Theoretical Computer Science 410 (2009) 1573–1580

Contents lists available at ScienceDirect

Theoretical Computer Science

journal homepage: www.elsevier.com/locate/tcs

# Market equilibria with hybrid linear-Leontief utilities

Xi Chen [a], Li-Sha Huang [a,*], Shang-Hua Teng [b,1]

[a] Institute for Theoretical Computer Science, Tsinghua University, Beijing, China

[b] Computer Science Department, Boston University, Boston, USA

ARTICLE INFO

*Keywords:*
Computational complexity
Market equilibrium
PPAD

ABSTRACT

We introduce a new family of utility functions for exchange markets. This family provides a natural and "continuous" hybridization of the traditional linear and Leontief utilities and might be useful in understanding the complexity of computing approximating market equilibria, although computing an equilibrium in a market with this family of utility functions, this is **PPAD**-hard in general. In this paper, we present an algorithm for finding an approximate Arrow–Debreu equilibrium when the Leontief components of the market are grouped, finite and well-conditioned.



## 1. Introduction

In recent years, the problem of computing and approximating market equilibria has attracted many researchers. In an exchange market, there is a set of traders and each trader comes with an initial endowment of commodities. They interact through some exchange process in order to maximize their own utility functions. In the state of an equilibrium, the traders can simply sell their initial endowments at a determined market price and buy commodities that maximize their utilities. Then, the market will clear — the price is so wisely set that the supplies exactly satisfy the demands. This price is called the *equilibrium price*.

Arrow and Debreu [1] proved the existence of equilibrium prices under some mild conditions. Since then, efficient algorithms have been developed for various families of utility functions.

### 1.1. From linear to Leontief utilities

Two popular families of utility functions are the *linear* and *Leontief* utilities. Both of them can be specified by an $m \times n$ demand matrix $\mathbf{D} = (d_{i,j})$, in an exchange market with $m$ goods and $n$ traders. If trader $j$, where $1 \le j \le n$, receives a bundle of goods $\mathbf{x}_j$, then its linear utility is $u_j(\mathbf{x}_j) = \sum_i x_{i,j}/d_{i,j}$, while its Leontief utility is $u_j(\mathbf{x}_j) = \min_i(x_{i,j}/d_{i,j})$. Both linear and Leontief utilities are members of a larger family of utility functions, referred to as *CES* utilities.

Although the two families of functions look similar, the complexities of computing market equilibria in these two settings might be very different. In the linear case, a market equilibrium can be approximated and computed in polynomial time, thanks to a collection of great algorithmic results by Nenakhov and Primak [12], Devanur, Papadimitriou, Saberi and Vazirani [6], Jain, Mahdian and Saberi [10], Garg and Kapoor [7], Jain [9], and Ye [13].

However, approximating market equilibria with Leontief utilities has proven to be hard, under some reasonable complexity assumptions. In particular, by analyzing a reduction of Codenotti, Saberi, Varadarajan and Ye [4] from Nash

* Corresponding author.
*E-mail addresses:* csxichen@gmail.com (X. Chen), huangstc@gmail.com (L.-S. Huang), steng@cs.bu.edu (S.-H. Teng).

1 Also affiliated with Akamai Technologies Inc. Cambridge, MA, USA.

0304-3975/$ – see front matter 
doi:10.1016/j.tcs.2008.12.030

equilibria to market equilibria, Huang and Teng [8] showed that approximating Leontief market equilibria is as hard as approximating Nash equilibria of general two-player games. Thus by a recent result of Chen, Deng and Teng [2], it is **PPAD**-hard to approximate a Leontief market equilibrium in fully polynomial time. In fact, the smoothed complexity of finding a Leontief market equilibrium cannot be polynomial, unless **PPAD** $\subset$ **RP**.

### 1.2. Hybrid linear-Leontief utilities and our results

In this paper, we introduce a new family of utility functions, and study the computation and approximation of equilibria in exchange markets with these utilities. Our work is partially motivated by the complexity discrepancy between linear and Leontief utilities. In our market model, each trader's utility function is a linear combination of a collection of Leontief utility functions. We parameterize such a utility function by the maximum number of commodities in its Leontief components. If the number of commodities in any of its Leontief components is at most $k$, we refer to it as a *k-wide linear-Leontief function*.

Intuitively, the new utility function combines an "easy" linear function with several "hard" Leontief utility functions. Clearly, a 1-wide linear-Leontief function is a linear function and thus, a market with 1-wide linear-Leontief utilities can be solved in polynomial time. On the other hand, for markets with general linear-Leontief utilities, finding an equilibrium is **PPAD**-hard.

