Auction Algorithms for Market Equilibrium under Weak Gross Substitute Demands

Jugal Garg, Edin Husić, and László Végh

Fisher market

- Set of m divisible goods G.
- Set of n agents A, each with budget b_i.

Goal: find prices such that market clears

supply = demand

- *Supply*: wlog there is one unit of each good $j \in G$,

$$\sum_{i \in A} e_{ij} = 1 \, .$$

Demand and equilibrium

A ***demand*** is a function $D_i : \mathbb{R}_+^{G+1} \to \mathbb{R}_+^G$;

$D_i(p, b_i)$ is the *preferred* bundle of an agent i at prices p and budget b_i.

Bundle = vector of goods.
Preferred or optimal or demanded.

Definition **[Market equilibrium]**:

We say that the prices $p \in \mathbb{R}_+^G$ and bundles $x_i \in \mathbb{R}_+^G$ form a ***market equilibrium*** if

- $x_i = D_i(p, b_i)$, and
- $\sum_{i=1}^{n} x_{ij} \leq 1$ with equality whenever $p_j > 0$, for all $j \in G$.

Tâtonnement

- Informally:
    1. Start with arbitrary prices $p \in \mathbb{R}_{+}^{G}$.
    2. Look at the excess demand (demand - supply) and "fix" the price of a single good:
       change the price of good j until the demand = supply on good j.
    3. Repeat.
- Dynamics for finding market equilibrium.
- Proposed by Walras in 1874 after observations of stock market.

Tâtonnement

- Informally:
    1. Start with arbitrary prices $p \in \mathbb{R}_+^G$.
    2. Look at the excess demand (demand - supply) and "fix" the price of a single good:
       change the price of good j until the demand = supply on good j.
    3. Repeat.
- Dynamics for finding market equilibrium.
- Proposed by Walras in 1874 after observations of stock market.

When does it converge?

Intuition. Suppose one price raises. Then we expect that the demand for that good falls.

Purchasing power is then diverted to the other goods; it is reasonable to assume that demand for other goods increases.

"Increasing a price of a good j will increase the demand for other goods $G \setminus \{j\}$"

(Weak) Gross Substitutes

"Increasing a price of a good j will increase the demand for other goods $G \setminus \{j\}$"

Definition **[Weak gross substitutes]**:

Consider price vectors $p, q \in \mathbb{R}_+^G$ such that $p \leq q$ (pointwise). Demand D_i of agent i satisfies *weak gross substitutes* property if

$$\text{for } x_i = D_i(p, b_i) \text{ and } y_i = D_i(q, b_i) \text{ it holds } y_{ij} \geq x_{ij} \text{ whenever } p_j = q_j.$$

GS if strict inequality holds.

(Weak) Gross Substitutes

"Increasing a price of a good j will increase the demand for other goods $G \setminus \{j\}$"

Definition **[Weak gross substitutes]**:

Consider price vectors $p, q \in \mathbb{R}_+^G$ such that $p \leq q$ (pointwise). Demand D_i of agent i satisfies *weak gross substitutes* property if

$$\text{for } x_i = D_i(p, b_i) \text{ and } y_i = D_i(q, b_i) \text{ it holds } y_{ij} \geq x_{ij} \text{ whenever } p_j = q_j.$$

GS if strict inequality holds.

- Introduced by Arrow, Block and Hurwicz in 1958, 1959.
- They showed that *tâttonement* converges to an equilibrium if *aggregate* demand satisfies **GS**.

$$\sum_{i \in A} D_i(p, b_i)$$

Examples of WGS utilities/demands

Assume agent i is equipped with a concave utility function $u_i : \mathbb{R}^G_+ \to \mathbb{R}_+$, then we have

$$D_i(p, b_i) := D^{u_i}(p, b_i) := \arg\max\{u_i(x_i) : p^\top x_i \le b_i, x \ge 0\}$$

Linear (additive) utility

$$u(x) = v^\top x \quad \text{for } v \in \mathbb{R}^G_+$$

$$D^u(p, b) = \arg\max\{v^\top x : p^\top x \le b\}$$

$$= \{x \in \mathbb{R}^G : \frac{v_j}{p_j} \text{ is maximum, and } p^\top x = b\}$$

$$MBB = \max_{j \in G} \frac{v_j}{p_j}$$

Constant elasticity of substitution

$$u(x) = \left(\sum_j \beta_j^{\frac{1}{\sigma}} x_j^{\frac{\sigma - 1}{\sigma}} \right)^{\frac{\sigma}{\sigma - 1}}$$

$$D^u(p, b) = x \text{ where}$$

$$x_j = \frac{\beta_j p_j^{-\sigma} b}{\sum_k \beta_k p_k^{1-\sigma}}.$$

The demand (utility) is GS iff $\sigma \ge 1$.

