Polynomial Time Algorithms to Find an Approximate Competitive Equilibrium for Chores*

Shant Boodaghians† Bhaskar Ray Chaudhury‡ Ruta Mehta §

Abstract

Competitive equilibrium with equal income (CEEI) is considered one of the best mechanisms to allocate a set of items among agents fairly and efficiently. In this paper, we study the computation of CEEI when items are chores that are disliked (negatively valued) by agents, under 1-homogeneous and concave utility functions which includes linear functions as a subcase. It is well-known that, even with linear utilities, the set of CEEI may be non-convex and disconnected, and the problem is PPAD-hard in the more general exchange model. In contrast to these negative results, we design a FPTAS: A polynomial-time algorithm to compute ε-approximate CEEI where the running-time depends polynomially on $\frac{1}{\varepsilon}$.

Our algorithm relies on the recent characterization due to Bogomolnaia et al. (2017) of the CEEI set as exactly the KKT points of a non-convex minimization problem that have all coordinates non-zero. Due to this *non-zero* constraint, naïve gradient-based methods fail to find the desired local minima as they are attracted towards zero. We develop an *exterior-point method* that alternates between guessing *non-zero* KKT points and maximizing the objective along supporting hyperplanes at these points. We show that this procedure must converge quickly to an approximate KKT point which then can be mapped to an approximate CEEI; this exterior point method may be of independent interest.

When utility functions are linear, we give explicit procedures for finding the exact iterates, and as a result show that a stronger form of approximate CEEI can be found in polynomial time. Finally, we note that our algorithm extends to the setting of un-equal incomes (CE), and to mixed manna with linear utilities where each agent may like (positively value) some items and dislike (negatively value) others.

*The full version of the paper can be accessed at https://arxiv.org/abs/2107.06649

†University of Illinois at Urbana-Champaign.
‡University of Illinois at Urbana-Champaign.
§University of Illinois at Urbana-Champaign.

1 Introduction

Allocating a set of items among agents in a non-wasteful *(efficient)* and agreeable *(fair)* manner is an age old problem extensively explored within economics, social choice, and computer science. An allocation based on *competitive equilibria (CE)* has emerged as one of the best mechanisms for this problem due its remarkable fairness and efficiency guarantees [AD54, Var74, BMSY17]. The existence and computation of competitive equilibria has seen much work when all the items are goods, *i.e.* liked (positively valued) by agents. However, when items are chores, *i.e.* disliked (negatively valued) by agents, the problem is relatively less explored even though it is as relevant in every day life; for example dividing teaching load among faculty, job shifts among workers, and daily household chores among tenants.

In this paper, we study the problem of computing competitive equilibria with *equal income* (CEEI) [Var74, BMSY17] for chore division, where a set of m *divisible* chores has to be allocated among a set of agents. Agents receive *payments* for doing chores, and are required to earn a minimum amount, and under *equal income*, these amounts are the same.[1] A competitive equilibrium (CE) for chores consists of a payment per-unit for each chore, and an allocation of chores to agents such that every agent gets her *optimal bundle*, *i.e.,* the disutility-minimizing bundle subject to fulfilling her earning requirement. Typically, agent preferences are represented by a monotone and concave *utility function* [AD54, BMSY17], that is negative and decreasing in case of chores. Equivalently, we consider *disutility functions*, namely $D_i : \mathbb{R}_+^m \to \mathbb{R}_+$ for agent i, that is monotone increasing and convex. We assume disutility functions to be 1-homogeneous as otherwise the problem is known to be intractable [CT09, CGMM20]. We note that 1-homogeneous functions form a rich class that includes the well-studied linear and CES functions as special cases.

The computational complexity of CE is well-understood when items are goods, *e.g.,* [DPSV08, CDDT09, CPY17, VY11, CDG$^+$17, Rub18] (see Section 4 for a detailed discussion): for 1-homogeneous utilities, the famous Eisenberg-Gale [EG59] convex programming formulation and its dual are known to give equilibrium allocation and prices respectively. As a consequence the set of CE is convex, and the ellipsoid and/or interior point methods would find an approximate CE in polynomial-time, assuming utility functions are *well-behaved*. When utility functions are further restricted to be linear, there are many (strongly) polynomial time combinatorial algorithms known [DPSV08, Orl10], even for the more general *exchange model* where agents want to *exchange* items they own to optimize their utilities [DM15, DGM16, GV19].

Although goods and chores problems seem similar, results for chores are surprisingly contrasting: Even in the restricted case of linear disutilities, the set of CEEI can be non-convex and disconnected [BMSY17, BMSY19], and in the exchange model computing a CE is PPAD-hard [CGMM20]. No polynomial time algorithms are known to find CEEI with chores, except for when number of agents or number of chores is a constant [BS19, GM20].[2] We note that the combinatorial approaches known for the goods case [DPSV08, Orl10, Vég12] seem to fail due to disconnectedness of the CEEI set (see Remark 3.1 for further explanation). In light of these results, computing exact CEEI may turn out to be hard even with linear disutilities, but what about an approximate CEEI?

We resolve the above question by designing an FPTAS for the more general class of 1-homogeneous disutilities. Specifically, we design an algorithm to find ε-approximate CEEI in time polynomial in $\frac{1}{\varepsilon}$ and bit-size of the input instance parameters. We remark that many of the above bottlenecks exist even when we focus on approximate CEEI. In particular, the set of approximate CEEI can be non-convex and disconnected. And the fundamental bottleneck in generalizing the combinatorial algorithm explained in Remark 3.1 still persists. Despite these challenges, we are able to design an FPTAS to find an approximate CEEI, and extend it to more general valuations than linear which includes CES valuation functions.

Our algorithm crucially builds on the characterization of Bogomolnaia et al. [BMSY17], which states that the set of CEEI is exactly the *strictly positive* local-minima (KKT points) of a non-convex formulation, namely minimize the *product of disutilities (equivalently* $\sum_i \log d_i$*)* over the space of *feasible* disutility vectors. The set of feasible disutility vectors may not be convex, but they can be made convex by allowing overallocation. Unfortunately, standard interior-point methods for finding local optimum, such as gradient descent, will fail at ensuring the strict positivity constraint, since the gradient of the objective is attracted towards the minimum disutility coordinate. This difficulty is not alleviated by barrier function methods either. A possible fix is to introduce additional constraints to avoid zeros, but then we loose the CEEI characterization.

The above issues would not arise if we *maximize* $\sum_i \log d_i$ instead of minimizing it. Motivated from this observation, we design an *exterior-point method* that tries to maximize the objective *outside* of the feasible region, starting from an outside point that is below the lower-hull. However, we are faced with two crucial difficulties: (i) now the *outside region* is

[1] The earning requirement of an agent can also be thought of as her importance/weight compared to others, and thereby under *equal income* all agents have the same weight.

[2] These algorithms are based on enumeration from a cleverly designed set of candidates. Similar approaches are known for goods manna when the number of items or agents is a constant [DK08, GMSV15], while the general case is PPAD-hard even to approximate [CT09, Rub18]

truly non-convex, and (ii) we must ensure that we do find a desired local minimum from the inside.

Our exterior-point method handles the above issues by repeatedly guessing candidate solutions, and checking if they are local minima for the problem inside the feasible region by verifying if the gradient is parallel to some supporting hyperplane. If not, it goes on to try another such candidate, while ensuring it is always increasing along the objective function. Thus, the objective acts as a potential function, and we can bound convergence rates by the size of objective improvement at each step. This method may be of independent interest. We terminate search when the supporting hyperplane direction is *approximately* equal to the gradient, in a multiplicative sense, and argue that such an approximate KKT point suffices to guarantee an approximate CEEI.

The crucial step in each iteration of this procedure is to find the nearest feasible point in the disutility space, which allows us to find a boundary point along with a supporting hyperplane at it. When disutility functions are linear, we argue that both distance minimization and supporting hyperplane computation can be solved exactly, leading to a stronger form of approximate CEEI.

For the case of general 1-homogeneous and convex disutility functions, the nearest feasible point must be found by interior point methods. We assume black-box access to the disutility functions' value and partial derivatives. This approximate nearest-point computation introduces errors in the local optimum and the supporting hyperplane both, that are tricky to handle. We show how to handle these extra errors by modifying the algorithm, and argue that a slight weakening of approximately competitive equilibria can still be guaranteed. As expected, these guarantees, including successful application of the interior point method, rely on the disutility functions being "well-behaved", and the running time of the algorithm depends logarithmically on continuity parameters of the disutility functions, namely, the Lipschitz constants for lower-bounding and upper-bounding the partial derivatives.

Extensions. Finally, we argue how our algorithm easily extends to the setting of un-equal income (CE) when max to min income/weight ratios is polynomially bounded. Another natural extension we consider is to mixed manna, where each agent may like some items and dislike others. Again, using the characterization of [BMSY17], every instance can be put into one of the three categories, namely positive, negative, and null. We argue that the instance in the positive category can be solved using the Eisenberg-Gale convex program [EG59], and those in null have a trivial solution. For instances in the negative category, we discuss how our algorithm can be extended with simple modifications.

Linear Disutilities with Infinities. We note that, [CGMM20] that shows PPAD-hardness for the linear exchange model allows an agent to have infinite disutility for some chores indicating they do not have skills to do the chore in a reasonable amount of time. Our algorithm extends to this model as well, since their sufficiency conditions to ensure existence of equilibrium dictates that every component of the bipartite graph between agents and chores with finite disutility edges should be a complete bipartite graph. They show that even CEEI may not exist without this condition, and checking if it exists is NP-hard. Under this condition, it suffices to find CEEI for each of the connected component separately where there are no agent-chore pairs with infinite disutility.

In order to convey the main ideas cleanly we mainly focus on CEEI with chores in what follows, and refer the reader to the full version of the paper [BCM21] for the extensions.

2 Model and Our Results

In the chore division problem, a set of m *divisible* chores $[m] := \{1, \ldots, m\}$ is to be allocated to a set of n agents $[n] := \{1, \ldots, n\}$. It is without loss of generality to assume that exactly one unit of each chore needs to be allocated. Agent i's preferences (over chores) is represented by a non-negative, non-decreasing, and convex disutility function $D_i \colon \mathbb{R}^m_{\geq 0} \to \mathbb{R}_{\geq 0}$.[3] We denote by x_{ij} the fraction of item j that is allocated to agent i, and we denote $\boldsymbol{x}_i := (x_{i1}, \ldots, x_{im})$. We assume that D_i's are 1-homogeneous, *i.e.*

$$D_i(a \cdot \boldsymbol{x}_i) = a \cdot D_i(\boldsymbol{x}_i) \quad \text{for all } \boldsymbol{x}_i, \text{ and all } a \geq 0. \tag{2.1}$$

If $D_i(\cdot)$ is linear, then it is represented by $D_i(\boldsymbol{x}_i) = \sum_{j \in [m]} D_{ij} \cdot x_{ij}$ where $D_{ij} \in (0, \infty)$ is the disutility of agent i per unit of chore j.[4] Equivalently, we write $D_i(\boldsymbol{x}_i) = \langle \boldsymbol{D}_i, \boldsymbol{x}_i \rangle$ where $\boldsymbol{D}_i = (D_{i1}, D_{i2}, \ldots, D_{im})$. We also use $\overrightarrow{D}(\boldsymbol{x})$ to denote the disutility vector $(D_1(\boldsymbol{x}_1), D_2(\boldsymbol{x}_2), \ldots, D_n(\boldsymbol{x}_n))$.

[3] Typically, agents' preferences for chores are represented by non-positive, non-increasing, and concave utility functions since agents dislike chores [BMSY17]. By taking the negation of these utility functions we get non-negative, non-decreasing, convex disutility functions that agents want to minimize.

[4] If for some (i, j) pair $D_{ij} = 0$ then chore j can be freely allocated to agent i, and can be removed. Infinite disutilities can be handled as discussed in the introduction.

Competitive equilibrium with equal income (CEEI) At a CE with chores, payments are linear, and the j-th chore pays p_j per unit of the chore assigned. Let $\boldsymbol{p} = (p_1, \ldots, p_m)$ denote the vector of payments, and then the payment to agent i is $\langle \boldsymbol{p}, \boldsymbol{x}_i \rangle$. Each agent seeks to minimize their disutility subject to being paid at least 1 unit. We note that, under *equal income*, the exact value being paid is immaterial so long as all agents get paid the same amount. Prices $\boldsymbol{p}$ and allocation $\boldsymbol{x} = (\boldsymbol{x}_1, \boldsymbol{x}_2, \ldots, \boldsymbol{x}_n)$ are said to be at CEEI if all the chores are fully allocated when every agent consumes her least-disliked bundle with payment at least 1, *i.e.,* an optimal bundle. Formally [Var74, BMSY17]

$(E1)$ *(equal payments)* for all agents i and i' we have $\langle \boldsymbol{x}_i, \boldsymbol{p} \rangle = \langle \boldsymbol{x}_{i'}, \boldsymbol{p} \rangle$, and

$(E2)$ *(optimal bundle)* for all $i \in [n]$, we have $D_i(\boldsymbol{x}_i) \leq D_i(\boldsymbol{y}_i)$ for all $\boldsymbol{y}$ s.t. $\langle \boldsymbol{y}_i, \boldsymbol{p} \rangle \geq \langle \boldsymbol{x}_i, \boldsymbol{p} \rangle$, and

$(E3)$ *(feasible allocation)* for all $j \in [m]$, we have $\sum_{i \in [n]} x_{ij} = 1$.

