CS787: Advanced Algorithms Final Project

Topic: Market Equilibria **Presenters:** Holly Esquivel and Chin-Ya Huang

17.1.1 Introduction

We introduce the concept of *Market Equilibria* related to game theory. In the market, there exists sellers which sell divisible goods and buyers who have a desire to purchase some set of goods. Each buyer places a value on each of the goods in his list that relates to the importance of purchasing that specific good. At market equilibrium, no party (seller or buyer) has incentive to deviate from their current position and all goods are sold. This means that no seller has incentive to increase or decrease the price for a specific good, and no buyer has incentive to change what or how much of each good she is buying.

The are several different types of markets that exist today. We will focus our attention on Fisher's Markets, Resource Allocation Markets, and Eisenberg-Gale(EG) Markets. Figure 17.1.1 shows a Venn diagram of some of the markets we will discuss. As you can see EG markets encompass many different types of markets, thus many problems are able to be solved via EG programs including network flow problems. While resource allocation markets (not pictured) allow us to consider combinatorial markets. In later sections, we discuss in detail a flow market over a directed graph where buyers are source-sink pairs.

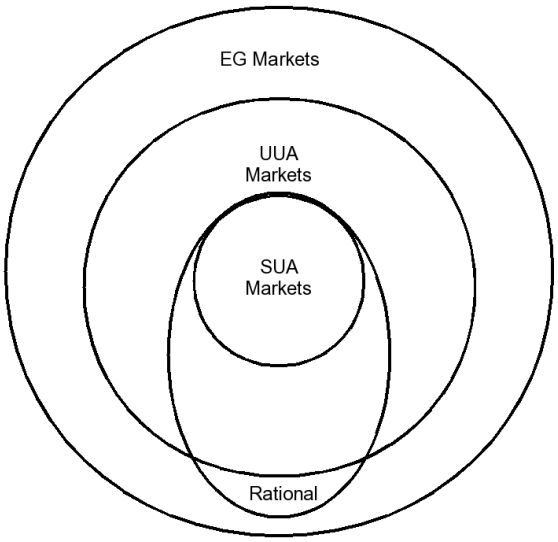

Figure 17.1.1.1: Venn diagram of markets to be discussed

17.1.1.1 Fisher's Market Model

Fisher Markets are widely known in economics and play an important role in game theory. We describe the market setup below that will be used throughout the remainder of this lecture.

- n buyers (Agents), $A = 1, ..., n$
- B is the set of k indivisible goods,
- g_j is the quantity of good j
- $m_1, \cdots, m_n$ is the money possessed by the n buyers
- $u_{i,j}$ is the utility derived by buyer i upon receiving one unit of good j.
- u_i is total utility derived by i
- $x_{i,j}$ is the amount of good j that is purchased by buyer i
- x_i is total bundle of goods purchased by i

In order to find market equilibrium it is necessary to find prices $p_1, ..., p_k$, which maximize the total utility of each buyer in the market. We define the total utility, u_i, of buyer i as $u_i = \sum_{j=1}^{k} u_{i,j} x_{i,j}$. We represent this program as:

$$\max \Pi_{i \in A} u_i^{m_i} \tag{17.1.1.1}$$

In order for an equilibrium to exist we make the following assumption:

Assumption 17.1.1.1 *For every good j, there is a buyer i such that $u_{i,j} > 0$.*

Secondly, we will only consider a market equilibrium in which all goods are sold, and we call the corresponding prices at this equilibrium point *market clearing prices.* The formal definition is as follows:

Definition 17.1.1.2 *A market equilibrium has an optimal price vector, P, such that*
a) demand equals supply for every good, $\forall j \sum_i x_{i,j} = g_j$
b) the utility of buyer i, u_i, must be maximized such that he does not exceed his budget, m_i, thus $\forall i \sum_j p_j x_{i,j} \leq m_i$

We note that equilibrium optional solution can be found to this problem using Fisher's markets but we will describe a solution in Section 17.1.3 which uses a more general EG market program.

