.

3. Let T be the first time-step when the LS enters a raised group. Prior to T, while the allocation is not pEF1, the algorithm either performs a direct chore transfer from BS to LS, or performs price-rise on the group of b. Lemma 9 shows that the groups are raised exactly once and in order of $N_1, \ldots, N_R$. Lemma 10 shows the algorithm runs for $\mathsf{poly}(n, m)$ steps before T.

4. Once the LS enters a raised group, there are no more price-rise steps. The algorithm performs chore transfers from the BS to LS via alternating paths with at most 3 agents. Lemma 13 and Lemma 14 show that the algorithm performs at most $\mathsf{poly}(n, m)$ steps after T and terminates with a pEF1 allocation.

5. Finally, we note that the allocation is always fPO, since (i) Algorithm 1 returns a market outcome which is fPO, and (ii) any transfer of chores happens along mBB edges.

4 EF+PO allocation of divisible chores

In this section, we state our second result:

Theorem 2. *Given a bivalued fair division instance* (N, M, C) *of divisible chores with all* $c_{ij} \in \{a, b\}$ *for some* $a, b \in \mathbb{R}^+$, *an EF+PO allocation can be computed in strongly polynomial-time.*

Due to space constraints, we present the high-level intuition and defer details to Appendix B. Similar to the indivisible case, we start by partitioning agents into groups with desirable properties like in Lemma 3. Using a balanced flow network formulation (Devanur et al. 2008), we obtain a fractional allocation $\mathbf{x}$, prices $\mathbf{p}$, and agent groups $\{N_i\}_{i \in [R]}$ such that $(\mathbf{x}, \mathbf{p})$ is PO and each N_i is pEF.

We show that there exists some r^* such that raising the prices of the first r^* groups by a factor of $k = a/b > 1$ allows us to obtain a pEF allocation by *draining* chores from the set of all big spenders B (agents with maximum *total* spending) to the set of all least spenders L, when the chores of agents in B are all on mBB for agents in L. That is, we take chores from all agents in B at a uniform rate ρ_B and distribute them among all agents in L at a uniform rate ρ_L. Note that since groups are pEF, if an agent $i \in N_r$ is a BS, then every agent in N_r is also a BS.

Suppose we raise the first r groups. If it is impossible for the algorithm to obtain a pEF allocation, one of two mutually exclusive cases must be true:

1. Raised groups have *too much* total spending. The spending of agents in L rises to spending level of N_r before the spending of agents in B falls to the level of N_r.

2. Raised groups have *too little* total spending. The spending of agents in B falls to the level of N_{r+1} before the spending of agents in L rises to the level of N_{r+1}.

We show that there exists r^* such that raising the first r^* groups results in neither too much nor too little spending. Informally, raising the first group cannot result in too much spending. Inductively, if raising r groups does not result in too much spending, neither does raising $r + 1$ groups. Finally, we show that in raising $R - 1$ groups we cannot have too little spending. Thus, there must be some $r^* \in [R - 1]$ such that raising the first r^* groups results in exactly enough spending to allow for chores to be drained from the first r^* groups to the rest in a manner that culminates in a pEF allocation. As before, transferring chores along mBB ensures the allocation is PO.

5 Discussion

In this paper, we presented a strongly polynomial-time algorithm for computing an EF1+fPO allocation of chores to agents with bivalued preferences, constituting the first non-trivial result for the EF1+PO problem for chores. Our algorithm is novel and relies on several involved arguments. Given that the general case is a challenging open problem, we believe extending our algorithm and its analysis to the class of k-ary chores is an interesting and natural next step. Another interesting question is whether we can compute an EFX allocation in this setting. We also presented a strongly polynomial-time algorithm for computing an EF+PO allocation of divisible bivalued chores. Computing an EF+PO allocation of divisible chores in polynomial-time is also a compelling direction for future work.

References

Amanatidis, G.; Birmpas, G.; Filos-Ratsikas, A.; Hollender, A.; and Voudouris, A. A. 2020. Maximum Nash Welfare and Other Stories About EFX. In *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, (IJCAI)*, 24–30.

Aziz, H. 2020. The Hylland-Zeckhauser Rule Under Bi-Valued Utilities. *CoRR*, abs/2006.15747.

Aziz, H.; Biró, P.; Lang, J.; Lesca, J.; and Monnot, J. 2019. Efficient reallocation under additive and responsive preferences. *Theoretical Computer Science*, 790: 1 – 15.

Babaioff, M.; Ezra, T.; and Feige, U. 2021. Fair and Truthful Mechanisms for Dichotomous Valuations. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(6): 5119–5126.

Barman, S.; Krishnamurthy, S. K.; and Vaish, R. 2018a. Finding Fair and Efficient Allocations. In *Proceedings of the 19th ACM Conference on Economics and Computation (EC)*, 557–574.

Barman, S.; Krishnamurthy, S. K.; and Vaish, R. 2018b. Greedy Algorithms for Maximizing Nash Social Welfare. In *Proceedings of the 17th International Conference on Autonomous Agents and MultiAgent Systems (AAMAS)*, 7–13.

Bhaskar, U.; Sricharan, A. R.; and Vaish, R. 2020. On Approximate Envy-Freeness for Indivisible Chores and Mixed Resources. *CoRR*, abs/2012.06788.

Bliem, B.; Bredereck, R.; and Niedermeier, R. 2016. Complexity of Efficient and Envy-Free Resource Allocation: Few Agents, Resources, or Utility Levels. In *Proceedings of the 25th International Joint Conference on Artificial Intelligence (IJCAI)*, 102–108.

Bogomolnaia, A.; and Moulin, H. 2004. Random Matching Under Dichotomous Preferences. *Econometrica*, 72(1): 257–279.

Bogomolnaia, A.; Moulin, H.; Sandomirskiy, F.; and Yanovskaya, E. 2017. Competitive Division of a Mixed Manna. *Econometrica*, 85(6): 1847–1871.

Brânzei, S.; and Sandomirskiy, F. 2019. Algorithms for Competitive Division of Chores. *CoRR*, abs/1907.01766.

Budish, E. 2011. The Combinatorial Assignment Problem: Approximate Competitive Equilibrium from Equal Incomes. *Journal of Political Economy*, 119(6): 1061 – 1103.