It is known that the set of CEEI may be nonconvex, or even disconnected [BMSY17]. In light of this fact, and the PPAD-hardness of CE in the linear-exchange model [CGMM20], we turn our attention to approximately competitive equilibria. We formalize the notion of ε-CEEI as follows:

DEFINITION 2.1. *Prices $\boldsymbol{p}$ and allocation $\boldsymbol{x}$ are termed a ε-CEEI for an $\varepsilon \geq 0$, if and only if*

(1) for all agents i and i', we have $(1-\varepsilon) \cdot \langle \boldsymbol{x}_i, \boldsymbol{p} \rangle \leq \langle \boldsymbol{x}_{i'}, \boldsymbol{p} \rangle$, and

(2) for all $i \in [n]$, we have and $(1-\varepsilon) \cdot d_i(\boldsymbol{x}_i) \leq d_i(\boldsymbol{y}_i)$ for all $\boldsymbol{y}$ such that $\langle \boldsymbol{y}_i, \boldsymbol{p} \rangle \geq \langle \boldsymbol{x}_i, \boldsymbol{p} \rangle$, and

(3) for all $j \in [m]$, we have $1 - \varepsilon \leq \sum_{i \in [n]} x_{ij} \leq 1 + \varepsilon$.

It is well known that CEEI satisfy well-sought-after fairness and efficiency notions of *envy-freeness* and *Pareto-optimality* respectively. An allocation $\boldsymbol{x}$ is said to be envy-free (EF) if every agent prefers their own bundle over that of any other agent. And it is said to be Pareto-optimal (PO) if no other allocation Pareto-dominates it, *i.e.,* there is no feasible allocation $\boldsymbol{y}$ such that $D_i(\boldsymbol{y}_i) \leq D_i(\boldsymbol{x}_i)$ for all i, and for some agent k, $D_i(\boldsymbol{y}_i) < D_i(\boldsymbol{x}_i)$. In the full version of the paper [BCM21], we show that an ε-CEEI allocation approximately guarantees these properties.

Our main contribution in this paper is an FPTAS – a polynomial time algorithm to find an ε-CEEI when the disutility functions of the agents are *L-well-behaved*; We say that the disutility function D_i is L-well-behaved if (i) for all $j \in [m]$, we have $|D_i(\boldsymbol{x} + \delta \cdot \boldsymbol{e}_j) - D_i(\boldsymbol{x})| \geq \delta / L$ and (ii) for all $i \in [n]$ and all $\boldsymbol{x}, \boldsymbol{y} \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$, we have $|D_i(\boldsymbol{x}) - D_i(\boldsymbol{y})| \leq L \cdot ||\boldsymbol{x} - \boldsymbol{y}||_2$.

THEOREM 2.1. *Given black-box access to 1-homogeneous and convex disutilities $D_1, \ldots, D_n$ that are L-well-behaved, and also to their partial derivatives, there is an algorithm that finds an ε-CEEI in time polynomial in n, m, $1/\varepsilon$, and $\log(L)$.*

However, in this paper, we only explain how our result holds in the simpler setting when agents have linear disutility functions. The proof of Theorem 2.1 can be found in the full version of the paper [BCM21]. We also remark that for linear disutilities, we are able to prove the following stronger guarantee in Section 6.

THEOREM 2.1. *Given an instance I with linear disutility functions represented by $D_{11}, \ldots, D_{nm} > 0$, and an $\varepsilon > 0$, a stronger ε-CEEI can be computed in time* $\text{poly}\left(n, m, \log\left(\frac{\max_{ij} D_{ij}}{\min_{ij} D_{ij}}\right), \frac{1}{\varepsilon}\right)$ *where no error is incurred in the last two conditions,* i.e., $(\boldsymbol{x}, \boldsymbol{p})$ *that satisfies (1), $(E2)$, and $(E3)$.*

More importantly, our algorithm is an *exterior point method* that builds on tools from continuous optimization to find an *approximate KKT point*, which may be of independent interest. Next we give an overview of this method and it's analysis.

3 Overview of the Algorithm and Analysis

Our algorithm builds on the following characterization of CEEI due to Bogomolnaia et al. [BMSY17]: Analogous to the convex program of Eisenberg and Gale [EG59], the CEEI in the case of bads are characterized as local minima to the product of disutilities. However, this optimization program is over *disutility space*, rather than *allocation space*.

Formally, we let $\mathcal{F}$ denote the set of feasible allocations, namely

$$\mathcal{F} := \{\boldsymbol{x} \in \mathbb{R}^{nm} \,|\, \textstyle\sum_i x_{ij} = 1 \;\; \forall\, j,\; x_{ij} \geq 0 \;\; \forall\, i, j\} \;. \tag{3.2a}$$

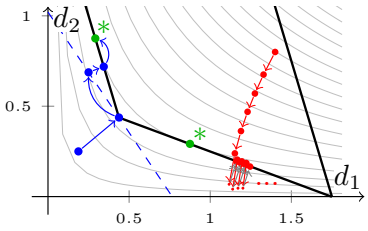

Figure 1: *A representation of both the exterior point method (in blue), and the pitfalls of gradient descent (in red) in minimizing the objective inside the feasible region. The light gray lines denote the level sets of the objective, orthogonal to the gradient.* ***Blue****: For the exterior point method, we start outside the region, find a nearest point and supporting hyperplane, jump to a new exterior point, and repeat until we find one of the two on-face local optima (green).* ***Red****: For the gradient descent, we start inside the region and quickly accelerate towards the $d_2 \geq 0$ boundary, which we wish to avoid.*

The disutility space $\mathcal{D}$ will be the set of all disutility profiles which can be attained over $\mathcal{F}$, or

$$\mathcal{D} := \{\boldsymbol{d} = (d_1, \ldots, d_n) \in \mathbb{R}^n \mid \exists\, \boldsymbol{x} = (\boldsymbol{x}_1, \ldots, \boldsymbol{x}_n) \in \mathcal{F} :\ D_i(\boldsymbol{x}_i) = d_i \ \forall\, i\} \ . \tag{3.2b}$$

In all that follows, we will distinguish between disutilities as functions and as variables by the upper- and lower-case symbols respectively. When disutility functions are linear, $\mathcal{D}$ is a polytope. However, for more general convex disutility functions, $\mathcal{D}$ may not be a convex set. We will remedy this by working instead with the extended feasible region $\mathcal{D}+\mathbb{R}^n_{\geq 0}$, the Minkowski sum, which is convex (Claim 5.1). This is the set of all disutility profiles which are *at least as bad* as some feasible profile, *i.e.,* disutility profiles attainable at over-allocations of the chores.

The characterization of Bogomolnaia et al. [BMSY17] states that any disutility profile $\boldsymbol{d}$ which is a local minimum (KKT point) to the following non-convex minimization program is the disutility profile of *some* CEEI, and the prices and allocation of this CEEI can be found by understanding $\boldsymbol{d}$ in allocation-space.

$$\min_{\boldsymbol{d} \in \mathcal{D}} \prod_{i=1}^n d_i \quad \text{s.t. } d_i > 0 \ \forall\, i \ .$$

Note that minimizing over the set $\mathcal{D}$ is equivalent to minimizing over the extended set $\mathcal{D} + \mathbb{R}^n_{\geq 0}$. And the KKT points to this program are equivalent to the KKT points for the minimization of the logarithm of objective, $\mathcal{L}(\boldsymbol{d}) := \sum_{i=0}^n \log(d_i)$. Hence the above program can equivalently stated as,

$$\min_{\boldsymbol{d} \in \mathcal{D}+\mathbb{R}^n_{\geq 0}} \mathcal{L}(\boldsymbol{d}) = \textstyle\sum_{i=0}^n \log(d_i) \quad \text{s.t. } d_i > 0 \ \forall\, i \ . \tag{3.3}$$

Primary difficulty. The open constraints $d_i > 0$ are both fundamental to the above characterization, and the source of the main difficulty of the problem. Any disutility profile $\boldsymbol{d}$ with a zero coordinate are trivial optima to these minimization problems, but are economically meaningless since no fairness or efficiency properties can be guaranteed. Furthermore, any naïve interior-point attempt at finding local minima are attracted by these constraints: the gradient of the objective $\mathcal{L}$ at $\boldsymbol{d}$ is inversely proportional to $\boldsymbol{d}$ componentwise, since $\frac{\partial}{\partial d_i}\mathcal{L} = 1/d_i$. This has the effect of accelerating gradient descent towards the $d_i \geq 0$ constraint for the smallest d_i value. See Figure 1 (red) for an illustration. This effect is robust to barrier methods at the boundaries, and thus gradient-following methods are not helpful in this task. The same problem afflicts attempts at strengthening the constraint to $d_i > \eta$ for some small η, as the dual variables for these constraint break the CEEI characterization. We circumvent this issue by designing an iterative *exterior-point method* that always increases the log-sum $\mathcal{L}$.

REMARK 3.1. *A natural question is if the combinatorial methods known for computing CE in linear Fisher (exchange) model with goods, e.g., [DPSV08, Orl10], extend to chores with linear disutilities? Unfortunately, they do not. In particular, the non-convexity and disconnectedness of the CEEI set is a primary difficulty in extending any algorithm from the goods setting to the chores setting. For example, these methods rely on the fact that CE allocation and prices changes continuously with the the (money) endowments of the agents [MV07], which is not true with chores. A chore division instance may have multiple disconnected equilibria some of which may disappear as we change these parameters, and as a result the said methods may get stuck. This fundamental bottleneck persists even if we restrict ourselves to approximate-CEEI.*

In the rest of this section, we will outline our approach first for linear disutility functions, and then afterwards in the general case. In the linear setting, many sub-routines can be solved exactly. Thus, it requires less technical detail to present, and serves as a good intuition for the more involved general case that we address later.

3.1 Linear Disutilities: Relating Approximate KKT to Approximate CEEI The main insights of our result are that (1) approximate KKT points allow for approximately competitive equilibria to be constructed, and (2) approximate KKT points can be found, despite the difficulty described above about ensuring strict positivity constraints. We begin here by formally defining the approximate KKT conditions.

Recall, KKT points are local optima where the cone of normal vectors of the tight constraints contains the function's gradient. Equivalently, there exists a supporting hyperplane at the local optimum whose normal vector is parallel to the function's gradient. Formally, $\boldsymbol{d}$ is a KKT point for the problem $\min_{\boldsymbol{d}\in\mathcal{D}} \mathcal{L}(\boldsymbol{d})$ if there exists a normal vector $\boldsymbol{a}$ such that: $\langle \boldsymbol{a}, \boldsymbol{y}\rangle = \langle \boldsymbol{a}, \boldsymbol{d}\rangle\}$ is a supporting hyperplane for $\mathcal{D} + \mathbb{R}^n_{\geq 0}$, and there exists some $c > 0$ such that $\boldsymbol{a} = c \cdot \nabla\mathcal{L}(\boldsymbol{d})$, *i.e.* $a_i = c/d_i$ for all i.

Since our procedure is iterative, it will converge in the limit to a KKT point, but only approximately after finitely many iterations. We show that after a polynomial number of iterations, it finds an *approximate KKT point*, defined below, with inverse-polynomial error.

DEFINITION 3.1. (γ-APPROXIMATE KKT) *For $\gamma \geq 1$, we say a point $\boldsymbol{d}$ along with the normal direction $\boldsymbol{a}$ is a γ-KKT point for problem* (3.3) *if*

(1) $\boldsymbol{d} \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$, (2) $\gamma_i^{-1} \leq a_i \cdot d_i \leq \gamma_i$ for all i, and (3) $\mathcal{D} + \mathbb{R}^n_{\geq 0} \subseteq \{\boldsymbol{y} \in \mathbb{R}^n | \langle \boldsymbol{a}, \boldsymbol{y}\rangle \geq \langle \boldsymbol{a}, \boldsymbol{d}\rangle\}$.

Informally, each entry of $\boldsymbol{a}$ is a γ-approximation of $\mathbf{1}/\boldsymbol{d}$, the gradient of $\mathcal{L}$, and $\boldsymbol{a}$ is normal to a supporting hyperplane for $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ at $\boldsymbol{d}$. Furthermore, we say $\boldsymbol{d}$ is a γ-KKT point if there exists a vector $\boldsymbol{a}$ such that $(\boldsymbol{d}, \boldsymbol{a})$ satisfy the above conditions.

Recall, when disutilities are linear, $\mathcal{D}$ is a linear polytope and is therefore convex. Hence, the above definition need not be defined over $\mathcal{D} + \mathbb{R}^n_{\geq 0}$, but we introduce it as it will be necessary later.

We outline here the first insight of our result, that approximate local minima give approximate equilibria. In the original analysis of the Eisenberg-Gale program [EG59] for goods, and more notably in the proof of [BMSY17] for chores, the relationship between competitive equilibria and local maxima hinges on the gradient being inversely proportional to the marginal (dis)utility-per-dollar incurred. Intuitively, the KKT conditions enforce that the payment to each agent (in the chores setting) is perfectly balanced by their disutility incurred, and their payment is equal to that of any other player. It can be shown that if some player is paid more, then the KKT conditions are violated. Thus, we can conclude condition (1) of Definition 2.1, with $\varepsilon = 0$. Condition (2) is argued using the fact that an agent is only allocated her minimum disutility-per-dollar chores, and (3) is true by definition since the allocation lies in $\mathcal{F}$.

To extend this argument to the approximate setting, it suffices to observe that when multiplicative error is introduced in the gradient direction, then this argument suffers only multiplicatively. A γ-sized error bound in the gradient direction allows for some player to be paid γ less than the unit, and another γ more, which allows us to show that γ-KKT points satisfy condition (1) of Definition 2.1 with $\varepsilon = (1 - \gamma^2)$. As above, condition (2) is argued similarly with the same ε, and condition (3) holds with $\varepsilon = 0$, again by feasibility. Formally, by extending the argument of Bogomolnaia et al. [BMSY17], we show the following.

THEOREM 3.1. *Let $(\boldsymbol{d}, \boldsymbol{a})$ be a $(1+\varepsilon)$-KKT point for the problem of minimizing $\mathcal{L}(\boldsymbol{d})$ subject to $\boldsymbol{d} \in \mathcal{D}$, and $\mathcal{L}(\boldsymbol{d}) > -\infty$. Let $\boldsymbol{x} \in \mathcal{F}$ be any allocation that realizes $\boldsymbol{d}$, i.e. $D_i(\boldsymbol{x}_i) = d_i$ for all i. Then there exists payments $\boldsymbol{p} = (p_1, \ldots, p_m)$ such that $(\boldsymbol{x}, \boldsymbol{p})$ form a stronger 2ε-CEEI, where no error is incurred in the last two conditions,* i.e., *$(\boldsymbol{x}, \boldsymbol{p})$ satisfies (1), $(E2)$, and $(E3)$.*

Furthermore, when disutilities are linear, the allocation $\boldsymbol{x}$ and payments $\boldsymbol{p}$ can be computed exactly in polynomial time from the disutility profile $\boldsymbol{d}$ and normal vector $\boldsymbol{a}$.