17.1.1.2 Game Theoretic Properties

We have presented a market of sellers and buyers which interact with each other in order to come to a stable solution of the presented problem. Solving problems in this iterative fashion between two parties is often known as taking a game theoretic approach to problem solving. Along with game

theory comes the notion of efficiency, fairness, and competition monotonicity. These three metrics are used to define the performance of an algorithm relative to the solution it produces. Efficiency is defined as the ratio of algorithm's solution to the optimal solution. Next, fairness is evaluated in terms of the utility received per unit amount of money spent. This is often referred to as bang-per-buck and must satisfy min-max and max-min fairness. Lastly, competition monotonicity states that increasing the amount of money, m_i, that buyer i has can not increase the utility of the other $n-1$ buyers. We note that all of markets we present can be evaluated based on these metrics, but the results are omitted for brevity.

17.1.1.2.1 Resource Allocation Markets and KKT Conditions

Resource allocation markets are a type of market that can be accurately represented in a EG program. We start by describing how this market is similar to Fisher's Model. We are given a set of finite resources R, where every $r_i \to \mathbb{Z}^+$. Let there be n buyers as before that have a finite amount of money, m_i for each buyer i. The difference is that a buyer wishes to construct as many objects, f_i, as possible, that can be made by requesting different subsets of the resources in R. The value of f_i is calculated as $f_i = \sum_{j=1}^{k_i} x_{i,j}$, where k_i is the number of possible subsets buyer i can use. We say that a feasible solution must not violate any capacity constraints over R. In contrast this equilibrium solution has the property that a resource r_i only has a positive price if it is used to capacity. This problem can be represented as a convex program, which then can be formed into an EG program. The convex program is as follows:

$$\begin{gathered} \max \sum_{a_i \in A} m_i \log f_i \qquad\qquad s.t. \\ \forall a_i \in A : f_i = \textstyle\sum_{j=1}^{k_i} x_{i,j} \\ \forall r \in R : \textstyle\sum_{(ij):r \in S_{ij}} x_{i,j} \leq capacity(r) \\ \forall a_i \in A, 1 \leq j \leq k_i : x_{i,j} \geq 0 \end{gathered} \tag{17.1.1.2}$$

We argue that the EG algorithm will provide the market with an optimal equilibrium solution. In order for this equilibrium to be an optional solution to the convex program in the case of non-linear utilities the solution must satisfy the KKT conditions. Unlike in the linear case, where only the primal or dual variables are necessary to solve complementary slackness conditions, here both are necessary. This is because in EG Markets buyers have a fixed budget and they try to maximize the amount they buy; unlike in LP-based algorithms where buyers have fixed demands and wish to get the desired demand satisfied at the lowest cost. The four KKT conditions relate to stationarity, primal feasibility, dual feasibility and complementary slackness.

Given the convex program described above, if we apply Lagrangian variables to the second set of conditions ($p_r, r \in R$), we can formulate the following the KKT conditions which completes the necessary information to turn this market problem into an EG program. The solutions for $x_{i,j}$'s and p_r must follow these KKT conditions:

1. $\forall r \in R$: $p_r \geq 0$

2. $\forall r \in R : p_r > 0 \Rightarrow \sum_{(ij):r \in S_{ij}} x_{i,j} = capacity(r)$

3. $\forall a_i \in A, 1 \leq j \leq k_i : \frac{m_i}{f_i} \leq \sum_{r \in S_{ij}} p_r$

4. $\forall a_i \in A, 1 \leq j \leq k_i : x_{ij} > 0 \Rightarrow \frac{m_i}{f_i} \leq \sum_{r \in S_{ij}} p_r$

We show a proof that an optimal equilibrium exists for a similar convex program in Section 17.1.3.

17.1.2 Algorithm for Single-Source Multiple-Sinks Markets

Primal-dual based algorithms can be used to satisfy the KKT conditions, and these algorithms can be viewed as executing an auction to find an equilibrium solution. These algorithms can be solved in polynomial time. More specifically, the algorithm is described as an ascending price auction in which the buyers are sinks and sellers are edges. Buyers have fixed budgets and tried to maximize the receiving flows. Sellers try to extract as high price as possible for traffic flowing on their edges. Further, sellers act in a highly coordinated manner - all edges in a particular cut (e.x. $(S, \bar{S})$ raise prices simultaneously while prices of the remaining edges remain unchanged.

