Auction Algorithms for Market Equilibrium with Weak Gross Substitute Demands and Their Applications

Jugal Garg
University of Illinois at Urbana-Champaign, IL, USA

Edin Husić
Department of Mathematics, The London School of Economics and Political Science, UK

László A. Végh
Department of Mathematics, The London School of Economics and Political Science, UK

Abstract

We consider the Arrow–Debreu exchange market model where agents' demands satisfy the weak gross substitutes (WGS) property. This is a well-studied property, in particular, it gives a sufficient condition for the convergence of the classical tâtonnement dynamics. In this paper, we present a simple auction algorithm that obtains an approximate market equilibrium for WGS demands. Such auction algorithms have been previously known for restricted classes of WGS demands only. As an application of our technique, we obtain an efficient algorithm to find an approximate spending-restricted market equilibrium for WGS demands, a model that has been recently introduced as a continuous relaxation of the Nash social welfare (NSW) problem. This leads to a polynomial-time constant factor approximation algorithm for NSW with budget separable piecewise linear utility functions; only a pseudopolynomial approximation algorithm was known for this setting previously.

2012 ACM Subject Classification Theory of computation → Design and analysis of algorithms; Theory of computation → Algorithmic game theory and mechanism design

Keywords and phrases auction algorithm, weak gross substitutes, Fisher equilibrium, Gale equilibrium, Nash social welfare

Digital Object Identifier 10.4230/LIPIcs.STACS.2021.33

Related Version *Full Version*: https://arxiv.org/abs/1908.07948

Funding Jugal Garg was supported by the NSF CAREER Award 1942321. Edin Husić and László A. Végh were supported by the European Research Council (ERC) under the European Union's Horizon 2020 research and innovation programme (grant agreement ScaleOpt–757481).

Acknowledgements We would like to thank anonymous referees for their comments and suggestions that have helped to improve the presentation of the paper.

1 Introduction

Market equilibrium is a fundamental and well-established notion to analyze and predict the outcomes of strategic interaction in large markets. In the classic Arrow-Debreu exchange model, a set of agents arrive at the market with initial endowments of divisible goods. A market equilibrium comprises a set of prices and allocations of goods to the agents such that each agent spends their income from selling their initial endowment on a bundle that maximizes their utility, and the market clears: demand of each good meets its supply. This model was first studied by Walras in 1874 [61], who also introduced a natural market dynamics, called the *tâtonnement* process. A continuous version of the process was shown to converge to an equilibrium if the utility functions satisfy the *weak gross substitutability (WGS)* property, namely, that if the prices of some goods increase and the others remain unchanged, then the demand for the latter goods may not decrease (see Arrow, Block, and Hurwitz [3], Arrow and Hurwitz [6], and references therein). However, Scarf [59] showed, using an example of Leontief utilities, that tâtonnement may not always converge to an equilibrium. We refer the reader to [54, Chapter 17] on the stability of the tâtonnement process.

The polynomial-time computability of market equilibrium for WGS utilities was first established by Codenotti, Pemmaraju, and Varadarajan [25]. Later, a simple ascending-price algorithm using *global demand queries* was given by Bei, Garg, and Hoefer [9]. Further, Codenotti, McCune, and Varadarajan [23] have shown that a simple discrete variant of the tâtonnement algorithm converges to an approximate equilibrium (see also [57, Section 6.3]). This was followed by a number of papers providing tâtonnement algorithms for various classes of utility functions and restricted models, some of them substantially weakening the need for central coordination among agents, see e.g., [7, 19, 20, 27, 37].

However, most of these algorithms still rely on global demand queries, and hence they are less realistic. In a sense, they require a central authority (responsible for updating prices) to have some general information about the demands of all agents in the market.

Auction algorithms. In this paper, we focus on an even simpler subclass of tâtonnement-type algorithms, called *auction algorithms*. Whereas prices in tâtonnement may increase as well as decrease, in auctions prices may only go up. Auction algorithms are appealing due to their simplicity and distributed nature: under simple "ground rules" the agents outbid each other and in the process converge to an approximate market equilibrium. Unlike the above mentioned works, these algorithms do not require a central authority and need only minimal coordination between the agents. Further, these algorithmic frameworks are quite robust and easily allow for various extensions and generalizations. For exchange market models, the first such algorithm was established for linear utilities by Garg and Kapoor [44] (see also [57, Section 5.12]). The algorithm was later improved [45] and generalized to separable concave gross substitute utility functions [47], to a subclass of non-separable gross-substitutes called *uniformly separable* [46], and to a production model with linear production constraints and linear utilities [50].

There is a long history of auction algorithms both in the optimization and in the economics literature. Bertsekas [11, 12] introduced auction algorithms for assignment and transportation problems. Closely related algorithms were introduced for markets with indivisible goods, by Kelso and Crawford [52], and Demange, Gale, and Sotomayor [30]. We discuss markets with indivisible goods later in this section.

Our contributions. Our first main contribution is an auction algorithm that computes an approximate market equilibrium for arbitrary WGS utilities, given via demand oracles, settling an open question from [46]. This result shows that for WGS utilities, this restricted class of tâtonnement algorithms already suffices to obtain an equilibrium. The result affirms the natural intuition that the WGS property is geared for auction algorithms. A main invariant in auction algorithms is that at every price increase, the agents will still hold on to the goods they have purchased previously at the lower prices. This property is almost identical to the definition of the WGS property; nevertheless, making an auction algorithm work for general WGS utilities requires some careful technical ideas. The previously mentioned auction algorithms operate with two prices for each good, a lower price p_j and a higher price $(1+\epsilon)p_j$. For linear utilities, [44] maintains that all purchases are maximum bang-per-buck goods with respect to the lower or higher price. This idea can be extended to separable [45] and to uniformly separable utilities [47], but does not work if the utilities are genuinely non-separable. For this general case, our main technical idea is to maintain subsets of optimal bundles for each agent with respect to some individual prices. These individual prices can be different for each agent but fall between the higher and lower prices p and $(1+\epsilon)p$.

This results in the first "agent-driven" algorithm for the entire range of WGS utilities that avoids the need of a central authority, where each agent uses only their own black-box oracle `FindNewPrices` (Section 3), which depends only on their own preference to *outbid* another agent on a particular good. The process of outbidding another agent can also be implemented in an uncoordinated manner. Overall, this lessens the level of coordination needed in the market, making it more plausible mechanism in a decentralized environment.

We also study auction algorithms for multiple models of *Fisher markets*. These are a special case of exchange markets where every agent arrives with a fixed budget instead of an endowment of goods. A particular motivation comes from recent study of the *Nash social welfare (NSW)* problem: allocating indivisible goods to agents so that the geometric mean of their utilities is maximized. This problem is NP-hard already for simple classes of utilities, and there has been a considerable recent literature on approximation algorithms for the problem and its extensions. Cole and Gkatzelis [28] gave the first constant-factor approximation for linear utilities, followed by further work with stronger guarantees as well as extensions for other utility classes [1, 2, 8, 17, 26, 28, 38, 40, 41].

The algorithm in [28] and many others start by studying a continuous relaxation corresponding to a specific market equilibrium problem with *spending restrictions*: namely, if the price p_i of good i is above 1, then the amount of good i sold is decreased to $1/p_i$ from the initial total amount of 1. Whereas a market equilibrium with spending restrictions can be obtained via a convex program for linear utilities [26], it becomes challenging to find for more general utilities: currently known cases are budget-additive valuations [38] and separable piecewise-linear concave (SPLC) utility functions [2]. The set of equilibria in the former case turned out to be not even convex.

In this paper, we show that auction algorithms are particularly well-suited for spending restricted equilibrium computation: once the price of a good goes above one, we can naturally decrease the total available amount of these goods within the auction framework. This enables us to find simple approximation algorithms for spending restricted equilibria for a broad class of utility functions, including the models above as well as their common generalization: *budget-SPLC*. A surprising feature here is that we do not even have to make the standard non-satiation assumption. Moreover, our algorithm can be used to obtain a constant-factor approximation for maximizing NSW in polynomial-time when agents have budget-SPLC utilities and goods come in multiple copies. The previous algorithm for this setting in [17] runs in pseudopolynomial time. We expect that our algorithm for finding approximate spending restricted equilibria will find more applications for the NSW and other related problems.

Markets with indivisible goods. Auction algorithms have been widely studied in the context of markets with *indivisible goods*. Equilibria may not always exists in markets with indivisible goods. The class of (discrete) gross substitute utilities was introduced by Kelso and Crawford [52]. For this class, an equilibrium is guaranteed to exist, and an approximate equilibrium can be efficiently found via a simple auction algorithm, extending [29]. It turned out that the discrete gross substitutes property is essentially a necessary and sufficient condition for the auction algorithm to work. We refer the reader to the survey by Paes Leme [53] on the role of gross substitute utilities in markets with indivisible goods, and their connections to discrete convex analysis.