This theorem is proven in Section 6.1, Theorem 6.1, and the first part of it does not require that the disutilities be linear. However, as we will see below, γ-KKT points can only be guaranteed when disutilities are linear, and the definitions will need to be modified for the general case. The allocation and prices can be efficiently computed when disutilities are linear because they are the solutions to linear feasibility problems. With this theorem in hand, it remains therefore to compute $(1 + \varepsilon)$-KKT points, discussed next.

3.2 Linear Disutilities: Exterior Point Methods for Approximate KKT Points. Here we discuss our approach to find approximate-KKT point in polynomial time; formal details are presented in Section 6.2. As discussed above, it is tempting to hope that interior-point methods will find local minima efficiently, but they will not work in this setting. Instead, we will

rely on the geometry of the feasible space and objective function to allow us to repeatedly make guesses at KKT points, all the while increasing along the objective $\mathcal{L}(\boldsymbol{d}) = \sum_i \log(d_i)$, which we treat as a potential function. This potential will ensure that if we do not find approximate KKT points, then we make significant progress, dependent on the degree of precision γ needed. By bounding the values that the potential can take, this will suffice to show that the procedure is an FPTAS. Refer to Figure 1 (blue) for a pictorial representation of the algorithm.

Our "guesses" at KKT points are made by starting with an exterior, infeasible point $\boldsymbol{d}$, and finding the nearest feasible point $\boldsymbol{d}_*$ to it. Formally, $\boldsymbol{d}_*$ is the solution to $\min_{\boldsymbol{y}\in\mathcal{D}} \|\boldsymbol{y} - \boldsymbol{d}\|_2^2$. Using the fact that it is the nearest point to $\boldsymbol{d}$ in the ℓ_2 sense, we show that $\boldsymbol{a} = \boldsymbol{d}_* - \boldsymbol{d}$ is normal to a supporting hyperplane for $\mathcal{D}$ at $\boldsymbol{d}_*$ (Lemma 6.1). Notice that, so long as $\boldsymbol{d}_* \geq \boldsymbol{d}$ componentwise, then this all still holds when replacing $\mathcal{D}$ with $\mathcal{D} + \mathbb{R}^n_{\geq 0}$. Furthermore, this will ensure that we are increasing along the potential $\mathcal{L}$.

It remains to find the start of the next iterate, while ensuring that we are increasing in the $\mathcal{L}$ direction. Note that we have that the hyperplane $\{\boldsymbol{y} \in \mathbb{R}^n | \langle \boldsymbol{a}, \boldsymbol{y}\rangle = \langle \boldsymbol{a}, \boldsymbol{d}_*\rangle\}$ is supporting for $\mathcal{D}$, and therefore none of the points on this hyperplane are in the interior of $\mathcal{D}$. Thus, we can choose our next starting point to be the $\mathcal{L}$-maximizing point on this hyperplane. Since $\boldsymbol{d}_*$ is also feasible, this will ensure that we are increasing in the $\mathcal{L}$ direction, and that we are starting from a new exterior, infeasible point. This $\mathcal{L}$-maximizer on the hyperplane can be found efficiently, since we have a closed form for it: the maximizer on the hyperplane will be the point at which $\nabla\mathcal{L}$ is proportional to $\boldsymbol{a}$, and we show that it is exactly a rescaling of $(1/a_1, \ldots, 1/a_n)$ (Claim 6.5).

Thus, the algorithm is iterative, and each round $k \geq 0$ proceeds as follows:

0. $\boldsymbol{d}^k$ is the infeasible point "lying below" $\mathcal{D}$ starting the round. $\boldsymbol{d}^0$ is any infeasible point.
1. Set $\boldsymbol{d}^k_*$ to be the nearest feasible point to $\boldsymbol{d}^k$, *i.e.* the solution to $\min_{\boldsymbol{d}\in\mathcal{D}+\mathbb{R}^n_{\geq 0}} \|\boldsymbol{d} - \boldsymbol{d}^k\|_2^2$.
2. Set $\boldsymbol{a}^k \propto \boldsymbol{d}^k_* - \boldsymbol{d}_k$, rescaled so that $\langle \boldsymbol{a}^k, \boldsymbol{d}^k_*\rangle = n$.
3. Define $\boldsymbol{d}^{k+1}$ to be $(1/a_1, \ldots, 1/a_n)$, the maximizer of $\mathcal{L}(\boldsymbol{d})$ subject to $\langle \boldsymbol{a}^k, \boldsymbol{d}\rangle = n$.
4. Stop if $\boldsymbol{d}^{k+1}$ is "close enough" to $\boldsymbol{d}^k_*$, otherwise repeat.

We initialize the procedure at any infeasible point "lying below" the feasible region. When disutilities are linear, this can be found by noticing that we can lower-bound the disutilities over the feasible region, and picking an allocation which assigns half of the lower bound to each agent (Claim 6.4). The normalization in Step 2 ensures that if $\boldsymbol{a}^k$ is approximately parallel to the gradient $\nabla\mathcal{L}(\boldsymbol{d}^k_*)$, then it is also of the right magnitude. The notion of "close enough" in Step 4 is multiplicative, as it measures increase in the potential function $\sum_{i=1}^n \log(d_i)$.

The Potential Function, and Convergence Rates. As discussed above, we wish to use $\mathcal{L}(\boldsymbol{d}) = \sum_{i=1}^n \log(d_i)$ as a potential function to measure the progress of the algorithm. For each iteration $k \geq 0$, we will have $\mathcal{L}(\boldsymbol{d}^k) \leq \mathcal{L}(\boldsymbol{d}^k_*) \leq \mathcal{L}(\boldsymbol{d}^{k+1})$ (Claim 6.5). This first inequality is due to the observation that the nearest feasible point to $\boldsymbol{d}^k$ Pareto dominates it, and $\mathcal{L}$ is monotone increasing in each coordinate. The second inequality is by construction, as $\boldsymbol{d}^{k+1}$ maximizes $\mathcal{L}$ on a hyperplane that contains $\boldsymbol{d}^k_*$.

It remains then to argue that progress along $\mathcal{L}$ is rapid, relative to its range. We noted above that the stopping condition in Step 4 is multiplicative. Formally, we stop when the ℓ_1 norm of the logarithmic difference, *i.e.* $\sum_{i=1}^n \left|\log\left((\boldsymbol{d}^k_*)_i/(\boldsymbol{d}^{k+1})_i\right)\right|$ is at most ε. When this log-distance is more than ε, we show that the objective $\mathcal{L}$ increases by at least $\Omega(\varepsilon^2/n^2)$ (Lemma 6.3).

Conversely, when this log-distance is upper-bounded by ε, we will show that $(\boldsymbol{d}^k_*, \boldsymbol{a}^k)$ form a $(1+\varepsilon)$-KKT point (Lemma 6.2). Thus, since $\log(D_i(\boldsymbol{x}_i))$ is bounded over the feasible region, we will be able to bound the maximum number of iterations as a polynomial in $1/\varepsilon$, n, and $-\mathcal{L}(\boldsymbol{d}^0)$. This allows us to argue that an approximate equilibrium may be found in polynomially many iterations.

Implementing Iterations in Polynomial Time. We have argued above that an approximate KKT point, and therefore an approximate equilibrium, can be found in polynomially many iterates of the exterior point method. However, it remains to show that each step can be solved efficiently.

With the exception of Step 1 above, the rest of the algorithm is arithmetic, which can be easily performed. The minimization problem in Step 1 may pose a problem in general, if we expect an exact minimum. This is the source of the extra care needed in the general case. However, in the case of linear disutilities, we show that the minimization problem is actually a quadratic program with a semidefinite bi-linear form over $\mathcal{F}$ space (Lemma 6.5), and methods for finding exact solutions to such programs have long been known [KTK80].

Finally, we note that although each step in our algorithm generates polynomial sized rational numbers wrt it's parameters, one needs to be careful about how their bit-sizes grow. This can be taken care of by rounding down the $\boldsymbol{d}^k$ to a nearest rational vector with polynomial bit-size at the end of each iteration. Note that this step will ensure that $\boldsymbol{d}^k$ lies below $\mathcal{D}$ and we also argue why the bound on the iterations still hold: Since at every iteration k of our algorithm, the value of each d_i^k can be lower bounded using the value of the potential at $\boldsymbol{d}^k$ and the upper bound on the maximum disutility values in $\boldsymbol{d}^k$,[5] such a rounding is possible without hitting the $d_i \geq 0$ boundary, and while ensuring at least $\Omega(\varepsilon^2/n^2)$ increase in the potential $\mathcal{L}(\cdot)$. However, to convey the main important technical ideas, in Section 6 we focus on bounding number of arithmetic operations.

Putting all the above together, we get an FPTAS to compute stronger approximate CEEI where the last two conditions of ε-CEEI are satisfied with $\varepsilon = 0$ (Theorem 6.2).

3.3 General 1-Homogeneous Disutilities In general, the disutility functions are 1-homogeneous and convex, and are given as a *value oracle* black-box, along with a value oracle for their partial derivatives. In this section we outline the new issues that arise in extending our algorithm, and their resolutions.

At a high-level the issues are as follows: First, to find the nearest points we need to employ *interior point* methods which returns approximate solutions, and in turn we incur error in the hyperplane as well as the gradient. Secondly, in order to use the interior point method, we will have to work in the allocation space and can not work with the disutility space directly. This causes problems as convex constraints in disutility space need not be convex in allocation space. We elaborate these two issues and also highlight how we overcome them. Finally, we give an overview of the entire algorithm by putting everything together.

Finding Approximate Nearest Point. Recall, we have defined $\overrightarrow{D}(\boldsymbol{x}) := (D_1(\boldsymbol{x}_1), \ldots, D_n(\boldsymbol{x}_n))$. The natural program to find the nearest point in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ to a point $\boldsymbol{d}$ below $\mathcal{D}$ (or equivalently outside $\mathcal{D} + \mathbb{R}^n_{\geq 0}$) requires finding a $\boldsymbol{x} \in \mathcal{F}'$ that minimizes $||\overrightarrow{D}(\boldsymbol{x}) - \boldsymbol{d}||_2^2$, where $\mathcal{F}' := \{\boldsymbol{y} \in \mathbb{R}^{nm}_{\geq 0} \mid \sum_{i\in[n]}(\boldsymbol{y})_{ij} \geq 1 \text{ for all } j \in [m]\}$. Unfortunately the objective function is not necessarily convex[6]. One way to ensure it's convexity is to put additional constraints of the form $D_i(\boldsymbol{x}_i) \geq \boldsymbol{d}_i$ for all $i \in [n]$. But again, since the disutility functions $D_i(\cdot)$ are convex, these constraints create non-convex feasible region.

We come up with an alternative formulation for finding the approximate nearest point which is convex. The crucial observation is the fact that given any point $\boldsymbol{d}$ outside $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ there exists no point $\boldsymbol{d}' \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$, such that $\boldsymbol{d}$ *Pareto-dominates*(coordinate-wise larger or equal) $\boldsymbol{d}'$, i.e., $\boldsymbol{d}'$ does not belong in the negative orthant centered at $\boldsymbol{d}$. Therefore, a point $\boldsymbol{d} \in \mathbb{R}^n_{\geq 0}$ can Pareto-dominate any point $\boldsymbol{d}' \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$ if and only if $\boldsymbol{d} \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$. We now show how to use this fact to come up with a convex program to find the nearest point in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$. Our goal is to find a vector $\boldsymbol{\beta} \in \mathbb{R}^n$ of smallest magnitude and a point $\boldsymbol{d}' \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$ such that the point $\boldsymbol{d} + \boldsymbol{\beta}$ Pareto-dominates $\boldsymbol{d}'$: Note that this is only possible when $\boldsymbol{d} + \boldsymbol{\beta} \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$. Since $||\boldsymbol{\beta}||_2^2$ is minimum, $\boldsymbol{d} + \boldsymbol{\beta}$ is the nearest point in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ to $\boldsymbol{d}$. Formally,

$$
\begin{aligned}
\text{minimize} \quad & \sum_{i\in[n]} ((\boldsymbol{\beta})_i)^2 & \\
\text{subject to} \quad & \sum_{i\in[n]} z_{ij} \geq 1, & \forall j \in [m] \\
& z_{ij} \geq 0, & \forall i \in [n], \forall j \in [m] \\
& D_i(\boldsymbol{z}_i) - (\boldsymbol{d})_i - (\boldsymbol{\beta})_i \leq 0, & \forall i \in [n],
\end{aligned}
$$

It is easy to verify that the above program minimizes a convex function over a convex domain. The above convex program returns point $\boldsymbol{z} \in \mathcal{F}'$ such that $\overrightarrow{D}(\boldsymbol{z})$ is the nearest point in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ to $\boldsymbol{d}$. Unfortunately, this program cannot be solved exactly in polynomial time and therefore we need to argue about how to extract an approximate-CEEI given an approximate nearest neighbour.

[5]Note that we start with a $\boldsymbol{d}^0$ where each agent has a non-negligible disutility, and at any point in time, the disutilities of the agents in $\boldsymbol{d}^k$ are upper-bounded (as the disutility vector lies below $\mathcal{D}$), implying that there cannot be a significant increase in the disutility of any agent throughout the algorithm. Also, since the sum of logs of the disutilities $\mathcal{L}(\cdot)$ is increasing throughout the algorithm, we can conclude that there cannot be a significant decrease in the disutility of any agent throughout the algorithm, implying that the disutilities in $\boldsymbol{d}^k$ are also lower bounded.