The algorithm starts by denoting flow demanded by a sink as $t_i = m_i/\text{rate}(t_i)$. Where $\text{rate}(t_i)$ as the shortest path (least cost) path from a source s to destination t_i. Alternating it adjusts the flows and then the prices, attempting to satisfy all KKT conditions. Initially the prices of all edges are set to 0, and then each sink demands an infinite flow (because of the low cost) which results in an infeasible solution because the demand exceeds supply (i.e. over-saturation). If demand exceeds supply in a cut, the price of all edges in that cut are increased to force sinks to reduce their demands. When the demand exactly equals supply, the edge in this cut stops raising prices, and claim themselves sold at those prices. The detail algorithm for finding the next cut is presented in figure 17.1.2.

Given a graph, $G = (V, E)$ our goal is to find the equilibrium flow and edge prices the graph. An example of single-source multiple-sink markets is unicast traffic. Consider the UW-Madison campus (single source) with students requesting access to pages all over the Internet (multiple sinks), such as facebook.com, gmail.com, google.com, hotmail.com. Following the algorithm above, we note that as the number of iterations increases, different cuts in the graph are found. Thus, eventually all sinks will be satisfied.

Theorem 17.1.2.1 *Flow f and the prices found by the algorithm constitute an equilibrium flow and prices.*

Proof: We show that the solution is an equilibrium by creating graph G', that contains G with an extra sink t_e connected to another sink t, thus there exist edge (t, t_e) between them. The capacity is set to be $m_i/\text{rate}(t_e)$. We let $s \in V$ be the source node and $T = t_1, ..., t_r$ be the set of sink nodes, also called terminals. f is the resulting feasible flow in G after the maximum $s - t$ flow is found in G' and all flow from t_e to t is ignored. We will show that flow f and the prices found satisfy all KKT conditions.
a) Since each of the cuts $(S_i, \overline{S_i} \cup \{t\}$, for $0 \leq i < k$ is saturated in $G^{'}$ by flow $f^{'}$, each of the cuts $c_0, c_1, ..., c_{k-1}$ is saturated by f. Hence, all edges having non-zero prices must be saturated.
b) The cost of the cheapest path to terminal $t' \in T$ is $\text{rate}(t^{'})$. Clearly, every flow to $t^{'}$ uses a pat

of this cost.
c) Since the flow sent to $t' \in T$ is $m_i/\text{rate}(t')$, the money of each terminal is fully spent ■

Theorem 17.1.2.2 *The above algorithm finds equilibrium edge prices and flows using $O(r^2)$ max-flow computations, where r is the number of sinks.*

Proof: We know that the solution is an equilibrium solution by theorem 17.1.2.1. As the number of iterations increase so does the number of cuts. With each cut that is found the number of remaining sinks that must be satisfied is reduced. We note that the most expensive part of the algorithm is the subroutine which takes at most r max-flow computations. We will have to perform this subroutine r, if only one sink is satisfied per cut. Resulting in $O(r^2)$ max-flow computations possible. ■

Inputs: $\text{Cut}(S, \bar{S})$ in G whose price is being raised in the current iteration.
Output: Price $p*$ and next cut $(U, \bar{U})$.

1. $C \leftarrow (V, t)$
2. $p \leftarrow \text{price}(C)$
3. While $\text{cut}(p) \neq C$ do:
   (a) $C \leftarrow \text{cut}(p)$
   (b) $p \leftarrow \text{price}(C)$
4. Output (C, p)

Figure 17.1.2.2: Subroutine for finding the next cut in the graph for Single-Source Multi-Sink Markets

17.1.3 Eisenberg-Gale Markets

Over time, convex programs have been extensively studied. Examples such as scalable utilities, Leontief utilities and homothetic utilities, all fall into this category and can be solved using EG markets since EG markets are a generalization of many markets. Any convex program in the form:

$$\sum_{i \in A} m_i \log u_i$$

is of Eisenberg-Gale-type convex program. These programs can be solved in polynomial time given packing constraints with non-negative coefficients.

