CS 234r: Markets for Networks and Crowds **Lecture 2**
Pricing, Walrasian Equilibrium, Tâtonnement

Markets and Money

Broadly speaking, a market is defined by some collection of items up for trade, along with the buyers and sellers who want to do the trading. In this lecture, we'll focus on markets with indivisible goods, and in particular on situations where each buyer and seller is interested in trading just a single item. We call this a *matching market*. For example:

- Housing markets
- Assignment of workers to jobs

In a market like this, we can think of each buyer as having preferences over the items. We can represent these preferences as dollar amounts: if a certain buyer values a certain item at \$4, we think of that as meaning that the buyer is indifferent between having the item and having \$4. Given a market and the preferences of all participants, a natural goal is to match buyers with items to maximize the total value of the outcome.

Algorithmically, this is the same as finding a maximum weighted matching in a bipartite graph, which can be done in polynomial time. In principle, we could use such an algorithm to determine market outcomes.

Note: The algorithm will find an efficient matching, but it might be that not every participant is happy with the outcome. If we were to resolve real economic markets using an algorithm for maximum weighted matching, the participants might benefit by lying about their preferences.

Goal: Instead of simply choosing a market outcome, design a market where the participants are individually incentivized to select the optimal matching.

Idea: Use payments!

Combinatorial Markets

Let's formalize these ideas. In a *combinatorial market*, there is a set N of n *agents* (buyers) and a set M of m *goods* (or *items*). There is one indivisible copy of each good. Think: cars. We typically think of each good as being sold by a separate seller, but this won't be important for now — we'll talk about this more in the next lecture.

Def: A *valuation function* assigns a non-negative value to each set of goods: $v : 2^M \to \mathbb{R}_{\geq 0}$. Each agent i has a valuation function v_i. We will assume valuation functions are

- *monotone*: $v(S) \leq v(T)\ \forall S \subseteq T$, and
- *normalized* so that $v(\emptyset) = 0$.

Def: An *allocation* x is a partition of the goods among the agents, with possibly some goods left unallocated. We write x_i for the set of goods allocated to agent i.

Def: The *social welfare* of an allocation x

is $\sum_i v_i(x_i)$. An allocation that maximizes social welfare is said to be *efficient*.

Goal: Find an allocation x that maximizes social welfare. We will call this the *allocation problem*.

Special Case: Matchings

Note: In the special case that each agent can be allocated at most one item, an allocation is precisely a matching. In this case, a valuation function simply assigns a value to each item. The allocation problem is equivalent to finding a maximum-weight bipartite matching.

To see that this is a special case of a combinatorial market, suppose that each valuation function is *unit-demand*. A valuation v is unit-demand if it assigns a value $v(j)$ to each item $j \in M$, and then for any set of items S we have

$$v(S) = \max_{j \in S} v(j).$$

That is, each agent gets value from at most one of the items allocated to him. When valuation functions are unit-demand, it is without loss of generality to allocate at most one item to each agent.

Example:

We will use this running example throughout. There are 3 items, $\{a, b, c\}$, and 3 agents:

- Alice: $v(a) = \$2$, $v(b) = \$3$, $v(c) = \$0$
- Bob: $v(a) = \$0$, $v(b) = \$2$, $v(c) = \$4$
- Charlie: $v(a) = \$0$, $v(b) = \$4$, $v(c) = \$5$

Alice and Bob are both unit-demand agents. (Alice $\leftarrow \{a\}$, Bob $\leftarrow \{c\}$, Charlie $\leftarrow \{b\}$) is an allocation, with social welfare 10.

Walrasian Equilibrium

Left to their own devices, each agent would naturally want to take the item they value the most, but this might cause some items to be overdemanded. To coordinate the agents' preferences, we introduce prices.

Imagine that every good $j \in M$ has a *price* $p_j \geq 0$. If agent i is allocated a set of goods x_i, he must pay $\sum_{j \in x_i} p_j$ in exchange for receiving those goods.

Def: the *utility* of agent i, given that he is allocated set x_i, is $v_i(x_i) - \sum_{j \in x_i} p_j$. We'll write this as $u_i^p(x_i)$.

Each agent wants to maximize his or her own utility.

Def: the *demand correspondence* of agent i at prices p, $D_i(p)$, is the set of utility-maximizing sets of goods. That is, $D_i(p)$ equals

$$\{S \ : \ u_i^p(S) \geq u_i^p(T) \ \forall T \subseteq M\}.$$

Def: a *Walrasian Equilibrium* (also known as a competitive equilibrium, or a price equilibrium) is a choice of item prices p, plus an assignment x, such that

- every agent is allocated a demanded set: $x_i \in D_i(p)$ for all i.
- every item with positive price is sold: if $j \notin x_i$ for all i, then $p_j = 0$.

