Journal of Artificial Intelligence Research 78 (2023) 1201-1219 Submitted 07/2023; published 12/2023

Competitive Equilibria with a Constant Number of Chores

Jugal Garg JUGAL@ILLINOIS.EDU
Peter McGlaughlin MCGLGHL2@ILLINOIS.EDU
University of Illinois, Urbana-Champaign, IL, USA

Martin Hoefer MHOEFER@EM.UNI-FRANKFURT.DE
Marco Schmalhofer SCHMALHOFER@EM.UNI-FRANKFURT.DE
Goethe University, Frankfurt am Main, Hessen, Germany

Abstract

We study markets with mixed manna, where m divisible goods and chores shall be divided among n agents to obtain a competitive equilibrium. Equilibrium allocations are known to satisfy many fairness and efficiency conditions. While a lot of recent work in fair division is restricted to linear utilities and chores, we focus on a substantial generalization to separable piecewise-linear and concave (SPLC) utilities and mixed manna. We first derive polynomial-time algorithms for markets with a constant number of items or a constant number of agents. Our main result is a polynomial-time algorithm for instances with a constant number of chores (as well as any number of goods and agents) under the condition that chores dominate the utility of the agents. Interestingly, this stands in contrast to the case when the goods dominate the agents utility in equilibrium, where the problem is known to be PPAD-hard even without chores.

1. Introduction

The allocation of a set of divisible items to a set of agents in a *fair* and *efficient* manner is the main challenge in fair division, a prominent field in economics with a variety of well-established concepts and techniques (Moulin, 2003). *Algorithms* for fair division have recently prompted a large amount of research interest in AI, due to many important applications arising from computer-aided decision making in various parts of society (Brandt, Conitzer, Endriss, Lang, & Procaccia, 2016, Part II). Standard criteria for fair and efficient allocation in markets include envy-freeness (EF; no agent prefers the bundle of goods from another agent), proportionality (PROP; every agent gets a bundle that has at least her "average" value), and Pareto-optimality (PO; no allocation gives an agent strictly more utility without decreasing another agent's utility). Interestingly, all these criteria are achieved in a *competitive equilibrium from equal incomes (CEEI)*, an equilibrium allocation in a market where every agent has \$1 of (fake) money.

For more than two decades, the computation of competitive equilibria (with and without equal incomes) has been one of the main lines of research in fair division, and, more broadly, at the intersection of economics and computer science (Nisan, Roughgarden, Tardos, & Vazirani, 2007, Chapters 5+6). An intriguing recent development in this area is the consideration of *chores* and, more generally, *mixed manna*. In an allocation problem with mixed manna there are goods and chores. Goods are desired by at least one of the agents (e.g., cake), while chores are undesirable for all agents (e.g., job shifts, cleaning tasks). In particular, chores are not disposable. The goal again is to satisfy fairness criteria

such as EF, PROP, and/or PO. The consideration of mixed manna substantially generalizes our understanding of fair division and represents an intriguing challenge for algorithms to computing such allocations when they exist.

In a seminal contribution (Bogomolnaia, Moulin, Sandomirskiy, & Yanovskaia, 2017) the existence of competitive equilibria under general conditions for instances with mixed manna was established. Moreover, even for mixed manna, CEEI retain their attractive fairness properties. Clearly, this raises a natural question from a computational perspective, which we study in this paper: *Under which conditions can competitive equilibria be computed in polynomial time for markets with mixed manna*?

The answers depend on whether we consider instances with only goods, only chores or, more generally, true mixed manna. For only goods, markets with linear utilities allow even strongly polynomial-time algorithms (Orlin, 2010; Garg & Végh, 2019). For additively separable piecewise-linear concave (SPLC) utilities, the problem is PPAD-hard for both goods (Chen & Teng, 2009) and chores (Chaudhury, Garg, McGlaughlin, & Mehta, 2021).

1.1 Contribution and Outline

In this paper, we provide polynomial-time algorithms for computing competitive equilibria in Fisher markets with mixed manna. The introduction of the formal model and preliminary results are given in Section 2. As a first set of results, we show a polynomial-time[1] algorithm to compute equilibria in markets with SPLC utilities when the number of agents or items (i.e., goods and bads) is constant. SPLC utilities are quite general and applicable as they model natural properties like decreasing marginals while maintaining (piecewise) linearity; see, e.g., (Garg, Mehta, Sohoni, & Vazirani, 2015). The discussion of these results is given in Section 3. We note that this is the first polynomial-time algorithm to compute a competitive equilibrium of mixed manna with linear or SPLC utilities under any assumptions. Our main result is then presented in Section 4 – an efficient algorithm for computing competitive equilibria in *negative* instances with arbitrary many agents, goods, and a constant number of chores. The agents can have SPLC utilities for goods, but we assume linear utilities for chores. Negativity is a condition that implies that chores dominate the utility of the agents (for a formal definition, see Section 2). This is a notable contrast to *positive* instances with SPLC utilities for goods, where computation of an equilibrium is PPAD-hard, even without chores (Chen & Teng, 2009). Interestingly, our result also applies to the symmetric case, i.e., in positive instances, when there is a constant number of linear valued goods and an arbitrary number of SPLC valued bads.

Extended abstracts of this work have appeared in the proceedings of AAMAS 2020 (Garg & McGlaughlin, 2020) and SAGT 2021 (Garg, Hoefer, McGlaughlin, & Schmalhofer, 2021).

1.2 Further Related Work

The literature on fair division with only goods is vast, and a complete review is beyond the scope of this paper. Instead, we refer the reader to the books (Brams & Taylor, 1996; Robertson & Webb, 1998; Moulin, 2003), and focus on the case of mixed manna.

1. Polynomial w.r.t. instance size and number of segments.

Manna Type	Valuations	Restrictions	Complexity	Source
goods	linear	–	strongly poly.	(Orlin, 2010; Garg & Végh, 2019)
goods	SPLC	–	PPAD-hard	(Chen & Teng, 2009)
bads	SPLC	–	PPAD-hard	(Chaudhury et al., 2021)
bads	linear	m or n const	poly.-time	(Branzei & Sandomirskiy, 2019; Chaudhury et al., 2021)
mixed	linear	ϵ-approx. CE	$\tilde{\mathcal{O}}(n^4m^2/\epsilon)$	(Chaudhury, Garg, McGlaughlin, & Mehta, 2022)
mixed	SPLC	m or n const	poly.-time	Theorem 2
mixed	SPLC goods, linear bads	#bads=const, neg. instance	poly.-time	Theorem 3
	linear goods, SPLC bads	#goods=const, pos. instance		

Table 1: Related work on the complexity of competitive equilibria computation in Fisher markets and our results.

While most of the work in fair division focuses on goods, there are a few works for the case of bads, e.g., (Azrieli & Shmaya, 2014; Brams & Taylor, 1996; Robertson & Webb, 1998; Su, 1999). The study of competitive division with a mixed manna was initiated by Bogomolnaia et al. (2017). They establish equilibrium existence and show further properties, e.g., that multiple, disconnected equilibria may exist, and polynomial-time computation is possible if there are either two agents or two items with linear utility functions (Bogomolnaia, Moulin, Sandomirskiy, & Yanovskaia, 2019).

On the algorithmic side, an algorithm to compute a competitive allocation of goods under linear utilities was given by Orlin (2010), and Garg and Végh (2019). Both algorithms have strongly polynomial runtime. For SPLC utilities, the problem was shown to be PPAD-hard for goods by Chen and Teng (2009) and for bads by Chaudhury et al. (2021). An algorithm to compute a competitive allocation in only-bads instances with linear utility functions was given by Branzei and Sandomirskiy (2019). Their algorithm runs in strongly polynomial time if either the number of agents or bads is constant. We generalize this to mixed manna with SPLC utilities without achieving strongly polynomial running time. The key idea in Branzei and Sandomirskiy (2019) is to characterize allocation graphs of equilibrium allocations, and to use the obtained properties for reducing the exponentially large set of possible allocation graphs to a set of polynomial size if the number of agents or items is constant. The reduced set of graphs is then enumerated. We instead, in the constant number of agents case, introduce a threshold value for every agent, which can be directly derived from the optimal bundles condition using Karush-Kuhn-Tucker conditions.