The Cobb-Douglas utility

$$u(x) = \prod_j x_j^{\alpha_j} \text{ s.t. } \sum_j \alpha_j = 1, \alpha_j \ge 0$$

$$D^u(p, b) = x \text{ where}$$

$$x_j = b\alpha_j / p_j.$$

Complexity of finding an equilibrium?

For WGS utilities:

- The first polytime algorithm. Codenotti, Pemmaraju, and Varadarajan [2005]
- A simple ascending price algorithm. Bei, Garg, and Hoefer [2019]
- A discrete variant of tâttonement converges to an approximate equilibrium. Codenotti, McCune, and Varadarajan [2005]
- A lot more…

Complexity of finding an equilibrium?

For WGS utilities:

- The first polytime algorithm. Codenotti, Pemmaraju, and Varadarajan [2005]
- A simple ascending price algorithm. Bei, Garg, and Hoefer [2019]
- A discrete variant of tâttonement converges to an approximate equilibrium. Codenotti, McCune, and Varadarajan [2005]
- A lot more…

Outside of WGS:

- In general, hopeless. Finding equilibria when utilities are "just outside" gross substitutability is PPAD-complete. Chen, Paparas, Yannakakis [2013]
- Polynomial time algorithms for particular classes of utilities.

Auction Algorithms

- A subclass of *tâttonement* where prices only go up.
- Under simple “ground rules” the agents outbid each other and converge to an approximate equilibrium.
- Does not require a central authority.
- Robust: small changes allow for various extensions and generalisations.

Auction Algorithms

- A subclass of *tâttonement* where prices only go up.
- Under simple "ground rules" the agents outbid each other and converge to an approximate equilibrium.
- Does not require a central authority.
- Robust: small changes allow for various extensions and generalisations.

- Auction algorithms for *assignment* and *transportation* problems.
  Bertsekas [1981, 1990].
- A long history of auction algorithms for markets with *indivisible goods.*
  Kelso and Crawford [1982], Demange, Gale and Sotomayor [1986].
- Auction algorithm for market equilibrium in exchange market with *linear utilities*.
  Garg and Kapoor [2004]
- Extended to *restricted* subclasses of WGS utilities.
  Garg, Kapoor and Vazirani [2004], Garg and Kapoor [2007].
- **Open:** Design auction algorithm for whole WGS?

Auction algorithm for finding approximate market equilibria in Fisher markets when agents have WGS demands.

Algorithm overview and “ground rules”

- We maintain *market prices* p;
  - Increases only by factor $(1+\epsilon)$.
  - A part $l_j > 0$ of each good is sold at p_j, and the rest is sold at $(1+\epsilon)p_j$. (All goods are fully sold.)

Global

Algorithm overview and “ground rules”

- We maintain *market prices* p;
  - Increases only by factor $(1+\epsilon)$.
  - A part $l_j > 0$ of each good is sold at p_j, and the rest is sold at $(1+\epsilon)p_j$. (All goods are fully sold.)

Global

- Agent i maintains *individual prices* $p^{(i)}$ such that $p_j \le p_j^{(i)} \le (1+\epsilon)p_j$.
  1. Throughout, i owns a bundle c_i such that $c_i \le x_i = D_i(p^{(i)}, b_i)$.
  2. If $p_j^{(i)} < (1+\epsilon)p_j$ agent i pays p_j for the amount c_{ij}.
  3. Otherwise, ($p_j^{(i)} = (1+\epsilon)p_j$) agent i pays $(1+\epsilon)p_j$ for c_{ij}.

Local for agent

Algorithm overview and "ground rules"

- We maintain *market prices* p;
  - Increases only by factor $(1+\epsilon)$.
  - A part $l_j > 0$ of each good is sold at p_j, and the rest is sold at $(1+\epsilon)p_j$. (All goods are fully sold.)