Whereas the definitions of discrete gross substitutes and continuous WGS utilities are very similar, there does not appear to be a direct connection between these notions. The main difference is in the utility concepts: for indivisible markets, the standard model is to maximize the valuation minus the price of the set at given prices, whereas the standard divisible market models operate with *fiat money*: the prices appear via the budget constraints but not in the utility value. Still, our result can be interpreted as the continuous analogue of the strong link between auction algorithms and the gross substitutes property for markets with indivisible goods: we show that auction algorithms are applicable for the entire class of WGS utilities for markets with divisible goods. We suspect that the converse should also be true, namely, that the applicability of auction algorithms should be limited to WGS utilities. In contrast, tâtonnement algorithms have been successfully applied beyond the WGS class, see e.g. [19, 20, 37].

Let us also comment on the oracle model we use. Typically, (continuous) WGS utilities in the literature are given in an explicit form such as CES or Cobb-Douglas utilities. This is in contrast with the discrete WGS setting, where the common model is via a value or demand oracle [53], since direct preference elicitation, that is, the explicit description of the valuation function would be exponential. The class of continuous WGS functions also appears to be very rich and expressive, and hence an oracle approach seems more appropriate to devise algorithms for this class. In our model, the agent preferences are represented via a *demand oracle* (Definition 3).

The auction algorithm relies on the more powerful `FindNewPrices` subroutine, which can be seen as a strengthening of the demand oracle, incorporating a mechanism for price increments. There are various ways to implement such a subroutine: we use a simple iterative application of the demand oracle for the case of bounded price elasticities; we use a convex programming approach for Gale demand systems; and we devise a combinatorial algorithm for budget-additive SPLC utilities.

Further related work. The existence of a market equilibrium is always guaranteed under some mild assumptions, as shown by Arrow and Debreu [4], using Kakutani's fixed point theorem. The computational aspects of finding a market equilibrium have been extensively studied in the theoretical computer science community over the last two decades, establishing hardness results as well as polynomial-time algorithms for certain cases. We refer the reader to [14, 18, 24, 31, 34, 42, 49, 60, 63, 43] for an overview of the literature.

The other famous dynamics to study market equilibrium is *proportional response* where in each round agents bid on goods in proportional to the utility they receive from them in the previous round. The goods are then allocated in proportion of the agents' bids. It has been shown that proportional response converges to market equilibrium in a variety of Fisher markets [13, 21, 22, 64], and some special cases of exchange markets [15, 16, 62].

The rest of the paper is structured as follows. Section 2 defines the exchange market model and provides examples of WGS demand systems. Section 3 presents the auction algorithm for exchange markets. Section 4 discusses the applicability of the algorithm to the Fisher market model, spending restricted equilibrium, Gale demand systems, and the NSW problem. Several proofs and some significant arguments are deferred to the Appendices, as indicated at the respective parts. For missing proofs and other details, we refer the reader to the full version [39].

2 Models and concepts

Notation $[k]$ denotes $\{1,2,\dots,k\}$, and $\mathbb{1}^k$ denotes the k dimensional vector with all entries 1. We use $\mathbb{1}$ if the dimension is clear from the context. We consider an *exchange market* with a set of agents $A=[n]$ and divisible goods $G=[m]$. Each agent $i\in[n]$ arrives at the market with an initial endowment of goods $e^{(i)}\in\mathbb{R}^m_+$. Thus, the total amount of good $j\in[m]$ is e_j where $e=\sum_{i=1}^n e^{(i)}$; w.l.o.g. $e_j>0$. Given a non-negative price vector $p\in\mathbb{R}^m_+$, the budget of agent i at prices p is defined as $b_i=b_i(p)=p^\top e^{(i)}$. It follows that $p^\top e=\sum_i p^\top e^{(i)}=\sum_i b_i$.

We now define the market equilibrium using *demand systems.* A *bundle* x is a non-negative vector $x\in\mathbb{R}^m_+$. A *demand system* is a function $D:\mathbb{R}^{m+1}_+\to 2^{\mathbb{R}^m_+}$; $D(p,b)$ denotes the set of preferred bundles of an agent at prices p and budget b. Bundles in $D(p,b)$ are called the *optimal* or *demand* bundles at prices p and budget b. This corresponds to the standard concept of a demand function, except that we do not assume the uniqueness of a preferred bundle. For example, in case of a linear utility function $u(x)=\sum_{j\in G} v_j x_j$, $D(p,b)$ includes all fractional assignment of goods maximizing v_j/p_j with a total price b. If $|D(p,b)|=1$ for all $(p,b)\in\mathbb{R}^{m+1}$ we say that D is *simple*, and use $D(p,b)$ to denote the unique bundle. We include the budget b in the definition of the demand system, even though for exchange markets the budget of agent i is uniquely defined by the prices as $p^\top e^{(i)}$. This formalism will be useful for our algorithm where the budgets are defined according to a slightly different set of prices.

▶ **Definition 1** (Market equilibrium). *Let D_i denote the demand system of agent $i\in A$. We say that the prices $p\in\mathbb{R}^m_+$ and bundles $x^{(i)}\in\mathbb{R}^m_+$ form a* market equilibrium *if (i) $x^{(i)}\in D_i(p,p^\top e^{(i)})$, and (ii) $\sum_{i=1}^n x^{(i)}_j\le e_j$, with equality whenever $p_j>0$, for all $j\in G$.*

That is, p and optimal bundles $x^{(i)}$ form an equilibrium if no good is overdemanded and goods at a positive price are fully sold. Note that this implies that every agent fully spends their budget.

▶ **Definition 2.** *Let $(p,b)\in\mathbb{R}^{m+1}_+$ and $x\in D(p,b)$. If for any $p'\ge p$ and $b'\ge b$ there exists $y\in D(p',b')$ such that $y_j\ge x_j$ whenever $p'_j=p_j$, we say that the demand system D satisfies the* weak gross substitutes (WGS) *property.*

We will also say that $D(p,b)$ is a WGS demand system. In the context of the tâtonnement process, the weak gross substitutes property is usually defined with respect to the *aggregate* excess demand function of all agents. We use the stronger requirement of having a WGS demand system for each individual agent. The previous auction algorithms [46, 47] have also used WGS on the level of agents as this seems to be the necessary condition that allows agents to update their bundles individually, as opposed to tâtonnement, where the prices adjustments react to the aggregate demands.

▶ **Definition 3** (Demand oracle). *For a WGS demand system $D(p,b)$, a* demand oracle *requires two vectors $(p,b),(p',b')\in\mathbb{R}^{m+1}_+$ such that $(p',b')\ge(p,b)$, and a vector $x\in D(p,b)$. The output is a vector $y\in D(p',b')$ such that that $y_j\ge x_j$ whenever $p'_j=p_j$.*

In other words, the oracle provides the allocations guaranteed by the definitions of WGS systems. The complex form of the definition is due to the possible non-uniqueness of demand bundles. For simple demand systems, the input to the oracle is simply a vector $(p',b')\in\mathbb{R}^{m+1}_+$, and the output is the unique vector $y\in D(p',b')$.

For exchange markets, we will make the following assumptions:

▶ **Assumption 4** (Scale invariance). *For every agent i, $D_i(p,b_i)=D_i(\alpha p,\alpha b_i)$ for all $\alpha>0$.*

▶ **Assumption 5** (Non-satiation). *For all demand systems, and for every $(p, b) \in \mathbb{R}_+^{m+1}$, and every $x \in D(p, b)$, we have $p^\top x = b$.*

In scale invariance, we require that the demand is homogeneous of degree 0; informally, the demand does not depend on the currency. This is a standard assumption in microeconomics and exchange markets, see e.g. [5, 33, 35, 55].

Non-satiation states that in every optimal bundle the agents must fully spend their budgets. This is a standard assumption for exchange markets as it is necessary for the fundamental theorems of welfare economics (see e.g. [54, Chapter 16]). However, we note that we do not require this assumption for spending restricted Fisher markets.

Approximate equilibria. We define the concept of an ϵ-equilibrium in exchange markets that our algorithm finds. We require that each agent gets an approximate optimal bundle and market clears approximately.

▶ **Definition 6** (Approximate equilibrium). *For an $\epsilon > 0$, the prices $p \in \mathbb{R}^m$ and bundles $x^{(i)} \in \mathbb{R}_+^m$ form an* ϵ-approximate market equilibrium *if*

(i) $x^{(i)} \leq z^{(i)}$ *for some* $z^{(i)} \in D_i(p^{(i)}, p^\top e^{(i)})$*, where* $p \leq p^{(i)} \leq (1+\epsilon)p$*,*

(ii) $\sum_{i=1}^n x_j^{(i)} \leq e_j$*, and*

(iii) $\sum_{j=1}^m p_j \left(e_j - \sum_{i=1}^n x_j^{(i)}\right) \leq \epsilon p^\top e$.

That is, every agent owns a subset of their optimal bundle at prices that are within a factor $(1 + \epsilon)$ from p, and all goods are nearly sold: the value of the unsold goods is at most an ϵ fraction of the total value of the goods. The total value of the goods "taken away" from the near-optimal bundles of the agents is $\sum_{i=1}^n p^\top(z^{(i)} - x^{(i)})$. Parts (i) and (iii), together with the fact that ${p^{(i)}}^\top z^{(i)} \leq p^\top e^{(i)}$ for all i, imply that this amount is $\leq 2\epsilon p^\top e$.