Theorem 17.1.3.1 *At market equilibrium the following conditions hold:*

1. *the allocation of goods forms a convex set*
2. *market clearing prices exist*
3. *equilibrium utilities and prices are unique*

Many Fisher markets can be transformed into EG programs. We show such a transformation using the Fisher market described in Section 17.1.1.1 and prove the theorem 17.1.3.1.

$$\max \Pi_{i \in A} u_i^{m_i}$$

$$\begin{gathered} \max \sum_{i \in A} m_i \log u_i \qquad s.t. \\ \forall i \in A : u_i = \textstyle\sum_{j \in B} x_{i,j} u_{i,j} \\ \forall j \in B : \textstyle\sum_{i \in x_{ij}} \leq 1 \\ \forall i \in A, \forall j \in B : x_{i,j} \geq 0 \end{gathered} \tag{17.1.3.3}$$

Next we apply the KKT conditions as described in 17.1.1.2.1.

1. $\forall j \in B$: $p_j \geq 0$
2. $\forall j \in B : p_j > 0 \Rightarrow \sum_{i \in A} x_{i,j} = 1$
3. $\forall a_i \in A, \forall j \in B : \frac{u_{i,j}}{p_i} \leq \frac{\sum_{j \in B} u_{i,j} x_{i,j}}{A_i}$
4. $\forall a_i \in A, \forall j \in B : x_{i,j} > 0 \Rightarrow \frac{u_{i,j}}{p_i} \leq \frac{\sum_{j \in B} u_{i,j} x_{i,j}}{A_i}$

Finally we must prove that there exists clearance prices by reducing the problem.

$$0 < \frac{A_i u_{ij}}{\sum_{j \in B} u_{ij} x_{ij}} \leq p_j$$

$$\frac{u_{ij}}{p_j} x_{ij} = \frac{\sum_{j \in B} u_{ij} x_{ij}}{A_i} x_{ij}$$

$$\frac{A_i u_{ij} x_{ij}}{\sum_{j \in B} u_{ij} x_{ij}} = p_j x_{ij}$$

$$\frac{A_i \sum_{j \in B} u_{ij} x_{ij}}{\sum_{j \in B} u_{ij} x_{ij}} = \sum_{j \in B} p_j x_{ij}$$

which reduces to

$$A_i = \textstyle\sum_{j \in B} p_j x_{ij}$$

Given the above simplification we can see that theorem 17.1.3.1 in fact holds for all three conditions. This concludes the proof.

We note that E-G markets have several interesting properties, such as, the fact that given all rational inputs, the resulting solution has rational solutions. Also, the number of constraints defining M need not be finite and the EG program can still capture the optimal solution.

17.1.3.1 SUA and UUA Markets

Uniform utility allocation and *submodular utility allocation* markets are two submarkets of EG markets. Given a function v as covering closure, we define a market $M(v)$. An instance of $M(v)$ specifies the money possessed by each agent. Let m_i be the money possessed by each agent i under an instance I. Then, the object is to find utilities of buyers, u, and the prices p_j for each subset $S \subseteq A$ such that:

1. $\forall S \subseteq A$: $p_j \geq 0$
2. $\forall S \subseteq A$: $p_j > 0 \Rightarrow u$ makes S tight.
3. $\forall i \in A$: $\frac{m_i}{u_i} = \text{rate}(i)$
   where for $i \in A$, $\text{rate}(i) = \sum_{S: i \in S} p_j$, and this is the rate at which agent i gets one unit of utility.

Further, this market is called **uniform utility allocation market (UUA)**.

Further, Consider the following convex program corresponding to instance I of market $M(v)$.

$$\begin{gathered} \max \sum_{i \in A} m_i \log u_i, \quad s.t. \\ \forall S \subseteq A : \sum_{i \in S} u_i \leq v(s), \\ \forall i \in A : u_i \geq 0 \end{gathered} \tag{17.1.3.4}$$

where $u_{i,j}$ is 0 or 1,

Further, $v(S) + v(T) \geq v(S \cap T) + v(S \cup T)$ for any sets $S, T \subseteq A$

When v is a concave function, $M(v)$ stands for **submodular utility allocation market (SUA)** and the above statement holds. Thus, we obtain a convex program, which we will describe in further detail because of the natural nature of problems to fall into this category especially in economic markets.

