Combinatorial Algorithms for Linear Fisher Markets
In this talk

- Fisher market model and Arrow-Debreu market model
- Combinatorial algorithm solving linear Fisher market
  - Similar to matching and maximum flow

- Not in exam
Irving Fisher, 1891

- Defined a fundamental market model
- Special case of Walras’ model
Several buyers with different utility functions and moneys.
Several buyers with different utility functions and moneys. Find equilibrium prices.
Linear Fisher Market

- $B = n$ buyers, money $m_i$ for buyer $i$
- $G = g$ goods, w.l.o.g. unit amount of each good
Linear Fisher Market

- $B = n$ buyers, money $m_i$ for buyer $i$
- $G = g$ goods, w.l.o.g. unit amount of each goods
- $u_{ij}$: utility derived by $i$ on obtaining one unit of $j$

® Desirability
Linear Fisher Market

- $B = n$ buyers, money $m_i$ for buyer $i$
- $G = g$ goods, w.l.o.g. unit amount of each good
- $u_{ij}$: utility derived by $i$ on obtaining one unit of $j$
- Total utility of $i$,

$$v_i = \sum_j u_{ij} x_{ij}$$

$x_{ij} \in [0,1]$
Linear Fisher Market

- $B = n$ buyers, money $m_i$ for buyer $i$
- $G = g$ goods, w.l.o.g. unit amount of each good
- $u_{ij}$: utility derived by $i$ on obtaining one unit of $j$
- Total utility of $i$
  \[ v_i = \sum_j u_{ij} x_{ij} \]
  \[ x_{ij} \in [0,1] \]
- Find market clearing prices.
Market clearing allocation

- Budget constraint
  - $\sum_j p_j x_{ij} \leq m_i$
  - The buyer cannot spend more than what he has

- Optimality
  - $x$ maximize $\sum_j u_{ij} x_{ij}$ for every buyer $i$
  - no other bundle of goods that satisfies the budget constraint is more desirable

- Market clearing
  - There is neither deficiency nor surplus of any goods:
    $\forall j, \sum_i x_{ij} = 1$
Prices and utilities

$100

$60

$20

$140

Price

$20

$40

$10

$60

utilities
Bang per buck

- $100
- $60
- $20
- $140

$20 10/20
$40 20/40
$10 4/10
$60 2/60
Bang per buck

- Utility of $1 worth of goods
- Buyers will only buy goods providing maximum bang per buck

\[
\max_{i,j} \frac{u_{ij}}{p_j}
\]
Equality subgraph

$100$ can get utility of $50$
Arrow-Debreu market

- Each agent is assigned a bundle of goods instead of an amount of money
- They sell their own goods to other agents then buy their desirable goods
- Fisher market can be seen as a special case of AD market
  - Money can be seen as a kind of goods
Arrow-Debreu Theorem, 1954

- Established existence of market equilibrium under very general conditions using a deep theorem from topology - Kakutani fixed point theorem.

- Provides a mathematical ratification of Adam Smith’s “invisible hand of the market”.
Need algorithmic ratification!!
Linear Fisher Market

- \( B = n \) buyers, money \( m_i \) for buyer \( i \)
- \( G = g \) goods, w.l.o.g. unit amount of each good
- \( u_{ij} \): utility derived by \( i \) on obtaining one unit of \( j \)
- Total utility of \( i \),
  \[
  v_i = \sum_j u_{ij} x_{ij}
  \]
  \( x_{ij} \in [0,1] \)
- Find market clearing prices.
Can equilibrium allocations be captured via an LP?

- Set of feasible allocations:

\[ \forall j \sum_i x_{ij} \leq 1 \]

\[ \forall i, j \quad x_{ij} \geq 0 \]
Does equilibrium optimize a global objective function?

- **Guess 1:** Maximize sum of utilities, i.e.,

\[
\max \sum_i u_i(x) = \max \sum_i \sum_j u_{ij} x_{ij}
\]

- **Problem:** \(u_i(x)\) and \(2 \times u_i(x)\) are equivalent utility functions.
However,

Maximize $u_1(x) + \sum_{i \neq 1} u_i(x)$ does not necessarily

maximize $2u_1(x) + \sum_{i \neq 1} u_i(x)$

$\forall j \sum_i x_{ij} \leq 1$

$\forall i, j \quad x_{ij} \geq 0$
Guess 2: Product of utilities.

Maximize $u_1(x) \times \prod_{i \neq 1} u_i(x)$

maximizes $2u_1(x) \times \prod_{i \neq 1} u_i(x)$

$\forall j \sum_{i} x_{ij} \leq 1$

$\forall i, j \; x_{ij} \geq 0$
However, suppose a buyer with $200 is split into two buyers with $100 each. And same utility function.

Clearly, equilibrium should not change.
However,

Maximize $u_1(x) \times \prod_{i \neq 1} u_i(x)$ does not necessarily

maximize $u_1(x)^2 \times \prod_{i \neq 1} u_i(x)$

$\forall j \sum_i x_{ij} \leq 1$

$\forall i, j \ x_{ij} \geq 0$
Money of buyers is relevant.