Example:

Consider our running example. Price vector $(p_a = 0, p_b = 1, p_b = 2)$, along with allocation (Alice $\leftarrow \{a\}$, Bob $\leftarrow \{c\}$, Charlie $\leftarrow \{b\}$), together form a Walrasian Equilibrium.

A Walrasian equilibrium is essentially an assignment of prices to goods that equates

supply and demand. On the demand side, each buyer obtains their most desired set of goods. On the supply side, every good is sold to someone (except possibly if the good is "worthless," in the sense that no agent wants it even if it is free). We say that the market *clears*.

As it turns out, a Walrasian equilibrium not only clears the market, but it does so in a socially efficient way.

Theorem 1 (First Welfare Theorem) *If (x, p) is a Walrasian equilibrium, then x maximizes social welfare.*

Proof: Say (x, p) is a WE, and let y be any other allocation. For each agent i, we know $v_i(x_i) - \sum_{j \in x_i} p_j \geq v_i(y_i) - \sum_{j \in y_i} p_j$. Take a sum over all agents to get

$$\sum_i v_i(x_i) - \sum_i \sum_{j \in x_i} p_j \geq \sum_i v_i(y_i) - \sum_i \sum_{j \in y_i} p_j.$$

Since all items with positive price are allocated under x, we actually have that $\sum_i \sum_{j \in x_i} p_j$ is equal to $\sum_{j \in M} p_j$, which cannot be less than $\sum_i \sum_{j \in y_i} p_j$. So

$$\sum_i \sum_{j \in x_i} p_{x_i} \geq \sum_i \sum_{j \in y_i} p_{y_i},$$

from which we conclude

$$\sum_i v_i(x_i) \geq \sum_i v_i(y_i),$$

so x must be optimal. □

Finding Walrasian Equilibria in Matching Markets

In this section we will restrict our attention to the special case of matching markets. That is, we will assume that all agent valuations are unit-demand. Later we will see how to relax this assumption.

To build some intuition about Walrasian equilibria, here are some simple results; the proofs are left as an exercise. The first result states that any Walrasian price vectors can be paired with any Walrasian allocation to form an equilibrium. (Proof hint: use the first welfare theorem.)

Lemma 2 (Mix-and-Match) *If (x, p) and (y, q) are Walrasian equilibria in a matching market, then (x, q) and (y, p) are as well.*

Because of the Mix-and-Match Lemma, we tend to associate Walrasian equilibria with their price vectors. The following lemma shows that these prices vectors form a complete lattice. In particular, there is a minimum equilibrium and a maximum equilibrium. (Proof hint: use the Mix-and-Match Lemma.)

Theorem 3 *In a matching market, the set of Walrasian price vectors form a complete lattice.*

Example: Consider our running example. The minimal equilibrium is price vector ($p_a = 0, p_b = 1, p_b = 2$), along with allocation (Alice $\leftarrow \{a\}$, Bob $\leftarrow \{c\}$, Charlie $\leftarrow \{b\}$). The maximal equilibrium is ($p_a = 2, p_b = 3, p_b = 4$), with the same allocation.

So now we have some basic understanding of what Walrasian equilibria look like. Can we find one? Does an equilibrium even exist? The following simple procedure is useful for constructing a Walrasian Equilibrium. It is called "Tâtonnement," which translates roughly to "grasping around blindly." The

basic idea is to raise prices on over-demanded items until the market reaches equilibrium.

We will begin by presenting a simple version that modifies prices in discrete jumps, but obtains only an approximate equilibrium.

Def: the δ-approximate demand correspondence of agent i given prices p, $D_i^\delta(p)$, is the set of sets of goods that maximize agent i's utility, up to an additive δ error. That is, $D_i^\delta(p)$ is equal to

$$\{S : u_i^p(S) \geq u_i^p(T) - \delta \ \forall T \subseteq M\}$$

Def: an δ-approximate Walrasian equilibrium is an allocation x and price vector p such that $x_i \in D_i^\delta(p)$ for all i, and all unsold items have price 0.

Algorithm:

1. Let $\delta > 0$ be arbitrarily small.
2. Start with $p_j = 0$ for all $j \in M$, and $x_i = \emptyset$ for all $i \in N$.
3. If $u_i^p(x_i) \geq u_i^p(S) - \delta$ for all i and for all $S \subseteq M$, stop and return (x, p)
4. Choose any i and $S \in D_i(p)$ such that $u_i^p(x_i) < u_i^p(S) - \delta$
5. Choose some $j \in S \backslash x_i$, increase p_j by δ.
6. Allocate S to agent i: set $x_i = S$, and set $x_k = x_k \backslash S$ for all $k \neq i$.
7. Continue on line 3.