The definition (i) can be seen as a natural extension of the corresponding approximate optimality conditions in [44, 46, 47]. For linear utilities, [44] requires the approximate maximum bang-per-buck condition $v_{ij}/p_j \leq (1 + \epsilon)v_{ik}/p_k$ for any agent i, goods j and k such that $x_{ik} > 0$. Thus, one can set approximate prices $p \leq p^{(i)} \leq (1 + \epsilon)p$ for each agent for which they purchase maximum bang-per-buck goods.

Condition (iii) corresponds to the definition of approximate equilibrium in [32] and [48]. This notion is weaker than the ones used in [44, 46, 47]. The most important difference is that the latter papers guarantee that each agent recovers approximately their optimal utility. Such a property could be achieved by strengthening the bound in (iii) from $\epsilon p^\top e$ to $\epsilon p_{\min} e_{\min}$, where $p_{\min}$ is the minimum price and $e_{\min}$ is the smallest total fractional amount in the initial endowment of any agent. However, this would come at the expense of substantially worse running time guarantees in our algorithmic framework.

2.1 Examples of WGS demand systems

A standard way to implement a demand oracle is via an explicitly given utility function. Assume the agent is equipped with a concave utility function $u : \mathbb{R}_+^m \to \mathbb{R}_+$. The set of demand bundles at prices p and budget b it given as the set of optimal solutions of

$$\max \quad u(x) \quad \text{s.t.} \quad p^\top x \leq b; \quad x \geq 0\,. \tag{1}$$

Then, $D(p, b) := D^u(p, b) = \arg\max_{x \in \mathbb{R}_+^m} \{u(x) : p^\top x \leq b\}$. We say that a utility function is WGS if the corresponding demand system is WGS. Most models studied in the literature assume strictly concave utilities and thus have a unique optimal solution; a notable exception is the case of linear utility functions. If the solution is not unique, we can implement the demand oracle for inputs $(p, b), (p', b')$ and $x \in D(p, b)$ by imposing the constraints that $u(y)$ equals the optimal utility in $D(p', b')$, and $y_i \geq x_i$ for every i with $p'_i = p_i$. Thus, the optimal demand system can also be implemented via convex programming (we now ignore the question of numerical precision).

We now present some classical examples of WGS utilities studied in the literature:

- For $v \in \mathbb{R}^m_+$ the *linear (additive) utility* is given by $u(x) = v^\top x$. Then, $D^u(p, b) = \arg\max\{v^\top x : p^\top x \leq b\}$.
- The *constant elasticity of substitution (CES)* utility is defined by $u(x) = \left(\sum_j \beta_j^{\frac{1}{\sigma}} x_j^{\frac{\sigma-1}{\sigma}}\right)^{\frac{\sigma}{\sigma-1}}$, where $\sum_j \beta_j = 1$. Then, $D(b, p) = \{x\}$ for the unique optimal bundle x given by $x_j = \dfrac{\beta_j p_j^{-\sigma} b}{\sum_k \beta_k p_k^{1-\sigma}}$. It is well-known that CES demand system satisfies the WGS property iff $\sigma > 1$.
- The *Cobb-Douglas* utility function is given by $u(x) = \prod_j x_j^{\alpha_j}$ where $\sum_j \alpha_j = 1$, $\alpha \geq 0$. The unique optimal bundle is therefore $x_j = b\alpha_j / p_j$ and $D^u(p, b) = \{x\}$. The Cobb-Douglas utility function satisfies the WGS property for any parameter choices.
- The *nested CES* utility function is defined recursively (see [49]). Any CES function is a nested CES function. If $g, h_1, \ldots, h_t$ are nested CES functions, then $f(x) = \max g(h_1(x^1), \ldots, h_t(x^t))$ over all $x^1, \ldots, x^t$ such that $\sum_{k=1}^t x^k = x$, is a nested CES function. In a well-studied special case (see e.g., [51]), each good j can only be used in at most one of the h_i's.

Conic combinations of demand systems. Given two WGS utility functions u and u', the demand system corresponding to their sum $u + u'$ may not be WGS. On the other hand, consider two simple WGS demand systems D and D' and nonnegative coefficients λ, λ'. Then it is easy to see that $\lambda D + \lambda' D'$ is also a simple WGS demand system. This enables the construction of some interesting demand systems. For example, [55] has studied hybrids of CES and Cobb-Douglas demands, where the demand system is given as a conic combination of the two. [1]

$$x_j = \frac{b}{p_j}\left[\epsilon\alpha_j + (1-\epsilon)\frac{\beta_j p_j^{1-\sigma}}{\sum_k \beta_k p_k^{1-\sigma}}\right], \text{ for some } 0 \leq \epsilon \leq 1 \text{ and } \sigma > 1 \ .$$

Note that if $D = D^u$ and $D' = D^{u'}$ for some concave utility functions u and u', the demand system $\lambda D + \lambda' D'$ in general does *not* correspond to the utility function $\lambda u + \lambda' u'$. In fact, it is unclear if one can find explicitly utility functions corresponding to such conic combinations. Our model does *not* require the demand system to be given in the form $D = D^u$ for some function u.

Price elasticity of demands. One possible implementation of the key subroutine `FindNewPrices` (Section 3) relies on the *(price) elasticity of the demands.*[2] The standard definition of the elasticity for good j with respect to the price of good k is $e_{j,k} = \partial \log x_j(p, b)/\partial \log p_k$, where $x_j(p, b)$ is the (unique) demand for good j at prices p and budget b. The WGS

[1] We note that this demand function does not seem to correspond to a nested CES utility function.
[2] No finite lower bound exists on the elasticity of linear demand systems. If we are buying a positive amount of good j, then j maximizes v_k/p_k. If there is another good ℓ with $v_j/p_j = v_\ell/p_\ell$, then if we increase p_j but leave the other prices unchanged, then $x'_j = 0$ for every optimal bundle x' w.r.t. the new prices. Consequently, for this case, we have another way to implement `FindNewPrices`.

property guarantees that $e_{j,k} \geq 0$ if $j \neq k$, and consequently, $e_{k,k} \leq 0$. The definition below corresponds to $e_{k,k} \geq -f$ for all $k \in [m]$, for the more general model of non-simple demand systems.

▶ **Definition 7.** *Consider a WGS demand system $D(p, b)$. For some $f > 0$, we say that the* elasticity of $D(p, b)$ is at least $-f$, *if for any $\mu \geq 0$, $j \in [m]$, $(p, b) \in \mathbb{R}_+^{m+1}$ and $x \in D(p, b)$, if we define p' as $p'_j = p_j(1 + \mu)$ and $p'_k = p_k$ for $k \in [m] \setminus \{j\}$, then there exists a bundle $x' \in D(p', b)$ such that $x'_j \geq \frac{1}{(1+\mu)^f} x_j$.*

In can be shown that the CES demand system with parameter $\sigma > 1$ has elasticity at least $-\sigma$, and the Cobb-Douglas demand system has elasticity at least -1.

Separable and uniformly separable WGS utility functions. The auction algorithm in [44] was later extended in [47] to separable WGS utility functions, that is, $u = \sum_{j \in G} u_j$ where each u_j is a WGS utility function depending only on good j. This model was further generalized in [46] to *uniformly separable* WGS utility functions, that is, $\frac{\partial u(x)}{\partial x_j} = f_j(x_j) g(x)$, where each f_j is a strictly decreasing function. This class already includes CES and Cobb-Douglas utilities; however, it does not appear to extend to demand systems obtained as their conic combinations, where even the explicit form of the utility function is unclear. Further, the running time bound stated in [46] is unbounded for the CES and Cobb-Douglas cases; see the full version of the paper for further discussion.

3 Auction algorithm for exchange markets

The algorithm (shown in Algorithm 1) uses the accuracy parameter $0 < \epsilon < 0.25$, and returns a 4ϵ-approximate equilibrium. We initialize all prices $p_j = 1$ and the prices will only increase during the algorithm, in increments by a factor $(1 + \epsilon)$. This initialization is enabled by Assumption 4 that guarantees the existence of market clearing prices where all positive prices are ≥ 1.[3]

We maintain a price vector p called the *market prices*; the budget of agent $i \in [n]$ is $b_i = p^\top e^{(i)}$ at the current prices. Further, every agent $i \in [n]$ maintains individual prices $p^{(i)}$ such that $p \leq p^{(i)} \leq (1 + \epsilon)p$. At any point of the algorithm, agent i owns a bundle $c^{(i)}$ of the goods such that $c^{(i)} \leq x^{(i)}$ for some $x^{(i)} \in D_i(p^{(i)}, b_i)$. Some amount of good j is sold at the lower price p_j, and some at the higher price $(1 + \epsilon)p_j$. The price agent i has to pay for good j is the higher price $(1 + \epsilon)p_j$ if $p_j^{(i)} = (1 + \epsilon)p_j$ and the lower price p_j otherwise. (Note that this is in contrast with [44] and the other previous auction algorithms where i may pay p_j for some amount of good j and $(1 + \epsilon)p_j$ for another amount.)

