Arrow-Debreu Model – Linear Case

Arrow-Debreu Model

- A – set of agents; G – set of goods; |A|=m, |G|=n.
- Each agent i comes to market with an initial endowment of goods: (e_{i1},e_{i2}, ..., e_{in})
- May assume wlog that total amount of each good is unit: for $1 \leq j \leq n$, $\sum_{i=1}^{m} e_{ij} = 1$.
- Linear case: Each agent has linear utility for these good: utility of agent i for x_{ij} amount good j: $u_{ij}x_{ij}$
- Problem: Find prices **p**=(p_1,...,p_m) for goods so that if each agent sells initial endowment at these prices and buys optimal bundle, market clears.

Arrow-Debreu Model

- Generalizes Fisher market
- For agent i, let $a_i = \sum_{j=1}^{m} e_{ij}$.
- Let a_{min} = min a_i
- p_{max} : maximum price assigned to a good by an algorithm
- v_{min}, v_{max}: the minimum and maximum utility values u_{ij} over all agents and goods.

Auction-Based Algorithm

- We will discuss auction-based algorithm for the linear case of Arrow-Debreu model
- Will find an approximate equilibrium: will find prices **p** such that the market clears and each agent gets a bundle that provides utility at least $(1-\varepsilon)^2$ the utility of her optimal bundle

Auction-Based Algorithm

- Denote by **p** the vector of prices of goods at any point in the algorithm. Initially, **p**=(1,1,…,1).
- All goods initially unsold
- Each agent is given money = to value of endowment. As **p** changes, the algorithm recomputes the value of each player's initial endowment and updates her money accordingly
- At any point in the algorithm, part of good j is sold at price p_j and part is sold at price $p_j(1 + \varepsilon)$

Auction-Based Algorithm

- Run of algorithm partitioned into iterations
- An iteration terminates with the price of a good being raised by a factor of $(1 + \varepsilon)$
- Each iteration is partitioned into rounds.
- Within a round, the algorithm considers agents one-by-one in some fixed order, say 1,2,...,m.
- If agent *i* has no surplus money, algorithm moves to next agent. Otherwise, find an optimal good at current prices, say good *j*. Then the algorithm executes operation *outbid*

Auction-Based Algorithm

- Outbid: Buy good *j* from agents who have it at price p_j. Sell it to *i* at price $p_j(1 + \varepsilon)$
- This can end in two ways:
  1) Agent i runs out of surplus money. Then the algorithm moves to the next agent.
  2) No agent has good j at price p_j anymore. If so, the algorithm raises the price of good j to $p_j(1 + \varepsilon)$, terminates the iteration, recomputes agents' money and starts the next iteration.
- When current round ends, check if total surplus is less than $\epsilon a_{\min}$; if so, alg. terminates.

Auction-Based Algorithm

- At termination, the algorithm gives the unsold goods to an arbitrary agent to ensure that the market clears.
- Outputs all allocations and terminating prices $\mathbf{p}$
- However, some goods might have been sold at slightly higher price, so agents get only approximately optimal bundles.

Theoretical Guarantees

Lemma 5.23 *The number of rounds executed in an iteration is bounded by*

$$O\left(\frac{1}{\epsilon}\log\frac{np_{\max}}{\epsilon a_{\min}}\right).$$

Lemma 5.24 *The total number of iterations is bounded by*

$$O\left(\frac{n}{\epsilon}\log p_{\max}\right).$$

Theoretical Guarantees

Lemma 5.25 *Relative to terminating prices, each agent gets a bundle of goods that provides her utility at least $(1-\epsilon)^2$ times the utility of her optimal bundle.*

Theorem 5.26 *The algorithm given above finds an approximate equilibrium for the linear case of the Arrow–Debreu model in time*

$$O\left(\frac{mn}{\epsilon^2}\log\frac{nv_{\max}}{\epsilon a_{\min}v_{\min}}\log\frac{v_{\max}}{v_{\min}}\right).$$