The algorithm for solving SUA markets is described in figure 17.1.3.1. Like Single-Source Multi-Sink markets, the SUA algorithm can best be viewed as an ascending price auction. Sellers keep raising their bids until a set of prices is found that results in demand being equal to capacity. At this point it becomes the active set, it declares itself sold and fixes its price. Sellers can not continue to increase their prices if they are under-saturated. The process continues until there are no more sets left which can become active sets indicating that the allocations and prices have been set for the market.

1. $T \leftarrow A$
2. $r \leftarrow 0$
3. While $T \neq \emptyset$
   $p \leftarrow 0$
   Raise p uniformly until a set becomes active. Let $S \subset T$ be a minimal active set.
   $p_T \leftarrow p$
   $r \leftarrow r + p$
   $\forall i \in T - S$: rate$(i) \leftarrow r$
   $T \leftarrow S$
4. $\forall i \in A$: $u_i \leftarrow m_i/\text{rate}(i)$

Figure 17.1.3.3: Algorithm for an SUA Market

17.1.3.2 Fairness and Efficiency of SUA Markets

This section discusses why the algorithm presents an equilibrium solution that is both fair and efficient. Briefly, we conclude with the following Lemmas of SUA about fairness and efficiency, and for more detail, please read [1].

Lemma 17.1.3.2 *The equilibrium for a SUA market is max-min fair and min-max fair. Further, as addressed previously, fairness is considered in game-theoretic markets since all parties seek to maximize their utility. SUA market algorithm forces the set $A = M(I)$ to be tight, thus an efficiency of 1 is obtained in the SUA market. More specifically, efficiency $= 1$ means the total utility by derived by the agents in $M(I)$ is equivalent to the total utility in the optimal solution.*

Lemma 17.1.3.3 *A UUA market has efficiency 1 iff it is a SUA market. This is true because in a UUA market the function v is not guaranteed to be concave, thus it is possible to construct a solution $M(I)$ that would not be tight. Resulting in an efficiency ratio of less than one.*

Lemma 17.1.3.4 *Let g be a feasible utility allocation for v such that $g(S) > v(S) - \delta$, where $\delta > 0$. Then g cannot tighten $S \cup i, j$ This Lemma can be proved using the properties of SUA markets described above and the Lemmas previously stated.*

Lemma 17.1.3.5 *$g(S) > v(S) - \delta$. According to the idea of convex combination $(1 - \lambda)g + \lambda h$, where $\lambda \in [0, 1]$ and the maximization of market equilibrium (i.e. $\sum_{i \in A} m_i \log u_i$), we could prove this Lemma by finding an instance I of market $M(v)$ where equilibrium utilities do not make the set $M(I)$ tight.*

17.1.4 Solving for Market Equilibria using Distributed Algorithms

In prior sections, we have discussed centralized algorithms for solving for market equilibriums. Although many problems can be successfully solved using these centralized algorithms, they do not accurately represent the distributed nature of some problems. In this section, we briefly introduce a

distributed algorithm known as *weak approximate market equilibrium* which decomposes the market equilibrium problem into *indirect utility functions*. Unlike the previously described algorithms, *weak approximate market equilibrium* provides an approximate market equilibrium solution in polynomial time. In particular, it can be solved in almost linear with respect to the number of goods, k, in the system. A full description and proof can be found in [5].

This distributed algorithm allows for buyers and sellers to interact in an iterative fashion. The buyers do not need to submit their utility functions to a central authority, instead sellers directly tell buyers the price of a particular good and buyers respond with the corresponding demand. The algorithm does tell the seller how to adjust their prices according to the supply and demand they are

We will use the same notion for markets as described above with the addition of the following terms:

- $\tilde{u}$ is the indirect utility function of
- K_i is the feasible allocations buyer i may receive
- π is the price vector for the goods
- Δ is a scale value
- α is the degree of the utility function u where $0 \leq \alpha \leq 1$

Definition 17.1.4.1 *(Indirect utility function) For a buyer, i, $\tilde{u_i}$ is the maximum achievable utility for a given price and the money, m_i, that i has. It can be calculated as follows*

$$\tilde{u_i}(\pi, e) = \max u_i(x) | x \in K_i, \pi \cdot x \leq e$$

In order for the algorithm to be applicable, we need to ensure that the indirect utility functions($\tilde{u_i}$) are convex in π. Do ensure this we restrict the prices π to a selected and limited convex set. This allows us to describe our approximate market equilibrium and the associated algorithm 17.1.4.