We consider the agents one-by-one. If an agent i has surplus money, they use the subroutine `FindNewPrices` to update their prices $p^{(i)}$ and bundle $x^{(i)}$, by maintaining $x_j^{(i)} \geq c_j^{(i)}$ – this latter requirement turns out to be the main challenge. They will then try to purchase $x_j^{(i)} - c_j^{(i)}$ amount of good j in the `Outbid` procedure. They start by purchasing any unsold amount of good at price p_j. If they still need more, then they will outbid other agents who have been paying the lower price p_j for this good, by offering the higher price $(1 + \epsilon)p_j$. Once good j is sold only at the higher price $(1 + \epsilon)p_j$, we increase the price of the good. If no price is increased, we move to the next agent. Otherwise, we announce the new prices p and repeat. The algorithm terminates once the total surplus of the agents drops below $3\epsilon p^\top e$. At this point, we can conclude that the current prices and allocations form a 4ϵ-approximate equilibrium.

[3] Even though there might be goods priced at 0 in an equilibrium, we can always find an ϵ-approximate equilibrium where all prices are positive.

We express the running time of the algorithm in terms of the running time T_F of the subroutine `FindNewPrices`, as well as the upper bound on the ratio $p_{\max}/p_{\min}$ of the largest and smallest nonzero prices at any ϵ-equilibrium. Such an upper bound may be obtained for the specific demand systems. Alternatively, one can follow the approach of the papers [23, 25] by adding a dummy agent with a Cobb-Douglas demand system and an initial endowment of a small fraction of all goods. In the presence of such an agent, we can obtain a strong bound on $p_{\max}/p_{\min}$, at the expense of obtaining a slightly worse approximation guarantee (see the full version of the paper).

Note that for (approximate-)equilibrium prices p, αp also gives (approximate-)equilibrium prices with the same allocation, for any $\alpha > 0$. In our algorithm, the minimum price will remain at most $1+\epsilon$ throughout, see Lemma 10.

▶ **Theorem 8.** *Let T_F be an upper bound on the running time of the subroutine `FindNewPrices`. Algorithm 1 finds a 4ϵ-approximate market equilibrium in time $O\left(\frac{nmT_F}{\epsilon^2}\cdot\log\left(\frac{p_{\max}}{p_{\min}}\right)\right)$.*

There are various options for implementing `FindNewPrices`. A simple price can be implemented increment procedure for the case of bounded elasticities; recall the elasticity bound f from Definition 7. Using this subroutine and Lemma 13, we obtain the following overall bound.

▶ **Theorem 9.** *If all agents have elasticity at least $-f$ for some $f > 0$, then an ϵ-approximate equilibrium can be computed in time $O\left(\frac{nm^2f\cdot T_D}{\epsilon^2}\cdot\log\left(\frac{p_{\max}}{p_{\min}}\right)\right)$, where T_D is the time needed for one call to the demand oracle.*

As noted earlier, there are demand systems (such as linear) where the flexibility parameter cannot be bounded. However, in case the demand system is given in the form (1) via a utility function that is homogeneous of degree one, we can obtain an implementation of `FindNewPrices` by solving a convex program. This is in particular applicable for Cobb-Douglas and CES utilities with $\sigma > 1$. One could find further possible ways for implementing `FindNewPrices` for particular demand systems; e.g., we give a simple direct procedure for linear utilities, and for budget-SPLC utilities. For details, see the full version of the paper.

The full version also contains an overview of the running times of previous auction algorithms.

Invariants. Let us now summarize the invariant properties maintained throughout the algorithm. We say that a bundle y dominates the bundle x if $x \le y$.

(a) Each good is partitioned into three parts according to the price it is being sold at:
- amount w_j is the unsold part of the good,
- amount l_j is sold at the lower price p_j, and
- amount h_j is sold at the higher price $(1+\epsilon)p_j$.

Moreover, $w_j + l_j > 0$, i.e., there is always a part of the good that is unsold or owned by an agent at the lower price.

(b) The unsold amount w_j of each good is non-increasing. If $w_j > 0$ then $p_j = 1$.

(c) The budget of agent i is $b_i = p^\top e^{(i)}$. Each agent i maintains prices $p^{(i)}$ such that $p \le p^{(i)} \le (1+\epsilon)p$, and owns a bundle $c^{(i)}$ that is dominated by a bundle $x^{(i)} \in D_i(p^{(i)}, b_i)$.

(d) For the amount $c_j^{(i)}$ of good j, agent i pays
- price p_j for goods in $L_i := \{j \in [m] : p_j^{(i)} < (1+\epsilon)p_j\}$, and
- the price $(1+\epsilon)p_j$ for goods in $H_i := \{j \in [m] : p_j^{(i)} = (1+\epsilon)p_j\} = [m] \setminus L_i$.

Algorithm 1 Auction algorithm for exchange markets.

Input: Demand systems D_i, and the endowment vectors $e^{(i)}$, and $\epsilon \in (0, 0.25)$.
Output: A 4ϵ-approximate market equilibrium.

Initialization: $\forall i, j$ set $p_j \leftarrow 1$, $p_j^{(i)} \leftarrow 1$, $c_j^{(i)} \leftarrow 0$, $w_j = e_j = \sum_i e_j^{(i)}$, and $l_j = 0$;
NewIt **for** $i \in [n]$ **do** `// recompute the budgets and surpluses`
  $b_i \leftarrow p^\top e^{(i)}$; $s_i \leftarrow b_i - \sum_{j \in L_i} c_j^{(i)} p_j - \sum_{j \in H_i} c_j^{(i)} (1+\epsilon) p_j$
if $\sum_{i=1}^{n} s_i \leq 3\epsilon p^\top e$ **then return** p, $\{p^{(i)}\}_{i \in [n]}$ and $\{c^{(i)}\}_{i \in [n]}$;
NewStp **for** $i \in [n]$ *with* $s_i > 0$ **do** `// step for agent` i
  $(\tilde{p}, y) \leftarrow$ `FindNewPrices`$(i, p^{(i)}, p, \epsilon, c^{(i)}, b_i)$;
  for $j = 1$ **to** m **do**
    if $p_j^{(i)} < (1+\epsilon) p_j$ *and* $\tilde{p}_j = (1+\epsilon) p_j$ **then** `// Case 1`
      $s_i \leftarrow s_i - c_j^{(i)} \cdot \epsilon p_j$; $l_j \leftarrow l_j - c_j^{(i)}$; `//` i `pays` $(1+\epsilon)p_j$ `instead of` p_j
      `Outbid`$(i, j, y_j - c_j^{(i)})$;
    else if $p_j^{(i)} = (1+\epsilon) p_j$ *and* $\tilde{p}_j = (1+\epsilon) p_j$ **then** `// Case 2`
      `Outbid`$(i, j, y_j - c_j^{(i)})$;
    `// Skip the goods with` $p_j^{(i)} < (1+\epsilon) p_j$ `and` $\tilde{p}_j < (1+\epsilon) p_j$`.` `Case 3`
  $p^{(i)} \leftarrow \tilde{p}$; flag $\leftarrow 0$;
  for $j \in [m]$ *with* $w_j + l_j = 0$ **do**
    $p_j \leftarrow (1+\epsilon) p_j$; $l_j = e_j$; `// price increase`
    foreach $k \in [n]$ **do** $p_j^{(k)} \leftarrow (1+\epsilon) p_j$;
    flag $\leftarrow 1$;
  if *flag* $= 1$ **then go to** NewIt;

Procedure Outbid(i, j, t).

`//` t `is the amount of good` j `agent` i `wants to outbid.`
if $w_j > 0$ **then** `// a part of` j `is unsold`
  $\tau = \min\{w_j, t\}$;
  $w_j \leftarrow w_j - \tau$; $c_j^{(i)} \leftarrow c_j^{(i)} + \tau$; $t \leftarrow t - \tau$;
  $s_i \leftarrow s_i - \tau \cdot (1+\epsilon) p_j$; `// here` $p_j = 1$ `always`
while $t > 0$ *and* $l_j > 0$ **do**
  Let $k \in [n]$ be such that $c_j^{(k)} > 0$ and $p_j^{(k)} = p_j$. Set $\tau = \min\{c_j^{(k)}, t\}$;
  $c_j^{(k)} \leftarrow c_j^{(k)} - \tau$; $c_j^{(i)} \leftarrow c_j^{(i)} + \tau$; `//` i `outbids` k
  $s_k \leftarrow s_k + \tau \cdot p_j$; $s_i \leftarrow s_i - \tau \cdot (1+\epsilon) p_j$; $l_j \leftarrow l_j - \tau$; $t \leftarrow t - \tau$;

In accordance with (d), the *surplus* of agent i is $s_i := b_i - \sum_{j \in L_i} c_j^{(i)} p_j - \sum_{j \in H_i} c_j^{(i)} (1+\epsilon) p_j$.