[6]The natural sufficient condition for composition of two convex functions to be convex is if the outer function is monotone in the variables. We do not have this with our current objective function.

Approximate Supporting Hyperplane and $(\lambda, \gamma, \delta)$-KKT Points. In polynomial time, we can only find an approximate nearest neighbour of a point $\boldsymbol{d}$ in $\mathcal{D}+\mathbb{R}^n_{\geq 0}$. Therefore, our supporting hyperplanes will also be approximate, and therefore we need to redefine the approximate KKT points that we can compute. Let $\overrightarrow{D}(\boldsymbol{z}^*)$ be the nearest point in $\mathcal{D}+\mathbb{R}^n_{\geq 0}$ to $\boldsymbol{d}$. Then, $\overrightarrow{D}(\boldsymbol{z}^*)-\boldsymbol{d}$ is normal to a supporting hyperplane of $\mathcal{D}+\mathbb{R}^n_{\geq 0}$ at $\overrightarrow{D}(\boldsymbol{z}^*)$, i.e., $\langle \overrightarrow{D}(\boldsymbol{z}^*)-\boldsymbol{d}, \boldsymbol{y}\rangle = \langle \overrightarrow{D}(\boldsymbol{z}^*)-\boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}^*)\rangle$ is a supporting hyperplane of $\mathcal{D}+\mathbb{R}^n_{\geq 0}$ at $\overrightarrow{D}(\boldsymbol{z}^*)$. Since we have access only to an approximate nearest neighbour of $\boldsymbol{d}$, say $\overrightarrow{D}(\boldsymbol{z}')$, we wish to have $\langle \overrightarrow{D}(\boldsymbol{z}')-\boldsymbol{d}, \boldsymbol{y}\rangle = \langle \overrightarrow{D}(\boldsymbol{z}')-\boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}')\rangle$ as an approximate supporting hyperplane, i.e. $\langle \overrightarrow{D}(\boldsymbol{z}')-\boldsymbol{d}, \boldsymbol{y}\rangle \geq \langle \overrightarrow{D}(\boldsymbol{z}')-\boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}')\rangle - \delta$ for all $\boldsymbol{y} \in \mathcal{D}+\mathbb{R}^n_{\geq 0}$ for a sufficiently small δ.

With this, we introduce the notion of $(\lambda, \gamma, \delta)$-KKT points.

DEFINITION 3.2. ($(\lambda, \gamma, \delta)$-APPROXIMATE KKT) *We say $(\boldsymbol{a}, \boldsymbol{d}, \boldsymbol{x})$, i.e., a point $\boldsymbol{d}$ along with the normal direction $\boldsymbol{a}$ and a pre-image $\boldsymbol{x}$ is a $(\lambda, \gamma, \delta)$-KKT point with $\lambda \geq 1$, $\gamma \geq 1$, and $\delta > 0$, for the minimization problem on $\mathcal{D}+\mathbb{R}^n_{\geq 0}$ if*

1. *$x_{ij} \geq 0$ for all $i \in [n]$ and $j \in [m]$, and $\lambda^{-1} \leq \sum_{i\in[n]} x_{ij} \leq \lambda$ for all $j \in [m]$,*
2. *$\boldsymbol{d} = \overrightarrow{D}(\boldsymbol{x})$ and $\gamma_i^{-1} \leq a_i \cdot d_i \leq \gamma_i$ for all $i \in [n]$, and*
3. *and $\mathcal{D}+\mathbb{R}^n_{\geq 0} \subseteq \{\boldsymbol{y} \in \mathbb{R}^n | \langle \boldsymbol{a}, \boldsymbol{y}\rangle \geq \langle \boldsymbol{a}, \boldsymbol{d}\rangle - \delta = n - \delta\}$.*

Informally, all chores are almost fully allocated, each entry of $\boldsymbol{a}$ is a γ-approximation of $\mathbf{1}/\boldsymbol{d}$, the gradient of $\mathcal{L}$, and $\boldsymbol{a}$ is a δ-approximately-supporting hyperplane for $\mathcal{D}+\mathbb{R}^n_{\geq 0}$.

In the full version of the paper [BCM21], we show that a $(\lambda, \gamma, \delta)$-KKT point with where $\lambda = 1+\varepsilon/2^{\text{poly}(n,m)}$, $\gamma = 1+\varepsilon$ and $\delta = \varepsilon/2^{\text{poly}(n,m)}$ can be mapped to a $\varepsilon^{1/6}$-CEEI. The proof emulates the proof in [BMSY17], and consequently matches the proof in the linear case.

However, some subtle problems arise when generalizing the algorithm from the linear case to determine a $(\lambda, \gamma, \delta)$-KKT point. Firstly, the convergence of the entire algorithm relies crucially on the fact that the potential $\mathcal{L}(\boldsymbol{d})$ never decreases at any point. For this, we require that we have $\overrightarrow{D}(\boldsymbol{z}')$ Pareto-dominate $\boldsymbol{d}$. We can ensure this by first computing an arbitrary approximate nearest point $\boldsymbol{z}''$ and then increase the consumption of certain chores in $\boldsymbol{z}''$ to get $\boldsymbol{z}'$ such that $\overrightarrow{D}(\boldsymbol{z}')$ Pareto-dominates $\boldsymbol{d}$. Since we know that $\overrightarrow{D}(\boldsymbol{z}^*)$ Pareto-dominates $\boldsymbol{d}$, and $||\overrightarrow{D}(\boldsymbol{z}'')-\overrightarrow{D}(\boldsymbol{z}^*)||_2$ is small, the increase in consumption of the chores will also be small.

Secondly, the hyperplane $\langle \overrightarrow{D}(\boldsymbol{z}')-\boldsymbol{d}, \boldsymbol{y}\rangle = \langle \overrightarrow{D}(\boldsymbol{z}')-\boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}')\rangle$ can be a good approximation of the hyperplane $\langle \overrightarrow{D}(\boldsymbol{z}^*)-\boldsymbol{d}, \boldsymbol{y}\rangle = \langle \overrightarrow{D}(\boldsymbol{z}^*)-\boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}^*)\rangle$ (or equivalently δ is inverse-exponentially small) only if $||\boldsymbol{d}-\overrightarrow{D}(\boldsymbol{z}^*)||_2$ is significantly larger than $||\overrightarrow{D}(\boldsymbol{z}^*)-\overrightarrow{D}(\boldsymbol{z}')||_2$. Therefore, if at any point in our algorithm, we have $||\boldsymbol{d}-\overrightarrow{D}(\boldsymbol{z}')||_2 \leq M\varepsilon$ for a sufficiently large M, where $\varepsilon \geq ||\overrightarrow{D}(\boldsymbol{z}')-\overrightarrow{D}(\boldsymbol{z}^*)||_2$, then we stop and return a pre-image of $\boldsymbol{d}$ (note that as the disutility functions are 1-homogeneous, this can be done by appropriately scaling the consumption of chores for each agent).

Finally, and most importantly, we need to ensure that the approximate supporting hyperplanes do not introduce point with excessive over-allocation. Let $\langle \boldsymbol{a}^*, \boldsymbol{y}\rangle = n$ and $\langle \boldsymbol{a}', \boldsymbol{y}\rangle = n$ represent the hyperplanes $\langle \overrightarrow{D}(\boldsymbol{z}^*)-\boldsymbol{d}, \boldsymbol{y}\rangle = \langle \overrightarrow{D}(\boldsymbol{z}^*)-\boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}^*)\rangle$ and $\langle \overrightarrow{D}(\boldsymbol{z}')-\boldsymbol{d}, \boldsymbol{y}\rangle = \langle \overrightarrow{D}(\boldsymbol{z}')-\boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}')\rangle$ respectively after appropriate scaling, i.e., $\boldsymbol{a}^\ell = \big(n/\langle \overrightarrow{D}(\boldsymbol{z}^\ell)-\boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}^\ell)\rangle\big)\cdot(\overrightarrow{D}(\boldsymbol{z}^\ell)-\boldsymbol{d})$ for $\ell \in \{*, '\}$. Since we are dealing with approximate supporting hyperplane[7], the point maximizing $\mathcal{L}$, say $\boldsymbol{d}'$ on $\langle \boldsymbol{a}', \boldsymbol{y}\rangle = n$, maybe contained in the strict interior of $\mathcal{D}+\mathbb{R}^n_{\geq 0}$. Also note that in this case, the nearest point in $\mathcal{D}+\mathbb{R}^n_{\geq 0}$ to $\boldsymbol{d}'$ is $\boldsymbol{d}'$ itself, and therefore the distance between $\boldsymbol{d}'$ and its approximate nearest point in $\mathcal{D}+\mathbb{R}^n_{\geq 0}$ is significantly smaller than $M\varepsilon$ and our algorithm will return the point $\boldsymbol{d}'$, the normal to the hyperplane $\boldsymbol{a}'$ and its pre-image, say $\boldsymbol{x}'$ in the very next iteration. Now note that while conditions (2) and (3) in Definition 3.2 are satisfied, condition (1) may not be satisfied. In particular, there could be chores that are significantly over-allocated! At first this may seem to be counter-intuitive as the hyperplane $\langle \boldsymbol{a}', \boldsymbol{y}\rangle = n$ is a good approximation of the exact supporting hyperplane $\langle \boldsymbol{a}^*, \boldsymbol{y}\rangle = n$, and, the point $\boldsymbol{d}^*$ that maximizes $\mathcal{L}$ on $\langle \boldsymbol{a}^*, \boldsymbol{y}\rangle = n$ lies outside $\mathcal{D}+\mathbb{R}^n_{\geq 0}$ and as a result no chores are over allocated in a pre-image of $\boldsymbol{d}^*$. However, we show that the disutility profiles of the point $\boldsymbol{d}^*$ maximizing $\mathcal{L}$ on the hyperplane $\langle \boldsymbol{a}^*, \boldsymbol{y}\rangle = n$ and the point $\boldsymbol{d}'$ maximizing $\mathcal{L}$ on the hyperplane $\langle \boldsymbol{a}', \boldsymbol{y}\rangle = n$ can be very far apart even if $||\boldsymbol{a}'-\boldsymbol{a}^*||_2$ is small[8]. This is primarily due to the fact that $\boldsymbol{d}' = \big(\frac{1}{a'_1}, \frac{1}{a'_2}, \ldots, \frac{1}{a'_n}\big)$ and $\boldsymbol{d}^* = \big(\frac{1}{a^*_1}, \frac{1}{a^*_2}, \ldots, \frac{1}{a^*_n}\big)$, and even though $|a'_i - a^*_i| \leq \varepsilon$ for

[7] $\langle \boldsymbol{a}', \boldsymbol{y}\rangle \geq n - \delta'$ for all $\boldsymbol{y} \in \mathcal{D}+\mathbb{R}^n_{\geq 0}$, where $\delta' = \frac{\delta\langle \overrightarrow{D}(\boldsymbol{z}')-\boldsymbol{d}, \overrightarrow{D}(\boldsymbol{z}')\rangle}{n}$.

[8] In fact $||\boldsymbol{a}'-\boldsymbol{a}^*||_2$ will be small as $||\boldsymbol{d}'-\boldsymbol{d}^*||_2$ is significantly small.

all $i \in [n]$, $1/a_i'$ and $1/a_i^*$ can be very far apart. We circumvent this issue by showing that if there are some chores that are significantly over-allocated in $\boldsymbol{x}'$, then we can find an allocation $\boldsymbol{x}''$ from $\boldsymbol{x}'$ by reducing consumption of the over-allocated chores and re-allocating some of the not-over-allocated chores such that $\overrightarrow{D}(\boldsymbol{x}'') \in \mathcal{D} + \mathbb{R}^n_{\geq 0}$ and $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{x}'')\rangle < n - \delta'$, which is a contradiction to the fact that $\langle \boldsymbol{a}, \boldsymbol{y}\rangle = n$ is an approximate supporting hyperplane to $\mathcal{D} + \mathbb{R}^n_{\geq 0}$. This is where the bulk of the error analysis is required.

We now outline the entire procedure.

Putting it Together. Similar to the case with linear disutilities, the algorithm is iterative. In each iteration $k \geq 0$,

0. $\boldsymbol{d}^k$ is the infeasible point "lying below" $\mathcal{D}$ at the start of round k. $\boldsymbol{d}^0$ is any infeasible point.
1. Find $\boldsymbol{x}^k_+$ such that $\overrightarrow{D}(\boldsymbol{x}^k_+)$ is an ε-approximate nearest feasible point to $\boldsymbol{d}^k$ in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$, s.t. $(\boldsymbol{d}^k_+)_i \geq (\boldsymbol{d}^k)_i$ for all $i \in [n]$ and then *round up* $\boldsymbol{d}^k_+$ to the nearest rational point with polynomial bit size.
2. If $||\boldsymbol{d}^k_+ - \boldsymbol{d}^k||_2 \leq M \cdot \varepsilon$, then return $(\boldsymbol{a}^{k-1}, \boldsymbol{d}^k, \boldsymbol{x}^k)$ where $\boldsymbol{x}^k$ is a pre-image of $\boldsymbol{d}^k$ obtained by rescaling $\boldsymbol{x}^k_+$ appropriately, i.e., $(\boldsymbol{x}^k)_i \leftarrow (\boldsymbol{x}^k_+)_i \cdot \frac{(\boldsymbol{d}^k)_i}{(\boldsymbol{d}^k_+)_i}$ for all $i \in [n]$.
3. Set $\boldsymbol{a}^k \propto \boldsymbol{d}^k_+ - \boldsymbol{d}_k$, rescaled so that $\langle \boldsymbol{a}^k, \boldsymbol{d}^k_+\rangle = n$.
4. Define $\boldsymbol{d}^{k+1}$ to be $(1/a_1, \ldots, 1/a_n)$, the maximizer of $\mathcal{L}(\boldsymbol{d})$ subject to $\langle \boldsymbol{a}^k, \boldsymbol{d}\rangle = n$.
5. Return $(\boldsymbol{a}^k, \boldsymbol{d}^k_+, \boldsymbol{x}^k_+)$ if $\boldsymbol{d}^{k+1}$ is "close enough" to $\boldsymbol{d}^k_*$, otherwise repeat.