Chakrabarty, D.; Khanna, S.; and Li, S. 2015. On (1, e)-Restricted Assignment Makespan Minimization. In *Proceedings of the 26th Annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, 1087–1101.

Chaudhury, B. R.; Garg, J.; McGlaughlin, P.; and Mehta, R. 2020. Competitive Allocation of a Mixed Manna. *CoRR*, abs/2008.02753.

Devanur, N. R.; Papadimitriou, C. H.; Saberi, A.; and Vazirani, V. V. 2008. Market Equilibrium via a Primal–Dual Algorithm for a Convex Program. *J. ACM*, 55(5).

Foley, D. 1967. Resource Allocation and the Public Sector. *Yale Economic Essays*, 7(1): 45–98.

Freeman, R.; Sikdar, S.; Vaish, R.; and Xia, L. 2019. Equitable Allocations of Indivisible Goods. In *Proceedings of the 28th International Joint Conference on Artificial Intelligence (IJCAI)*, 280–286.

Freeman, R.; Sikdar, S.; Vaish, R.; and Xia, L. 2020. Equitable Allocations of Indivisible Chores. In *Proceedings of the 19th International Conference on Autonomous Agents and MultiAgent Systems*, AAMAS '20, 384–392. Richland, SC: International Foundation for Autonomous Agents and Multiagent Systems. ISBN 9781450375184.

Garg, J.; and Murhekar, A. 2021. Computing fair and efficient allocations with few utility values. *In Proc. 14th Symp. Algorithmic Game Theory (SAGT)*. To appear.

Garg, N.; Kavitha, T.; Kumar, A.; Mehlhorn, K.; and Mestre, J. 2009. Assigning Papers to Referees. *Algorithmica, v.x, x-x (2009)*, 58.

Golovin, D. 2005. Max-min fair allocation of indivisible goods. *Technical Report, CMU-CS-05-144*.

Lipton, R. J.; Markakis, E.; Mossel, E.; and Saberi, A. 2004. On Approximately Fair Allocations of Indivisible Goods. In *Proceedings of the 5th ACM Conference on Electronic Commerce (EC)*, 125–131.

Mas-Colell, A.; Whinston, M.; and Green, J. 1995. *Microeconomic Theory*. Oxford University Press.

Murhekar, A.; and Garg, J. 2021. On Fair and Efficient Allocations of Indivisible Goods. *Proceedings of the AAAI Conference on Artificial Intelligence*, 35(6): 5595–5602.

Nisan, N.; Roughgarden, T.; Tardos, E.; and Vazirani, V. 2007. *Algorithmic Game Theory*. Cambridge University Press. ISBN 9781139466547.

Orlin, J. B. 2010. Improved Algorithms for Computing Fisher's Market Clearing Prices: Computing Fisher's Market Clearing Prices. In *Proceedings of the Forty-Second ACM Symposium on Theory of Computing*, STOC '10, 291–300. New York, NY, USA: Association for Computing Machinery. ISBN 9781450300506.

Steinhaus, H. 1949. Sur La Division Pragmatique. *Econometrica*, 17(1): 315–319.

Varian, H. R. 1974. Equity, envy, and efficiency. *Journal of Economic Theory*, 9(1): 63 – 91.

Vazirani, V. V.; and Yannakakis, M. 2021. Computational Complexity of the Hylland-Zeckhauser Scheme for One-Sided Matching Markets. In *Proceedings of the 12th Innovations in Theoretical Computer Science Conference, (ITCS)*.

Woeginger, G. J. 1997. A polynomial-time approximation scheme for maximizing the minimum machine completion time. *Operations Research Letters*, 20(4): 149 – 154.

A Missing Proofs from Section 3

Lemma 3. *Algorithm 1 returns in* $\text{poly}(n, m)$*-time a market outcome* $(\mathbf{x}, \mathbf{p})$ *with agents partitioned into groups* $N_1, \ldots, N_R$*, with the following properties:*

(i) For all low-cost chores $j \in M$, $p_j = 1$, and for all high-cost chores $j \in M$, $p_j = k$.

(ii) The mBB ratio α_i of every agent i is 1.

(iii) Let H_r be the collection of agents N_r and chores allocated to them in $(\mathbf{x}, \mathbf{p})$. Then each H_r is a *partial component* of some agent. That is, for each $r \in [R]$, there is an agent $h_r \in H_r$ s.t. H_r comprises of all agents and chores not in $\bigcup_{r' < r} H_{r'}$ reachable through alternating paths from h_r. Further, h_r is the big spender among agents not in $\bigcup_{r' < r} H_{r'}$:

$$h_r \in \mathsf{argmax}_{i \notin (\bigcup_{r' < r} H_{r'})} \min_{j \in \mathbf{x}_i} \mathbf{p}(\mathbf{x}_i \setminus j)$$

(iv) The spending (up to removal of the largest chore) $f(r)$ of the big spender in H_r weakly decreases with r. Here $f(r) = \max_{i \in H_r} \min_{j \in \mathbf{x}_i} \mathbf{p}(\mathbf{x}_i \setminus j)$.

(v) Each *group is pEF1*, i.e., an agent does not pEF1-envy other agents in the same group.

(vi) For every $i \in H_r$ and $j \in H_{r'}$ with $r' < r$, $c_{ij} = k$.

(vii) All high-cost chores belong to H_R.

Proof. Properties (i) and (ii) follow from the construction of the initial allocation in Line 1.

Property (iii) follows from the construction of the set H_r in Line 10. The agent h_r is the agent b chosen in Line 4. Property (iv) then follows from the last result of (iii).

To see why (v) holds, we examine the loop in Lines 6-9. This loop terminates only once the big spender b in the component does not pEF1-envy any other agent in the component. Since every agent spends (up to the removal of one chore) less than the big spender, it must be that every agent does not pEF1-envy any other agent in the component. Thus, the component is pEF1.

Next, suppose for some $i \in H_r$ and some $j \in H_{r'}$ for $r' < r$, $c_{ij} = 1$. Then j is a low-cost chore and must be priced at 1. Hence, $\alpha_{ij} = c_{ij}/p_j = 1 = \alpha_i$, implying that $j \in \mathrm{mBB}_i$. However this means that i would have been added to $H_{r'}$, since there is an alternating path from $h_{r'}$ to i via $j \in H_{r'}$. This is a contradiction, thus showing (vi).

