Reminder:

Last few lectures: Single item auctions, full information & Bayesian. General mechanism - VCG. (Truthful bidding is dominant)

Next few lectures: Make statements about outcomes in auctions without strenuous calculus using smoothness framework.

**Smooth auctions:**

Set up:

- Outcome $a \in \Omega$
- Payment $p_i$ for player $i$
- Value $v_i(a)$ for each outcome
- Utility (quasi-linear) $u_i(a, p_i) = v_i(a) - p_i$
- Strategy space $S_i$ for player $i$
- $s = (s_1, \ldots, s_n)$ a vector of strategies.
- Outcome function $o: S_1 \times \ldots \times S_n \mapsto \Omega$
- Payment functions $p_i: S_1 \times \ldots \times S_n \mapsto \mathbb{R}$

Remarks: The strategy $s_i$ should be thought of as a set of bids for player $i$ on outcomes, often their willingness to pay. Previous notation for bids that are such "willingness to pay" was $b_i$.

Notation: Let $o(s)$ be the outcome function. Payment, value, utility functions may be written as $p_i(s), v_i(o(s)), u_i(o(s), p_i(s))$, respectively. The rest of the notes will write $v_i(s)$ to mean $v_i(o(s))$ and $u_i(s)$ to mean $u_i(o(s), p_i(s))$ when a mechanism (a tuple of outcome and payment functions) is given.

Example:

1. VCG - outcome: $\arg\max_a \sum_i b_i(a)$.

2. First price auction - outcome: $\arg\max_i b_i$, payment: $p_i = b_i$ if $i = \arg\max_i b_i$, $0$ otherwise.

Approach: Let’s see where we get using utility smoothness. Then we will define a new notion of smoothness for auction games.

**Smoothness, utility maximization games:**
Recall that a utility game is \((\lambda, \mu)\) smooth if \(\exists s^* \text{ s.t } \forall s \sum_i u_i(s^*_i, s_{-i}) \geq \lambda \text{OPT} - \mu \text{SW}(s)\).

Remarks:

- We will regard this as utility smoothness for the rest of these notes.
- \(\text{OPT} = \max_s \sum_i v_i(s)\). Note that \(\text{SW}(s^*)\) is not required to be equal to \(\text{OPT}\).
- \(\text{SW}(s) = \sum_i u_i(s),\) where \(u_i(s) = v_i(s) - p_i(s)\)

It is useful to see how this translates to an auction game. In an auction, the auctioneer is a player with a fixed strategy: to collect the money. His/her utility may be written as \(u_{\text{auctioneer}}(s) = \sum_i p_i(s)\). We add the auctioneer as a player to the utility game.

Translating utility smoothness inequality directly, this is

\[
\sum_i u_i(s^*_i, s_{-i}) + \left( \sum_i p_i(s) \right) \geq \lambda \text{OPT} - \mu \left( \sum_i u_i(s) + \sum_i p_i(s) \right)
\]

Remarks: The sum on \(i\) is over all players excluding the auctioneer.

**Smoothness, auction games:**

Now, in comparison, we define this new notion of smoothness for auction games. (motivation in future lectures)

**Definition.** An auction game is \((\lambda, \mu)\) smooth if \(\exists s^* \text{ s.t } \forall s,\)

\[
\sum_i u_i(s^*_i, s_{-i}) \geq \lambda \text{OPT} - \mu \sum_i p_i(s)
\]

Remarks: Sum on \(i\) is over all players, excluding the auctioneer. This is not that dissimilar to utility smoothness: Assuming \(u_i \geq 0\), we can think of a \((\lambda, \mu)\) smooth auction as \((\lambda, \mu + 1)\) smooth utility game, with the auctioneer added as a player. In future lectures we will see why this new definition of smoothness for auction games is natural.