The algorithm has polynomially many iterations, since similar to the case when agents have linear disutilities, if it does not terminate in iteration k, then the potential $\mathcal{L}$ increases by at least $\Omega(\varepsilon^2/n^2)$. And $\mathcal{L}$ is upper bounded. By arguing that every iteration can be done in polynomial time, we get an FPTAS.

3.4 Organization We give a brief road map of the rest of the paper. In what follows, we first discuss some related work on CE in Section 4 and state some fundamental results from [BMSY17] that we use crucially for our algorithm design in Section 5 . Thereafter, we present the FPTAS when agents have linear disutilities in Section 6 so that the reader gets a good idea of the meta-level algorithm.

In the full version of the paper [BCM21], (i) we elaborate the FPTAS when agents have general 1-homogeneous disutilities and (ii) also discuss the extensions of our results to the setting when the items to be divided contain both goods and bads (*mixed manna*) with linear valuations, and when agents have unequal income needs (*CE in Fisher model*).

4 Related Work

Competitive equilibrium (CE) has been a fundamental concept in several economic models since the time of Léon Walras [Wal74] in the 19th century. In this paper, we primarily focus on CEEI, which is a special case of CE in Fisher markets, which again is a special case of CE in exchange markets (also referred to as Arrow-Debreu markets). The existence of CE under some mild assumption was proved in the exchange setting by Arrow and Debreu [AD54] and independently by Mackenzie [McK54, McK59]. However, the proofs of existence used fixed point theorems and were non-constructive. In the last few decades, there has been substantial contribution from the computer science community in coming up with constructive algorithms to determine a CE. As mentioned in the introduction, there has been a long line of convex programs, interior point and combinatorial polynomial time algorithms for determining CE with goods in both Fisher and the exchange setting [CDG$^+$17, DGV16, NP83, DPSV08, Orl10, Vég12, DM15, DGM16, GV19, CCD13]. There are also hardness results known when agents have more general utility functions [CPY17, CDDT09, CT09, Rub18]. The existence and computational complexity of CE and its relaxations have been studied in discrete settings (with indivisible objects) as well [FGL16].

The study of CE with chores/ bads has not received similar extensive investigation. One plausible reason could be that this does not capture a natural market and such a setting is interesting only from a fair division perspective. Nevertheless, the CE with bads exhibits far less structure than the CE with goods as explained in the introduction. There are polynomial time enumerative algorithms known only when there are constant number of agents or chores [BS19, GM20]. Quite recently, [CGMM21] gave an LCP formulation for determining CEEI with *mixed manna* (goods and bads) when the utility functions are separable piecewise-linear and concave (SPLC) which includes linear.

5 Preliminaries

Recall the chore division problem formalized in Section 2 above: We seek to divide m divisible chores among n agents with convex, 1-homogeneous disutility functions $D_1, \ldots, D_n$, through the mechanism of *competitive equilibrium with equal income (CEEI)*. In this section we state a characterization of CEEI and certain properties of the disutility space that are crucial for our results.

In the case of dividing goods, the seminal work of Eisenberg and Gale [EG59] shows that any allocation that maximizes the Nash welfare — or equivalently the geometric mean of the utilities — is at a CEEI. Since the Nash welfare maximization is a convex program, an approximate CEEI can be determined by an ellipsoid algorithm. Unfortunately, in the case of dividing bads, the set of equilibria could be non-convex and therefore one cannot hope for convex program formulation that captures equilibria [BMSY17]. However, a recent result by Bogomolnaia et al. [BMSY17] show a similar, but non-convex formulation for an exact CEEI (Definition 2.1, with $\varepsilon = 0$) with chores. In particular, [BMSY17] show that the conditions of an exact CE hold if and only if the disutility profile is a critical point for the Nash welfare on the boundary of the feasible region. Formally:

THEOREM 5.1. ([BMSY17]) *Let $\mathcal{F}$ and $\mathcal{D}$ be the feasible space of allocations and disutility profiles as defined in* (3.2)*. For some $\boldsymbol{d} \in \mathbb{R}^n$, denote the Nash social welfare as $\mathrm{NSW}(\boldsymbol{d}) := \prod_{i=1}^n d_i$. Then $\boldsymbol{d}$ can be achieved by a CEEI if and only if the following conditions all hold: a) $\boldsymbol{d} \in \mathcal{D}$, b) $\mathrm{NSW}(\boldsymbol{d}) > 0$, and c) $\boldsymbol{d}$ satisfies the KKT conditions for the problem of minimizing NSW on $\mathcal{D}$. Equivalently, $\boldsymbol{d}$ is on the lower-boundary of $\mathcal{D}$, but not on the boundary of $\mathbb{R}^n_{\geq 0}$, and the gradient $\nabla \mathrm{NSW}(\boldsymbol{d})$ is parallel to some supporting hyperplane normal for $\mathcal{D}$ at the point $\boldsymbol{d}$.*

Note that when dis-utilities are linear functions, $\mathcal{D}$ is a linear polytope. When dis-utilities are general, 1-homogeneous, convex functions, the set $\mathcal{D}$ need not be convex. However, $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ is convex (Minkowski sum of two convex sets), and we will therefore use it as our feasible region in the analysis.

CLAIM 5.1. *$\mathcal{D} + \mathbb{R}^n_{\geq 0}$ is convex, when the disutility functions $D_1, \ldots, D_n$ are convex.*

In the following sections, we extend Theorem 5.1 to map approximate KKT points to approximate CEEI (Definition 2.1), and then design an algorithm to find an approximate KKT point.

6 Polynomial-Time Algorithm for ε-CEEI under Linear Disutilities

In this section we present an algorithm to find an ε-CEEI in time polynomial in $\frac{1}{\varepsilon}$ and the size of the input instance, when agents have linear disutility functions. Recall that, the linear function of agent i is represented by $D_i(\boldsymbol{x}_i) = \sum_{j=1}^m D_{ij} x_{ij}$, or equivalently $D_i(\boldsymbol{x}_i) = \langle \boldsymbol{D}_i, \boldsymbol{x}_i \rangle$ where $\boldsymbol{D}_i = (D_{i1}, D_{i2}, \ldots, D_{im})$.

Our algorithm will ensure a stronger notion of approximation where all the chores are exactly allocated, *i.e.,* condition 3 in Definition 2.1 is satisfied exactly. For this, the algorithm finds a γ-KKT point as defined in Definition 3.1. Let us first discuss how such a KKT point gives a stronger approximate CEEI in the next section, thereby extending Theorem 5.1.

6.1 Approximate KKT Suffices to get Approximate CEEI

We begin with some notation: as we often use element-wise inverse of a vector, for any two n-dimensional vectors $\boldsymbol{x} = (x_1, \ldots, x_n)$ and $\boldsymbol{y} = (y_1, \ldots, y_n)$, we denote

$$\boldsymbol{x}/\boldsymbol{y} := (x_1/y_1, \ldots, x_n/y_n) \,.$$

Recall that we are interested in finding local minima for the logarithm of the Nash social welfare

$$\mathcal{L}(\boldsymbol{d}) := \log(\mathrm{NSW}(\boldsymbol{d})) = \sum_{i=1}^n \log(d_i) \,. \tag{6.4}$$

Observe that $\nabla \mathcal{L}(\boldsymbol{d}) = \mathbf{1}/\boldsymbol{d}$. From Definition 3.1, recall the γ-KKT point, $\gamma \geq 1$, for minimizing $\mathcal{L}$ on $\mathcal{D}$: point $\boldsymbol{d}$ on the boundary of $(\mathcal{D} + \mathbb{R}^n_{\geq 0})$, such that it has $\{\boldsymbol{y} \mid \boldsymbol{a}^\top \cdot \boldsymbol{y} \geq n\}$ as a supporting hyperplane for $\mathcal{D} + \mathbb{R}^n_{\geq 0}$, where $\boldsymbol{a} \in \mathbb{R}^n$ approximates $\nabla \mathcal{L}(\boldsymbol{d})$ coordinate-wise, i.e., $\forall i, \gamma^{-1} \leq \frac{a_i}{1/d_i} \leq \gamma$.

We emulate here the proof of Bogomolnaia et al. [BMSY17] to show that approximate KKT points give approximate CEEI. As stated in the overview, we wish to show the following.

THEOREM 6.1. *Let $(\boldsymbol{d}, \boldsymbol{a})$ be a $(1+\varepsilon)$-KKT point for the problem of minimizing $\mathcal{L}(\boldsymbol{d})$ subject to $\boldsymbol{d} \in \mathcal{D}$, and $\mathcal{L}(\boldsymbol{d}) > -\infty$. Let $\boldsymbol{x} \in \mathcal{F}$ be any allocation that realizes $\boldsymbol{d}$, i.e. $D_i(\boldsymbol{x}_i) = d_i$ for all i. Then there exists payments $\boldsymbol{p} = (p_1, \ldots, p_m)$ such*

that $(\boldsymbol{x}, \boldsymbol{p})$ *form a stronger* 2ε*-CEEI, where no error is incurred in the last two conditions,* i.e., $(\boldsymbol{x}, \boldsymbol{p})$ *satisfies (1),* $(E2)$*, and* $(E3)$*.*

Furthermore, when disutilities are linear, the allocation $\boldsymbol{x}$ *and payments* $\boldsymbol{p}$ *can be computed exactly in polynomial time from the disutility profile* $\boldsymbol{d}$ *and normal vector* $\boldsymbol{a}$*.*

Proof. Let $\gamma = (1+\varepsilon)$, then it suffices to show that γ-KKT gives $(1-\gamma^{-2})$-CEEI since $2\varepsilon > (1-\gamma^{-2})$ for $\varepsilon > 0$. Recall we have defined $\overrightarrow{D}(\boldsymbol{x}) := (D_1(\boldsymbol{x}_1), \ldots, D_n(\boldsymbol{x}_n))$, and sets $\mathcal{F}$ and $\mathcal{D}$ are as in (3.2), namely, the set of feasible allocations and the set of feasible disutility profiles, in general.

Defining and Computing the Allocation and Prices. Let $\boldsymbol{d}$ be the disutility profile of the approximate KKT point. Since $\mathcal{D} + \mathbb{R}^n_{\geq 0} \subseteq \{\boldsymbol{y} \mid \boldsymbol{a}^\top \boldsymbol{y} \geq \langle \boldsymbol{a}, \boldsymbol{d}\rangle\}$ and the entries of $\boldsymbol{a}$ are positive, then $\boldsymbol{d} \in \mathcal{D}$, by minimality. Now, consider any allocation $\boldsymbol{z}$ in $\mathcal{F}$, such that $\overrightarrow{D}(\boldsymbol{z}) = \boldsymbol{d}$.

For the second part of the statement of the theorem, we must show that $\boldsymbol{z}$ can be computed, as this will be the allocation of the approximate CEEI. In fact, it suffices to find an allocation vector $\boldsymbol{x}$ which simultaneously satisfies the non-negativity constraints of $\mathcal{F}$, and the linear equality constraints of $\mathcal{F}$ along with $\overrightarrow{D}(\boldsymbol{x}) = \boldsymbol{d}$. This can be solved by linear programming techniques in polynomial time.

We wish now to compute the prices at the allocation, for which we will need separating hyperplanes. To this end, define the set $S_\lambda := \{\boldsymbol{x} \in \mathbb{R}^{nm} \mid \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{x})\rangle \leq \lambda\}$. As the disutility functions are convex and continuous, we can conclude that set S_λ is closed, convex, and non-empty for all $\lambda > 0$, since $S_\lambda \ni \mathbf{0}$. When disutilities are linear, S_λ is in fact a closed half-space, since

$$\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{x})\rangle \leq n \iff \textstyle\sum_{i=1}^n a_i \sum_{j=1}^m D_{ij} x_{ij} \leq n\,.$$

Now, because $\langle \boldsymbol{a}, \boldsymbol{y}\rangle \geq \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z})\rangle$ for all $y \in \mathcal{D}$, we can conclude that S_λ does not intersect $\mathcal{F}$ for any $\lambda < \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z})\rangle$. Denote $S^* := S_{\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z})\rangle}$. The set S^* must be tangent to $\mathcal{F}$, since the D_i's are continuous, but $\boldsymbol{z} \in \mathcal{F} \cap S^*$. See Figure 2 for an illustration. Thus, there exists a half-space $H_{\boldsymbol{c}} := \{\boldsymbol{x} \mid \langle \boldsymbol{c}, \boldsymbol{x}\rangle \geq b\}$ which separates the two sets, *i.e.* $\mathcal{F} \subseteq H_{\boldsymbol{c}}$, and $S^* \subseteq \mathrm{cl}(H_{\boldsymbol{c}}^{\complement})$. Also, note that we must have $\langle \boldsymbol{c}, \boldsymbol{z}\rangle = b$. Note that when disutilities are linear, we have $c_{ij} = a_i D_{ij}$, and $b = n$ as the hyperplane separating $\mathcal{F}$ and S^* is $\langle a, \overrightarrow{D}(\boldsymbol{x})\rangle = n$.

Finally, we can define the prices at the allocation. Let $p_j := \min_i c_{ij}$, and let $\boldsymbol{p} := (p_1, \ldots, p_m)$. See Figure 2 for an illustration of the supporting hyperplanes $\langle \boldsymbol{a}, \boldsymbol{y}\rangle = \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z})\rangle$ in $\mathcal{D}$ and $\langle \boldsymbol{c}, \boldsymbol{x}\rangle = \langle \boldsymbol{c}, \boldsymbol{z}\rangle$ in $\mathcal{F}$.

It remains then to show that the allocation $\boldsymbol{z}$ and the price vector $\boldsymbol{p}$ satisfy the conditions in Definition 2.1 where the last two are satisfied without any error, since we have argued already that they can be computed efficiently.