Next, we show (vii). Suppose a high-cost chore j belongs to H_r for $r < R$. We know j is priced at k. Then for an agent $i \in H_R$, $\alpha_{ij} = c_{ij}/p_j = 1 = \alpha_i$, implying that $j \in \mathrm{mBB}_i$. However this means i belongs to H_r since there is an alternating path from h_r to i via j, which is a contradiction.

Finally, we argue that in making a group starting with the a new big spender in N' outside of previously established pEF1 components, we do not disturb the previously established groups. Let N_r be the current group being made, and let $N_{r'}$, where $r' < r$ be a previously established component. Let b_r and $b_{r'}$ be the big spenders in N_r and $N_{r'}$, respectively. We show that no agent in $N_{r'}$ gains or loses a chore. Note that there are no high-cost chores in $N_{r'}$, as high-cost chores are mBB for all agents so a group containing a high-cost chore must be the last group. Since all chores are priced 1 and $N_{r'}$ is pEF1, every agent in $N_{r'}$ has total spending either $\mathbf{p}(\mathbf{x}_{b'})$ or $\mathbf{p}(\mathbf{x}_{b'}) - 1$. We also have that $\mathbf{p}(\mathbf{x}_b) \leq \mathbf{p}(\mathbf{x}_{b'})$ by construction. Thus, it cannot be that b pEF1-envies any agent in $N_{r'}$, so agents in $N_{r'}$ will not receive any chores. Agents in $N_{r'}$ also will not lose any chores, since they can only transfer chores to another agent in $N_{r'}$. However, we have already shown that agents in $N_{r'}$ cannot receive chores since they will not be pEF1-envied by b,

so these transfers are impossible. Thus, all previous groups must be undisturbed while establishing a later group. □

Lemma 4. *Algorithm 1 terminates in time* $\mathsf{poly}(n, m)$.

Proof. Consider the time needed to make the component of the big spender pEF1. We show that this takes time $\mathsf{poly}(n, m)$. Note that when the identity of the big spender b does not change, a transfer of chore j strictly increases the level of j. Thus, while the big spender b is unchanging, chore j can be transferred at most n times. Thus, the total number of chore transfers with the same big spender is at most mn.

We now bound the total number of times the identity of the big spender may change. Let $\mathbf{x}^t$ be the allocation at iteration t, and $\mathbf{p}$ be the price vector. Furthermore, let $j_{(i,t)}$ denote the most expensive chore of agent i at time t.

We claim that if an agent b ceases to be the BS at iteration t and subsequently becomes the BS again at t', then her spending (up to removal of one chore) strictly goes down, i.e., $\mathbf{p}(\mathbf{x}_b^{t'} \setminus j_{(b,t')}) < \mathbf{p}(\mathbf{x}_b^t \setminus j_{(b,t)})$. To see this, suppose for sake of contradiction that $\mathbf{p}(\mathbf{x}_b^{t'} \setminus j_{(b,t')}) \geq \mathbf{p}(\mathbf{x}_b^t \setminus j_{(b,t)})$. Note that b must lose some chore at time t to cease being the BS. Then, since the spending of b up to one chore at time t' is greater than at time t, b must also have gained a chore between t and t'. Let the most recent time b gained a chore be $t'' \in (t, t')$, and let b' (resp. b'') be the BS at time t' (resp. t''). Note that the spending of the big spender up to one chore weakly decreases with each iteration, so $\mathbf{p}(\mathbf{x}_{b'}^{t'} \setminus j_{(b',t')}) \leq \mathbf{p}(\mathbf{x}_{b''}^{t''} \setminus j_{(b'',t'')}) \leq \mathbf{p}(\mathbf{x}_b^t \setminus j_{(b,t)})$. Since b receives a chore j at time t'', b must be envied by b'' at time t'' (Line 6). This gives us $\mathbf{p}(x_b^{t''}) < \mathbf{p}(\mathbf{x}_{b''}^{t''} \setminus j_{(b'',t'')})$. As j is the last chore gained by b, we have $\mathbf{p}(\mathbf{x}_b^{t'} \setminus j_{(b,t')}) \leq \mathbf{p}(\mathbf{x}_b^{t'} \setminus j) \leq \mathbf{p}(\mathbf{x}_b^{t''}) < \mathbf{p}(\mathbf{x}_{b''}^{t''} \setminus j_{(b'',t'')}) \leq \mathbf{p}(\mathbf{x}_b^t \setminus j_{(b,t)})$, giving us our contradiction.

Thus, if an agent i ceases to be the BS at time t and later becomes the BS again at t', the spending of i is strictly smaller at t' than at t. It follows that i has strictly smaller disutility at t' since the prices have not been changed. In any allocation $\mathbf{x}$, if s_i (resp. t_i) is the number of chores in $\mathbf{x}_i$ with cost b (resp. a) by i, the disutility of i is $u_i = s_i + t_i k$. Since $0 \leq s_i, t_i \leq m$, the number of different disutility values i can have in any allocation is at most $O(m^2)$. Thus, for any agent i, the number of times her disutility decreases is at most $O(m^2)$. Thus, an agent can become the big spender only $O(m^2)$ times. Hence, the number of identity changes of the BS is at most $O(m^2 n)$.