The Outbid subroutine. An important subroutine, described in Procedure `Outbid`, controls how the ownership of goods may change. If agent k has paid price p_j on a certain amount of good j, then agent i may take over some of this amount by offering a higher price $(1+\epsilon)p_j$. Possibly $i = k$, in which case the agent outbids herself. We also incorporate into the procedure the case when a certain amount of a good is being purchased for the first time. Note that $p_j = 1$ at this point due to invariant (b).

Main iterations. The algorithm is partitioned into iterations. Each iteration finishes when the price of a good increases from p_j to $(1+\epsilon)p_j$. At every such event, the budgets b_i of the agents also increase. Therefore, at the start of an iteration each agent i recomputes their budget at line NewIt. An iteration is further partitioned into steps, which are single executions of the main for loop in Algorithm 1. The algorithm terminates as soon as the total surplus drops below $3\epsilon p^\top e$.

Steps. Suppose we are considering agent i. By invariant (c), the agent is buying a bundle $c^{(i)} \le x^{(i)}$ for some $x^{(i)} \in D_i(p^{(i)}, b_i)$. The subroutine $\texttt{FindNewPrices}(i, p^{(i)}, p, \epsilon, c^{(i)}, b_i)$ delivers new prices $\tilde{p}$ and a bundle y such that

(A) $y \ge c^{(i)}$ for $y \in D_i(\tilde{p}, b_i)$, and

(B) $p^{(i)} \le \tilde{p} \le (1+\epsilon)p$, and $\tilde{p}_j = (1+\epsilon)p_j$ whenever $y_j > (1+\epsilon)\, c_j^{(i)}$.

Condition (A) says that agent i still wants whatever they own even at the increased prices $\tilde{p}$. Condition (B) is the crucial one for the outbid. It guarantees that $\tilde{p} \ge p^{(i)}$, and whenever an agent wants to buy more of some good than they already own at least by a factor $1+\epsilon$, then they are willing to pay the higher price $(1+\epsilon)p_j$ for it. (They might already be paying the increased price to start with if $p_j^{(i)} = (1+\epsilon)p_j$. In this case $\tilde{p}_j = (1+\epsilon)p_j = p_j^{(i)}$.) The description of this subroutine is given in the full version of the paper. Observe that FindNewPrices will make progress whenever $c^{(i)}$ is far from $x^{(i)}$ for some agent i. When they are very close for each agent i, then we have already reached an approximate equilibrium.

The above properties suggest the following update rules for each good $j \in [m]$.

Case 1. $p_j^{(i)} < (1+\epsilon)p_j$ *and* $\tilde{p}_j = (1+\epsilon)p_j$. The good j was in L_i and needs to be moved to H_i, i.e., agent i used to pay p_j but now is willing to pay the higher price for j. Agent i first outbids themselves for the amount $c_j^{(i)}$ they already own and starts paying $p_j(1+\epsilon)$ for this amount. Additionally, agent i outbids on good j up to the amount they want and that is available from the other agents.

Case 2. $p_j^{(i)} = (1+\epsilon)p_j$ *and* $\tilde{p}_j = (1+\epsilon)p_j$. The good j was in H_i and stays in H_i, i.e., agent i continues to pay the higher price. The agent i still keeps the amount $c_j^{(i)}$ of good j that they already had and outbids for as much as they can from the other agents.

Case 3. $p_j^{(i)} < (1+\epsilon)p_j$ *and* $\tilde{p}_j < (1+\epsilon)p_j$. The good j remains in L_i, i.e., agent i continues to pay the lower price. By (B), we must have $c_j^{(i)} \le y_j \le (1+\epsilon)c_j^{(i)}$; the agent will not seek to buy more of these goods.

The cases above have covered all possibilities since $p_j^{(i)} \le \tilde{p}_j$. Note that in the first two cases the agent will own $\min(y_j, l_j + w_j)$ amount of good j, whereas they will own $c_j^{(i)}$ amount in the third case. Once all of the goods have been considered we set $p^{(i)} = \tilde{p}$, $x^{(i)} = y$, and update $c^{(i)}$ as the current allocation. If $w_j + l_j = 0$ for some j then $h_j = e_j$, i.e., the whole j is sold at the higher price $p_j(1+\epsilon)$. For each such good j we increase the market price p_j to $(1+\epsilon)p_j$, and for all agents k we set $p_j^{(k)} = p_j$ for the new increased p_j; finally, we set $l_j = e_j$ and $h_j = 0$. The step ends.

3.1 Analysis

The missing proofs are presented in the full version. Here, we analyze the running time.

▶ **Lemma 10.** *The smallest price* $\min_{j \in G}\{p_j\}$ *remains at most* $(1+\epsilon)$ *throughout the algorithm.*

Next, we give a bound on the number of iterations, using the same basic idea of organizing the steps into rounds as in [44]. A *round* consists of going over all agents exactly once in the main "for" loop and doing a step for each agent; i.e, a round comprises at most n steps.

▶ **Lemma 11.** *The number of rounds in an iteration is at most* $2/\epsilon$.

Proof. Let us fix an iteration and denote with p the market prices at the start of the iteration. Consider a step of an agent i within the iteration. If from a good j, i buys everything that is available at the cheaper price p_j, then the market price of j increases and the iteration finishes. So for the rest of the proof we assume that the market price increase does not happen; consequently, the budget of each agent is unchanged and agent i gets the amount of each good it desires.

Let φ denote the total amount of money spent at a certain point of this iteration that is spent by the agents on higher price goods. That is, $\varphi = (1+\epsilon)\sum_{i=1}^{n}\sum_{j\in H_i} c_j^{(i)} p_j$.

▷ Claim 12. Let s_i denote the surplus of agent i at the beginning of their step. Then the value of φ increases at least by $s_i - 2.25\epsilon b_i$ in the step of agent i.

Proof. Recall Cases 1-3 in the description of the step. Let T_k be the set of goods that fall into case k, that is, $T_1 \cup T_2 \cup T_3 = [m]$.

- If $j \in T_1$, then $(1+\epsilon)p_j y_j$ amount will be added to φ in the `Outbid` subroutine: In this case, the agent also outbids itself, moving the good from L_i to H_i.
- If $j \in T_2$, then $(1+\epsilon)p_j(y_j - c_j^{(i)})$ amount will be added to φ in the `Outbid` subroutine.
- If $j \in T_3$, then we do not increase φ. Nevertheless, (B) guarantees that $\tilde{p}_j(y_j - c_j^{(i)}) \le \epsilon \tilde{p}_j c_j^{(i)}$. Consequently,

$$\sum_{j\in T_3} \tilde{p}_j(y_j - c_j^{(i)}) \le \epsilon \tilde{p}^\top c^{(i)}. \tag{2}$$

Also note that $\tilde{p}_j = (1+\epsilon)p_j$ if $j \in T_1 \cup T_2$. Assumption 5 on non-satiation guarantees that $\tilde{p}^\top y = b_i$. Let $\Delta\varphi$ denote the increment in φ; this can be lower bounded as

$$\begin{aligned}\Delta\varphi &= \sum_{j\in T_1} \tilde{p}_j y_j + \sum_{j\in T_2} \tilde{p}_j(y_j - c_j^{(i)}) = \tilde{p}^\top y - \sum_{j\in T_3} \tilde{p}_j y_j - \sum_{j\in T_2} \tilde{p}_j c_j^{(i)} \\ &\ge b_i - \sum_{j\in T_3} \tilde{p}_j(y_j - c_j^{(i)}) - \tilde{p}^\top c^{(i)} \ge b_i - (1+\epsilon)\tilde{p}^\top c^{(i)} \ ,\end{aligned}$$

using (2). The money spent by the agent at the beginning of the step is $b_i - s_i$. Good j is purchased at price at least p_j according to (d), and $\tilde{p}_j \le (1+\epsilon)p_j$. Consequently, $\tilde{p}^\top c^{(i)} \le (1+\epsilon)(b_i - s_i)$. With the above inequality and using that $\epsilon < 0.25$, we obtain $\Delta\varphi \ge b_i - (1+\epsilon)^2(b_i - s_i) \ge s_i - 2.25\epsilon b_i$. ◁

As long as $\sum_{i=1}^n s_i > 3\epsilon p^\top e$, the claim guarantees that φ increases in every round by at least $3\epsilon p^\top e - 2.25\epsilon\sum_{i=1}^n b_i > 0.5\epsilon p^\top e$. Since $\varphi \le p^\top e$, the number of rounds is at most $2/\epsilon$. ◀

Proof of Theorem 8. In their steps, agents use their surpluses to outbid for the goods. We bound the number of repeats in the "while" cycles (lines 5–8) in all calls to `Outbid` in a given iteration. When `Outbid`(i,j,t) is called, the "while" loop is repeated until $t=0$ or good j is sold only at the higher price. Moreover, `Outbid`(i,j,t) possibility sets some $c_j^{(k)}$ to zero. The total number of such events within a single iteration is bounded by nm – each agent loses a good through the outbid at most once before the prices increases and iteration finishes.