Satisfying Condition (1) in Definition 2.1. We want to show that for all agents i and i', we have $\gamma^{-2} \cdot \langle \boldsymbol{z}_i, \boldsymbol{p}\rangle \leq \langle \boldsymbol{z}_{i'}, \boldsymbol{p}\rangle$. But first we make some simple but crucial observations about the price vector $\boldsymbol{p}$.

CLAIM 6.1. *We have* $\sum_{j\in[m]} p_j = \langle \boldsymbol{c}, \boldsymbol{z}\rangle = b$*.*

Proof. $\langle \boldsymbol{c}, \boldsymbol{x}\rangle \geq \langle \boldsymbol{c}, \boldsymbol{z}\rangle = b$ for all $\boldsymbol{x} \in \mathcal{F}$ by definition. Also, since $\boldsymbol{z} \in \mathcal{F}$, we can claim that $b = \min_{\boldsymbol{x}\in\mathcal{F}} \langle \boldsymbol{c}, \boldsymbol{x}\rangle$. Observe that $\min_{x\in\mathcal{F}} \langle \boldsymbol{c}, \boldsymbol{x}\rangle$ is obtained by assigning each chore fully to the agent that has the smallest c_{ij} value for it. Therefore, we have that $\min_{\boldsymbol{x}\in\mathcal{F}} \langle \boldsymbol{c}, \boldsymbol{x}\rangle = \sum_{j\in[m]} \min_{i\in[n]} c_{ij} = \sum_{j\in[m]} p_j$ (by the definition of p_j). ∎

Now, consider the half-space $H_{\boldsymbol{p}} = \{\boldsymbol{x} \in \mathbb{R}^{nm}_{\geq 0} \mid \sum_{i\in[n], j\in[m]} p_j \cdot x_{ij} \geq b\}$. We first observe that this half-space is entirely contained in $H_{\boldsymbol{c}}$.

CLAIM 6.2. *We have* $H_{\boldsymbol{p}} \subseteq H_{\boldsymbol{c}}$*.*

Proof. Consider any point $\boldsymbol{x} \in H_{\boldsymbol{p}}$. We have $b \leq \sum_{i\in[n]} \sum_{j\in[m]} x_{ij} \cdot p_j$. Since $p_j \leq c_{ij}$ for all $i \in [n]$, we have that $\sum_{i\in[n]} \sum_{j\in[m]} x_{ij} p_j \leq \sum_{i\in[n]} \sum_{j\in[m]} x_{ij} \cdot c_{ij}$, implying that $\sum_{i\in[n]} \sum_{j\in[m]} x_{ij} \cdot c_{ij} \geq b$, i.e., $\langle \boldsymbol{c}, \boldsymbol{x}\rangle \geq b$. Therefore $\boldsymbol{x} \in H_{\boldsymbol{c}}$. ∎

Finally, note that every point $\boldsymbol{x} \in \mathcal{F}$ is also contained in $H_{\boldsymbol{p}}$.

CLAIM 6.3. *Consider any* $\boldsymbol{x} \in \mathcal{F}$*. Then* $\boldsymbol{x} \in H_{\boldsymbol{p}}$*.*

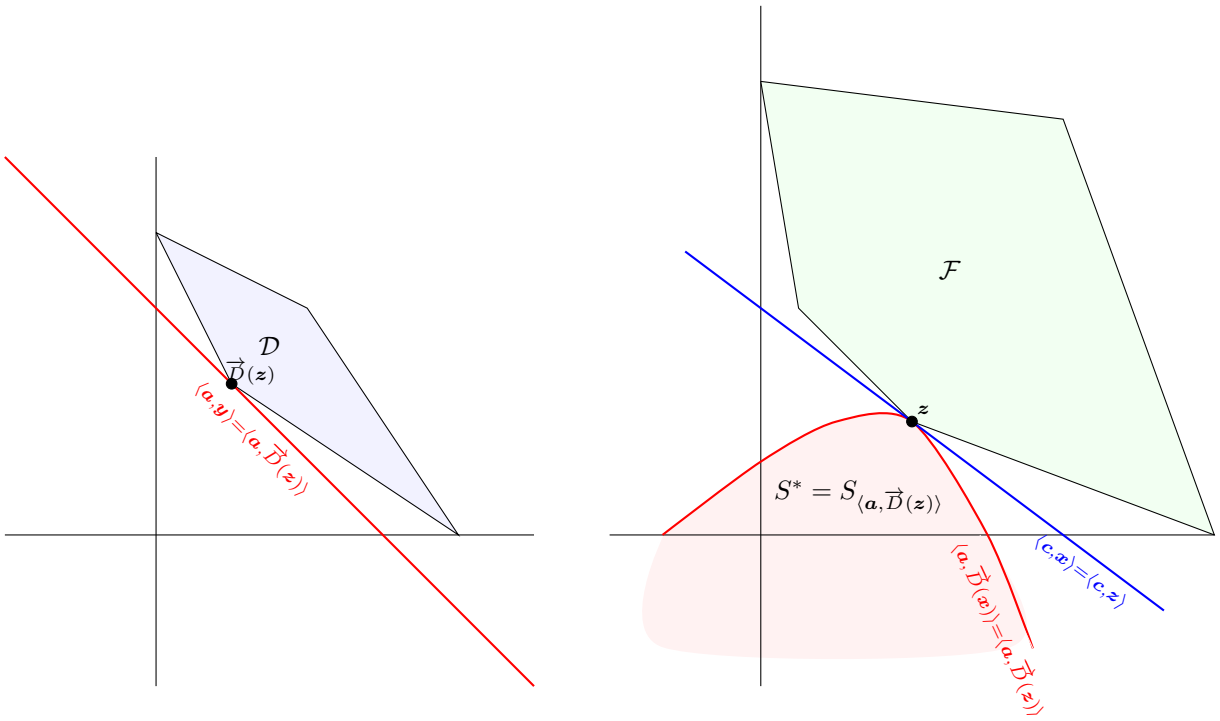

Figure 2: Illustration of the supporting hyperplanes: $\boldsymbol{z} \in \mathcal{F}$ is a point such that $(\overrightarrow{D}(\boldsymbol{z}), \boldsymbol{a})$ satisfies the approximate KKT conditions in Definition 3.2. Thus, we have a supporting hyperplane $\langle \boldsymbol{a}, \boldsymbol{y} \rangle = \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle = n$ of $\mathcal{D}$ such that $\gamma^{-1} \leq a_i \cdot D_i(z_i) \leq \gamma$ (left). The figure on the right describes the set $S^* = S_{\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle}$ and the hyperplane $\langle \boldsymbol{c}, \boldsymbol{x} \rangle = \langle \boldsymbol{c}, \boldsymbol{z} \rangle$ that separates $\mathcal{F}$ from S^*. Note that $\boldsymbol{z} \in \mathcal{F} \cap S^*$ and the curve $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{x}) \rangle = \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle$ coincides with the hyperplane $\langle \boldsymbol{c}, \boldsymbol{x} \rangle = \langle \boldsymbol{c}, \boldsymbol{z} \rangle$ when the disutility functions are linear.

Proof. Consider any $\boldsymbol{x} \in \mathcal{F}$. We have

$$\begin{aligned} \sum_{i \in [n], j \in [m]} x_{ij} \cdot p_j &= \sum_{j \in [m]} p_j \cdot \sum_{i \in [n]} x_{ij} \\ &= \sum_{j \in [m]} p_j && (\sum_{i \in [n]} x_{ij} = 1 \text{ as } \boldsymbol{x} \in \mathcal{F}) \\ &= b && \text{(by Claim 6.1)} \end{aligned}$$

Therefore $\boldsymbol{x} \in H_{\boldsymbol{p}}$. ∎

Now, we are ready to show that $\gamma^{-2} \cdot \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle \leq \langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle$. Assume otherwise and say we have $\gamma^{-2} \cdot \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle > \langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle$. Then we could replace the allocation as follows: Construct $\hat{\boldsymbol{z}}$ by setting $\hat{\boldsymbol{z}}_{i'} = \frac{1}{2} \boldsymbol{z}_{i'}$, and $\hat{\boldsymbol{z}}_i = \left(1 + \frac{\langle \boldsymbol{z}_{i'}, \boldsymbol{p} \rangle}{2 \langle \boldsymbol{z}_i, \boldsymbol{p} \rangle}\right) \boldsymbol{z}_i$. Since $\boldsymbol{z} \in \mathcal{F}$, we have $\sum_{i \in [n], j \in [m]} p_j z_{ij} = b$. Also note that

$$b = \textstyle\sum_{i \in [n], j \in [m]} p_j z_{ij} = \sum_{i \in [n], j \in [m]} p_j \hat{z}_{ij} \,,$$

since the payment subtracted from agent i' is equal to the payment added to agent i and so $\hat{\boldsymbol{z}} \in H_p$.

By Claim 6.2, we have that $\hat{\boldsymbol{z}} \in H_{\boldsymbol{c}}$. Recall that $H_{\boldsymbol{c}}$ is a separating half-space between S^* and $\mathcal{F}$, *i.e.*, $\mathcal{F} \subseteq H_{\boldsymbol{c}}$ and $S^* \subseteq \text{cl}(H_{\boldsymbol{c}}^{\complement})$, implying that for every point $\boldsymbol{x} \in H_{\boldsymbol{c}}$ we have $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{x}) \rangle \geq \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}) \rangle$. Since $\hat{\boldsymbol{z}} \in H_{\boldsymbol{c}}$, we have

$\langle \boldsymbol{a}, \overrightarrow{D}(\hat{\boldsymbol{z}})\rangle \geq \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z})\rangle$. However,

$$\begin{aligned}\langle \boldsymbol{a}, \overrightarrow{D}(\hat{\boldsymbol{z}})\rangle - \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z})\rangle &= -\frac{1}{2}a_{i'}D_{i'}(\boldsymbol{z}_{i'}) + \frac{\langle \boldsymbol{z}_{i'}, \boldsymbol{p}\rangle}{2\langle \boldsymbol{z}_i, \boldsymbol{p}\rangle}a_i D_i(\boldsymbol{z}_i)\\ &\leq -\frac{1}{2}\gamma^{-1} + \frac{\langle \boldsymbol{z}_{i'}, \boldsymbol{p}\rangle}{2\langle \boldsymbol{z}_i, \boldsymbol{p}\rangle}\cdot\gamma\\ &< -\tfrac{1}{2}\gamma^{-1} + \tfrac{1}{2}\gamma^{-1} = 0\,, \qquad \text{(as } \langle \boldsymbol{z}_{i'}, \boldsymbol{p}\rangle/\langle \boldsymbol{z}_i, \boldsymbol{p}\rangle < \gamma^{-2}\text{)}\end{aligned}$$

which is a contradiction. The first inequality is due to the definition of γ-approximate KKT, which dictates that $\gamma^{-1} \leq a_\ell \cdot D_\ell(\boldsymbol{z}_\ell) \leq \gamma$ for all $\ell \in [n]$.

Satisfying Condition (2) in Definition 2.1 Exactly (i.e., Condition 2). We want to show that for all $i \in [n]$, we have $D_i(\boldsymbol{z}_i) \leq D_i(\boldsymbol{y})$ for all y such that $\langle \boldsymbol{y}, \boldsymbol{p}\rangle \geq \langle \boldsymbol{z}_i, \boldsymbol{p}\rangle$. Let us assume that there exists a $\boldsymbol{y}$ such that $D_i(\boldsymbol{z}_i) > D_i(\boldsymbol{y})$ and $\langle \boldsymbol{y}, \boldsymbol{p}\rangle \geq \langle \boldsymbol{z}_i, \boldsymbol{p}\rangle$. We define a new allocation $\boldsymbol{z}' = (\boldsymbol{z}_1, \boldsymbol{z}_2, \ldots, \boldsymbol{z}_{i-1}, \boldsymbol{y}, \boldsymbol{z}_{i+1}, \ldots, \boldsymbol{z}_n)$. First note that $\sum_{i\in[n],j\in[m]} z'_{ij}\cdot p_j \geq \sum_{i\in[n],j\in[m]} z_{ij}\cdot p_j = b$ as $\langle \boldsymbol{y}, \boldsymbol{p}\rangle \geq \langle \boldsymbol{z}_i, \boldsymbol{p}\rangle$. Therefore $\boldsymbol{z}' \in H_{\boldsymbol{p}}$. By Claim 6.2, we have that $\boldsymbol{z}' \in H_{\boldsymbol{c}}$. Recall that $H_{\boldsymbol{c}}$ is a separating half-space between S^* and $\mathcal{F}$, *i.e.*, $\mathcal{F} \subseteq H_{\boldsymbol{c}}$ and $S^* \subseteq \mathrm{cl}(H^{\complement}_{\boldsymbol{c}})$, implying that for every point $\boldsymbol{x} \in H_{\boldsymbol{c}}$ we have $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{x})\rangle \geq \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z})\rangle$. Since $\boldsymbol{z}' \in H_{\boldsymbol{c}}$, we have $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}')\rangle \geq \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z})\rangle$. However, since $D_i(\boldsymbol{y}) < D_i(\boldsymbol{z}_i)$ and $a_i \geq \gamma^{-1}/D_i(\boldsymbol{z}_i) > 0$ (by the definition of approximate KKT point), we have that $\langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z}')\rangle < \langle \boldsymbol{a}, \overrightarrow{D}(\boldsymbol{z})\rangle$, which is a contradiction.

Satisfying Condition (3) in Definition 2.1 Exactly (i.e., Condition 2). Since $\boldsymbol{z} \in \mathcal{F}$, we have that $\sum_{i\in[n]} z_{ij} = 1$ for all $i \in [n]$. ∎

This concludes the proof that an approximate-CEEI can be determined from approximate-KKT points in polynomial time. In the next subsection, we outline a polynomial time algorithm that determines an approximate-KKT point.

6.2 Algorithm, and Convergence Guarantees We show that approximate-KKT points can be found in polynomial time. We begin with an overview of the procedure, and later show how the steps are implemented. The idea is to perform an exterior-point procedure outside of the feasible region, which produces a sequence of guesses for approximate KKT points, while increasing along the objective. Due to the nature of the objective function, we alternate between finding supporting hyperplanes, and finding NSW-maximizing points on these hyperplanes, until we find a point whose gradient is approximately in line with the supporting hyperplane.