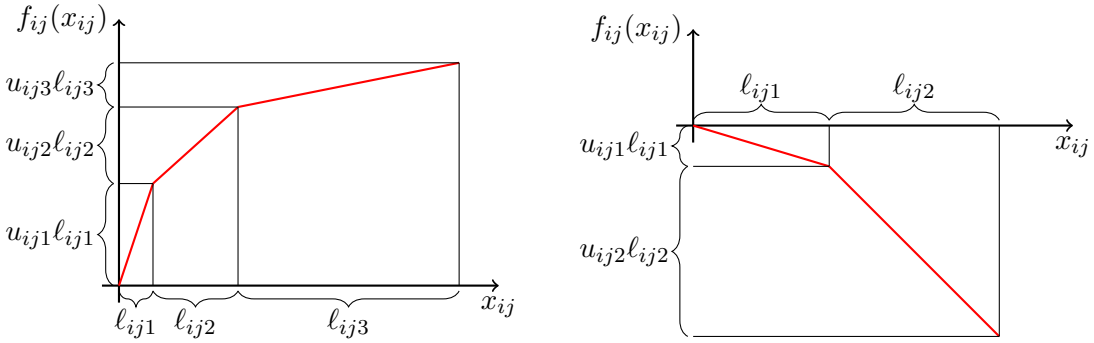

Figure 1: Piecewise linear concave utility $f_{ij}(x_{ij})$ of agent i for an x_{ij}-fraction of *good* j (left), and *bad* j (right).

Then, we use a cell decomposition on these threshold values to enumerate all meaningful allocation graphs. In the constant number of items case, decomposing the space of prices of items allows us to enumerate all meaningful allocation graphs.

Chaudhury et al. (2022) provided an algorithm to compute approximate competitive equilibria in mixed manna instances with linear utilities. Their algorithm achieves ϵ accuracy in time $\tilde{\mathcal{O}}(n^4m^2/\epsilon)$. For SPLC utilities and mixed manna, a simplex-like algorithm was given by Chaudhury et al. (2021). Table 1 summarizes related results and our contribution.

Another related line of research considers matching markets, initiated by the seminal work of Hylland and Zeckhauser (1979). In such a market, the goal is to fractionally allocate each agent exactly one unit of items. Alaei, Khalilabadi, and Tardos (2017) provided a polynomial-time algorithm to compute equilibria for matching markets when the number of agents is constant.

2. Preliminaries

2.1 Fair Division with Mixed Manna

We consider fair division of mixed manna, in which there is a set $N = [n]$ of n agents and a set $M = [m]$ of m divisible items. We strive to divide the items among the agents. Without loss of generality, we may assume that there is a unit amount of each item. A fractional allocation $x = \{x_1, \ldots, x_n\}$ assigns each agent $i \in N$ a bundle of items $x_i = (x_{i1}, \ldots, x_{im})$, where $x_{ij} \in [0, 1]$ is the amount of item j agent i receives. An allocation is feasible if all items are fully assigned, i.e., $\forall j \in M$, $\sum_{i \in N} x_{ij} = 1$. For the rest of the paper, we assume all allocations are feasible and use $\mathcal{X}$ to denote the set of feasible allocations.

Each agent $i \in N$ has a utility function u_i that maps the received bundle to a numerical value. In this work, we assume all utility functions are additively separable over items, piecewise linear, and concave (SPLC). Formally, agent i's utility for receiving x_{ij} amount of item $j \in M$ is given by the piecewise linear and concave function $f_{ij}(x_{ij})$, and the total utility for the bundle x_i is given by the sum of all utilities on single items, i.e., $u_i(x_i) = \sum_{j \in M} f_{ij}(x_{ij})$. Further, we assume normalized utilities, that is, $f_{ij}(0) = 0$. For a

function f_{ij}, we denote its number of linear segments by $|f_{ij}|$, and write (i, j, k) to refer to its k-th segment with slope u_{ijk} and length ℓ_{ijk} (see Fig. 1).

In contrast to the familiar case of disposable goods where $f_{ij} \geq 0$, a mixed manna allows $f_{ij} \in \mathbb{R}$, i.e., an agent may get *positive or negative* utility for an item. We assume each agent labels each item either an (individual) *good* or *bad*. If item j is a good for agent i (like in Fig. 1, left), then $f_{ij} \geq 0$ and $u_{ij1} > u_{ij2} > \cdots > u_{ijs} \geq 0$, $s = |f_{ij}|$, which implies concavity and captures the classical property of decreasing marginal utilities. Otherwise, if j is a bad for agent i, then $f_{ij} \leq 0$ and $0 \geq u_{ij1} > u_{ij2} > \cdots > u_{ijs}$ (like in Fig. 1, right). Note that two agents i, i' might disagree on the label of a given item j, e.g., j can be a good for i and a bad for i'. However, we call an item good if it is a good for at least one agent, and bad otherwise. The global set of goods is denoted by $M^+ = \{j \in M : \exists i \in N \; u_{ij1} > 0\}$, and the global set of bads is $M^- = M \setminus M^+$. For simplicity of the technical exposition, we further assume that $u_{ijk} \neq 0$ for all segments.[2] This is a reasonable assumption, since in real world scenarios it seems rather rare that an agent does not care (entirely) about getting an item or doing some work.

Instance Types. In (Bogomolnaia et al., 2017), the authors show that every fair division instance with mixed manna falls into one of three *types*: positive, negative, or null. The type roughly indicates whether there is a 'surplus' of goods or bads.

More formally, let $N^+ = \{i \in N : \max_{j \in M} u_{ij1} > 0\}$ be the set of *attracted* agents, i.e., agents that each have at least one good, whereas $N^- = N \setminus N^+$ is the set of *repulsed* agents that have only bads. We use $\mathcal{U}$ for the set of agent utilities over all feasible allocations, i.e., if $u \in \mathcal{U}$, then $u = (u_1(x_1), \ldots, u_n(x_n))$ for some $x \in \mathcal{X}$. Next, we define $\Gamma_+ = \mathbb{R}_+^{N^+} \times \{0\}^{N^-}$. Note that in Γ_+ attracted agents benefit (the $\mathbb{R}_+^{N^+}$ portion), without harming any repulsed agents (the $\{0\}^{N^-}$ portion). Also, let $\Gamma_{++} = \mathbb{R}_{++}^{N^+} \times \{0\}^{N^-}$ be the relative interior of Γ_+.

Definition 1. *A fair division instance is called*

- *positive if* $\mathcal{U} \cap \Gamma_{++} \neq \emptyset$,
- *negative if* $\mathcal{U} \cap \Gamma_+ = \emptyset$,
- *null if* $\mathcal{U} \cap \Gamma_+ = \{\mathbf{0}\}$.

In a positive instance, we can ensure all attracted agents receive strictly positive utility without harming any repulsed agents (who dislike all items). Conversely, in a negative instance, no feasible allocation gives *all* attracted agents non-negative utility without harming any repulsed agents. Finally, in null instances, the only feasible allocations which give all agents non-negative utility satisfy $u_i(x_i) = 0$, $\forall i \in N$.

2. While we conjecture that conceptually all our ideas can be applied also when $u_{ijk} = 0$ is allowed, the analysis of such segments generates a lot of technicalities, which we chose to avoid to aid the readability.

Determining the Instance Type. We can determine the type of a given instance with SPLC utilities in polynomial time by solving the following linear program (LP).

$$\begin{aligned} \max \quad & t \\ \text{s.t.} \quad & \sum_{j,k} u_{ijk} x_{ijk} \geq t, \quad \forall i \in N^+ \\ & \sum_{i \in N^+, k} x_{ijk} = 1, \quad \forall j \in M \\ & 0 \leq x_{ijk} \leq \ell_{ijk}, \quad \forall i \in N^+,\ j \in M,\ k \leq |f_{ij}| \end{aligned} \tag{1}$$

For an interpretation of this LP, one can imagine that single segments of the utility functions f_{ij} are allocated to agents. Variable x_{ijk} represents the amount of the k-th segment of the utility function f_{ij} allocated to agent i (see also Fig. 1). The solution t gives a lower bound on any attracted agent's utilities by the first set of constraints. The second set of constraints simply requires that all items are fully allocated among attracted agents, and the third set of constraints ensures that segments aren't over allocated. Note that by concavity of the utility functions f_{ij} and optimality of t, one can assume w.l.o.g. that the segments of each utility function are allocated in a consecutive way, i.e., if $x_{ijk} > 0$ then $x_{ijk'} = \ell_{ijk'}$ for all $k' < k$.