We further focus on *grouped hybridizations* in which the commodities of the exchange market are divided into groups. For any of these groups, each trader has a Leontief utility function over the commodities in the group. A trader's utility function is then the summation of all its Leontief utilities. If each group has at most $k$ commodities, we refer to the utility functions (in the exchange market) as *grouped k-wide linear-Leontief functions*.

An exchange market with grouped linear-Leontief functions can be viewed as a linear combination of several smaller Leontief markets, one for each group of commodities. In an equilibrium, the supplies exactly satisfy the demands in each of these Leontief markets. However, a trader can invest the surplus it earned from one Leontief market to other Leontief markets.

We present two algorithmic results on the computation and approximation of equilibria in markets with hybrid linear-Leontief utilities:

- In Fisher's model, we show that an equilibrium of an exchange market with $n$ traders, $M$ commodities and hybrid linear-Leontief utility functions can be computed in $O(\sqrt{Mn}(M+n)^3L)$ time, where $L$ is the bit-length of the input data.
- We also present an algorithm for finding an approximate equilibrium in a *well-conditioned* Arrow–Debreu market with grouped linear-Leontief functions (see Section 4 for details). While the upper bound that we can prove is exponential, the time complexity of this algorithm is closely related to an interesting sampling problem (see Section 4.2 for details).

In this paper, we only prove the first result for grouped linear-Leontief utilities. It is easy to extend the proof to the general case. In the Arrow–Debreu model, we notice that, due to a recent result of Chen, Deng and Teng [3] on the complexity of sparse two-player games, approximating equilibria in fully polynomial-time is **PPAD**-hard even for Arrow–Debreu markets with 10-*wide* linear-Leontief utilities.

### 1.3. Notations

We will use bold lower-case Roman letters such as $\mathbf{x}$, $\mathbf{a}$, $\mathbf{b}_j$ to denote vectors. Whenever a vector, say $\mathbf{a} \in \mathbb{R}^n$ is present, its components will be denoted by lower-case Roman letters with subscripts, such as $a_1, a_2, \ldots, a_n$. Matrices are denoted by bold upper-case Roman letters such as $\mathbf{A}$ and scalars are usually denoted by lower-case Roman letters. We will also use the following notation in the paper:

- $\mathbb{R}^m_+$: the set of $m$-dimensional vectors with non-negative real entries;
- $\mathbb{P}^n$: the set of vectors $\mathbf{x} \in \mathbb{R}^n_+$ with $\sum_{i=1}^n x_i = 1$;
- $\langle \mathbf{a}|\mathbf{b} \rangle$: the dot-product of two vectors in the same dimension;
- $\|\mathbf{x}\|_p$: the $p$-norm of vector $\mathbf{x}$, that is, $(\sum_i |x_i|^p)^{1/p}$; and $\|\mathbf{x}\|_\infty = \max_i |x_i|$.

## 2. Grouped linear-Leontief markets

Assume there are $n$ traders in the market, denoted by $\mathbf{T} = \{1, \ldots, n-1, n\}$. The market has $m$ groups of commodities, denoted by $\mathbf{G} = \{G_1, \ldots, G_m\}$, and each group $G_j$ contains $k_j$ kinds of commodities.

Trader $i$'s initial endowment of goods is a collection of $m$ vectors: $\{\mathbf{e}^i_j \in \mathbb{R}^{k_j}_+, 1 \le j \le m\}$, where $e^i_{j,k}$ is the amount of good $k$ in group $j$ held by trader $i$. For each group $j$, we use matrix $\mathbf{E}_j = (\mathbf{e}^1_j, \ldots, \mathbf{e}^n_j)$ to denote the traders' initial endowments in this group. We also assume that the amount of each commodity is normalized to 1, i.e., $\sum_{i=1}^n e^i_{j,k} = 1$ for all $j : 1 \le j \le m$ and $k : 1 \le k \le k_j$. Similarly, the allocation to trader $i$ is also a collection of $m$ vectors, denoted by $\mathbf{x}^i = \{\mathbf{x}^i_j \in \mathbb{R}^{k_j}_+, 1 \le j \le m\}$.

Trader $i$'s utility function $u_i$ is described by vectors $\{\mathbf{d}^i_j \in \mathbb{R}^{k_j}_+, 1 \le j \le m\}$ and $\mathbf{a}^i \in \mathbb{R}^m_+$: given an allocation $\mathbf{x}^i = \{\mathbf{x}^i_j, 1 \le j \le m\}$, we have