Global

- Agent i maintains *individual prices* $p^{(i)}$ such that $p_j \leq p_j^{(i)} \leq (1+\epsilon)p_j$.
  1. Throughout, i owns a bundle c_i such that $c_i \leq x_i = D_i(p^{(i)}, b_i)$.
  2. If $p_j^{(i)} < (1+\epsilon)p_j$ agent i pays p_j for the amount c_{ij}.
  3. Otherwise, ($p_j^{(i)} = (1+\epsilon)p_j$) agent i pays $(1+\epsilon)p_j$ for c_{ij}.

Local for agent

Consider the agents one-by-one. If agent has surplus, she will try use it to get more goods by outbidding.

Main ingredient

- Throughout, i owns a bundle c_i such that $c_i \leq x_i = D_i(p^{(i)}, b_i)$.

FindNewPrices$(p^{(i)}, c^{(i)}, b_i)$ delivers new prices $\tilde{p}$:

A. $y \geq c_i$ for $y = D_i(\tilde{p}, b_i)$, and

B. $p^{(i)} \leq \tilde{p} \leq (1+\epsilon)p$, where $\tilde{p}_j = (1+\epsilon)p_j$ whenever $y_j > (1+\epsilon)c_{ij}$.

Agent willing to outbid when she wants more

Main ingredient

- Throughout, i owns a bundle c_i such that $c_i \leq x_i = D_i(p^{(i)}, b_i)$.

FindNewPrices$(p^{(i)}, c^{(i)}, b_i)$ delivers new prices $\tilde{p}$:

A. $y \geq c_i$ for $y = D_i(\tilde{p}, b_i)$, and

B. $p^{(i)} \leq \tilde{p} \leq (1+\epsilon)p$, where $\tilde{p}_j = (1+\epsilon)p_j$ whenever $y_j > (1+\epsilon)c_{ij}$.

Agent willing to outbid when she wants more

Can be implemented in various ways:

- Linear (additive) utilities: direct algorithm and/or convex programming approach.
- CES and Cobb-Douglas: solve a convex program.
- WGS demands with bounded elasticity: adjustment procedure.

Algorithm: initialisation

- Initialisation: pick low enough prices so some agents demands all of the goods.
- Algorithm is partitioned into **iterations**;
  each iteration finishes when price of a good increases from p_j to $(1+\epsilon)p_j$ (when $l_j = 0$).
- An iteration is partitioned into **steps:**

  If agent has surplus, she will try use it to get more goods by outbidding.

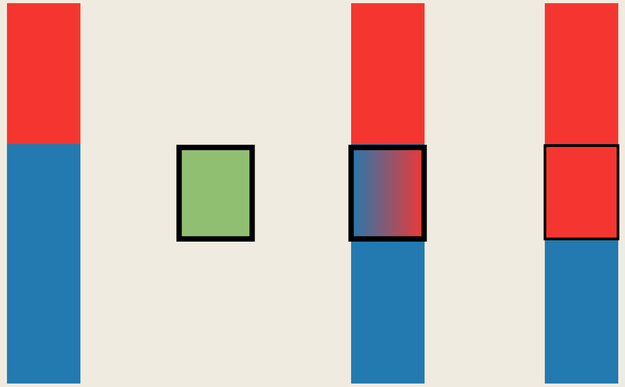
Algorithm: initialisation

- Initialisation: pick low enough prices so some agents demands all of the goods.
- Algorithm is partitioned into **iterations**;
  each iteration finishes when price of a good increases from p_j to $(1+\epsilon)p_j$ (when $l_j = 0$).
- An iteration is partitioned into **steps:**

If agent has surplus, she will try use it to get more goods by outbidding.

Outbid: pay higher price $p_j(1+\epsilon)$ to take a part of j currently sold at p_j.

Goods change the owner only through the outbid.