Hence, the number of "while" calls is at most nm plus the total number of calls to `Outbid`. This is at most m in each step, and thus nm in each round. According to Lemma 11, the number of repeats "while" calls in every iteration is $2nm/\epsilon$; each repeat takes $O(1)$ time. The same bound holds for the 'if' calls in lines 1–4 in `Outbid`.

Every step calls the procedure `FindNewPrices` exactly once. Therefore, the time taken by `FindNewPrices` in an iteration is $O(nT_F/\epsilon)$. According to Lemma 10, the minimum price remains at most $1+\epsilon$ throughout. Hence, the number of iterations is bounded by $O(m \log_{1+\epsilon}(p_{\max}/p_{\min})) = O(\frac{m}{\epsilon}\log(p_{\max}/p_{\min}))$. The claimed running time bound follows, using also $T_F = \Omega(m)$ since the output needs to return an m-dimensional vector of goods.

It is left to show that the prices p and bundles $c^{(i)}$ form a 4ϵ-approximate market equilibrium. The first two properties in the definition are clear: $c^{(i)}$ is dominated by an optimal bundle with respect to the prices $p^{(i)}$, and no good is oversold. At termination, the total surplus of the agents is bounded by $3\epsilon p^\top e$. However, this surplus is computed assuming that some goods are sold at price p_j and others at price $(1+\epsilon)p_j$. Decreasing the price of the latter goods to p_j releases an additional excess of at most $\epsilon p^\top e$. Consequently, $\sum_{j=1}^m p_j(e - \sum_{i=1}^n c_j^{(i)}) \le 4\epsilon p^\top e$. ◀

3.2 Implementing FindNewPrices

We now describe the subroutine `FindNewPrices`$(i, p^{(i)}, p, \epsilon, c^{(i)}, b_i)$. Recall that the outputs are new prices $\tilde{p} \ge p^{(i)}$ and a bundle y with

(A) $y \ge c^{(i)}$ for $y \in D_i(\tilde{p}, b_i)$, and

(B) $p^{(i)} \le \tilde{p} \le (1+\epsilon)p$, and $\tilde{p}_j = (1+\epsilon)p_j$ whenever $y_j > (1+\epsilon)\, c_j^{(i)}$.

Let us assume that the demand system D_i has elasticity at least $-f$ for some $f > 0$. Our Algorithm 2 for this case is a simple price increment procedure. First, we obtain $y \in D_i(p^{(i)}, b_i)$ from the demand oracle with $y \ge c^{(i)}$. This is possible due to invariant (c), which guarantees that $c^{(i)} \le x^{(i)}$ for some $x^{(i)} \le D_i(p^{(i)}, b_i)$. Then, the demand oracle is able to return a bundle y such that $y \ge x^{(i)} \ge c^{(i)}$. Then, we iterate the following step. As long as (B) is violated for a good j, we increase its price by a factor $(1+\epsilon)^{1/f}$ until it reaches the upper bound $(1+\epsilon)p_j$.

Algorithm 2 Finding new prices.

Input: $i, p^{(i)}, p, \epsilon, c^{(i)}, f, b_i$.
Output: Prices $\tilde{p}$ and bundle y.

Initialization: $\tilde{p} \leftarrow p^{(i)}$;
Obtain $y \in D_i(\tilde{p}, b_i)$ from the demand oracle with $y \ge c^{(i)}$;
while $\exists j : \tilde{p}_j < (1+\epsilon)p_j$ *and* $y_j > (1+\epsilon)c_j^{(i)}$ **do**
  $\tilde{p}_j \leftarrow \min\{(1+\epsilon)^{1/f}\tilde{p}_j, (1+\epsilon)p_j\}$;
  Obtain $y' \in D_i(\tilde{p}, b_i)$ from the demand oracle such that $y'_k \ge y_k$ for $k \ne j$;
  $y \leftarrow y'$;
return $(\tilde{p}, y)$;

▶ **Lemma 13.** *Assume the demand system D_i has elasticity at least $-f$ for some $f > 0$. Algorithm 2 terminates with $\tilde{p}$ and y satisfying* (A) *and* (B) *in time $O(mf \cdot T_D)$, where T_D is the time for a call to the demand oracle.*

We will assume that $T_D = \Omega(m)$, since the demand oracle needs to output an m-dimensional vector.

Proof. The bound on the number of iterations is clear: since we have $p \le \tilde{p} \le (1+\epsilon)p$ throughout, the price of every good can increase at most f times. Condition (A) is satisfied due to the WGS property and the bound on the demand elasticity. When increasing $\tilde{p}_j$, the demand y_k for $k \ne j$ is non-decreasing as guaranteed by the demand oracle. Further, y_j may decrease only by a factor $(1+\epsilon)$, and since we had $y_j > (1+\epsilon)c_j^{(i)}$ before the price update, we still have $y_j > c_j^{(i)}$ after the price update. Condition (B) is satisfied at termination since the while loop keeps running as long as it is violated. Checking the while condition each time requires $O(m)$ time; however, this will be dominated by the time T_D according to the comment on $T_D \ge m$ above. ◀

As explained in Section 3, this is only one of the possible ways of implementing `FindNewPrices`. A convex programming approach for utilities that are homogeneous of degree 1 can be developed. For example, for CES with parameter $\sigma > 1$, the running time of Algorithm 2 depends linearly on σ, whereas the running time of the convex programming is independent on this parameter. Nevertheless, for small values of σ the simple price increment procedure may be preferable to solving a convex program.

Further, more direct approaches for implementing `FindNewPrices` may be possible for particular demand systems. For Cobb-Douglas demands with parameter vector $\alpha^{(i)}$, it is easy to devise an $O(m)$ time algorithm implementing the procedure. The algorithm relies on the fact that the optimal bundle is the bundle that allocates $\alpha_j^{(i)} b_i$ money for good j. Hence, each price can be set independently of the others. Similarly, there is $O(m)$ procedure for implementing `FindNewPrices` for linear utilities; recall from Section 2.1 that the elasticity is unbounded in this case.

4 Fisher markets and the Nash social welfare problem

Fisher markets are a well-studied special case of exchange markets, where the initial endowment of agent i is $\delta_i e$ for $\delta_i > 0$ and therefore the relative budgets of the agents are independent of the prices. With appropriate normalization of the prices, we can assume that agent i arrives with a fixed budget b_i and that there is exactly one unit of each good. At an equilibrium, the agents spend these budgets on their most preferred goods at the given prices. Let us now assume that the demand systems are given via utility functions as in (1). Eisenberg and Gale [36] gave a convex programming formulation of the market equilibrium problem for linear utilities. Eisenberg [35] showed that the optimal solutions to the following convex program are in one-to-one correspondence with the market equilibria assuming that the utility functions are homogeneous of degree one, that is, $u_i(\alpha x) = \alpha u_i(x)$ for any $\alpha > 0$.

$$\max \sum_{i=1}^{n} b_i \log u_i(x^{(i)}) \quad \text{subject to} \quad \sum_{i=1}^{n} x_j^{(i)} \le 1, \quad \forall j = 1, \ldots, m. \tag{3}$$

We note that the equilibrium prices are given by the optimal Lagrange multipliers.

The Nash social welfare problem. In the Nash social welfare (NSW) problem, we need to allocate m indivisible items to n agents ($m \ge n$), with agent i equipped with a utility function on the subsets of goods. The goal is to find a partition $S_1 \cup S_2 \cup \ldots S_n = [m]$ of the goods in order to maximize the geometric mean of the utilities, $\left(\prod_{i=1}^n u_i(S_i)\right)^{1/n}$. This problem is NP-hard already for additive utilities, that is, if $u_i(S) = \sum_{j\in S} v_{ij}$.

The first constant factor approximation for this problem was given by Cole and Gkatzelis [28]. Their approach was to first solve a continuous relaxation that corresponds to a divisible market problem, and round the fractional optimal solution. The natural relaxation is exactly the program (3) above with all $b_i = 1$. For linear utilities, we can use the natural continuous extension $u_i(x) = \sum_{j\in S} v_{ij}x_{ij}$ of the additive utility function. However, it is easy to see that this relaxation has an unbounded integrality gap. Cole and Gkatzelis [28] introduced the notion of *spending restricted equilibrium* that we now define in a slightly more general form.

▶ **Definition 14.** *Suppose there are n agents with demand systems $D_i(p, b_i)$ and fixed budgets $b \in \mathbb{R}^n_+$. Further, let us be given bounds $t \in (0,\infty)^m$. The prices $p \in \mathbb{R}^m$ and allocations $x^{(i)} \in D_i(p, b_i)$ form a* Spending Restricted (SR) *equilibrium with respect to t, if $\sum_i x_j^{(i)} = \min\{1, t_j/p_j\}, \forall j \in [m]$.*

Note that the set of equilibria can be non-convex already for budget-additive utilities as shown in [38].

At given prices p, we let $a_j(p) = a_j = \min\{1, t_j/p_j\}$ denote the *available amount* of good j. That is, the amount of money spent on good j is bounded by t_j. By setting $t_j = \infty$ for all j, the above reduces to the standard definition of Fisher market equilibrium.