Thus, the time needed to make the component of the big spender pEF1 is $O(m^3 n^2)$. Since we need construct at most n pEF1 components, the total time needed is $O(m^3 n^3)$. In conclusion, Algorithm 1 terminates in $\mathsf{poly}(n, m)$ time. □

Lemma 5. *The spending of the least spender does not decrease in the run of Algorithm 2.*

Proof. The spending of the LS cannot decrease during a price-rise step. Suppose t is a chore transfer step and s be the spending of the LS at t. If $t \leq T$, transfers are from the BS b to the LS ℓ. Further, this happens only if b pEF1-envies ℓ. Thus, b cannot spend less than s after the chore

transfer due to the pEF1 condition. If $t > T$, chore transfers are along alternating paths $b \to j' \to i \to j \to \ell$, where $j' \in \mathbf{x}_b^t \cap \text{mBB}_i$ and $j \in \mathbf{x}_i^t \cap \text{mBB}_\ell$, and b pEF1-envies ℓ. As discussed in Section 3.4, the spending of i does not change during this chore transfer, hence i cannot spend less than s after the transfer. As before, b cannot spend less than s after the transfer due to the pEF1 condition. Thus, the spending of the least spender does not decrease in the run of Algorithm 2. □

Lemma 6. *Throughout the run of Algorithm 2, each group N_r remains pEF1.*

Proof. We prove this lemma in two parts: before T, and after T, the latter is proved in Lemma 13 (iii). For the run of the algorithm prior to T, we proceed by induction. As the base case, we know from Lemma 3 that the groups $\{N_r\}_{r \in [R]}$ are each pEF1 at the end of MakeInitGroups. As inductive hypothesis, assume at time-step t, each group is pEF1. After the event at t, a group can cease to be pEF1 potentially because of a (i) chore transfer, or (ii) price-rise.

1. *Chore transfer step.* From the algorithm it is clear that any chore transfer prior to T happens from the big spender b to ℓ. Let N_r be a group. We consider four cases:
   - Both b and ℓ lie outside N_r, then the bundles of agents in N_r are unaffected, and the allocation continues to remain pEF1 after the transfer.
   - Both b and ℓ lie in N_r. However, if this happens then the algorithm terminates with a pEF1 allocation, since N_r is pEF1 at time-step t by assumption.
   - b lies in N_r and ℓ does not. Let $i \neq b$ be an agent in N_r and $(\mathbf{x}, \mathbf{p})$ be the allocation at t. Since N_r is pEF1, we have $\mathbf{p}(\mathbf{x}_i) \geq \mathbf{p}(\mathbf{x}_b \setminus j)$ for some $j \in \mathbf{x}_b$. Since N_r is pEF1, b does not pEF1-envy i at t. Since b loses a chore at t, b does not pEF1-envy i after the transfer. Further, since b is the big spender in H_r, $\mathbf{p}(\mathbf{x}_i \setminus j') \leq \mathbf{p}(\mathbf{x}_b \setminus j)$ for some $j' \in \mathbf{x}_i$. Thus, $\mathbf{p}(\mathbf{x}_i' \setminus j') \leq \mathbf{p}(\mathbf{x}_b')$, where $\mathbf{x}'$ is the allocation after the transfer. Hence, i does not pEF1-envy b after the transfer. Thus, N_r remains pEF1 after the transfer.
   - ℓ lies in N_r and b does not. Let $i \neq \ell$ be an agent in N_r and $(\mathbf{x}, \mathbf{p})$ be the allocation at t. Since N_r is pEF1, we have $\mathbf{p}(\mathbf{x}_i \setminus j) \leq \mathbf{p}(\mathbf{x}_\ell)$ for some $j \in \mathbf{x}_i$. Since ℓ gains a chore j' at time t, the allocation $\mathbf{x}'$ after the event at t satisfies $\mathbf{p}(\mathbf{x}_\ell' \setminus j') = \mathbf{p}(\mathbf{x}_\ell) \leq \mathbf{p}(\mathbf{x}_i) = \mathbf{p}(\mathbf{x}_i')$. Hence ℓ does not pEF1-envy i after the transfer. Further, since ℓ gains an additional chore and N_r is pEF1 at t, i will not pEF1-envy ℓ after the transfer. Thus, N_r remains pEF1 after the transfer.

2. *Price-rise step.* Suppose N_r undergoes a price-rise at t. At t, N_r is pEF1 by assumption. Since the prices of all chores in N_r are raised by the same factor, it continues to be pEF1 after the price-rise.

In both cases, we see that N_r continues to remain pEF1 after the event at t. We conclude by induction that each group remains pEF1 throughout the run of the algorithm. □

We denote by $(\mathbf{x}^t, \mathbf{p}^t)$ the allocation and price vector at time-step t.

Lemma 8. *If at any point in the run of Algorithm 2 prior to T, the big spender lies in a group which contains a former least spender, then the allocation is pEF1.*

Proof. Let t be the first time-step when the big spender (BS) b lies in a group N_r which also contains an agent i, who was a least spender (LS) most recently at time-step $t' \leq t$. If $t' = t$, then N_r contains both LS and the BS, in which case the allocation must be pEF1 due to Lemma 6. Thus, $t' < t$.

Suppose N_r underwent a price-rise at a time-step t'' between t' and t, then the BS h at t'' must be in N_r. If that happens, then at t'', the big spender lies in a group which also contains a former LS i. Since $t'' < t$, this contradicts the definition of t. Thus, N_r does not undergo a price-rise between t' and t. We next show that:

Claim 1. *b cannot be a former least spender, unless the allocation is already pEF1.*

Proof. For sake of contradiction, assume b was an LS most-recently at time-step $t_0 < t$. From the properties of the algorithm (Lemma 7), b ceased to be an LS at t_0 because she received a chore j. Subsequent to this, b cannot receive another chore, since t_0 is the most recent time-step when b was an LS, and only an LS can receive a chore (Lemma 7). Thus we must have $\mathbf{x}_b^t \subseteq \mathbf{x}_b^{t_0} \cup \{j\}$. Further, $b \in H_r$ does not undergo price-rise between t_0 and t, otherwise it would contradict the definition of t. Hence we obtain: $\mathbf{p}^t(\mathbf{x}_b^t \setminus j) \leq \mathbf{p}^{t_0}(\mathbf{x}_b^{t_0})$.