What's going on in this algorithm? At any point in time, each buyer i has a provisional allocation x_i, and each item j has a provisional price p_j. Initially the prices are 0 and the allocation is empty. On each iteration, some buyer who is not "approximately satisfied" (condition on line 3) can choose a preferred set of items for themselves at the current prices (line 4). This buyer i can take those items for herself, possibly taking them away from other buyers if necessary (line 6). The "cost" of taking the items is that the price of one of the items taken (i.e., that the buyer didn't already have) goes up by δ (line 5).

Note: We could just as easily change line 5 to increase the price of *every* $j \in S \backslash x_i$ by δ, this wouldn't affect correctness of Tâtonnement. Also, if $S \backslash x_i = \emptyset$, no prices increase.

The above description is more general than matching markets, and one could imagine running Tâtonnement on any combinatorial market. However, Tâtonnement does not always succeed in finding an equilibrium in a combinatorial market; we will see an example of that later. But for the special case of matching markets, Tâtonnement does always converge to a Walrasian equilibrium.

Theorem 4 *In a matching market, the Tâtonnement process must terminate. Moreover, it terminates at a (δ)-approximate Walrasian equilibrium.*

Proof: Prices only rise, and some price rises on each iteration of the algorithm. The algorithm must therefore terminate eventually, since an item whose price is sufficiently high is not included in any agent's demanded set(s).

By line 3, if the algorithm terminates, it must be that $x_i \in D_i^\delta$ for each agent i. So it only remains to show that each unallocated item has price 0. We claim that once an item is allocated, it never again becomes unallocated. This will complete the proof, since

then the only unallocated items are items that were never allocated, and thus never had their prices incremented on line 5.

Suppose for contradiction that the algorithm unallocates some item j on some iteration of lines 3-7; say iteration t. This can only occur if an agent i with $j \in x_i$ is selected in line 4, but $j \notin y_i$. In this case, item j would be unallocated on line 6. However, in a matching market, allocations are singletons; it must therefore be that $x_i = \{j\}$. Moreover, it must be that $\{j\}$ was the allocation assigned to agent i on a previous iteration, say $t' < t$. So, on iteration t', it must have been that $\{j\} \in D_i(p)$. Since the prices of other items can only increase, and p_j increased only by δ on iteration t', $\{j\}$ must be in $D_i^\delta(p)$ on iteration t. This contradicts the fact that agent i was selected on line 4 in iteration t. □

Bonus Material: Exact Walrasian Equilibrium

The Tâtonnement algorithm described in the previous section only finds an approximate Walrasian equilibrium. We can make this approximation arbitrarily good by taking δ as small as desired. In the limit as $\delta \to 0$, this has the net effect of allowing prices to rise continuously. In this limit, the algorithm finds an exact Walrasian equilibrium for matching markets. This argument lets us conclude that an exact Walrasian Equilibrium always exists for matching markets.

To formalize this, suppose (x^δ, p^δ) is the output of the Tâtonnement process for a given choice of δ, say restricted to $\delta \in (0, 1]$. Consider the limit $\delta \to 0$. As there are only finitely many choices of allocations, some allocation x must occur for infinitely many δ. Restrict attention to only those values of δ for which $x^\delta = x$; this produces a subsequence of values of δ with limit 0. For each of these infinitely many δ's there is a price vector p^δ; moreover, these prices vectors lie in a bounded range (since no price can be greater than all agents' values for all items). There must therefore be a subsequence of δ's for which the associated price vectors converge to some vector of limit prices, say p, as $\delta \to 0$. But this then implies that (x, p) is an ϵ-approximate Walrasian equilibrium for all sufficiently small ϵ (since, as $\delta \to 0$, it can be made arbitrarily close to a $(m \cdot \delta)$-approximate equilibrium). Therefore (x, p) is an exact Walrasian equilibrium as well.

Beyond Matching

The Tâtonnement algorithm is defined for arbitrary combinatorial markets, not just matching markets. However, in general, a WE is not always guaranteed to exist. Why does Tâtonnement fail?

Recall our proof that Tâtonnement finds an approximation Walrasian equilibrium for matching markets: we needed to show that once an item is allocated, it is never unallocated. This is true for matching markets, but is not true in general.

Example: Suppose our market has two goods $\{L, R\}$, a left shoe and a right shoe.

- Alice would like to purchase the shoes only as a pair. Her valuation is $v_A(\{L, R\}) = 5$, but $v_A(L) = v_A(R) = 0$.
- Bob is interested in any single shoe (perhaps Bob is a dog). His valuation is $v_B(L) = v_B(R) = v_B(\{L, R\}) = 3$.