The algorithm in [28] first computes a spending restricted equilibrium for linear Fisher markets with bounds $t_j = 1$, and show that this can be rounded to an integer solution of cost at most $2e^{1/e}$ times the optimal NSW solution. Note that the spending restrictions cannot be directly added to the formulation (3) since they involve the Lagrange multipliers p. An SR-equilibrium in [28] was found via an extension of algorithms by Devanur et al. [31] and Orlin [58] for linear Fisher markets.

Subsequent work by Cole et al. [26] showed that a spending restricted equilibrium for the linear markets can be obtained as an optimal solution of a convex program (extending a convex formulation of linear Fisher market equilibrium that is different from (3)), and also improved the approximation guarantee to 2 (the current best factor is 1.45 [8]). However, this convex formulation is only known to work for linear utility functions.

Further work has studied the NSW problem for more general utility functions, following the same strategy of first solving a spending-restricted market equilibrium problem then rounding. Anari et al. [2] studied NSW with *separable, piecewise-linear concave (SPLC)* utilities. The paper [38] studied *budget-additive valuations*, that correspond to the utility function $u_i(x) = \min(c_i, \sum_j u_{ij}x_j)$. Both papers find (exact or approximate) solutions to the corresponding spending-restricted market equilibrium problem via fairly complex combinatorial algorithms.

The Gale demand systems. The demand systems of the market models in [2, 38] do not exactly correspond to (1). In [38] one needs additional conditions on the agents being "thrifty"; in [2] a "utility market model" is used. In both cases, the total spending of the agents can be below their budgets. A natural unified way of capturing these equilibrium concepts is via *Gale demand systems*, defined as

$$G^u(p, b) = \arg\max_{x\in\mathbb{R}^m_+} b \log u(x) - p^\top x\,. \tag{4}$$

We call $b \log u(x) - p^\top x$ the *Gale objective function*. It is easy to verify using Lagrangian duality that if all u_i's are concave functions, and the utility functions correspond to the Gale demand systems $D_i(p, b) = G^{u_i}(p, b)$, then the program (3) always finds a market equilibrium; see [56] for details. Moreover, if the utilities are homogeneous of degree one, then this equilibrium coincides with the equilibrium for the "standard" demand systems given by (1). For general concave utility functions, the optimal bundles stay within the budget b (that is, $p^\top x \le b$), but may not exhaust it. Finding a spending-restricted equilibrium for

Gale demand systems appears to be the right setting for NSW; in fact, the concepts used by [2] and [38] correspond to the Gale equilibrium in these settings, and moreover, these Gale demand systems admit the WGS property. On contrary, the demand systems arising from the previously mentioned utility functions do not satisfy the WGS property in the usual setting (1).

We refer the reader to the paper by Nesterov and Shikhman [56] on Gale demand systems as well as the more general concept of Fisher-Gale equilibrium; they also give a tâtonnement type algorithm for finding such an equilibrium.

Approximate spending-restricted equilibrium. We use an extension of Definition 6 as our approximate SR-equilibrium notion. The main difference is that we require all goods to be fully consumed.

▶ **Definition 15** (Approximate SR-equilibrium). *Let $t \in [1,\infty]^m$. For an $\epsilon > 0$, the prices $p \in \mathbb{R}^m$ and bundles $x^{(i)} \in \mathbb{R}^m_+$ form an* ϵ-approximate SR-equilibrium *w.r.t. t if*

(i) $x^{(i)} \le z^{(i)}$ *for some* $z^{(i)} \in D_i(p^{(i)}, b_i)$*, where* $p \le p^{(i)} \le (1+\epsilon)p$,

(ii) $\sum_{i=1}^n x_j^{(i)} = a_j := \min\{1, t_j/p_j\}$ *for all j, and*

(iii) $\sum_{j=1}^m p_j \left(\sum_{i=1}^n z_j^{(i)} - a_j\right) \le \epsilon \sum_{i=1}^n b_i$.

We note that whereas an equilibrium will always exist for WGS utilities, the existence of an SR-equilibrium is a nontrivial question. For example, suppose an agent i has budget b_i and Cobb-Douglas utility function $\prod_{j=1}^m (x_j^{(i)})^{\beta_j}$, where $\sum_j \beta_j = 1$, such that $\beta_k > \frac{1}{b_i}$ for some k with $t_k = 1$. Then the agent i would like to spend at least $\beta_k b_i > 1$ on good j for any prices p, but the total money that can be spent on this good is capped at 1. Hence, there doesn't exist any SR-equilibrium in this case.

While we do not have general necessary and sufficient conditions on the existence of an SR-equilibrium, we show that the objectives previously studied in the context of NSW admit an SR-equilibrium. In the case of budget-additive utilities, we have all $t_j = 1$, and all $b_i = 1$. An ϵ/n-approximate SR-equilibrium satisfies the required accuracy in [38]. Whereas [2] computes an exact SR-equilibrium, an approximate SR-equilibrium is sufficient to obtain a (slightly worse) approximation guarantee.

We show that our algorithmic framework is applicable to compute an ϵ-equilibrium for *budget-SPLC*, the common generalization of the models in [2] and [38]. Using a similar rounding as in [38], we obtain a constant-factor approximation algorithm for maximizing NSW in polynomial-time when agents have budget-SPLC utilities and goods come in multiple copies. The previous algorithm for this setting in [17] runs in pseudopolynomial time. For the special case of additive utilities, [10] gives such an algorithm.

References

1 Nima Anari, Shayan Oveis Gharan, Amin Saberi, and Mohit Singh. Nash social welfare, matrix permanent, and stable polynomials. In *Proceedings of the 8th Innovations in Theoretical Computer Science Conference (ITCS)*, volume 67, page 36. Schloss Dagstuhl–Leibniz-Zentrum fuer Informatik, 2017.

2 Nima Anari, Tung Mai, Shayan Oveis Gharan, and Vijay V Vazirani. Nash social welfare for indivisible items under separable, piecewise-linear concave utilities. In *Proceedings of the 29th annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2274–2290. SIAM, 2018.

3 Kenneth J Arrow, Henry D Block, and Leonid Hurwicz. On the stability of the competitive equilibrium, II. *Econometrica: Journal of the Econometric Society*, pages 82–109, 1959.

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

5 Kenneth J Arrow and Leonid Hurwicz. On the stability of the competitive equilibrium, I. *Econometrica: Journal of the Econometric Society*, pages 522–552, 1958.

6 Kenneth J Arrow and Leonid Hurwicz. Competitive stability under weak gross substitutability: The "Euclidean distance" approach. *International Economic Review*, 1(1):38–49, 1960.

7 Noa Avigdor-Elgrabli, Yuval Rabani, and Gala Yadgar. Convergence of tâtonnement in Fisher markets. *arXiv preprint*, 2014. `arXiv:1401.6637`.

8 Siddharth Barman, Sanath Kumar Krishnamurthy, and Rohit Vaish. Finding fair and efficient allocations. In *Proceedings of the 2018 ACM Conference on Economics and Computation (EC)*, pages 557–574. ACM, 2018.

9 Xiaohui Bei, Jugal Garg, and Martin Hoefer. Ascending-price algorithms for unknown markets. *ACM Transactions on Algorithms (TALG)*, 15(3):37:1–37:33, 2019.

10 Xiaohui Bei, Jugal Garg, Martin Hoefer, and Kurt Mehlhorn. Earning and utility limits in Fisher markets. *ACM Trans. Economics and Comput.*, 7(2):10:1–10:35, 2019.

11 Dimitri P Bertsekas. A new algorithm for the assignment problem. *Mathematical Programming*, 21(1):152–171, 1981.

12 Dimitri P Bertsekas. The auction algorithm for assignment and other network flow problems: A tutorial. *Interfaces*, 20(4):133–149, 1990.

13 Benjamin Birnbaum, Nikhil Devanur, and Lin Xiao. Distributed algorithms via gradient descent for Fisher markets. In *Proceedings of the 12th Conf. Electronic Commerce (EC)*, pages 127–136, 2011.

14 William C Brainard and Herbert E Scarf. How to compute equilibrium prices in 1891. *American Journal of Economics and Sociology*, 64(1):57–83, 2005.

15 Simina Brânzei, Nikhil R. Devanur, and Yuval Rabani. Proportional dynamics in exchange economies. *CoRR*, abs/1907.05037, 2019. `arXiv:1907.05037`.

16 Simina Brânzei, Ruta Mehta, and Noam Nisan. Universal growth in production economies. In *Advances in Neural Information Processing Systems 31: Annual Conference on Neural Information Processing Systems 2018, NeurIPS 2018, 3-8 December 2018, Montréal, Canada*, page 1975, 2018.

17 Bhaskar Ray Chaudhury, Yun Kuen Cheung, Jugal Garg, Naveen Garg, Martin Hoefer, and Kurt Mehlhorn. On fair division for indivisible items. In *Proceedings of the 38th IARCS annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 25:1–25:17. Springer, 2018.

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

19 Yun Kuen Cheung, Richard Cole, and Nikhil R Devanur. Tâtonnement beyond gross substitutes? Gradient descent to the rescue. *Games and Economic Behavior*, 2019.