Algorithm: step of agent i

By invariant (1) agent i owns $c_i \le x_i = D_i(p^{(i)}, b_i)$.
FindNewPrices$(p^{(i)}, c^{(i)}, b_i)$ delivers new prices $\tilde{p}$:

A. $y \ge c_i$ for $y = D_i(\tilde{p}, b_i)$, and

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

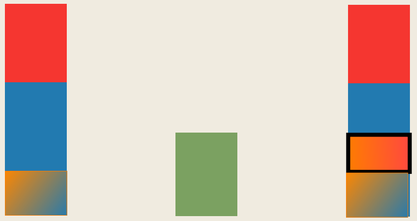
Algorithm: step of agent i

By invariant (1) agent i owns $c_i \le x_i = D_i(p^{(i)}, b_i)$.
FindNewPrices$(p^{(i)}, c^{(i)}, b_i)$ delivers new prices $\tilde{p}$:

A. $y \ge c_i$ for $y = D_i(\tilde{p}, b_i)$, and

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

- $\forall j \in G$ do the corresponding update:
  - $p_j^{(i)} < (1+\epsilon)p_j$ and $\tilde{p}_j = (1+\epsilon)p_j$.
    Agent i starts paying $(1+\epsilon)p_j$ for c_{ij} instead of p_j.
    Then i outbids up to y_j and what is available at p_j from the other agents.



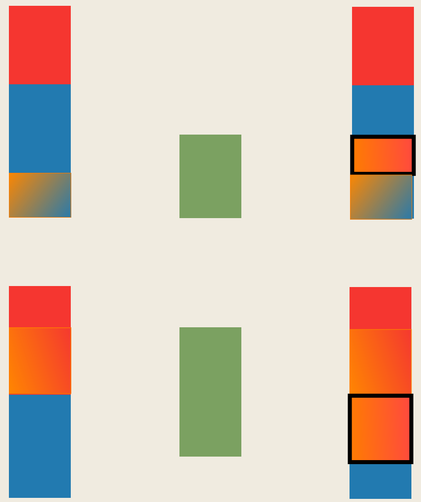
Algorithm: step of agent i

By invariant (1) agent i owns $c_i \le x_i = D_i(p^{(i)}, b_i)$.
FindNewPrices$(p^{(i)}, c^{(i)}, b_i)$ delivers new prices $\tilde{p}$:

A. $y \ge c_i$ for $y = D_i(\tilde{p}, b_i)$, and

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

- $\forall j \in G$ do the corresponding update:
  - $p_j^{(i)} < (1+\epsilon)p_j$ and $\tilde{p}_j = (1+\epsilon)p_j$.
    Agent i starts paying $(1+\epsilon)p_j$ for c_{ij} instead of p_j.
    Then i outbids up to y_j and what is available at p_j from the other agents.
  - $p_j^{(i)} = (1+\epsilon)p_j$ and $\tilde{p}_j = (1+\epsilon)p_j$.
    Agent i keeps paying the higher price $(1+\epsilon)p_j$ for c_{ij}
    and outbids for she wants y_j and can.



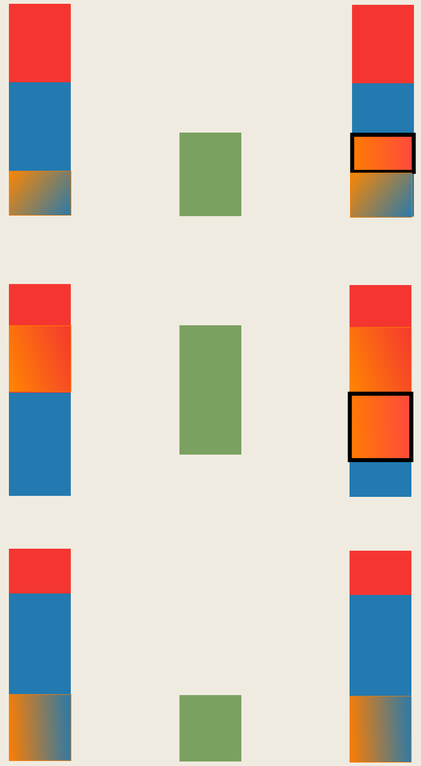
Algorithm: step of agent i

By invariant (1) agent i owns $c_i \le x_i = D_i(p^{(i)}, b_i)$.
FindNewPrices$(p^{(i)}, c^{(i)}, b_i)$ delivers new prices $\tilde{p}$:

A. $y \ge c_i$ for $y = D_i(\tilde{p}, b_i)$, and

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

- $\forall j \in G$ do the corresponding update:
  - $p_j^{(i)} < (1+\epsilon)p_j$ and $\tilde{p}_j = (1+\epsilon)p_j$.
    Agent i starts paying $(1+\epsilon)p_j$ for c_{ij} instead of p_j.
    Then i outbids up to y_j and what is available at p_j from the other agents.
  - $p_j^{(i)} = (1+\epsilon)p_j$ and $\tilde{p}_j = (1+\epsilon)p_j$.
    Agent i keeps paying the higher price $(1+\epsilon)p_j$ for c_{ij}
    and outbids for she wants y_j and can.
  - $p_j^{(i)} < (1+\epsilon)p_j$ and $\tilde{p}_j < (1+\epsilon)p_j$.
    By (B) $c_j^{(i)} \le y_j \le (1+\epsilon)c_j^{(i)}$; the agent will not seek to buy more of j.