Proposition 1. *Let (t^*, x^*) be an optimal solution to (1). The sign of t^* determines the instance type:*

- *If $t^* > 0$, then the instance is positive.*
- *If $t^* = 0$, then the instance is null.*
- *If $t^* < 0$, then the instance is negative.*

Proof. First suppose that $t^* > 0$. Then in x^* all attracted agents receive strictly positive utility, while repulsed agents receive no allocation. Hence the instance is positive by Definition 1.

Next, suppose that $t^* = 0$. We want to show that the only feasible allocations x which give all agents non-negative utility satisfy $u_i(x_i) = 0,\ \forall i \in N$. For contradiction, suppose not. Then at least one agent $i \in N^+$ receives strictly positive utility $u_i(x_i) > 0$ and some other agent $i' \in N^+$ receives a total utility of $u_{i'}(x_{i'}) = 0$. We now construct an alternate allocation y so that $u_i(y_i) > 0,\ \forall i \in N^+$, contradicting the optimality of $t^* = 0$.

First, observe that for any $j \in M^+$, there is $i \in N^+$ such that $u_{ij1} > 0$. Therefore, we may assume that no agent $i' \in N^+$ with $u_{i'j1} < 0$ receives any part $x_{i'j} > 0$ of j. This is valid since reallocating $x_{i'j}$ to i, i.e., $y_{ij} = x_{ij} + x_{i'j}$ and $y_{i'j} = 0$ improves both agents' utilities.

Next, consider any agent $i \in N^+$ with a non-zero allocation, i.e., $x_i \neq \mathbf{0}$, such that $u_i(x_i) = 0$. Since $x_i \neq \mathbf{0}$, we must have $x_{ij}, x_{ij'} > 0$, for some $j \in M^+$ and $j' \in M^-$. If $u_{i'}(x_{i'}) = \epsilon > 0$ for some $i' \in N^+$, then we can transfer a fraction $\delta > 0$ of bad j' from agent i to agent i' to make both agents' utilities strictly positive. Let k be the first segment of $f_{i'j'}$ not fully allocated, i.e., $x_{i'j'k} < \ell_{i'j'k}$. Define $\delta = \min\left\{x_{ij'},\ \ell_{i'j'k} - x_{i'j'k},\ \frac{\epsilon}{2|u_{i'j'k}|}\right\}$. Note that $\delta > 0$ is the maximum fraction of item j' which can be shifted from agent i to agent i' without leaving the current segments for both agents. Shift fraction δ of item j'

from i to i', i.e., define a new allocation y where $y_{ij'} = x_{ij'} - \delta$ and $y_{i'j'} = x_{i'j'} + \delta$ (and $y = x$ anywhere else). Then $u_{i'}(y_{i'}) \geq \epsilon/2 > 0$, and $u_i(y_i) > 0$. We repeat this step for all agents with $x_i \neq \mathbf{0}$ and $u_i(x_i) = 0$.

After the steps above, for all $i \in N^+$, either $u_i(x_i) > 0$ or $x_i = \mathbf{0}$. If $u_i(x_i) > 0$ for all $i \in N^+$, then we reach a contradiction to that $t^* = 0$ is optimal. Therefore, assume that $x_i = \mathbf{0}$ for some $i \in N^+$. By definition of N^+, there is $j \in M^+$ such that $u_{ij1} > 0$. Further, all items are fully allocated in x, so there is $i' \in N^+$ with $x_{i'j} > 0$, and $u_{i'}(x_{i'}) = \epsilon > 0$. Let k be the last segment with $x_{i'jk} > 0$. Suppose we reallocate a portion of $x_{i'j}$ to agent i, i.e., $y_{ij} = \delta$ and $y_{i'j} = x_{i'j} - \delta$, where $\delta = \min\left(x_{i'jk}, \frac{\epsilon}{2|u_{i'jk}|}\right)$. Then $u_{i'}(y_{i'}) \geq \epsilon/2 > 0$ and $u_i(y_i) > 0$. Repeating this step for all $i \in N^+$ with $x_i = \mathbf{0}$ ensures that $u_i(x_i) > 0$ for all $i \in N^+$, which contradicts that $t^* = 0$ maximizes (1).

The above argument shows that if $t^* = 0$, then any feasible allocation x must satisfy $u_i(x_i) = 0,\ \forall i \in N^+$, so the instance is null. Finally, repeating the above arguments in case $t^* < 0$ shows that the instance must be negative. □

2.2 Competitive Equilibrium

We are interested in computing *competitive equilibria (CE)*. To define this notion, we turn a fair division instance into a market. We endow each agent $i \in N$ with a budget e_i of (virtual) currency. The signs of agents' budgets have to correspond to the instance type to ensure CE existence (see Theorem 1 below). In positive instances, we assume strictly positive budgets $e_i > 0$ for attracted agents $i \in N^+$ and $e_i = 0$ for repulsed agents $i \in N^-$. In negative instances, all budgets need to be strictly negative, and in null instances there is no money (i.e., $e_i = 0$ for all $i \in N$), and computing a CE is easy.[3] Hence, for the rest of the paper, we concentrate on positive and negative instances.

A competitive equilibrium consists of an allocation x and a vector of prices $p = (p_1, \ldots, p_m)$ for the items. In markets with mixed manna, the price of an item can be positive or negative. A price $p_j > 0$ represents a payment to receive a fraction of an item an agent enjoys, while $p_j < 0$ means that agents are paid to receive a fraction of an item they dislike. Nevertheless, we say i *buys* item j whenever $x_{ij} > 0$.

Definition 2 (Competitive Equilibrium)**.** *A pair (x^*, p^*) of allocation and prices is a competitive equilibrium iff:*

1.) *Items are fully allocated:* $\sum_{i \in N} x^*_{ij} = 1,\ \forall j \in M$.

2.) *Budgets are fully spent:* $\sum_{j \in M} x^*_{ij} p^*_j = e_i,\ \forall i \in N$.

3.) *Each agent $i \in N$ buys a utility-maximizing bundle:*

$$x^*_i \in \arg\max_{x_i \in \mathbb{R}^m_+} u_i(x_i), \quad s.t. \ \sum_{j \in M} x_{ij} p^*_j \leq e_i,\ x_{ij} \geq 0 \tag{2}$$

Our algorithms for computing CE apply even to scenarios with different budgets, where agents have different entitlements to the items (e.g., when dissolving a business partnership

3. Any feasible allocation that gives all agents non-negative utility can be seen as CE. We can compute such an allocation when solving the LP (1) to determine the instance type.

where one partner is more senior than another). The prominent special case of equal budgets, i.e., $|e_i| = e$ for all $i \in N$ with $e_i \neq 0$, is called *competitive equilibrium from equal incomes (CEEI)*.

Bogomolnaia et al. (2017) showed that CE exist under very general conditions and satisfy a number of fairness criteria. The following theorem summarizes the result in our context.

Theorem 1 (follows from Bogomolnaia et al., 2017)**.** *Suppose agents' utility functions are SPLC and budgets are chosen such that*

- *in positive instances,* $e_i > 0 \ \forall i \in N^+$ *and* $e_i = 0 \ \forall i \in N^-$*;*
- *in negative instances,* $e_i < 0 \ \forall i \in N$*;*
- *in null instances,* $e_i = 0 \ \forall i \in N$*.*

Then a competitive equilibrium (x^*, p^*) *always exists. The allocation* x^* *is Pareto-optimal, and in case of equal budgets also envy-free and proportional.*

Remark: Bogomolnaia et al. (2017) do not require budgets to be fully spent in their definition of CE. However, one can see that by choosing budgets according to the instance type as in the above theorem, the optimal bundles condition implies that budgets are fully spent[4].