Let ℓ be the LS at t. From Lemma 5, we know $\mathbf{p}^t(\mathbf{x}_\ell^t) \geq \mathbf{p}^{t_0}(\mathbf{x}_b^{t_0})$, since b was the LS at t_0. Putting it together, we get $\mathbf{p}^t(\mathbf{x}_\ell^t) \geq \mathbf{p}^t(\mathbf{x}_b^t \setminus j)$, i.e., $(\mathbf{x}, \mathbf{p})$ is pEF1 at t, since b is the BS at t. □

With Claim 1 in hand, we can now prove Lemma 8. Since b is not a former LS, b has not received a chore. Also, b has not undergone a price-rise. Hence: $\mathbf{p}^t(\mathbf{x}_b^t \setminus j) \leq \mathbf{p}^{t'}(\mathbf{x}_b^{t'} \setminus j') \leq \mathbf{p}^{t'}(\mathbf{x}_i^{t'})$, for some $j \in \mathbf{x}_b^t$ and $j' \in \mathbf{x}_b^{t'}$, since H_r is pEF1 at t' by Lemma 6. Finally, by Lemma 5, we get $\mathbf{p}^{t'}(\mathbf{x}_i^{t'}) \leq \mathbf{p}^t(\mathbf{x}_\ell^t)$, where ℓ is the LS at t. Putting these together, we get: $\mathbf{p}^t(\mathbf{x}_\ell^t) \geq \mathbf{p}^t(\mathbf{x}_b^t \setminus j)$, showing that $(\mathbf{x}, \mathbf{p})$ is pEF1 at t. □

Lemma 11. *At T, agents in $N_{<r}$ are pEF1 towards others.*

Proof. Let $t \leq T$ be the time-step at which N_r was raised, i.e., the most recent price-rise. Let h be the big spender at t, we know $h \in H_r$. Let i be an arbitrary agent in $N_{<r}$.

Since i is in a raised group $N_{r'}$, where $r' < r$, i cannot have gained a chore before T. Let $t' < t$ be the time at which $N_{r'}$ was raised. Then:

- if i gained a chore before t', then $N_{r'}$ cannot have been undisturbed at t'.
- if i gained a chore at time $t'' \in (t', T)$, then i had to have been the LS at t''. Hence at $t'' < T$, the LS was in a raised group. This contradicts the definition of T.

10

Hence i can only lose chores. Just after t', all chores of i are priced at k. Hence the spending of i after t' is always a multiple of k. Further, until t, h is undisturbed, and contains only chores of price 1. Let $c = \mathbf{p}^t(\mathbf{x}_h^t)$ be the number of (price 1) chores initially assigned to h.

Note that $N_{r'}$ must contain an agent who initially has at least c price 1 chores, since $N_{r'}$ undergoes price-rise before N_r. Since each group is pEF1, i must have at least $c-1$ price 1 chores. Hence just after t', we have $\mathbf{p}^{t'}(\mathbf{x}_i^{t'}) \geq k(c-1)$. After t', the spending of i potentially drops because of transfer of chores of price k.

At t, since h is the BS, we know: $\mathbf{p}^t(\mathbf{x}_i^t) - k \leq \mathbf{p}^t(\mathbf{x}_h^t) - 1$. Since the spending of i is a multiple of k, we have:

$$\mathbf{p}^t(\mathbf{x}_i^t) \leq k + k \cdot \left\lfloor \frac{\mathbf{p}^t(\mathbf{x}_h^t) - 1}{k} \right\rfloor = k + k \cdot \left\lfloor \frac{c-1}{k} \right\rfloor$$

Just before the time before t that i lost her last chore, since h is not the BS, the spending of i has to be at least $k+(c-1)$. Hence the spending of i at t is gk where:

- if $k|(c-1)$, then $g \in \{(c-1)/k, 1 + (c-1)/k\}$
- otherwise, $g = 1 + \lfloor (c-1)/k \rfloor$

Since i is an arbitrary agent in $N_{<r}$, we see that every agent in $N_{<r}$ spends almost the same amount (up to k). Hence there is no pEF1-envy among agents of $N_{<r}$ at t. From t to T, agents in $N_{<r}$ only lose chores, and the gap between the spendings of any two agents cannot ever exceed k. Hence at T, agents in $N_{<r}$ do not pEF1-envy each other. Since the LS is in $N_{<r}$ at T, we conclude that agents in $N_{<r}$ do not pEF1-envy any other agent. □

We see from the previous analyis that $\mathbf{p}^t(\mathbf{x}_i^t) \geq c-1$, where $c = \mathbf{p}^t(\mathbf{x}_h^t)$.

Lemma 12. *At T, agents in $N_{>r}$ are pEF1 towards others.*

Proof. Let $t \leq T$ be the time-step at which N_r was raised, i.e., the most recent price-rise. Let i be an arbitrary agent in $N_{>r}$. Since i is in an unraised group, i cannot have undergone a price-rise or ever lost a chore.

Suppose t' is the last time i gained a chore j. Then at t', i is a LS. Thus, $\mathbf{x}_i^{t'} \cup \{j\} = \mathbf{x}_i^T$, and $\mathbf{p}^T(\mathbf{x}_i^T \setminus j) = \mathbf{p}^t(\mathbf{x}_i^t)$. Since the spending of the LS weakly increases, $\mathbf{p}^t(\mathbf{x}_i^t) \leq \mathbf{p}^T(\mathbf{x}_\ell^T)$, where ℓ is the LS at T. Hence we obtain: $\mathbf{p}^T(\mathbf{x}_i^T \setminus j) \leq \mathbf{p}^T(\mathbf{x}_\ell^T)$, showing that i does not pEF1-envy the LS.

Suppose i did not ever gain a chore. Then, the bundle of i has been undisturbed from the start. Since h is the BS at t, we have $\mathbf{p}^T(\mathbf{x}_i^T \setminus j') = \mathbf{p}^t(\mathbf{x}_i^t \setminus j') \leq c-1$, for some $j' \in \mathbf{x}_i^T$. We know that ℓ spends at least $c-1$ at t. We know ℓ does not gain a chore. Is it possible that ℓ loses a chore after t?

Suppose ℓ loses a chore j for the last time at $t' \in (t, T)$. At t', ℓ was the BS. Since there are no price-rises after t, the spending of the BS upto removal of once chore weakly decreases. Hence $\mathbf{p}^T(\mathbf{x}_b^T \setminus j') \leq \mathbf{p}^{t'}(\mathbf{x}_\ell^{t'} \setminus j)$, where b is the BS at T. Since ℓ doesn't lose a chore after t', $\mathbf{p}^{t'}(\mathbf{x}_\ell^{t'} \setminus j) = \mathbf{p}^T(\mathbf{x}_\ell^T)$. Thus the allocation is pEF1.