Assume a utility function is written on each dollar in market.
Guess 3: Product of utilities over all dollars

\[
\text{Max } \prod_i u_i(x)^{m(i)}
\]

\[
\forall j \sum_i x_{ij} \leq 1
\]

\[
\forall i, j \quad x_{ij} \geq 0
\]
Eisenberg-Gale Program, 1959

\[
\begin{align*}
\max & \sum_i m(i) \log u_i \\
\text{s.t.} & \\
\forall i : u_i &= \sum_j u_{ij} x_{ij} \\
\forall j : \sum_i x_{ij} &\leq 1 \\
\forall ij : x_{ij} &\geq 0
\end{align*}
\]

This can be solved in polynomial time
However, we want combinatorial algorithms
An easier question

- Given prices $p$, are they equilibrium prices?
- If so, find equilibrium allocations.
At prices $p$, buyer $i$’s most desirable goods, $S_i = \arg \max_j \frac{u_{ij}}{p_j}$

Any goods from $S_i$ worth $m(i)$ constitute $i$’s optimal bundle
For each buyer, most desirable goods, i.e.

\[ S_i = \arg \max_j \left\{ \frac{u_{ij}}{p_j} \right\} \]
Network $N(p)$

The diagram illustrates a network with infinite capacities. The network consists of nodes labeled $t$, $m(1)$, $m(2)$, $m(3)$, $m(4)$, and $S$, with directed edges connecting them. The capacities are denoted by $p(1)$, $p(2)$, $p(3)$, and $p(4)$. The diagram indicates the flow from $t$ to $S$ with infinite capacities.
Max flow in $N(p)$

$p$: equilibrium prices iff both cuts saturated
Idea of algorithm

- “primal” variables: allocations
- “dual” variables: prices of goods

Approach equilibrium prices from below:
- start with very low prices; buyers have surplus money
- iteratively keep raising prices and decreasing surplus
An important consideration

- The price of a good never exceeds its equilibrium price

- Invariant: $s$ is a min-cut
Invariant: $s$ is a min-cut in $N(p)$

$p$: low prices
Idea of algorithm

- Iterations:
  
execute primal & dual improvements

Allocations → Prices

Allocations ↔ Prices
Key Algorithmic Idea

- Dual variables (prices) are raised greedily
- Yet, primal objects go tight and loose
- Balanced Flows: For limiting no. of such events
Max-flow in $N$

W.r.t. max-flow $f$, surplus($i$) = $m(i) - f(i,t)$
Max-flow in $N$

surplus vector = vector of surpluses w.r.t. $f$
Obvious potential function

- Total surplus money $= \ell_1$ norm of surplus vector

- Reduce $\ell_1$ norm of surplus vector by inverse polynomial fraction in each iteration

$$\ell_1(s_1, s_2, \ldots, s_n) = |s_1| + |s_2| + \ldots + |s_n|$$
Balanced flow

- A max-flow that minimizes $l_2$ norm of surplus vector.

- Makes surpluses as equal as possible.

$$l_2(s_1, s_2, \ldots, s_n) = \sqrt{s_1^2 + s_2^2 + \ldots + s_n^2}$$
Balanced flow

- A max-flow that minimizes $l_2$ norm of surplus vector.

- Makes surpluses as equal as possible.

- All balanced flows have same surplus vector.
Our algorithm

- Reduces $l_2$ norm of surplus vector by inverse polynomial fraction in each iteration.
Property 1

- $f$: max-flow in $N$.

- $R$: residual graph w.r.t. $f$.

- If surplus $(i) < \text{surplus}(j)$ then there is no path from $i$ to $j$ in $R$. 
Property 1

\[ R: \]

\[ \text{surplus}(i) < \text{surplus}(j) \]
Property 1

\[ R: \]

\[ \text{surplus}(i) < \text{surplus}(j) \]
Circulation gives a more balanced flow.
Property 1

Theorem: A max-flow is balanced iff it satisfies Property 1.
Algorithm for an iteration

- Construct $N'(I, J)$
- Raise prices in $J$
- New edge enters $N$
- OR a subset in I becomes tight
Network \( N(p) \)

- \( m \) \( \rightarrow \) bang-per-buck \( \leftarrow \) \( p \)
- \( \text{buyers} \) \( \rightarrow \) edges \( \leftarrow \) \( \text{goods} \)
Construct $N'(I, J)$

- Find a balanced flow in $N(p)$
  
  Let $d = \text{max surplus w.r.t. balanced flow}$

- $I = \text{buyers with surplus } d$

- $J = \text{goods desired by } I$

- Raise prices in $J$

- New edge enters $N$

- OR a subset in $I$ becomes tight
Network $N(p)$
Network $N(p)$

The diagram shows a network $N$ with nodes $I$ and $J$, and a transition $N'$. The notation $N - N'$ indicates a change or transition in the network structure.
- Construct $N'(I, J)$

- Raise prices in $J$
  - $N'$ is decoupled from $N - N'$

- New edge enters $N$

- OR a subset in $I$ becomes tight
Network $N(p)$

$N - N'$

$I$

$N'(I, J)$

$J$
Network $N(p)$

By Property 1, this edge did not carry any flow.
Hence Invariant is not violated by its removal.
Raise prices in \( J \)

- proportionately, so that edges in \( N' \) don’t change.