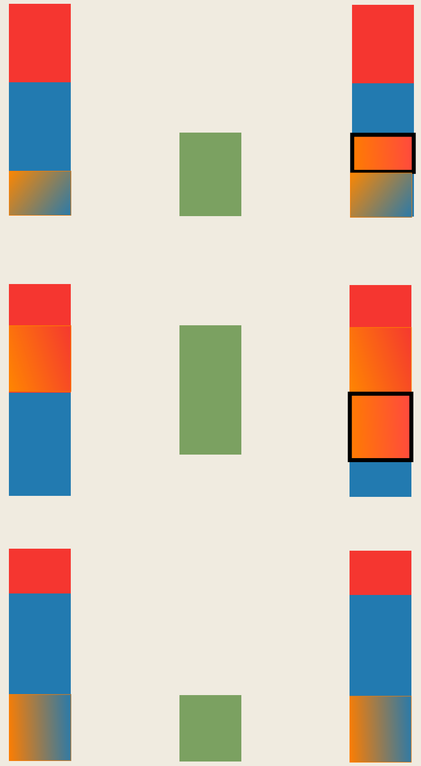
Algorithm: step of agent i

By invariant (1) agent i owns $c_i \le x_i = D_i(p^{(i)}, b_i)$.
FindNewPrices$(p^{(i)}, c^{(i)}, b_i)$ delivers new prices $\tilde{p}$:

A. $y \ge c_i$ for $y = D_i(\tilde{p}, b_i)$, and

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

- $\forall j \in G$ do the corresponding update:
  - $p_j^{(i)} < (1+\epsilon)p_j$ and $\tilde{p}_j = (1+\epsilon)p_j$.
    Agent i starts paying $(1+\epsilon)p_j$ for c_{ij} instead of p_j.
    Then i outbids up to y_j and what is available at p_j from the other agents.
  - $p_j^{(i)} = (1+\epsilon)p_j$ and $\tilde{p}_j = (1+\epsilon)p_j$.
    Agent i keeps paying the higher price $(1+\epsilon)p_j$ for c_{ij}
    and outbids for she wants y_j and can.
  - $p_j^{(i)} < (1+\epsilon)p_j$ and $\tilde{p}_j < (1+\epsilon)p_j$.
    By (B) $c_j^{(i)} \le y_j \le (1+\epsilon)c_j^{(i)}$; the agent will not seek to buy more of j.



Agent either gets y or increases the price of a good!

Running time

Iterations

In each iteration a price increases by factor $(1+\epsilon)$. Price of a good is at most $\sum_{i \in A} b_i$.

At most $O\left(\frac{m}{\epsilon} \log \frac{\sum_i b_i}{p_{\min}}\right)$ prices increases/iterations.

Running time

Iterations

In each iteration a price increases by factor $(1+\epsilon)$. Price of a good is at most $\sum_{i \in A} b_i$.

At most $O\left(\frac{m}{\epsilon} \log \frac{\sum_i b_i}{p_{\min}}\right)$ prices increases/iterations.

Steps

Consider n consecutive steps – **a round**.

Assume the price did not increase, agent i in her turn acquired all she wanted through outbid.

As outbid pays $(1+\epsilon)$ more, the amount of money spent on the goods increases.

Equivalently, the total surplus decreases by factor $(1+\epsilon)$ in each round.

Eventually, we either finish or increase the price.

Recap, comments and applications

- Auction algorithm for finding approximate market equilibria in exchange markets when agents have WGS demands.
- Generalizes to more general **exchange markets**.
- Generalizes to **spending-restricted** market equilibria, recently proposed as a relaxation of the discrete Nash Social Welfare problem.
- Can be extended to markets where WGS is satisfied only approximately.

Thank you!