To be precise, our algorithm starts from a point $\boldsymbol{d}^0$ very close to $\mathbf{0}$. Note that this point lies below $\mathcal{D}$. Then, we find the nearest point $\boldsymbol{d}^0_*$ in $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ to $\boldsymbol{d}^0$. We will address how to find this nearest point, and explain how to robustly handle approximation errors in finding this nearest point. In doing so, it will be helpful to find nearest points in the convex region $\mathcal{D} + \mathbb{R}^n_{\geq 0}$, but keeping in mind that the true optimum $\boldsymbol{d}^0_*$ lies in $\mathcal{D}$: to see this, note that $\boldsymbol{d}^0_*$ has to lie on the lower envelope of $\mathcal{D}$, and since it is the closest point in $\mathcal{D}$ to $\boldsymbol{d}^0$, it follows that $(\boldsymbol{d}^0_* - \boldsymbol{d}^0)$ is normal to a supporting hyperplane of $\mathcal{D}$ at $\boldsymbol{d}^0_*$. Furthermore, we show that $\boldsymbol{d}^0_*$ Pareto-dominates $\boldsymbol{d}^0$, thereby implying that the Nash welfare at $\boldsymbol{d}^0_*$ is larger than the Nash welfare at $\boldsymbol{d}^0_*$.

Let $\langle \boldsymbol{a}, \boldsymbol{y}\rangle = n$ be the supporting hyperplane of $\mathcal{D}$ at $\boldsymbol{d}^0_*$, where $\boldsymbol{a} \propto (\boldsymbol{d}^0_* - \boldsymbol{d}^0)$. Let $\boldsymbol{d}^1$ be a point on this hyperplane with maximum Nash welfare. Observe that at $\boldsymbol{d}^1$, we should have $\nabla\mathcal{L}$ proportional to $\boldsymbol{a}$, *i.e.*, $\boldsymbol{a} = \mathbf{1}/\boldsymbol{d}^1$, implying that $\boldsymbol{d}^1 = \mathbf{1}/\boldsymbol{a}$. Since $\langle \boldsymbol{a}, \boldsymbol{y}\rangle = n$ is a supporting hyperplane of $\mathcal{D}$ at $\boldsymbol{d}^0_*$ (a point on the lower envelope of $\mathcal{D}$), we have that $\boldsymbol{d}^1$ also lies below the lower envelop of $\mathcal{D}$. We prove that if the distance between $\boldsymbol{d}^0_*$ and $\boldsymbol{d}^1$ is small, then $\boldsymbol{d}^0_*$ is our approximate KKT-point, otherwise we have a new point $\boldsymbol{d}^1$ below $\mathcal{D}$, which has significantly higher Nash welfare than $\boldsymbol{d}^0$. We run the exact same steps from $\boldsymbol{d}^1$. We argue that such a procedure should eventually give us an approximate KKT point as there is significant increase in Nash welfare with every iteration of the algorithm whenever no approximate KKT point is found. The full description of the algorithm is given in Algorithm 1.

In what follows, define $\mathrm{RelDist}(\boldsymbol{x}, \boldsymbol{y}) := \sum_i |\log(x_i/y_i)|$. Notice that if $\mathrm{RelDist}(\boldsymbol{x}, \boldsymbol{y}) \leq \varepsilon$, then $(1+\varepsilon)^{-1} \leq x_i/y_i \leq (1+\varepsilon)$ for all i, since $\log(1+a) \leq a$ for all $a > -1$. We will find a point which is a $(1+\varepsilon)$-approximate KKT point following Algorithm 1.

Correctness. We begin by proving here that the algorithm truly returns an approximate KKT point and we will later show that (i) it will terminate in polynomially many iterations, (ii) each iteration can be implemented in polynomial time. To this end, we will need the following technical results, about the steps of the algorithm.

LEMMA 6.1. *Regardless of the geometry of $\mathcal{D}$, so long as $\mathcal{D} + \mathbb{R}^n_{\geq 0}$ is convex, we have that for each iteration $k \geq 0$ of Algorithm 1:*

Algorithm 1 Finding Approximate KKT

1: Let $\boldsymbol{d}^0$ be any infeasible, strictly positive, disutility profile, near $\mathbf{0}$
2: **while true do**
3: Set $\boldsymbol{d}^k_*$ to be the nearest dominating point in $\mathcal{D}$ to $\boldsymbol{d}^k$, *i.e.*

$$\arg\min \left\{ \|\boldsymbol{y} - \boldsymbol{d}^k\|_2^2 \;\middle|\; \boldsymbol{y} \in \mathcal{D} + \mathbb{R}^n_{\geq 0},\, \boldsymbol{y} \geq \boldsymbol{d}^k \right\}$$

4: **Set** $\boldsymbol{a}^k \leftarrow (\boldsymbol{d}^k_* - \boldsymbol{d}^k)$, the direction from $\boldsymbol{d}^k$ to $\mathcal{D}$
5: **Rescale** $\boldsymbol{a}^k$ so that $\langle \boldsymbol{a}^k, \boldsymbol{d}^k_* \rangle = n$
6: **Set** $\boldsymbol{d}^{k+1} \leftarrow \mathbf{1}/\boldsymbol{a}^k$
7: **if** $\mathrm{RelDist}(\boldsymbol{d}^{k+1}, \boldsymbol{d}^k_*) < \varepsilon$ **then**
8: **Return** $(\boldsymbol{d}^k_*, \boldsymbol{a}^k)$

1. *The hyperplane defined as* $\{\boldsymbol{y} \in \mathbb{R}^n | \langle \boldsymbol{a}^k, \boldsymbol{y} \rangle \geq \langle \boldsymbol{a}^k, \boldsymbol{d}^k_* \rangle\}$ *is supporting for* $\mathcal{D} + \mathbb{R}^n_{\geq 0}$*, at* $\boldsymbol{d}^k_*$.
2. *If* $\boldsymbol{d}^k$ *has strictly positive entries and does not lie in* $\mathcal{D} + \mathbb{R}^n_{\geq 0}$*, then* $\boldsymbol{d}^k_*$*,* $\boldsymbol{a}^k$*, and* $\boldsymbol{d}^{k+1}$ *have strictly positive entries, and* $\boldsymbol{d}^k_* \in \mathcal{D}$.

We skip the formal proof of this lemma as it is mostly technical and refer the reader to the full version of the paper [BCM21]. Informally, these hold due to the geometry of the feasible region, and ensure that each iterate is well-defined, and economically meaningful. To complete the proof of correctness, we show that we can efficiently find a starting point $\boldsymbol{d}^0$ which is strictly positive in every entry, and is infeasible. Thus, Lemma 6.1 will inductively show that every point is positive and well-defined.

CLAIM 6.4. *The point* $\boldsymbol{d}^0 = \frac{m\delta}{2n}\mathbf{1}$ *where* $\delta = \min_{ij} D_{ij}$ *is a strictly positive infeasible disutility profile.*

Proof. Since $D_{ij} \geq \delta$ for all i and j, any feasible dis-utility profile must assign disutility at least $m\delta/n$ to some agent. Therefore, it is impossible for every agent to have disutility $\frac{1}{2}m\delta/n$ at a feasible point. ∎

We now show that in the stopping condition, Algorithm 1 returns an approximate KKT point. Intuitively, this holds because the RelDist function in the stopping condition is designed to correctly captures the multiplicative error needed in the definition of approximate KKT.

LEMMA 6.2. *Algorithm 1 returns a* $(1+\varepsilon)$*-KKT point for minimizing* $\mathcal{L}$ *on* $\mathcal{D}$.

Proof. Suppose the algorithm terminates and returns $(\boldsymbol{d}^k_*, \boldsymbol{a}^k)$ on line 8. Note that we have $\boldsymbol{d}^{k+1} = \mathbf{1}/\boldsymbol{a}^k$ and $\mathrm{RelDist}(\boldsymbol{d}^{k+1}, \boldsymbol{d}^k_*) < \varepsilon$, implying that $\mathrm{RelDist}(\mathbf{1}/\boldsymbol{a}^k, \boldsymbol{d}^k_*) < \varepsilon$. Then, we have $(1+\varepsilon)^{-1} \leq a^k_i \cdot (d^k_*)_i \leq 1 + \varepsilon$ for all i. Also by Lemma 6.1, we have that $\langle \boldsymbol{a}^k, \boldsymbol{y} \rangle = n$ is a supporting hyperplane of $\mathcal{D}$ passing through $\boldsymbol{d}^k_*$. Therefore, the point $\boldsymbol{d}^k_*$ is a $(1+\varepsilon)$-KKT point as in Definition 3.1. ∎

In the rest of this section, we will argue that the number of iterations must be polynomial, and that each iteration can be solved in polynomial time, which will allow us to conclude the correctness and efficiency of the algorithm.

Polynomially Many Iterations. We show that in polynomially many iterations the algorithm finds an approximate KKT point. In particular, we show that (a) the log-NSW $\mathcal{L}$ is always increasing throughout Algorithm 1, and (b) it increases additively by $\mathrm{poly}(n, 1/\varepsilon)$ every time $\mathrm{RelDist}(\boldsymbol{d}^{k+1}, \boldsymbol{d}^k_*) \geq \varepsilon$. Bounding the range of $\mathcal{L}$ over the course of the iteration will then give our desired bound.

CLAIM 6.5. *Steps 3. and 6. always increase* $\mathcal{L}$*, the log-product of disutilities. Formally,* $\mathcal{L}(\boldsymbol{d}^{k+1}) \geq \mathcal{L}(\boldsymbol{d}^k_*) \geq \mathcal{L}(\boldsymbol{d}^k)$ *for all* $k \geq 0$.

Proof. By Lemma 6.1, $\boldsymbol{d}^k_* \geq \boldsymbol{d}^k$, coordinate-wise. Thus, since $\mathcal{L}$ is monotone increasing in each coordinate direction, $\mathcal{L}(\boldsymbol{d}^k_*) \geq \mathcal{L}(\boldsymbol{d}^k)$.

We prove that Step 6 is an improvement by showing that $\boldsymbol{d}^{k+1}$ is the maximizing point on the hyperplane $\langle \boldsymbol{a}^k, \boldsymbol{y} \rangle = n$, and therefore $\mathcal{L}(\boldsymbol{d}^{k+1}) \geq \mathcal{L}(\boldsymbol{d}^k_*)$.

Since $\mathcal{L}$ is a concave function, it is maximized on this hyperplane when $\nabla\mathcal{L}$ is proportional to $\boldsymbol{a}^k$, *i.e.* when $a^k_i = c/d_i$ for some $c > 0$, for all i. Since we need $\langle \boldsymbol{a}^k, \boldsymbol{d} \rangle = n$, it suffices to set $c = 1$. Thus, $\boldsymbol{d}^{k+1}$ is the $\mathcal{L}$-maximizing point on the supporting hyperplane which contains $\boldsymbol{d}^k_*$, and so this move is an $\mathcal{L}$-improvement. ∎

Using the above claims, next we show that $\mathcal{L}$ increases significantly in each iteration of our algorithm.

LEMMA 6.3. *If Algorithm 1 does not return at step 8, then the logarithm of the Nash social welfare increases by at least* $\frac{1}{16}(\varepsilon/n)^2$*, i.e.,* $\mathcal{L}(\boldsymbol{d}^{k+1}) - \mathcal{L}(\boldsymbol{d}^k) \geq \frac{1}{16}(\varepsilon/n)^2$.

Proof. Since $\mathcal{L}(\boldsymbol{d}_*^k) > \mathcal{L}(\boldsymbol{d}^k)$ by Claim 6.5, it suffices to show that if $\mathrm{RelDist}(\boldsymbol{d}^{k+1}, \boldsymbol{d}_*^k) > \varepsilon$, then $\mathcal{L}(\boldsymbol{d}^{k+1}) - \mathcal{L}(\boldsymbol{d}_*^k)$ is large. Let $A = \mathrm{diag}(\boldsymbol{a}^k)$, and note that $\langle \mathbf{1}, A\boldsymbol{d}\rangle = \langle \boldsymbol{a}^k, \boldsymbol{d}\rangle$, and furthermore, $A\boldsymbol{d}^{k+1} = \mathbf{1}$. Let $\boldsymbol{\Delta} = A\boldsymbol{d}_*^k - \mathbf{1}$, and notice that

$$\langle \mathbf{1}, \boldsymbol{\Delta}\rangle = \langle \mathbf{1}, A(\boldsymbol{d}_*^k - \boldsymbol{d}^{k+1})\rangle = 0$$

Note that $\boldsymbol{d}_*^k = (\mathbf{1} + \boldsymbol{\Delta})/\boldsymbol{a}^k$, where we take the quotient componentwise as is defined at the start of Section 6.1. With $\boldsymbol{d}^{k+1} = \mathbf{1}/\boldsymbol{a}^k$, this gives $\mathrm{RelDist}(\boldsymbol{d}_*^k, \boldsymbol{d}^{k+1}) = \mathrm{RelDist}((\mathbf{1} + \boldsymbol{\Delta}), \mathbf{1})$. Therefore, we know that $\sum_{i=1}^n |\log(1 + \Delta_i)| > \varepsilon$. We also get

$$\mathcal{L}(\boldsymbol{d}^{k+1}) - \mathcal{L}(\boldsymbol{d}_*^k) = \sum_{i=1}^n \log(1/a_i^k) - \log((1 + \Delta_i)/a_i^k) = -\sum_{i=1}^n \log(1 + \Delta_i)$$

Define:

$$F(z) := \begin{cases} \frac{1}{4}z^2 & \text{if } -1 < z \leq 1 \\ \frac{1}{2}z - \frac{1}{4} & \text{if } z \geq 1 \\ +\infty & \text{otherwise} \end{cases}$$