**Theorem 1.** An auction is \((\lambda, \mu)\) smooth implies a Nash equilibrium strategy profile \(s\) satisfies \(\text{SW}(s) \geq \frac{\lambda}{\max\{1, \mu\}} \text{OPT}\)

**Proof.** Let \(s\) be Nash strategy profile, and \(s^*\) a strategy profile that satisfies smoothness requirements.

Because \(s\) is Nash, \(u_i(s) \geq u_i(s^*_i, s_{-i})\). Summing over all players:

\[
\text{SW}(s) = \sum_i u_i(s^*_i, s_{-i}) + \sum_i p_i(s) \\
\sum_i (u_i(s) + p_i(s)) \geq \sum_i u_i(s^*_i, s_{-i}) + \sum_i p_i(s)
\]
\[ \sum \left( u_i(s) + p_i(s) \right) \geq \lambda \text{OPT} - \mu \sum p_i(s) + \sum p_i(s) \] by auction smoothness

\[ \sum u_i(s) + \mu \sum p_i(s) \geq \lambda \text{OPT} \]

\[ \max\{\mu, 1\} \left( \sum u_i(s) + \sum p_i(s) \right) \geq \lambda \text{OPT} \]

\[ \text{SW}(s) \geq \frac{\lambda}{\max\{1, \mu\}} \text{OPT} \]

Remark: Sum on \( i \) is over all players excluding the auctioneer.

**Generalization to Bayesian Nash:** In general, \( s_i^* \) for player \( i \) is computed with knowledge of other players’ values. In a Bayesian setting, we do not have this information. Restricting \( s_i^* \) such that it only depends on player \( i \)’s value allows us to prove the following theorem:

**Theorem 2.** If an auction is \((\lambda, \mu)\) smooth with an \( s_i^* \) such that \( s_i^* \) depends only on the value of player \( i \), this implies that a Bayesian Nash equilibrium satisfies \( \mathbb{E}[\text{SW}] \geq \frac{\lambda}{\max\{1, \mu\}} \mathbb{E}[\text{OPT}] \)

**Proof.** Idea is to put expectation operator around the proof of Theorem 1.

By definition, a strategy \( s(v) = (s_1(v_1), \ldots, s_n(v_n)) \) is now a function (or a distribution over functions, if randomized), as each player’s strategy depends on his/her own value. If such a function is a Bayesian Nash Equilibrium if \( \mathbb{E}_v[u_i(s_i^*, s_{-i})|v_i] \leq \mathbb{E}_v[u_i(s)|v_i] \), for all strategies \( s_i^* \in S_i \), where values \( v = (v_1, \ldots, v_n) \) is drawn from some distribution. Using this for \( s_i^* \), and taking also expectations over \( v_i \) we get:

\[ \mathbb{E}_v[u_i(s)] \geq \mathbb{E}_v[u_i(s_i^*, s_{-i})] \]

\[ \sum_i \mathbb{E}_v[u_i(s)] \geq \sum_i \mathbb{E}_v[u_i(s_i^*, s_{-i})] \quad \text{summing over players} \]

\[ \mathbb{E}_v\left[ \sum_i u_i(s) \right] \geq \mathbb{E}_v\left[ \sum_i u_i(s_i^*, s_{-i}) \right] \quad \text{linearity of expectation} \]

\[ \mathbb{E}_v\left[ \sum_i u_i(s) \right] \geq \mathbb{E}_v\left[ \lambda \text{OPT} - \mu \sum p_i(s) \right] \quad \text{by smoothness} \]

\[ \mathbb{E}_v\left[ \sum_i u_i(s) \right] + \mathbb{E}_v\left[ \mu \sum p_i(s) \right] \geq \mathbb{E}_v[\lambda \text{OPT}] \]

\[ \mathbb{E}_v[\text{SW}(s)] \geq \frac{\lambda}{\max\{1, \mu\}} \mathbb{E}_v[\text{OPT}] \]

Next time: Examples of auctions that satisfy \((\lambda, \mu)\) smoothness in this framework.