If ℓ did not lose any chore after t, then $\mathbf{p}^T(\mathbf{x}_\ell^T) = \mathbf{p}^t(\mathbf{x}_\ell^t) \geq c-1$. Since $\mathbf{p}^T(\mathbf{x}_i^T \setminus j') \leq c-1$, for some

$j' \in \mathbf{x}_i^T$ if i has not gained a chore, we see that in this case too i does not pEF1-envy the LS at T.

Since i was an arbitrary agent in $N_{>r}$, we conclude that no agent in $N_{>r}$ pEF1-envies any other agent. □

Lemma 13. *After T, the following are invariant:*

(i) Agents in $N_{<r}$ do not pEF1-envy any other agent.
(ii) Agents in $N_{>r}$ do not pEF1-envy any other agent.
(iii) Each group is pEF1.

Proof. Recall that Lemmas 11 and 12 shows that (i) and (ii) hold just after T, and Lemma 6 shows (iii) holds just after T. With this as our base case, we proceed by induction. Assume (i)-(iii) hold at some time step $t > T$. The event at t can either be:

- A direct chore transfer from $b \in N_r$ to $\ell \in N_{>r}$. Such a transfer does not newly cause any agent to pEF1-envy ℓ since ℓ gains a chore, nor any to agent pEF1-envy b since b is the BS. Also, b does not newly pEF1-envy any agent since she loses a chore, and neither does ℓ newly pEF1-envy any agent since she is the LS.
- A chore transfer of the form $b \rightarrow j' \rightarrow i \rightarrow j \rightarrow \ell$, where $i \in N_{>r}$, $j' \in \mathbf{x}_b^t$, $j \in \mathbf{x}_i^t$, and j is a chore that ℓ lost prior to T. Hence $p_j^t = p_{j'}^t = k$. Hence, $\mathbf{p}^{t+1}(\mathbf{x}_i^{t+1}) = \mathbf{p}^t(\mathbf{x}_i^t)$. Thus, this step effectively amounts to transferring a chore from b to ℓ. As argued in the previous paragraph, this does not cause any agent to newly start pEF1-envying another agent.

Hence, by induction, (i)-(iii) hold after t as well. □

B Missing Proofs from Section 4

Theorem 2. *Given a bivalued fair division instance (N, M, C) of divisible chores with all $c_{ij} \in \{a, b\}$ for some $a, b \in \mathbb{R}^+$, an EF+PO allocation can be computed in strongly polynomial-time.*

We prove Theorem 2 by showing that our Algorithm 4 computes an EF+PO allocation in polynomial-time.

B.1 Describing the Approach

First note that we can scale the costs of the chores so that they are in $\{1, k\}$. Since we are interested in an EF+PO allocation, we try to construct a (fractional) market outcome $(\mathbf{x}, \mathbf{p})$ on mBB which is price-envy-free (pEF) as opposed to pEF1. A market outcome $(\mathbf{x}, \mathbf{p})$ is said to be pEF if for all agents $i, h \in N$, $\mathbf{p}(\mathbf{x}_i) \geq \mathbf{p}(\mathbf{x}_h)$, i.e., all agents have equal spending. Similar to Lemma 1, we can show that if $(\mathbf{x}, \mathbf{p})$ is a pEF market outcome on mBB, then $\mathbf{x}$ is EF. Thus, our aim is to obtain a market outcome on mBB in which all agents have the same spending.

As in Algorithm 2, we begin with an initial market outcome $(\mathbf{x}, \mathbf{p})$ on mBB, with certain desirable properties. Specifically, we partition the agents into groups $N_1, \ldots, N_R$ so that each group spends the same amount, each group is a partial component of some agent, and the per-agent spending of the group N_r decreases with r. Now, if the allocation is not pEF, it must be the case that any agent in the *pool* of biggest spenders ($B = \{b : b \in \text{argmax}_{i \in N} \mathbf{p}(\mathbf{x}_i)\}$)

Algorithm 3: MakeInitGroupsDiv

Input: Fair division instance (N, M, C) with $c_{ij} \in \{1, k\}$
Output: Fractional alloc. $\mathbf{x}$, prices $\mathbf{p}$, agent groups $\{N_r\}_{r \in [R]}$

1: $(\mathbf{x}, \mathbf{p}) \leftarrow$ initial cost minimizing balanced flow fractional market allocation, where $p_j = c_{ij}$ for $j \in \mathbf{x}_i$.
2: $R \leftarrow 1, N' \leftarrow N$
3: **while** $N' \neq \emptyset$ **do**
4:      $b \leftarrow \mathrm{argmax}_{i \in N'} \mathbf{p}(\mathbf{x}_i)$ $\triangleright$ Biggest Spender
5:      $C_b \leftarrow$ Component of b $\triangleright$ See Definition 2
6:      $H_R \leftarrow C_b \cap (N' \cup \mathbf{x}_{N'})$ $\triangleright$ Partial component
7:      $N_R \leftarrow H_R \cap N$ $\triangleright$ Agent group
8:      $N' \leftarrow N' \setminus N_R, R \leftarrow R + 1$
9: **return** $(\mathbf{x}, \mathbf{p}, \{N_r\}_{r \in [R]})$

---

envies any agent in the pool of least spenders ($L = \{\ell : \ell \in \mathrm{argmin}_{i \in N} \mathbf{p}(\mathbf{x}_i)\}$), i.e., $\mathbf{p}(\mathbf{x}_b) > \mathbf{p}(\mathbf{x}_\ell)$ for any $b \in B$ and $\ell \in L$. The natural idea following Algorithm 2 is to reduce the pEF-envy by *draining* chores uniformly from B to L along mBB edges. Further, if the pool B has chores which are not on mBB for L, then we raise the prices of certain chores in B by a factor of k, which creates the required mBB edges following Lemma 9. Initially, B equals N_1, and L equals N_R. At a certain point in the run of the algorithm, B equals $\bigcup_{i \leq r} N_i$ and L equals $\bigcup_{i \geq r'} N_{r'}$, where $r \leq r'$. When eventually the pools of biggest and least spenders coincide, the algorithm terminates with a pEF allocation. This approach is similar to Algorithm 2 and terminates in polynomial time.