Prices of Goods and Bads. It is easy to see that in any CE (x^*, p^*), each good $j \in M^+$ has a strictly positive price $p_j^* > 0$, while each bad $j \in M^-$ has a strictly negative price $p_j^* < 0$. Assume $p_j^* \leq 0$ for a good $j \in M^+$. Then any agent $i \in N$ for which j is a good has infinite demand for j in (2) regardless of her budget e_i. On the other hand, assume $p_j^* \geq 0$ for a bad $j \in M^-$. Then no agent chooses to purchase j in (2) and hence item j is not allocated at all.

2.3 Optimal Bundles

Let us analyze the structure of an agent's optimal bundle in a CE. Note that for SPLC utilities, the optimization problem in (2) is an LP. We use variables x_{ijk} as agent i's allocation on the k-th segment of item j. Since the segment (i, j, k) has length ℓ_{ijk}, we have $0 \leq x_{ijk} \leq \ell_{ijk}$. Given a vector of prices p, agent i then solves the LP

$$\max_{x_i} \sum\nolimits_{j,k} u_{ijk} x_{ijk} \quad \text{s.t.} \quad \sum\nolimits_{j,k} x_{ijk} p_j \leq e_i, \quad 0 \leq x_{ijk} \leq \ell_{ijk}.$$

Like in LP (1), due to concavity of the functions f_{ij}, any optimal solution x_i of the LP allocates the segments (i, j, k) of each utility function f_{ij} in a consecutive way.

Bang and Pain Per Buck. Given prices p, we define agent i's *bang per buck* for the k-th segment of good $j \in M^+$ as $bpb_{ijk} = u_{ijk}/p_j$, and the *pain per buck* for the k-th segment

4. Assume otherwise, and consider an agent not spending her budget. Recall that we assume $u_{ijk} \neq 0$, so purchasing any fraction of an item must either strictly increase or strictly decrease an agent's utility. Thus, the agent not spending her budget could strictly increase her utility by either purchasing a larger amount of a good or a smaller amount of a bad.

of bad $j \in M^-$ as $ppb_{ijk} = u_{ijk}/p_j$. Note that *bpb* (*ppb*) gives the utility (disutility) per unit spending on a good (bad). By applying Karush-Kuhn-Tucker (KKT) conditions to the above LP, the following useful characterization can be obtained.

Proposition 2. *For any agent $i \in N$, a bundle x_i is optimal under prices p iff there exists a threshold value $\alpha_i \geq 0$, such that for each segment (i, j, k), the following hold:*

(g1) If $j \in M^+$ and $\frac{u_{ijk}}{p_j} < \alpha_i$, then $x_{ijk} = 0$.

(g2) If $j \in M^+$ and $\frac{u_{ijk}}{p_j} > \alpha_i$, then $x_{ijk} = \ell_{ijk}$.

(b1) If $j \in M^-$ and $\frac{u_{ijk}}{p_j} > \alpha_i$, then $x_{ijk} = 0$.

(b2) If $j \in M^-$ and $\frac{u_{ijk}}{p_j} < \alpha_i$, then $x_{ijk} = \ell_{ijk}$.

We define α_i as the smallest possible threshold value for agent i. Note that for each agent there always exists at least one segment (i, j, k) with $\frac{u_{ijk}}{p_j} = \alpha_i$.

Forced and Flexible Segments. Given a CE (x, p) and the corresponding α_i for each $i \in N$, we call a segment (i, j, k) *flexible* if $\frac{u_{ijk}}{p_j} = \alpha_i$, and denote the set of flexible segments with $\hat{F}$. If for a good $j \in M^+$ it holds $\frac{u_{ijk}}{p_j} > \alpha_i$, then we call the segment (i, j, k) *forced*. Similarly, we call the segment (i, j, k) forced for a bad $j \in M^-$ if $\frac{u_{ijk}}{p_j} < \alpha_i$. Note that due to Proposition 2 forced segments need to be fully allocated, i.e., $x_{ijk} = \ell_{ijk}$, while flexible segments can be allocated arbitrarily without violating the optimal bundles condition, i.e., $0 \leq x_{ijk} \leq \ell_{ijk}$. Note also that an item j might not have a flexible segment (i, j, k). Instead, every agent i has at least one flexible segment (i, j, k) (due to our definition of α_i).

3. Constant Number of Agents or Items

In this section, we discuss a polynomial time algorithm for computing CE in instances with SPLC utilities when there is a constant number of agents or a constant number of items. Our algorithm computes *all* CE in the following sense: For every possible CE (x, p), the algorithm returns at least one CE (x', p'), in which every agent achieves the same utility as in (x, p). It is worth noting that for an arbitrary number of agents or items there cannot exist such an algorithm since the number of different utility profiles corresponding to CE can be exponential in $\min\{m, n\}$ (Bogomolnaia et al., 2019). As an immediate corollary of our algorithm, we also get a polynomial bound on this number when m or n is constant. Being able to compute all CE also has the additional benefit of further optimization, e.g., one could select a CE that maximizes the minimum utility of an agent.

Theorem 2. *Suppose agents have SPLC utilities and there is a constant number of agents or items. Then we can compute all CE in polynomial time, i.e., for every CE our algorithm returns one with the same utility profile.*

To obtain the result, the treatment of SPLC utilities creates a number of technical challenges in the correct handling of forced and flexible segments.

We assume that the input is a market with agents, items, utilities, and budgets[5] (in accordance with the instance type). Our algorithm is based on the *cell decomposition* technique first used in the context of market equilibria by Devanur and Kannan (2008). It rests on the fact that k hyperplanes separate $\mathbb{R}^d$ into $O(k^d)$ non-empty regions or *cells*. If d is constant, then this creates only polynomially many cells, which can also be enumerated in polynomial time (see e.g. Avis & Fukuda, 1996). We choose hyperplanes in $\mathbb{R}^n$ or $\mathbb{R}^m$ (depending on which of both has constant dimension) so that each cell corresponds to a unique set of forced and a unique set of flexible segments for each agent. We call such a pair of sets a *utility-per-buck (UPB) configuration*. Then, for each such UPB configuration it remains to check whether there actually is a CE consistent with it.

3.1 Finding UPB Configurations

We present a cell decomposition to determine all meaningful UPB configurations, and show that if the number of agents or items is constant, we thereby obtain only polynomially many cells. Using polynomial-time algorithms for checking equilibria existence from Section 3.2, we get a polynomial-time algorithm to compute all CE.

Constant Number of Items. Let $m = |M|$ be a constant. Consider $\mathbb{R}^m$ with coordinates $p_1, \ldots, p_m$. For each tuple (i, j, j', k, k') where $i \in N$, $j \neq j' \in M$, $k \leq |f_{ij}|$ and $k' \leq |f_{ij'}|$, we create a hyperplane $u_{ijk}p_{j'} - u_{ij'k'}p_j = 0$. Each hyperplane divides $\mathbb{R}^m$ into regions with signs $>$, $=$, or $<$, where the sign determines whether i prefers the segment (i, j, k) or (i, j', k'), e.g., if $j, j' \in M^+$ then $u_{ijk}/p_j > u_{ij'k'}/p_{j'}$ in the $>$ region. A cell is the intersection of these half-spaces. One may think of a cell being represented by a tuple from $\{>, =, <\}^H$, where H is the set of all pairs of segments of the same agent. Thus a cell gives a partial ordering of the segments (i, j, k) of agent i according to bpb_{ijk} and ppb_{ijk} (partial in the sense that there might be multiple segments with the same bpb_{ijk} (ppb_{ijk}) value). Sort the segments (i, j, k) of goods in decreasing order of *bpb* for agent i and create the equivalence classes $G_1^i, \ldots, G_g^i$ with the same *bpb*. Similarly, create equivalence classes $B_1^i, \ldots, B_b^i$ of segments of bads with the same *ppb*, sorted in increasing order. Let ppb_l be the *ppb* of B_l^i, and bpb_l be the *bpb* of G_l^i.