In this example, Tâtonnement does not find

a Walrasian equilibrium. Consider the simple Tâtonnement process, with a small price increment δ. Starting at prices 0, Alice will choose both shoes on every iteration, and Bob will choose whichever shoe is cheaper. This will continue until the price of each shoe rises above \$2.50. At this point, Bob will take one shoe from Alice (say the left), raising its price further, and then Alice will discard her remaining shoe and take her demanded set which is $\emptyset$. The resulting allocation is (Alice $\leftarrow \emptyset$, Bob $\leftarrow \{L\}$) at prices (\$2.50+, \$2.50+). This is not an approximate Walrasian equilibrium, since the right shoe is unallocated but has a positive price.

What went wrong in this example? When proving that Tâtonnement finds a WE for matching markets, an important fact was that once a good is allocated, it is never unallocated. In the example above, this is precisely what fails: Alice ultimately discards one of her shoes.

We've argued that Tâtonnement does not find a Walrasian equilibrium. In fact, this is for a good reason: in this market, no Walrasian equilibrium exists. One way to see this is using the first welfare theorem. If we suppose (for contradiction) that a WE exists, it must be socially efficient, so it must allocate both shoes to Alice and no shoes to Bob. Since this is an equilibrium outcome, then Bob must not want either shoe, which means $p_L \geq 3$ and $p_R \geq 3$. But at these prices, the total price of both shoes is at least 6 which is greater than 5. This means that Alice does not actually demand both shoes, since she prefers getting nothing to getting both shoes at those prices. This is a contradiction, so no WE exists.

Ride-share services, Cobweb markets, and the Myth of the Invisible Auctioneer

We've shown that Tâtonnement can fail when goods are not subtitutes. What else can go wrong? One fundamental assumption underlying Tâtonnement is that the market doesn't actually clear until prices have settled at their equilibrium point. It is as if an invisible auctioneer is coordinating the prices, and lets the market know when prices have converged and trade actually occurs. In practice, one might imagine that sales are actually happening concurrently with price adjustment. Should we still expect prices to converge to a competitive equilibrium when that happens?

As it turns out, the way that buyers and sellers respond to prices in the short-run can dramatically influence market convergence. We can illustrate one issue with a toy model of a ride-share platform. In this toy model, there are n passengers in a neighborhood all wanting a ride. Write v_i for passenger i's value for a ride. There are also m drivers in the surrounding area who could potentially travel to that neighborhood to start looking for passengers. Say that driver j has cost c_j to travel to the neighborhood and pick up a passenger.

In this scenario there is only one type of good (a ride), the passengers are potential buyers, and the drivers are potential sellers. Unlike the combinatorial market model above, the sellers have a cost for providing the good. This is straightforward to add to the model: we'll simply add a constraint to the definition of Walrasian equilibrium: each seller is providing their desired set of goods at the given prices. (Our earlier definition of Walrasian equilibrium is then simply the special case where the seller costs are all 0.)

A competitive equilibrium is then simply a price p for which the number of buyers wishing to buy at price p (i.e., with $v_i \geq p$) equals the number of sellers wishing to sell at price p (i.e., with $c_j \leq p$), breaking indifferences arbitrarily. One can view a varying price p (e.g., surge pricing) as attempting to find an equilibrium price for a given supply and demand. However, as we will now see, if passengers and drivers respond to prices myopically then the resulting behavior can be erratic.

Example: Suppose passenger values are $(6, 5, 4, 3)$, and driver costs are $(6, 5, 2, 1)$. Let's say that, initially, all 4 passengers are looking for rides but there is only 1 driver available. The price that clears the market on the buyer side of the market, given the presence of only a single driver, is 6: this is the price at which only one passenger wants a ride (breaking indifference appropriately). There's actually some flexibility here, since any price between 5 and 6 would do, but let's choose 6 for the sake of the example.

When the price is 6, all 4 drivers are interested in providing a ride. So let's suppose all 4 drivers make their way to the currently high-priced neighborhood. But now there are 4 drivers available, and the equilibrium price for the buyers falls to 3. (Again, there is some flexibility here, but let's always choose the highest appropriate price, for the sake of the example.) The drivers with costs 5 and 6 no longer want to provide rides, so they soon leave and go elsewhere. But then, with 2 drivers remaining, the equilibrium price rises back up to 5, which compels one of the drivers to come back. The market has now reached an equilibrium.

The above example shows that prices can fluctuate up and down on their way to finding an equilibrium point. (In fact, one can modify the example so that the price doesn't converge at all, and instead diverges to ever greater extremes!) These types of price oscillation are typical of scenarios in which sellers respond to price changes, which then respond to the quantity of sellers, and vice-versa. This is known as a cobweb model, since prices slowly spiral around the supply and demand curves (i.e., the value and cost vectors). This type of asynchrony can present a barrier to equilibrium, when decentralized markets are left to their own devices. In this example, it would be preferable to dampen the price oscillations suggested by current supply and demand, in the hope of making a "soft landing" at the equilibrium point instead of repeatedly overshooting it.