However, we observe a stronger property of this approach, which we use to design Algorithm 4. We claim that effectively, this approach is equivalent to raising the prices of all chores in the first r^* groups, and then draining chores carefully, eventually resulting in a pEF allocation. We can simply iterate over $r = 1$ to $R - 1 \leq n$ as a guess for a right value of r^*. We now describe the parts of the algorithm and also prove the above argument.

B.2 Obtaining Initial Groups

Algorithm 3 obtains a partition of the set N of agents into groups $N_1, \ldots, N_R$ similar to Algorithm 1, with the strengthened condition that the partial components are price envy-free (pEF) rather than pEF1. The price vector $\mathbf{p}$ is set the same way as in Algorithm 1. Then all low-cost chores are priced at 1 and high-cost chores are priced at k. To allocate these chores, however, we use a balanced flow formulation (Devanur et al. 2008). With equal and sufficient endowments, this gives us an allocation with the key property that if agent i spends more than agent j, then there is no alternating path from i to j. Then, each b selected in Line 4 of Algorithm 3 is a biggest spender in its partial component, but also cannot spend more than any agent in its partial component. Thus, it must be that the partial component is pEF. Algorithm 3 runs in polynomial time as the balanced flow allocation is achieved by at most m max-flow computations and the remaining structure is analogous to Algorithm 1.

B.3 Discussion on Algorithm 4

Algorithm 4 proceeds similarly to Algorithm 2 with two key differences:

(1) Groups are raised simultaneously rather than one at a time.

(2) Transfer are made between a *pool of biggest spenders* and a *pool of least spenders*. That is, we transfer chores from *all* biggest spenders to *all* least spenders.

We show that there are always exists some $r^*, 1 \leq r \leq R - 1$, such that simultaneously raising the first r^* groups allows us to reach a pEF allocation by performing chore transfers along mBB edges from biggest spenders to least spenders. Recalling that the partial components associated with each group N_i are pEF, let $s(N_i)$ denote the *per agent spending* of N_i. By construction, we have that $s(N_i) \geq s(N_j)$ for $i \leq j$. Suppose we do an initial raise of the first r groups. The pool of biggest spenders B must be some union of raised groups $\bigcup_{i=1}^{\beta} N_i$, and the pool of least spenders L must be some union of unraised groups $\bigcup_{i=\lambda}^{R} N_i$. Algorithm 4 then proceeds to transfer chores out of agents in B at a uniform rate ρ_B and into agents in L at a uniform rate ρ_L, such that $|B|\rho_B = |L|\rho_L$. Then, $s(B)$ will fall towards $s(N_{\beta+1})$ while $s(L)$ rises towards $s(N_{\lambda-1})$. When we transfer enough chores so that $s(B)$ reaches $s(N_{\beta+1})$ or $s(L)$ reaches $s(N_{\lambda-1})$, whichever occurs first, we add the agents in that group to the respective pool. Note that in this process B and L are only growing. Once an agent enters B or L, she never leaves. In addition, we have that $s(B)$ is always decreasing while $s(L)$ is always increasing. When we have $s(B) = s(L)$, the allocation is pEF. However, to guarantee that transfers occur along mBB edges, we must have that all agents in B are in raised groups, and all agents in L are in unraised groups. Thus, we give the two conditions under which the algorithm *cannot* obtain a pEF allocation after raising the first r groups, as described earlier in Section 4.

- *Condition 1.* Raised groups have *too much* total spending. The spending of agents in L rises to spending level of N_r before the spending of agents in B falls to the level of N_r. That is, a raised group enters L.

- *Condition 2.* Raised groups have *too little* total spending. The spending of agents in B falls to the level of N_{r+1} before the spending of agents in L rises to the level of N_{r+1}. That is, an unraised group enters B.

If neither of these conditions hold, then B always contains only raised groups, and L always holds only unraised groups. It is clear then that we can perform transfers until the spending gap between B and L closes entirely and we have a pEF allocation. Specifically, this is possible when the algorithm reaches a point where B contains all the raised groups and L contains all the unraised groups (see Line 12). We show that is guaranteed for some $r^*, 1 \leq r^* \leq R - 1$.

Suppose we raise only the single group N_1. It cannot be that Condition 1 holds here, as L rising to the level of N_1 implies that the allocation is pEF. It must be that we have Condition 2, or we would be done as the allocation will be pEF and the algorithm would halt.

Algorithm 4: Computing an EF+fPO allocation

Input: Fair division instance (N, M, C) with $c_{ij} \in \{1, k\}$
Output: A fractional allocation $\mathbf{x}$

1: **for** $r \in [R-1]$ **do**
2:      $(\mathbf{x}, \mathbf{p}, \{N_r\}_{r \in [R]}) \leftarrow \mathsf{MakeInitGroupsDiv}(\mathsf{N}, \mathsf{M}, \mathsf{C})$
3:      Raise prices of chores of agents in $\{N_i\}_{i \in [r]}$ by a factor of k
4:      $B \leftarrow N_1$ ▷ Pool of Biggest Spenders
5:      $\beta \leftarrow 1$ ▷ Index of lowest group in B
6:      $L \leftarrow N_R$ ▷ Pool of Least Spenders
7:      $\lambda \leftarrow R$ ▷ Index of highest group of L
8:      **while** $\beta \leq r$ **and** $\lambda \geq r+1$ **do**
9:          $d_B \leftarrow |B| \cdot (s(B) - s(N_{\beta+1}))$
10:         $d_L \leftarrow |L| \cdot (s(N_{\lambda-1})) - s(L))$
11:         **for** agents $b \in B$ **do**
12:            **if** $\beta = r$ **and** $\lambda = r+1$ **then**
13:               $q \leftarrow \frac{|B| \cdot s(B) + |L| \cdot s(L)}{|N|}$
14:               $J \leftarrow$ subset of $\mathbf{x}_b$ s.t. $\mathbf{p}(J) = s(B) - q$
15:            **else if** $d_B \geq d_L$ **then**
16:               $J \leftarrow$ subset of $\mathbf{x}_b$ s.t. $\mathbf{p}(J) = \frac{d_L}{|B|}$
17:            **else**
18:               $J \leftarrow$ subset of $\mathbf{x}_b$ s.t. $\mathbf{p}(J) = \frac{d_B}{|B|}$
19:            **for** agents $\ell \in L$ **do**
20:               $\mathbf{x}_b \leftarrow \mathbf{x}_b \setminus \frac{1}{|L|} J$
21:               $\mathbf{x}_\ell \leftarrow \mathbf{x}_\ell \cup \frac{1}{|L|} J$
22:         **if** $(\mathbf{x}, \mathbf{p})$ is pEF **then**
23:            **return** $(\mathbf{x}, \mathbf{p})$
24:         **else if** $d_B \geq d_L$ **then**
25:            $L \leftarrow L \cup N_{\lambda-1}$
26:            $\lambda \leftarrow \lambda - 1$
27:         **else**
28:            $B \leftarrow B \cup N_{\beta+1}$
29:            $\beta \leftarrow \beta + 1$