Note that a cell uniquely determines the B_l^i's and G_l^i's. Further, agent i must purchase the classes G_l^i in increasing order, i.e., higher *bpb* first, and also earn from the classes B_l^i in increasing order, i.e., lower *ppb* first (due to Proposition 2). Thus, i's flexible segments are one or both of the final equivalence classes B_r^i and G_s^i that she purchases in a cell (both iff $ppb_r = bpb_s$). Let $B_{<l}^i = \cup_{z=1}^{l-1} B_z^i$ and define $G_{<l}^i$ similarly. Also let $B_{<1}^i = G_{<1}^i = \emptyset$. If i buys a strictly positive fraction of B_r^i, then $B_{<r}^i$ are forced segments. The same holds for goods. Thus, each choice of flexible segments B_r^i and/or G_s^i for each agent yields a unique UPB configuration.

We want to determine the forced and flexible segments. Therefore, in the rest of this section, we concentrate on negative instances where $e_i < 0$ for all $i \in N$. One can adapt the arguments for positive instances by swapping the roles of goods and bads. We partition each cell into sub-cells by adding two types of hyperplanes. First, we add $\sum_{(i,j,k) \in B_{<r}^i} \ell_{ijk} p_j -$

5. Alternatively, if the goal is to compute CEEIs for a fair division instance, we can determine the instance type in polynomial time by solving the LP (1) and then assign appropriate budgets.

$e_i = 0$, for each agent $i \in N$ and equivalence class $r \leq b$. Next, for each B^i_r and G^i_s, we add the hyperplanes $\sum_{(i,j,k)\in B^i_{\leq r}\cup G^i_{\leq s}} \ell_{ijk}p_j - e_i = 0$. Note that these hyperplanes relate agent i's budget and total spending after fully purchasing all of the segments in $B^i_{\leq r} \cup G^i_{\leq s}$. In a negative instance it holds $e_i < 0$, so that in the positive half-space $>$, i.e., $\sum_{(i,j,k)\in B^i_{\leq r}\cup G^i_{\leq s}} \ell_{ijk}p_j > e_i$, agent i has not earned her entire budget, and hence she must purchase more bads. Similarly, in the negative half-space $<$, i earns more than her budget, i.e., she purchased too many bads and needs to spend more on goods.

We can use a simple greedy algorithm based on the signs of the hyperplanes within a sub-cell to determine agent i's flexible segments. We discuss an overview, c.f. Algorithm 1 for a formal description.

In a negative instance, all agents must purchase some bads, so we start by purchasing B^i_l's in increasing order, i.e., lowest *ppb* first, until agent i earns her budget e_i. Suppose i earns her budget by fully purchasing $B^i_{<r}$ and some strict fraction or none of B^i_r. Now we want to check if agent i might purchase some goods. Any spending on goods must be offset by earning on bads. Clearly, i spends on the segments G^i_l if $bpb_l > ppb_r$, i.e., utility gained per unit spending is on G^i_l strictly greater than utility lost per unit earning on B^i_r. We define

$$\Sigma(r,s) = \sum_{(i,j,k)\in B^i_{\leq r}\cup G^i_{\leq s}} \ell_{ijk}p_j.$$

We now start purchasing segments greedily. Let B^i_r and G^i_s be the current segments i purchases of bads and goods respectively. That is, i either purchases none of B^i_r or a strict fraction of B^i_r, and similarly for G^i_s. As long as $bpb_s > ppb_r$, i.e., i's utility gained per unit spending on the current goods G^i_s is strictly larger than her utility lost per unit earning on the current bads B^i_r, she wants to buy as much as possible from G^i_s by earning from B^i_r. This is limited when either (or both) of G^i_s and B^i_r are fully consumed. We can check this based on the signs of the hyperplanes in the current sub-cell.

If $\Sigma(r,s) < e_i$, then i can fully purchase G^i_s with earnings from B^i_r, thereby not fully consuming B^i_r. Hence G^i_s gets fully allocated to i and B^i_r partly, and we proceed with the next class of goods G^i_{s+1}. Since the segments G^i_s are fully allocated, and B^i_r partly, we set G^i_s forced and B^i_r flexible.

If $\Sigma(r,s) > e_i$, then i cannot fully purchase G^i_s with earnings from B^i_r. Hence B^i_r gets fully allocated to i and G^i_s partly, and we proceed with the next class of bads B^i_{r+1}. Since the segments B^i_r are fully allocated, and G^i_s partly, we set B^i_r forced and G^i_s flexible.

Finally, if $\Sigma(r,s) = e_i$, then i can fully purchase G^i_s with earnings from exactly B^i_r. Hence both G^i_s and B^i_r get fully allocated to i, and we proceed with the next classes of bads and goods B^i_{r+1} and G^i_{s+1}. Since the segments B^i_r as well as G^i_s are fully allocated, we set both forced.

We repeat this procedure until we either reach classes B^i_r and G^i_s with $bpb_s \leq ppb_r$ or run out of B^i_r's or G^i_s's. If for the last classes B^i_r and G^i_s we have $bpb_s = ppb_r$, then i can still trade G^i_s against B^i_r, but her utility does not change. Hence in this case we set both B^i_r and G^i_s flexible.

For analyzing the running time of this greedy procedure, observe that in each iteration of the while-loop there is at least one segment of i added to the set of forced segments.

Algorithm 1: Forced and flexible segments F and $\hat{F}$ of agent i in a sub-cell

Find the smallest r s.t. $\sum_{(i,j,k)\in B^i_{\leq r}} \ell_{ijk}p_j < e_i$;

Set $s=1$, $F = B^i_{<r}$, $\hat{F} = \begin{cases} \emptyset & \text{if } \sum_{(i,j,k)\in B^i_{<r}} \ell_{ijk}p_j = e_i \text{ ,} \\ B^i_r & \text{otherwise.} \end{cases}$

while $bpb_s > ppb_r$ **do** // while worth to buy as much as possible from G^i_s

  if $\Sigma(r,s) < e_i$ **then** // G^i_s can be fully bought with earnings from inside B^i_r

    $F \leftarrow F \cup G^i_s$, $\hat{F} \leftarrow B^i_r$, $s \leftarrow s+1$;

  else if $\Sigma(r,s) > e_i$ **then** // G^i_s cannot be fully bought with earnings from B^i_r

    $F \leftarrow F \cup B^i_r$, $\hat{F} \leftarrow G^i_s$, $r \leftarrow r+1$;

  else // G^i_s can be fully bought with earnings from exactly B^i_r

    $F \leftarrow F \cup G^i_s \cup B^i_r$, $\hat{F} = \emptyset$, $s \leftarrow s+1$, $r \leftarrow r+1$;

  end

end

if $bpb_s = ppb_r$ **then** // G^i_s can be traded against B^i_r with no change in utility

  $\hat{F} \leftarrow G^i_s \cup B^i_r$;

end

return $F, \hat{F}$;

Hence the algorithm runs for at most mS rounds, where S is the maximum number of segments in any utility function of agent i. We obtain the following result.

Proposition 3. *Algorithm 1 finds an agent's forced and flexible segments within a sub-cell in polynomial time.*

Finally, we count the number of cells in both cell-decompositions. Let S be the maximum number of segments of any utility function. We created a hyperplane $u_{ijk}p_{j'} - u_{ij'k'}p_j = 0$, for each tuple (i,j,j',k,k'), $i \in N$, $j \neq j' \in M$, $k \leq |f_{ij}|$ and $k' \leq |f_{ij'}|$, and hence at most $n\binom{mS}{2} = O(nm^2S^2)$ in total. These divide $\mathbb{R}^m$ into at most $O((nm^2S^2)^m)$ many cells (note that m is constant). Now observe that the number of equivalence classes of segments in any cell is bounded by mS for each agent $i \in N$. Hence the total number of hyperplanes of the form $\sum_{(i,j,k)\in B^i_{\leq r}\cup G^i_{\leq s}} \ell_{ijk}p_j - e_i = 0$ is $O(nm^2S^2)$ in any cell, and they divide $\mathbb{R}^m$ into $O((nm^2S^2)^m)$ sub-cells. Hence the total number of sub-cells is $O((nm^2S^2)^{2m})$, which is polynomial for constant m.