Definition 17.1.4.2 (Weak $(1+\epsilon)$-approximate market equilibrium) *A price vector $\pi \in \Pi$ and bundles $x_i \in K_i$ for each buyer i are called a weak $(1+\epsilon)$-approximate market equilibrium in the exchange economy if*

1. *The utility-maximizing bundle under prices is $\pi : u_i x_i \geq \tilde{u_i}(\pi, \pi * g_i)$, must hold $\forall i \in A$*
2. *The demand is at most $(1+\epsilon)$ times the supply that is $\sum_i x_i \leq (1+\epsilon) \sum_i g_i$.*
3. *The market must be a clearing market, thus $pi * \sum_i g_i \leq \pi * \sum_i x_i$.*

Although we do not provide proof of this algorithm, it can be found in [5]. We note that this algorithm has several nice properties. It can be solved in polynomial time as well has giving an approximate solution in markets which otherwise would be difficult to solve for a feasible equilibrium solution.

1. Initialize all $p_j = 1$ for $1 \leq j \leq n$

2. Repeat for $N = \frac{n}{\Delta} \log_{1+\Delta} n$ iterations:
(a) Find $\alpha > 0$ such that $\alpha p \in \Pi$. Announce the new prices $\pi = \alpha p$. (b) Let each buyer i compute their demand, i.e. $x_i \in \text{argmax}\, u_i x | x \in K_i, \pi \cdot x \leq \pi \cdot g_i$. (c) Compute the aggregate demand $X = \sum_i x_i$ and let $\sigma = \frac{1}{\max_j X_j}$, where X_j is the aggregate demand for good j (d) Update the price of each good j: $p_j \leftarrow p_j(1 + \Delta \Sigma X_j)$.

3. Output the weighted average allocations for each i: $\bar{x_i} = \frac{\sum_{r=1}^{N} \sigma(r) x_i(r)}{\sum_{r=1}^{N} \sigma(r)}$ where $x_i(r)$ and $\sigma(r)$ the rth iteration values for x_i and σ.

4. Output the weighted average price vector $\bar{\pi} = \frac{\sum_{r=1}^{N} \sigma(r) \pi(r)}{\sum_{r=1}^{N} \sigma(r)}$

Figure 17.1.4.4: Algorithm for the convex program for solving for the $(1 + \epsilon)$-approximate market equilibrium

References

[1] K. Jain, and V. Vazirani. Eisenberg-Gale Markets: Algorithms and Game-Theoretic Properties. In *STOC*, 2007.

[2] D. Chakrabarty, N. Devanur, and V. Vazirani. New Results on Rationality and Strongly Polynomial Solvability in Eisenberg-Gale Markets. In *WINE*, 2006.

[3] B. Ccodenotti, and K. Varadarajan. Efficient Computation of Equilibrium Prices for Markets with Leontief Utilities. In *ICALP*, 2004.

[4] N. Devanur, C.H. Papadimitriou, A. Saberi, and V. Vazirani. Market Equilibrium via a Primal-Dual-Type Algorithm. In *FOCS*, 2002.

[5] L. Fleischer, R. Garg, S. Kapoor, R. Khandekar and A. Saberi. A Fast and Simple Algorithm for Computing Market Equilibria. In *WINE*, 2008.

[6] N. Chen, X. Deng, X. Sun, and A. Yao. Fisher Equilibrium Price with a class of Concave Utility Functions. In *ESA*, 2004.

[7] V. Vazirani. Combinatorial Algorithms for Market Equilibria In *Algorithmic Game Theory*, pages 103-134. Cambridge University Press, 2007.