Now suppose inductively that we have raised the first r groups and found that the allocation is not pEF and Condition 1 is impossible for the first r groups, i.e., Condition 2 holds. Then, it must also be that Condition 1 is impossible if we raise the first $r+1$ groups if the allocation is still not pEF. Let $p = s(N_{r+1})$ be the per-agent spending of N_{r+1} just before N_{r+1} undergoes price-rise. Since raising r groups results in Condition 2, we know that $s(B)$ falls to p *before* $s(L)$ rises to p. Surely then, in raising $r+1$ groups, $s(B)$ falls to pk before $s(L)$ rises to pk, as $pk > p$. So this shows Condition 1 cannot hold after raising the first $r+1$ groups, given that the allocation is not yet pEF.

Finally, suppose that we have raised the first $R-1$ groups, i.e., all but the last group, and the allocation is not pEF. Then, Condition 2 cannot hold, as $s(B)$ falling to the level of $s(N_R)$ implies that the allocation is pEF. Thus, if the allocation is not pEF, Condition 1 must hold, but this is impossible by our inductive argument. Hence it must be that for some r^*, $1 \leq r^* \leq R-1$, a pEF allocation was obtained.

Following this logic, Algorithm 4 checks each possible r until r^* is found and a pEF allocation is obtained. Further,

since all chore transfers along mBB, the allocation remains PO throughout. Thus the algorithm computes a pEF+PO allocation.

For each $r \in [R]$, the algorithm performs at most $O(n^2)$ transfers before adding an agent group to either B or L. A group cannot be added to both B and L, nor can it leave B or L after being added. Thus, since $R \leq n$, the total number of transfers performed to check if some r is correct is $O(n^3)$, so checking all possible r takes $O(n^4)$ transfers. Hence, Algorithm 4 terminates in strongly polynomial-time, proving Theorem 2.

B.4 Pseudocode Description

We describe the details of the pseudocode of Algorithm 4.

As described in Section B.1, Algorithm 4 iteratively searches for a right value r^* of the number of groups to raise. This corresponds to the for loop starting in Line 1. Line 2 calls Algorithm 3 to obtain initial groups as described in Section B.2.

Lines 4-5 initialize the pool of biggest spenders B to N_1 and set the index of the lowest group in B, β to 1. Lines 6-7 initialize the pool of least spenders L to N_R and set the index of the lowest group in L, λ to R.

The loop starting in Line 8 is active while the pool of biggest spenders doesn't contain an unraised component ($\beta \leq r$) and the pool of least spenders doesn't contain a raised component ($\lambda \geq r+1$). We then want to decide how much spending (chores) to *drain* from B to L. As described in Section B.3, either the per-agent spending of B can climb down to the per-agent spending of $N_{\beta+1}$, or the per-agent spending of L can climb up to the per-agent spending of $N_{\lambda-1}$. Line 9 computes the total amount of spending d_B to be transferred in the former case and Line 10 computes the total amount of spending d_L to be transferred in the latter case.

The loop starting in Line 11 considers agents $b \in B$ one by one and computes an appropriately priced bundle of chores that should be taken away from them. If B contains all the raised groups and L contains all the unraised groups, i.e., if $\beta = r$ and $\lambda = r+1$ (Line 12), then we are a step away from obtaining a pEF allocation. We compute the average spending of all agents, q in Line 13. This is the amount to which we want to bring every agent's spending. Hence, for each agent b in consideration, we compute a subset $J \subseteq \mathbf{x}_b$ with value $s(B) - q$ (Line 14), and distribute it evenly to all agents in L (Lines 20-21). Here, $\frac{1}{|L|} J$ denotes a (fractional) bundle of chores J' where $J'_{ij} = \frac{1}{|L|} J_{ij}$ for agent i and chore j (see Section 2 for the definition of a fractional allocation.)

Otherwise, if $d_B \geq d_L$, then the per-agent spending of L reaches the per-agent spending of $N_{\lambda-1}$ before the per-agent spending of B reaches the per-agent spending of $N_{\beta+1}$. Hence each agent $\ell \in L$ must obtain an additional spending of $\frac{d_L}{|L|}$ from agents in B. We do this by computing a subset J of $\mathbf{x}_b$ of value $\frac{d_L}{|B|}$ from every $b \in B$ (Line 16) and giving a $\frac{1}{|L|}$-sized fraction of J to ℓ (Line 20-21). Thus, ℓ obtains a spending $\frac{1}{|L|} \cdot \frac{d_L}{|B|}$ from every agent in B, which totals to $\frac{d_L}{|L|}$, the required amount. Now, we update the pool of least

spenders to include $N_{\lambda-1}$, and update λ (Lines 25-26).

Analogously, when $d_B < d_L$, the per-agent spending of B reaches the per-agent spending of $N_{\beta+1}$ before the per-agent spending of L reaches the per-agent spending of $N_{\lambda-1}$. Hence, each agent $b \in B$ loses an amount equalling $\frac{d_B}{|B|}$. For each b, we compute a set J with spending $\frac{d_B}{|B|}$ (Line 18), which is then transferred equally among all agents in L (Line 20-21). We also update the pool of biggest spenders to include $N_{\beta+1}$, and update β (Line 28-29).

We terminate if the allocation is pEF (Lines 22-23). If not, then we continue our search for a right value r^* by incrementing r (Line 1). The arguments of Section B.3 show that for some $1 \leq r \leq R - 1$, Algorithm 4 will terminate with a pEF+PO allocation.