Constant Number of Agents. Let $n = |N|$ be a constant, and define $\lambda_i = 1/\alpha_i$ for $i \in N$. Consider $\mathbb{R}^n$ with coordinates $\lambda_1, \ldots, \lambda_n$. For each tuple (j,i,i',k,k') where $j \in M$, $i,i' \in N$, $k \leq |f_{ij}|$ and $k' \leq |f_{i'j}|$, we create a hyperplane $u_{ijk}\lambda_i - u_{i'jk'}\lambda_{i'} = 0$. Each hyperplane divides $\mathbb{R}^n$ into regions with signs $>$, $=$, or $<$. From the sign we can determine which of the two segments (i,j,k) and (i',j,k') of item j has to be sold first. To see this, assume we are in the $>$ half-space for a good $j \in M^+$, i.e., $u_{ijk}\lambda_i > u_{i'jk'}\lambda_{i'}$. Now if agent i' buys a positive fraction $x_{i'jk'} > 0$ of segment (i',j,k'), then $u_{i'jk'}/p_j \geq \alpha_{i'}$ due to Proposition 2, or equivalently $u_{i'jk'}\lambda_{i'} \geq p_j$. From being in the $>$ half-space it follows $u_{ijk}\lambda_i > u_{i'jk'}\lambda_{i'} \geq p_j$ and hence $u_{ijk}/p_j > 1/\lambda_i = \alpha_i$, i.e., the segment (i,j,k) is forced. A

similar argument shows that if we are in the $<$ half-space for a bad $j \in M^-$ and agent i' buys a positive fraction of (i', j, k'), then (i, j, k) is also forced.

A *cell* is the intersection of these half-spaces. One may think of a cell being represented by a tuple from $\{>, =, <\}^H$, where H is the set of all pairs of segments of the same item. The $>, =, <$ relations inside a cell give a partial ordering on the $u_{ijk}\lambda_i$'s for each item j. We sort the segments of good $j \in M^+$ in the decreasing order of $u_{ijk}\lambda_i$, and partition them into equivalence classes $G_1^j, \ldots, G_g^j$ with the same $u_{ijk}\lambda_i$ value. Similarly, we create equivalence classes $B_1^j, \ldots, B_b^j$ for bad $j \in M^-$ by sorting the $u_{ijk}\lambda_i$ in increasing order. By the above discussion, if any part of G_t^j is sold, then all segments in $G_{t'}^j$ with $t' < t$ are forced. The same holds for $B_1^j, \ldots, B_b^j$. Let $G_{<t}^j = \cup_{l=1}^{t-1} G_l^j$ and define $B_{<t}^j$ similarly. Now since good j must be fully sold, the last class G_t^j of good j which is fully or partly sold is determined by the largest integer t such that $\sum_{(i,j,k)\in G_{<t}^j} \ell_{ijk} < 1$. The same holds for bads. The segments in $G_{<t}^j$ are anyway forced, while the segments of G_t^j are either forced or flexible depending on whether G_t^j is fully sold or not. In particular, if $\sum_{(i,j,k)\in G_{\leq t}^j} \ell_{ijk} > 1$, then G_t^j is not fully sold and hence the segments in G_t^j are flexible due to Proposition 2. Otherwise, i.e., if $\sum_{(i,j,k)\in G_{\leq t}^j} \ell_{ijk} = 1$, then G_t^j is fully sold and hence some of the segments in G_t^j might be forced and some flexible depending on the agents' allocations of other items. Anyway, all these segments need to be fully allocated and hence we can treat them as forced. The above discussion shows that in each cell we obtain a unique UPB configuration.

Finally, we count the number of cells in the decomposition. Let $S = \max_{i,j} |f_{ij}|$. Observe that the total number of hyperplanes created is at most $m\binom{nS}{2} = O(mn^2S^2)$, and they divide $\mathbb{R}^n$ into at most $O((mn^2S^2)^n)$ many cells, which is polynomial for constant n.

3.2 Checking Cells for Existence of Competitive Equilibria

In this section we show how to check if a cell admits a CE given the corresponding UPB configuration, i.e., the set of forced and flexible segments in the cell.

Proposition 4. *Given the set of forced and flexible segments F and $\hat{F}$ in a cell, it can be checked in polynomial time if there exists a CE (x, p) in the cell. If the cell contains at least one CE, then one of them can be computed in polynomial time.*

Proof. We set up a linear feasibility problem where any solution corresponds to a CE. Since forced segments are fully allocated and budgets are fully spent, the amount $\hat{e}_i$ of money agent i spends on flexible segments is given by

$$\hat{e}_i = e_i - \sum_{(i,j,k)\in F(i)} \ell_{ijk} p_j. \tag{3}$$

Analogously, since all items are fully allocated, the fraction $\hat{s}_j$ of item j allocated on flexible segments is given by

$$\hat{s}_j = 1 - \sum_{(i,j,k)\in F(j)} \ell_{ijk}. \tag{4}$$

Now let $G = (N, M, E)$ be a bipartite graph, where there exists an edge $(i, j) \in E$ between agent i and item j whenever i has a flexible segment on j, i.e., there exists a

number k^* such that (i, j, k^*) is flexible for i. We define $E^+ = \{(i, j) \in E : j \in M^+\}$ and $E^- = E \setminus E^+$ for the set of edges of goods and bads, respectively.

For each edge $(i, j) \in E$, denote with f_{ij} the money which i spends/earns (depending on the sign) on the flexible segment (i, j, k^*). By money conservation, we have

$$\hat{e}_i = \sum_{j \mid (i,j) \in E} f_{ij}, \quad \forall i \in N \tag{5}$$

as well as

$$\hat{s}_j p_j = \sum_{i \mid (i,j) \in E} f_{ij}, \quad \forall j \in M. \tag{6}$$

Moreover, segments cannot be oversold, so it must hold

$$0 \leq f_{ij} \leq \ell_{ijk^*} p_j, \quad \forall (i, j) \in E^+ \tag{7}$$

as well as

$$0 \geq f_{ij} \geq \ell_{ijk^*} p_j, \quad \forall (i, j) \in E^-. \tag{8}$$

Furthermore, Proposition 2 induces constraints on forced and flexible segments, i.e.,

$$u_{ijk} \lambda_i = p_j, \quad \forall (i, j, k) \in \hat{F} \tag{9}$$

and

$$u_{ijk} \lambda_i \geq p_j, \quad \forall (i, j, k) \in F. \tag{10}$$

Now, (3) to (10) together with the constraints of the cell[6] yield a linear feasibility problem (P) in the variables $(f_{ij})_{(i,j) \in E}$, $(p_j)_{j \in M}$, and $(\lambda_i)_{i \in N}$. This can be solved in polynomial time regardless of the number of agents, items, or bads.

Finally, observe that if (P) has a feasible solution, then the actual part x_{ijk^*} of the flexible segment (i, j, k^*) allocated to agent i in the CE can be derived from $x_{ijk^*} = f_{ij}/p_j$, while forced segments $(i, j, k) \in F(i)$ are always fully allocated to i, i.e., $x_{ijk} = \ell_{ijk}$. □

Although for a given cell, there might be a continuum of solutions that satisfy the conditions of a CE, we show that all these CE must yield the same vector of utilities for the agents.

Proposition 5. *If there is a CE (x^*, p^*) in a cell, then the utility $u_i(x_i)$ of an agent $i \in N$ is the same in every CE (x, p) inside the cell.*

Proof. Consider an arbitrary agent $i \in N$. The utility gained by i from forced segments is

$$u_i^{\text{forced}} = \sum_{(i,j,k) \in F(i)} u_{ijk} \ell_{ijk},$$

6. Recall that the cell might involve some strict inequalities. The feasibility of a system of (strict and weak) linear inequalities can be decided using an LP: Introduce one slack variable ϵ, and insert it to every strict inequality, thereby making all of them weak. Then solve the LP with objective to maximize ϵ. If $\epsilon > 0$, the result is a feasible solution for the original system of inequalities; otherwise it is infeasible.

and hence constant within a cell. Thus it suffices to show that the utility

$$u_i^{\text{flex}} = \sum_{j \mid (i,j) \in E} f_{ij} \frac{u_{ijk^*}}{p_j} = \hat{e}_i / \lambda_i$$

gained by i on flexible segments is also constant within a cell (recall that $\hat{e}_i$ is the variable representing i's spending on flexible segments).