- \( p \cdot x \), for each \( p \) in \( J \)
  - initialize \( x = 1 \)
  - raise \( x \)
- Construct $N'(I, J)$
- Raise prices in $J$
- New edge enters $N$
- OR a subset in $I$ becomes tight
Network $N(p)$
- Construct $N'(I, J)$
- Raise prices in $J$
- New edge enters $N$
  - Recompute balanced flow
  - Buyers in $N - N'$ having residual paths to $N'$ → Move to $N'$
- OR a subset in I becomes tight
Network $N(p)$
Network $N(p)$

$N - N'$

$I, J$

$N'(I, J)$
Network $N(p)$
- Construct $N'(I, J)$

- Raise prices in $J$

- New edge enters $N$
  - Recompute balanced flow
  - Buyers moved to $N'$ will have sufficiently large surplus

- OR a subset in $I$ becomes tight
Algorithm for an iteration

- Construct $N'(I, J)$
- Raise prices in $J$
- New edge enters $N$
- OR a subset in $I$ becomes tight
Tight set: \( p(S) = m(T) \)
Surplus of buyers in $T$ drops to 0
Assume $k$ sub-iterations.

Let $d_0 = d$. At the end of $l^{th}$ sub-iteration,

$$d_l = \min \{ \text{surplus}(i) \mid i \text{ is in } I \}.$$ 
So, $d_k = 0$. 

Network \( N(p) \)

\[ N - N' \]

\[ N'(I, J) \]

\[ I \]

\[ J \]
Construct $N'(I, J)$

Raise prices in $J$

New edge enters $N$

- Recompute balanced flow
- Move buyers in $N - N'$ having residual paths to $N'$
  - will have sufficiently large surplus

OR a subset in $I$ becomes tight
Network $N(p)$

$I$, $J$, $N - N'$, $N'(I, J)$
• Assume $k$ sub-iterations.

• Let $d_0 = d$. At the end of $l^{th}$ sub-iteration,

$$d_l = \min \{ \text{surplus}(i) \mid i \text{ is in } I \}. \text{ So, } d_k = 0.$$

• Decrease in $l_1$ norm in sub-iteration $l$
  is at least $(d_{l-1} - d_l)$

• Decrease in $l_2^2$ norm in sub-iteration $l$
  is at least $(d_{l-1} - d_l)^2$
Our algorithm

- Reduces $l_2$ norm of surplus vector by $1/n^2$ fraction in each iteration

- Polynomial time algorithm
An improved algorithm by Orlin ‘10

- More intuitive
- Scaling algorithm
- Similar to Hungarian algorithm for matchings
The residual network $N(p)$

- In the bipartite graph from buyer to goods

- For each equality edge $(i,j)$, $(i,j) \in N(p)$

- For every $(i,j)$ with $x_{ij} > 0$, there is an arc $(j,i) \in N(p)$
An example
For an augmenting path connecting a buyer and a goods both with surplus
For an augmenting path connecting a buyer and a goods both with surplus
In each scale, we maintain:

- The surplus of every buyer ≥0
- The surplus of every goods between [0, Δ]
- All $x_{ij}$ are multiples of Δ

Our aim in each scale:

- The surplus of every buyer < Δ

After that reduce Δ by one half
- In each phase, start from buyers with surplus $\geq \Delta$
- Find augmenting paths to goods with surplus $\geq \Delta$
If there is no such augmenting paths,

Define the ActiveSet to be the vertices reachable from buyers with surplus $\geq \Delta$

- Multiply the prices of goods in ActiveSet by the same factor $q$
- Until ActiveSet becomes larger or some goods in ActiveSet has surplus $\geq \Delta$
ActiveSet can become larger, since the goods not in it becomes more attractive
- It takes $O(m+n \log n)$ to find an augmenting path from a buyer.

- In each phase, the sum of all surplus is $O(n\Delta)$, so we just need to find $O(n)$ augmenting paths.
  - Because we start from the allocation of the previous phase with $2\Delta$.

- Total running time: $O(n(m+n \log n)(m_{\text{max}} + n \log U_{\text{max}}))$
  - $\Delta$ begins with $m_{\text{max}}/n$, and ends with $1/(8n^2U_{\text{max}})$. 
Arrow-Debreu market

- Each agent is assigned a bundle of goods instead of an amount of money
- They sell their own goods to other agents then buy their desirable goods
- Fisher market can be seen as a special case of AD market
- Still no combinatorial polynomial algorithms!