20 Yun Kuen Cheung, Richard Cole, and Ashish Rastogi. Tatonnement in ongoing markets of complementary goods. In *Proceedings of the 2012 ACM Conference on Electronic Commerce (EC)*, 2012.

21 Yun Kuen Cheung, Richard Cole, and Yixin Tao. Dynamics of distributed updating in Fisher markets. In *Proceedings of the 2018 ACM Conference on Economics and Computation, Ithaca, NY, USA, June 18-22, 2018*, pages 351–368, 2018.

22 Yun Kuen Cheung, Martin Hoefer, and Paresh Nakhe. Tracing equilibrium in dynamic markets via distributed adaptation. In *Proceedings of the 18th International Conference on Autonomous Agents and MultiAgent Systems, AAMAS '19, Montreal, QC, Canada, May 13-17, 2019*, pages 1225–1233, 2019.

23 Bruno Codenotti, Benton McCune, and Kasturi Varadarajan. Market equilibrium via the excess demand function. In *Proceedings of the 37th ACM symposium on Theory of Computing (STOC)*, pages 74–83. ACM, 2005.

24 Bruno Codenotti, Sriram Pemmaraju, and Kasturi Varadarajan. The computation of market equilibria. *Acm Sigact News*, 35(4):23–37, 2004.

25 Bruno Codenotti, Sriram Pemmaraju, and Kasturi Varadarajan. On the polynomial time computation of equilibria for certain exchange economies. In *Proceedings of the 16th annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 72–81. SIAM, 2005.

26 Richard Cole, Nikhil Devanur, Vasilis Gkatzelis, Kamal Jain, Tung Mai, Vijay V Vazirani, and Sadra Yazdanbod. Convex program duality, Fisher markets, and Nash social welfare. In *Proceedings of the 2017 ACM Conference on Economics and Computation (EC)*, pages 459–460. ACM, 2017.

27 Richard Cole and Lisa Fleischer. Fast-converging tatonnement algorithms for one-time and ongoing market problems. In *Proceedings of the 40th ACM symposium on Theory of Computing (STOC)*, pages 315–324. ACM, 2008.

28 Richard Cole and Vasilis Gkatzelis. Approximating the Nash social welfare with indivisible items. *SIAM J. Comput.*, 47(3):1211–1236, 2018.

29 Vincent P Crawford and Elsie Marie Knoer. Job matching with heterogeneous firms and workers. *Econometrica: Journal of the Econometric Society*, pages 437–450, 1981.

30 Gabrielle Demange, David Gale, and Marilda Sotomayor. Multi-item auctions. *Journal of Political Economy*, 94(4):863–872, 1986.

31 Nikhil Devanur, Christos Papadimitriou, Amin Saberi, and Vijay Vazirani. Market equilibrium via a primal–dual algorithm for a convex program. *Jounal of the ACM*, 55(5), 2008.

32 Nikhil R Devanur and Vijay V Vazirani. An improved approximation scheme for computing Arrow–Debreu prices for the linear case. In *Proceedings of the 23rd IARCS annual Conference on Foundations of Software Technology and Theoretical Computer Science (FSTTCS)*, pages 149–155. Springer, 2003.

33 Nikhil R Devanur and Vijay V Vazirani. The spending constraint model for market equilibrium: Algorithmic, existence and uniqueness results. In *Proceedings of the 36th ACM Symposium on Theory of Computing (STOC)*, volume 36, pages 519–528. ACM, 2004.

34 Ran Duan and Kurt Mehlhorn. A combinatorial polynomial algorithm for the linear Arrow-Debreu market. *Information and Computation*, 243:112–132, 2015.

35 Edmund Eisenberg. Aggregation of utility functions. *Management Science*, 7(4):337–350, 1961.

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

37 Lisa Fleischer, Rahul Garg, Sanjiv Kapoor, Rohit Khandekar, and Amin Saberi. A fast and simple algorithm for computing market equilibria. In *Proceedings of the 4th International Workshop on Internet and Network Economics (WINE)*, pages 19–30. Springer, 2008.

38 Jugal Garg, Martin Hoefer, and Kurt Mehlhorn. Approximating the Nash social welfare with budget-additive valuations. In *Proceedings of the 29th annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 2326–2340. SIAM, 2018.

39 Jugal Garg, Edin Husic, and László A. Végh. Auction algorithms for market equilibrium with weak gross substitute demands. *CoRR*, abs/1908.07948, 2019. `arXiv:1908.07948`.

40 Jugal Garg, Edin Husić, and László A. Végh. Approximating Nash social welfare under Rado valuations, 2020. `arXiv:2009.14793`.

41 Jugal Garg and Peter McGlaughlin. Improving Nash social welfare approximations. In *Proceedings of the 28th International Joint Conferences on Artificial Intelligence (IJCAI)*, 2019.

42 Jugal Garg, Ruta Mehta, Vijay V Vazirani, and Sadra Yazdanbod. Settling the complexity of Leontief and PLC exchange markets under exact and approximate equilibria. In *Proceedings of the 49th ACM Symposium on Theory of Computing (STOC)*, pages 890–901. ACM, 2017.

43 Jugal Garg and László A Végh. A strongly polynomial algorithm for linear exchange markets. In *Proceedings of the 51st Symp. Theory of Computing (STOC)*, 2019.

44 Rahul Garg and Sanjiv Kapoor. Auction algorithms for market equilibrium. *Mathematics of Operations Research*, 31(4):714–729, 2006.

45 Rahul Garg and Sanjiv Kapoor. Price roll-backs and path auctions: An approximation scheme for computing the market equilibrium. In *Proceedings of the 2nd International Workshop on Internet and Network Economics (WINE)*, pages 225–238. Springer, 2006.
46 Rahul Garg and Sanjiv Kapoor. Market equilibrium using auctions for a class of gross-substitute utilities. In *Proceedings of the 3rd International Workshop on Web and Internet Economics (WINE)*, pages 356–361. Springer, 2007.
47 Rahul Garg, Sanjiv Kapoor, and Vijay Vazirani. An auction-based market equilibrium algorithm for the separable gross substitutability case. In *Approximation, Randomization, and Combinatorial Optimization. Algorithms and Techniques*, pages 128–138. Springer, 2004.
48 Mehdi Ghiyasvand and James B Orlin. A simple approximation algorithm for computing Arrow–Debreu prices. *Operations Research*, 60(5):1245–1248, 2012.
49 K. Jain and K. Varadarajan. Equilibria for economies with production: Constant-returns technologies and production planning constraints. In *Proceedings of the 17th annual ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 688–697. SIAM, 2006.
50 Sanjiv Kapoor, Aranyak Mehta, and Vijay Vazirani. An auction-based market equilibrium algorithm for a production model. *Theoretical Computer Science*, 378(2):153–164, 2007.
51 Wouter J. Keller. A nested CES-type utility function and its demand and price-index functions. *European Economic Review*, 7:175–186, 1976.
52 Alexander S Kelso Jr and Vincent P Crawford. Job matching, coalition formation, and gross substitutes. *Econometrica: Journal of the Econometric Society*, pages 1483–1504, 1982.
53 Renato Paes Leme. Gross substitutability: An algorithmic survey. *Games and Economic Behavior*, 106:294–316, 2017.
54 Andreu Mas-Colell, Michael Dennis Whinston, Jerry R Green, et al. *Microeconomic theory*, volume 1. Oxford university press New York, 1995.
55 Kiminori Matsuyama and Philip Ushchev. Beyond CES: Three alternative cases of flexible homothetic demand systems. Buffett Institute Global Poverty Research Lab Working Paper No. 17-109, 2017.
56 Yurii Nesterov and Vladimir Shikhman. Computation of Fisher–Gale equilibrium by auction. *Journal of the Operations Research Society of China*, 6(3):349–389, 2018.
57 Noam Nisan, Tim Roughgarden, Eva Tardos, and Vijay V Vazirani. *Algorithmic game theory*. Cambridge University Press, 2007.
58 James B Orlin. Improved algorithms for computing Fisher's market clearing prices: Computing Fisher's market clearing prices. In *Proceedings of the 42nd ACM Symposium on Theory of Computing (STOC)*, pages 291–300. ACM, 2010.
59 Herbert Scarf. Some examples of global instability of the competitive equilibrium. *International Economic Review*, 1(3):157–172, 1960.
60 Vijay Vazirani and Mihalis Yannakakis. Market equilibrium under separable, piecewise-linear, concave utilities. *Jounal of the ACM*, 58(3):10, 2011.
61 Léon Walras. *Éléments d'économie politique pure, ou, Théorie de la richesse sociale*. F. Rouge, 1896.
62 Fang Wu and Li Zhang. Proportional response dynamics leads to market equilibrium. In *Proceedings of the 39th Symp. Theory of Computing (STOC)*, pages 354–363, 2007.
63 Yinyu Ye. A path to the Arrow-Debreu competitive market equilibrium. *Mathematical Programming*, 111(1-2):315–348, 2008.
64 Li Zhang. Proportional response dynamics in the Fisher market. *Theoretical Computer Science*, 412(24):2691–2698, 2011.