We say that a utility value $u \in \mathbb{R}$ is *competitive* for i, if $u = u_i(x_i)$ for some competitive equilibrium (x, p) consistent with the cell. Assume for contradiction that i has two different competitive utility values $u_i^{(1)} < u_i^{(2)}$ inside the cell. Then it must hold

$$u_i^{(1)} = \hat{e}_i^{(1)} / \lambda_i^{(1)} < \hat{e}_i^{(2)} / \lambda_i^{(2)} = u_i^{(2)},$$

where $(e_i^{(1)}, \lambda_i^{(1)})$ and $(e_i^{(2)}, \lambda_i^{(2)})$ are both part of feasible solutions to (P). Now, due to convexity of the solution space of (P), also all pairs

$$(1-\gamma) \begin{pmatrix} e_i^{(1)} \\ \lambda_i^{(1)} \end{pmatrix} + \gamma \begin{pmatrix} e_i^{(2)} \\ \lambda_i^{(2)} \end{pmatrix}$$

for $\gamma \in [0, 1]$ are part of feasible solutions to (P). This implies that all utility values in the set

$$U = \left\{ \frac{(1-\gamma)\hat{e}_i^{(1)} + \gamma \hat{e}_i^{(2)}}{(1-\gamma)\lambda_i^{(1)} + \gamma \lambda_i^{(2)}} \,:\, \gamma \in [0,1] \right\}$$

are also competitive for i. It is easy to see that U is a continuum. However, in (Bogomolnaia et al., 2017) it was shown that there is only a finite number of competitive utility profiles in every instance – a contradiction. Therefore, if a cell admits a CE, then the utility of an agent i is the same in all CE inside the cell. □

Since there is a unique utility profile for all CE in a given cell, we can also upper bound the number of distinct utility profiles of all CE. The next result extends Proposition 1 of Bogomolnaia et al., 2017 (see also Branzei & Sandomirskiy, 2019, Corollary 12).

Corollary 1. *If agents' utilities are SPLC and the number of agents or items is constant, then the number of distinct utility profiles of all CE is polynomial in the input size.*

4. Constant Number of Bads

In this section, we consider negative instances in which agents have linear utility functions for bads and SPLC utilities for goods. In this case, we can relax the requirement for a constant number of items. Instead, our result applies even when we have a constant number of *bads* (as well as any number of goods).

The same result can be obtained for the symmetric case, i.e., the instance is positive, agents have linear utilities for goods, SPLC utilities for bads, and there is a constant number of goods. The adjustment to our subsequent discussion is straightforward by changing the roles of goods and bads.

First, note that a linear utility function is SPLC with a single segment, i.e., $f_{ij}(x_{ij}) = u_{ij}x_{ij}$. Next, observe that if an agent i buys any fraction $x_{ij} > 0$ of bad $j \in M^-$, then also $x_{ij} < \ell_{ij} = \infty$, and hence $u_{ij}/p_j = \alpha_i$ due to Proposition 2. Therefore, any bad $j \in M^-$ agent i spends on has a *ppb* of exactly α_i, and this is minimum. We call these *minimum pain per buck bads*, denoted $mpb_i = \arg\min_{j \in M^-} u_{ij}/p_j$. In a negative instance where all agents must purchase some bads, this allows us to express the α_i's of the agents through the prices of purchased bads. In particular, for each agent i, we pick a representative bad $b(i) \in M^-$ and substitute $\alpha_i = u_{i,b(i)}/p_{b(i)}$. Using this substitution one can decompose according to prices of bads instead of λ_i's where $\lambda_i = 1/\alpha_i$ for all $i \in N$.

Finding UPB Configurations. The algorithm has the same basic structure as in Section 3.1: we use a cell decomposition to enumerate UPB configurations, and then check each cell for equilibrium existence. The difference lies in the cell decomposition. It can be seen as a hybrid of the techniques used in the two scenarios in Section 3.1.

In a negative instance, agents have negative budgets and must earn on some bads. First, we determine the mpb_i bads for each agent in a cell using a similar approach as for a constant number of items. This determines the set of bads each agent might purchase and the value of $\alpha_i = \min_{j \in M^-} u_{ij}/p_j$. For a constant number of agents, we used the variables $\lambda_i = 1/\alpha_i$ to determine mbb_i goods and mpb_i bads for each agent. Now we adapt the approach using the variables p_j for each bad $j \in mpb_i$, where $p_j/u_{ij} = 1/\alpha_i = \lambda_i$.

Theorem 3. *Suppose the instance is negative and that agents have linear utility functions for bads and SPLC utility functions for goods. If the number of bads is constant, then we can compute all CE in polynomial time, i.e., for each CE we obtain one with the same utility profile.*

Proof. Let $d = |M^-|$ be a constant. Consider $\mathbb{R}^d$ with coordinates $p_1, \ldots, p_d$. For each agent $i \in N$ and each pair of bads $j \neq j' \in M^-$ we introduce the hyperplane $u_{ij}p_{j'} - u_{ij'}p_j = 0$, which partitions $\mathbb{R}^d$ into regions with signs $>$, $=$, or $<$. The sign determines whether i prefers bad j over j', e.g., $u_{ij}/p_j < u_{ij'}/p_{j'}$ in the $<$ region. A cell is the intersection of these half-spaces. In each cell, we sort the bads j of agent i in increasing order of u_{ij}/p_j and create the equivalence classes $B_1^i, \ldots, B_b^i$. Now obviously B_1^i are the mpb_i bads of agent i in the cell. It remains to determine the forced and flexible segments of goods. Until now, we used $\binom{d}{2}$ hyperplanes for each agent, giving $O(nd^2)$ in total. Therefore, there are at most $O(n^d)$ cells.

Recall that we used the variables $\lambda_i = 1/\alpha_i$ in a cell-decomposition to determine the forced and flexible segments when the number of agents is constant. Now we follow a similar approach by expressing λ_i through prices of bads in mpb_i. In particular, for every agent i pick a representative bad $b(i) \in mpb_i$, and substitute $\lambda_i = 1/\alpha_i = p_{b(i)}/u_{i,b(i)}$. Note that $mpb_i \neq \emptyset$ for every agent i since the instance is negative and thus every agent must buy some bads.

We now determine the forced and flexible segments of goods for each agent in a given cell. For each tuple (j, i, i', k, k') where $j \in M^+$, $i, i' \in N$, $k \leq |f_{ij}|$ and $k' \leq |f_{i'j}|$, we create a hyperplane $\frac{u_{ijk}}{u_{i,b(i)}}p_{b(i)} - \frac{u_{i'jk'}}{u_{i',b(i')}}p_{b(i')} = 0$. This further divides a cell into sub-cells. Again, the sign of the half-space determines which of the two segments (i, j, k) and (i', j, k') of good j is sold first. Assume we are in the $>$ half space, i.e., $\frac{u_{ijk}}{u_{i,b(i)}}p_{b(i)} > \frac{u_{i'jk'}}{u_{i',b(i')}}p_{b(i')}$,

or equivalently $u_{ijk}/\alpha_i > u_{i'jk'}/\alpha_{i'}$. Now if agent i' buys a positive fraction $x_{i'jk'} > 0$ of segment (i', j, k'), then $u_{i'jk'}/p_j \geq \alpha_{i'}$ due to Proposition 2, or equivalently $u_{i'jk'}/\alpha_{i'} \geq p_j$. Since we are in the $>$ half-space, it follows $u_{ijk}/\alpha_i > p_j$, and thus (i, j, k) is forced. For each good $j \in M^+$, define $G_1^j, \ldots, G_g^j$ as the equivalence classes with the same $\frac{u_{ijk}}{u_{i,b(i)}} p_{b(i)}$ value, sorted in decreasing order. By the above discussion, if any part of G_t^j is sold, then all segments in $G_{t'}^j$ with $t' < t$ must be forced. Since each good must be fully sold, let t be the largest integer such that $\sum_{(i,j,k)\in G_{<t}^j} \ell_{ijk} < 1$, i.e., j becomes fully sold once agents purchase a strictly positive fraction of G_t^j. Now the segments in $G_{<t}^j$ are anyway forced, while the segments of G_t^j are either forced or flexible depending on whether G_t^j is fully sold or not. If $\sum_{(i,j,k)\in G_{\leq t}^j} \ell_{ijk} > 1$, then G_t^j is not fully sold and hence the segments in G_t^j are flexible. Otherwise, i.e., if $\sum_{(i,j,k)\in G_{\leq t}^j} \ell_{ijk} = 1$, then G_t^j is fully sold and hence the segments in G_t^j are also forced. Therefore, we determined unique forced and flexible segments in each sub-cell.