At $z = 0$, we have that $-z + F(z) = 0 = \log(1 + z)$ and $\frac{\mathrm{d}}{\mathrm{d}z}(-z + F(z)) = -1 = \frac{\mathrm{d}}{\mathrm{d}z}(-\log(1 + z))$. By comparing derivatives for the other values of $z > -1$, we can show that $-\log(1 + z) \geq -z + F(z)$ for all z. Thus,

$$\mathcal{L}(\boldsymbol{d}^{k+1}) - \mathcal{L}(\boldsymbol{d}_*^k) \;=\; -\sum_{i=1}^n \log(1 + \Delta_i) \;\geq\; \sum_{i=1}^n -\Delta_i + \sum_{i=1}^n F(\Delta_i) \;=\; \sum_{i=1}^n F(\Delta_i)$$

Now, since we have $\sum_{i=1}^n |\log(1 + \Delta_i)| > \varepsilon$, there must be some i such that $|\log(1 + \Delta_i)| > \varepsilon/n$. If $\Delta_i > 0$, then $\Delta_i \geq \log(1 + \Delta_i) \geq \varepsilon/n$. Conversely, if $\Delta_i < 0$, we being by noting that for $|z| < 0.5$, we have $-\log(1 + z) \leq -z + z^2$ for reasons similar to the above. Thus, we get

$$\varepsilon/n < -\log(1 + \Delta_i) \leq -\Delta_i + \Delta_i^2$$

We must have $\Delta_i > -1$, since the argument can't be negative, so we have $2|\Delta_i| > \Delta_i^2 - \Delta_i > \varepsilon/n$, or $\Delta_i < -\frac{1}{2}\varepsilon/n$. Noting that $F(z) \geq 0$ for all z, we can then conclude

$$\mathcal{L}(\boldsymbol{d}^{k+1}) - \mathcal{L}(\boldsymbol{d}_*^k) \geq \textstyle\sum_i F(\Delta_i) \geq \max_i F(\Delta_i) \geq \frac{1}{16}\varepsilon^2/n^2$$

as desired. ∎

Finally, to bound the number of iterations Algorithm 1 would take we need to bound the log-NSW value at the starting point, namely $\mathcal{L}(\boldsymbol{d}^0)$, where $\boldsymbol{d}^0 := \mathbf{1} \cdot \frac{m}{2n} \min_{i,j} D_{ij}$, as in Claim 6.4. We show the following.

LEMMA 6.4. *Starting at* $\boldsymbol{d}^0 := \mathbf{1} \cdot \frac{m}{2n} \min_{i,j} D_{ij}$*, Algorithm 1 finds a* $(1 + \varepsilon)$*-KKT point in*

$$O\left(\frac{n^3}{\varepsilon^2} \cdot \log\left(\frac{n \cdot \max_{i,j} D_{ij}}{\min_{i,j} D_{ij}}\right)\right)$$

many iterations.

Proof. If we can bound the range of the log-NSW objective, then the proof follows using Lemmas 6.2 and 6.3. Let M be such that $D_i(\boldsymbol{x}_i) \leq M$ for every agent i, at every feasible $\boldsymbol{x} \in \mathcal{F}$. Note that $M \leq m \cdot \max_{i,j} D_{ij}$.

Then we have that for any feasible $\boldsymbol{x}$, $\mathcal{L}(\boldsymbol{d}(\boldsymbol{x})) \leq n \log M$. Since each round of the above algorithm that doesn't terminate increases the log-NSW by at least $\frac{1}{16}(\varepsilon/n)^2$, then the total number of rounds possible is at most

$$16 \cdot \frac{n^2}{\varepsilon^2} \cdot (n \log(M) - \mathcal{L}(\boldsymbol{d}^0)) \leq \frac{16n^3}{\varepsilon^2} \cdot \left(\log(m \cdot \max_{i,j} D_{ij}) - \log(\tfrac{m}{2n} \min_{i,j} D_{ij})\right) ,$$

which gives the desired bound. ∎

Now that we have shown there are polynomially many iterations in our algorithm, it suffices to show that each iteration can be implemented in polynomial time to establish that Algorithm 1 is indeed polynomial time.

Implementing Each Iteration in Polynomial Time. To show that each iteration can be implemented in polynomial time, it suffices to show that the *nearest neighbour search* (step 3 in Algoritm 1) can be implemented in polynomial time.

LEMMA 6.5. *Each iteration of Algorithm 1 can be computed exactly in time polynomial in n, m, and the description complexity of the D_{ij}'s.*

Proof. Let $\overrightarrow{D}(\boldsymbol{x}) := (D_1(\boldsymbol{x}_1), \ldots, D_n(\boldsymbol{x}_n))$ as defined previously. Recall that disutility functions are linear, with $D_i(\boldsymbol{x}_i) := \sum_{j=1}^{m} D_{ij} x_{ij}$.

Let $\widehat{D}$ be the $n \times nm$ block-diagonal matrix such that $\widehat{D}\boldsymbol{x} = \overrightarrow{D}(\boldsymbol{x})$. To find the nearest-feasible disutility profiles, we will find the allocation $\boldsymbol{x}$ which minimizes the following convex quadratic program:

$$\min_{\boldsymbol{x} \in \mathcal{F}} \left\| \overrightarrow{D}(\boldsymbol{x}) - \boldsymbol{d}^k \right\|_2^2 = \min_{\boldsymbol{x} \in \mathcal{F}} \boldsymbol{x}^\top \left(\widehat{D}^\top \widehat{D} \right) \boldsymbol{x} - 2(\boldsymbol{d}^k)^\top \widehat{D}\boldsymbol{x} + (\boldsymbol{d}^k)^\top \boldsymbol{d}^k \, .$$

It was shown by Khachiyan et al. [KTK80] that this program can be solved exactly, with running time polynomial in the description complexity of the system. Thus, so long as $\widehat{D}$ and $\boldsymbol{d}^k$ have rational entries with polynomial description complexity (polynomial-sized numerators and denominators), the problem can be solved exactly in polynomial time, and the solution will have small description complexity.

The matrix $\widehat{D}$ consists of the D_{ij}'s and our running time is assumed to depend on their description complexity. ∎

Final Result. We now have all the ingredients to conclude that an approximate CEEI (Definition 2.1) can be computed in polynomial time. Lemma 6.4 bounds the number of iterations as a polynomial in n, $1/\varepsilon$, and the description complexity of the instance, Claim 6.4 shows how to find a good starting point, Lemma 6.5 shows that each iteration can be computed in polynomial time, with the same arguments, and Theorem 6.1 shows how to compute a ε-CEEI in polynomial time given the output of Algorithm 1. Thus, we conclude that Algorithm 1 is an FPTAS for finding 3ε-CEEI.

THEOREM 6.2. *Given linear disutility values $D_{11}, \ldots, D_{nm}$, Algorithm 1, along with 6.1, finds an ε-CEEI in time polynomial in n, m, $1/\varepsilon$, $\log(\frac{\max D_{ij}}{\min D_{ij}})$, and the description complexity of the D_{ij}'s.*

References

[AD54] Kenneth J Arrow and Gerard Debreu. Existence of an equilibrium for a competitive economy. *Econometrica: Journal of the Econometric Society*, pages 265–290, 1954.

[BCM21] Shant Boodaghians, Bhaskar Ray Chaudhury, and Ruta Mehta. Polynomial time algorithms to find an approximate competitive equilibrium for chores. *CoRR*, abs/2107.06649, 2021.

[BMSY17] Anna Bogomolnaia, Hervé Moulin, Fedor Sandomirskiy, and Elena Yanovskaia. Competitive division of a mixed manna. *Econometrica*, 85(6):1847–1871, 2017.

[BMSY19] Anna Bogomolnaia, Hervé Moulin, Fedor Sandomirskiy, and Elena Yanovskaia. Dividing bads under additive utilities. *Social Choice and Welfare*, 52(3):395–417, 2019.

[BS19] Simina Branzei and Fedor Sandomirskiy. Algorithms for competitive division of chores. arXiv:1907.01766 (To appear in Mathematics of Operations Research), 2019.

[CCD13] Yun Kuen Cheung, Richard Cole, and Nikhil Devanur. Tatonnement beyond gross substitutes? Gradient descent to the rescue. In *Proc. 45th Symp. Theory of Computing (STOC)*, pages 191–200, 2013.

[CDDT09] Xi Chen, Decheng Dai, Ye Du, and Shang-Hua Teng. Settling the complexity of Arrow-Debreu equilibria in markets with additively separable utilities. In *Proc. 50th Symp. Foundations of Computer Science (FOCS)*, pages 273–282, 2009.

[CDG+17] Richard Cole, Nikhil Devanur, Vasilis Gkatzelis, Kamal Jain, Tung Mai, Vijay Vazirani, and Sadra Yazdanbod. Convex program duality, Fisher markets, and Nash social welfare. In *Proc. 18th Conf. Economics and Computation (EC)*, 2017.

[CGMM20] Bhaskar Ray Chaudhury, Jugal Garg, Peter McGlaughlin, and Ruta Mehta. Dividing bads is harder than dividing goods: On the complexity of fair and efficient division of chores. *arXiv preprint arXiv:2008.00285*, 2020.

[CGMM21] Bhaskar Ray Chaudhury, Jugal Garg, Peter McGlaughlin, and Ruta Mehta. Competitive allocation of a mixed manna. In *Proc. 32nd Symp. Discrete Algorithms (SODA)*, 2021.

[CPY17] Xi Chen, Dimitris Paparas, and Mihalis Yannakakis. The complexity of non-monotone markets. *Journal of the ACM (JACM)*, 64(3):1–56, 2017.

[CT09] Xi Chen and Shang-Hua Teng. Spending is not easier than trading: on the computational equivalence of fisher and arrow-debreu equilibria. In *International Symposium on Algorithms and Computation*, pages 647–656. Springer, 2009.

[DGM16] Ran Duan, Jugal Garg, and Kurt Mehlhorn. An improved combinatorial polynomial algorithm for the linear Arrow-Debreu market. In *Proc. 27th Symp. Discrete Algorithms (SODA)*, pages 90–106, 2016.

[DGV16] Nikhil Devanur, Jugal Garg, and László Végh. A rational convex program for linear Arrow-Debreu markets. *ACM Trans. Econom. Comput.*, 5(1):6:1–6:13, 2016.

[DK08] Nikhil Devanur and Ravi Kannan. Market equilibria in polynomial time for fixed number of goods or agents. In *Proc. 49th Symp. Foundations of Computer Science (FOCS)*, pages 45–53, 2008.

[DM15] Ran Duan and Kurt Mehlhorn. A combinatorial polynomial algorithm for the linear Arrow-Debreu market. *Inf. Comput.*, 243:112–132, 2015.

[DPSV08] Nikhil Devanur, Christos Papadimitriou, Amin Saberi, and Vijay Vazirani. Market equilibrium via a primal–dual algorithm for a convex program. *J. ACM*, 55(5), 2008.

[EG59] Edmund Eisenberg and David Gale. Consensus of subjective probabilities: The pari-mutuel method. *The Annals of Mathematical Statistics*, 30(1):165–168, 1959.

[FGL16] Michal Feldman, Nick Gravin, and Brendan Lucier. Combinatorial walrasian equilibrium. *SIAM J. Comput.*, 45(1):29–48, 2016.

[GM20] Jugal Garg and Peter McGlaughlin. Computing competitive equilibria with mixed manna. In *Proceedings of the 19th International Conference on Autonomous Agents and Multiagent Systems, AAMAS '20, Auckland, New Zealand, May 9-13, 2020*, pages 420–428, 2020.

[GMSV15] Jugal Garg, Ruta Mehta, Milind Sohoni, and Vijay V. Vazirani. A complementary pivot algorithm for market equilibrium under separable, piecewise-linear concave utilities. *SIAM J. Comput.*, 44(6):1820–1847, 2015. Extended abstract appeared in STOC 2012.

[GV19] Jugal Garg and László A Végh. A strongly polynomial algorithm for linear exchange markets. In *Proc. 51st Symp. Theory of Computing (STOC)*, 2019.

[KTK80] Mikhail K Kozlov, Sergei P Tarasov, and Leonid G Khachiyan. The polynomial solvability of convex quadratic programming. *USSR Computational Mathematics and Mathematical Physics*, 20(5):223–228, 1980.

[McK54] Lionel McKenzie. On equilibrium in graham's model of world trade and other competitive systems. *Econometrica*, 22(2):147–161, 1954.

[McK59] Lionel W. McKenzie. On the existence of general equilibrium for a competitive market. *Econometrica*, 27(1):54–71, 1959.

[MV07] Nimrod Megiddo and Vijay V. Vazirani. Continuity properties of equilibrium prices and allocations in linear fisher markets. In *WINE*, volume 4858 of *Lecture Notes in Computer Science*, pages 362–367. Springer, 2007.

[NP83] E I Nenakov and M E Primak. One algorithm for finding solutions of the Arrow-Debreu model. *Kibernetica*, 3:127–128, 1983.

[Orl10] James Orlin. Improved algorithms for computing Fisher's market clearing prices. In *Proc. 42nd Symp. Theory of Computing (STOC)*, pages 291–300, 2010.

[Rub18] Aviad Rubinstein. Inapproximability of nash equilibrium. *SIAM J. Comput.*, 47(3):917–959, 2018.

[Var74] Hal Varian. Equity, envy and efficiency. *J. Econom. Theory*, 29(2):217–244, 1974.

[Vég12] László Végh. Strongly polynomial algorithm for a class of minimum-cost flow problems with separable convex objectives. In *Proc. 44th Symp. Theory of Computing (STOC)*, pages 27–40, 2012.

[VY11] Vijay Vazirani and Mihalis Yannakakis. Market equilibrium under separable, piecewise-linear, concave utilities. *J. ACM*, 58(3):10, 2011.

[Wal74] Léon Walras. *Éléments d'économie politique pure, ou théorie de la richesse sociale (Elements of Pure Economics, or the theory of social wealth)*. English version, Cambridge University Press, Lausanne, Paris, 1874. (1899, 4th ed.; 1926, rev ed., 1954, Engl. transl.).