Finally, let us analyze the overall number of cells. Let S be the maximum number of segments of any agents' utility function. We formed sub-cells by adding hyperplanes for each tuple (j, i, i', k, k') where $j \in M^+$, $i, i' \in N$, $k \leq |f_{ij}|$, $k' \leq |f_{i'j}|$. Hence we created at most $|M^+|\binom{nS}{2} = O(mn^2S^2)$ overall in any given cell, which partitions the cell into at most $O(m^d(nS)^{2d})$ sub-cells. As previously calculated, there are $O(n^d)$ cells. Hence the total number of sub-cells is $O(m^d n^{3d} S^{2d})$, which is poly(n, m, S) for constant d. □

Remark: If both goods and (constantly many) bads have SPLC utilities, we need to find agent i's flexible segments of bads B_r^i. When applying cell decomposition on prices, flexible segments are determined by ensuring an agent spends her entire budget, which obviously depends on both goods and bads. Thus, we cannot consider goods and bads separately as we have done in this proof. This applies even when the valuations for bads are SPLC with only two segments. Finding a polynomial-time algorithm in this case is an interesting open problem.

As mentioned at the beginning of this section, by switching roles of goods and bads we can also solve the symmetric case, i.e., computing CE in positive instances when there is a constant number of linear valued goods and an arbitrary number of SPLC valued bads.

Corollary 2. *Suppose the instance is positive and that agents have linear utility functions for goods and SPLC utility functions for bads. If the number of goods is constant, then we can compute all CE in polynomial time.*

5. Summary and Discussion

Our results refine the current understanding on the complexity of competitive equilibria computation in mixed manna fair division. We provide polynomial-time algorithms for mixed manna instances with SPLC valuations when the number of items m or the number of agents n is constant, while the problem is known to be PPAD-hard for arbitrary m and n, even for only-goods instances. Moreover, we identify conditions under which the problem becomes tractable for arbitrary m and n. In particular, we show that for arbitrary m and

n, polynomial-time computation is possible as long as the instance is negative (positive) and there is only a constant number of linear valued bads (goods).

Additionally to being polynomial, our algorithms are able to compute *all* CE in the given instance. Under this requirement, the result is also tight in the sense that one cannot hope for such an algorithm in instances with arbitrary m and n, not even with linear valuations (since there can be exponentially many CE).

As mentioned in Section 4, one of the main open problems is to relax the requirement of linear valued bads in Theorem 3 towards SPLC valued ones. Also remaining is the question whether polynomial time CE-computation is possible for an arbitrary number of chores with linear utilities. Another interesting question is whether there exist strongly polynomial-time algorithms for computing (all) CE in the types of instances we consider.

Acknowledgments

Jugal Garg and Peter McGlaughlin were supported by NSF grant CCF-1942321 (CAREER). Martin Hoefer was supported by DFG grant Ho 3831/5-1, and grants Ho 3831/6-1 and 7-1 within DFG research unit ADYN. We thank anonymous reviewers for helpful and constructive feedback on an earlier draft of this paper that led to significant improvements.

References

Alaei, S., Khalilabadi, P. J., & Tardos, É. (2017). Computing equilibrium in matching markets. In *Proceedings of the 2017 ACM Conference on Economics and Computation, EC 2017*, pp. 245–261.

Avis, D., & Fukuda, K. (1996). Reverse search for enumeration. *Discrete Applied Mathematics*, *65*(1-3), 21–46.

Azrieli, Y., & Shmaya, E. (2014). Rental harmony with roommates. *Journal of Economic Theory*, *153*, 128–137.

Bogomolnaia, A., Moulin, H., Sandomirskiy, F., & Yanovskaia, E. (2017). Competitive division of a mixed manna. *Econometrica*, *85*(6), 1847–1871.

Bogomolnaia, A., Moulin, H., Sandomirskiy, F., & Yanovskaia, E. (2019). Dividing bads under additive utilities. *Social Choice and Welfare*, *52*(3), 395–417.

Brams, S. J., & Taylor, A. D. (1996). *Fair division - from cake-cutting to dispute resolution*. Cambridge University Press.

Brandt, F., Conitzer, V., Endriss, U., Lang, J., & Procaccia, A. D. (2016). *Handbook of Computational Social Choice*. Cambridge University Press.

Branzei, S., & Sandomirskiy, F. (2019). Algorithms for competitive division of chores. arXiv:1907.01766.

Chaudhury, B. R., Garg, J., McGlaughlin, P., & Mehta, R. (2021). Competitive allocation of a mixed manna. In *Proceedings of the 2021 ACM-SIAM Symposium on Discrete Algorithms, SODA 2021*, pp. 1405–1424.

Chaudhury, B. R., Garg, J., McGlaughlin, P., & Mehta, R. (2022). Competitive equilibrium with chores: Combinatorial algorithm and hardness. In *Proceedings of the 2022 ACM Conference on Economics and Computation, EC 2022*, pp. 1106–1107.

Chen, X., & Teng, S. (2009). Spending is not easier than trading: On the computational equivalence of Fisher and Arrow-Debreu equilibria. In *Algorithms and Computation, 20th International Symposium, ISAAC 2009*, pp. 647–656.

Devanur, N. R., & Kannan, R. (2008). Market equilibria in polynomial time for fixed number of goods or agents. In *49th Annual IEEE Symposium on Foundations of Computer Science, FOCS 2008*, pp. 45–53.

Garg, J., Hoefer, M., McGlaughlin, P., & Schmalhofer, M. (2021). When dividing mixed manna is easier than dividing goods: Competitive equilibria with a constant number of chores. In *Algorithmic Game Theory - 14th International Symposium, SAGT 2021*, pp. 329–344.

Garg, J., & McGlaughlin, P. (2020). Computing competitive equilibria with mixed manna. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS 2020*, pp. 420–428.

Garg, J., Mehta, R., Sohoni, M. A., & Vazirani, V. V. (2015). A complementary pivot algorithm for market equilibrium under separable, piecewise-linear concave utilities. *SIAM Journal on Computing*, *44*(6), 1820–1847.

Garg, J., & Végh, L. A. (2019). A strongly polynomial algorithm for linear exchange markets. In *Proceedings of the 51st Annual ACM SIGACT Symposium on Theory of Computing, STOC 2019*, pp. 54–65.

Hylland, A., & Zeckhauser, R. (1979). The efficient allocation of individuals to positions. *Journal of Political Economy*, *87*(2), 293–314.

Moulin, H. (2003). *Fair division and collective welfare*. MIT Press.

Nisan, N., Roughgarden, T., Tardos, É., & Vazirani, V. V. (2007). *Algorithmic Game Theory*. Cambridge University Press.

Orlin, J. B. (2010). Improved algorithms for computing Fisher's market clearing prices. In *Proceedings of the 42nd ACM Symposium on Theory of Computing, STOC 2010*, pp. 291–300.

Robertson, J., & Webb, W. (1998). *Cake-Cutting Algorithms: Be Fair If You Can*. AK Peters, MA.

Su, F. E. (1999). Rental harmony: Sperner's lemma in fair division. *The American Mathematical Monthly*, *106*(10), 930–942.