algorithmorithmic Game Theory: Quick reference

vishvAs vAsuki

February 8, 2011

Based on [?].

Part I

Prelude

1 Notation

Pareto improvement: make some p_i better off without penalizing any p_j. (Strong) Pareto optimality.

2 Themes

Finding equilibria in games, making the ideal move.

The complexity of that effort. Natural game strategies to converge to equilibrium. Internet as an equilibrium with unknown game.

Learning in Games.

Mechanism design: make games with players with unknown, private utilities; with desired dominant strategy solutions.

Find the price of anarchy: Total cost (equilibrium with selfish agents) - min Total cost (selfless agents).

2.1 Applications

Any human or AI endeavour involving optimum allocation of resources, maybe using a market.

3 Games: introduction

3.1 Players, strategies, utilities

Players $P = \{p_i\}$. A strategy is not a move but an algorithmorithm to make moves; compare with decision procedure in decision theory in statistics ref. S_i: strategy set of p_i. Strategy vector (strategy profile): $s = (s_1, ..s_n)$. s_{-i}: s sans s_i.

3.1.1 Mixed/ randomized strategies

Independent mixed strategy of i: a Prob Distr over $S_i : D_i$.

Mixed strategy profile, perhaps p_i coordinated: Probability distribution over $\times_i S_i$: D.

3.1.2 Utility

Preference ordering of outcomes for i: Cost, utility of strategy:
$c_i(s) = -u_i(s)$. Compare with risk in decision theory in statistics ref.

3.1.2.1 eps dominated strategy

s_i is ϵ dominated by s_i' if, $\forall s_{-i}$:
$u_i(s_i', s_{-i}) \geq u_i(s_i, s_{-i}) + \epsilon$. Stronly vs weakly dominated strategy. Incomparable strategies: s_i may be better wrt some s_{-i}, but worse wrt some other s_{-i}'.

3.2 Simultaneous move game

Here, strategy = move. Maybe $\forall i, j, S_i = S_j$.

3.2.1 Standard (Explicit) form

Cost (or utility) matrix: $C_{i,j}$: costs (or utility) to P if they select (s_i, s_j). Memory $\Omega(|S_i|^n)$.

3.2.2 Solution concept

Rule for predicting how game will be played. Prediction is **solution** or value. Some solution concepts are better than others for certain games.

4 Solution concepts: examples

4.1 Dominant strategy solution

$\forall i$, $best(s_i)$ unique, independent of s_{-i}: $u(s_i, s'_{-i}) \geq u(s'_i, s'_{-i}) \forall s'$. Eg: Prisoner's dilemma; not: Battle of sexes. So, players individually select strategies. May not lead to optimal payoff for any p_i.

4.1.1 Prisoner's dilemma

Cost matrix $C = \begin{bmatrix} (4,4) & (1,5) \\ (5,1) & (2,2) \end{bmatrix}$. s = (2, 2) best for both, but unstable: If p_1 sets $s_1 = 2$, p_2 tempted to set $s_2 = 1$. $s = (1, 1)$ stable. Optimal selfish strategy of p_1 independent of p_2's actions. Can be extended to multi-player game.

4.1.2 Tragedy of commons

n players; 1 common bandwidth. Strategy about demanding a portion. So, ∞ strategies for each: $s_i \in [0, 1]$. If $\sum s_i > 1$, payoff for all 0; As $\sum s_i$ increases, utility decreases for all; so utility $u_i(s_i) = s_i(1 - \sum_{j \neq i} s_j)$. So, maximizing, best strategy: $s_i = (1 - \sum_{j \neq i} s_j)/2$. Unique stable soln: $s_i = (1 + n)^{-1} \forall i$. $\sum u_i = \frac{n}{(1+n)^2} \approx n^{-1}$; so tragedy. A cartel would be better!

4.1.3 2nd price auction

Aka Vickery. 1 shot Auction of an item: p_i's value for item is v_i; bids s_i; if p_i looses, $u_i = 0$; wins if $s_i = max_j s_j$. If p_i wins, $u_i = v_i - s_j$, where j's bid is next highest. Dominant strategy: $s_i = v_i$! Item auctioned to p_i who values it most: 'socially optimal outcome'.

4.1.3.1 Revelation principle

All p_i can give Game Designer (GD)
valuation function, let GD play game. Maybe need exponential communication for value function. Also, security needed.

4.2 Iterated elimination of str dominated strategies

Take
game matrix. For each player: Amongst the strategies left, eliminate all dominated strategies.

Sometimes left with many incomparable strategies.

Elimination for weakly dominated strategies could lead to elimination of Nash equilibrium strategies.

4.3 Nash equilibrium

Defn: D or $\{D_i\}$ where even if all p_i know all D_i, no treachery profitable. Maybe D not unique. So each p_i can decide D_i if he knows D_{-i}.

4.3.1 Pure strategy

s is Nash equilib if $\forall i, s' : u_i(s_i, s_{-i}) \geq u_i(s'_i, s_{-i})$. Eg: Battle of sexes. Includes dominant strategy solns.

4.3.1.1 (Anti) Coordination games

Battle of the sexes: p_1, p_2 gain when $s_1 = s_2$. Players select strategies together. $C = \begin{bmatrix} (4,5) & (1,1) \\ (2,2) & (5,4) \end{bmatrix}$. Multiple equilibria; Eg: $Pr(s = (1,1)) = 1$.

4.3.2 Randomized (mixed) strategies

Not Pure strategy s, but distr D. Risk neutral p_i maximize $u_i(D) = E_{s\sim D}[u_i(s)]$, with $Pr_{s\sim D}(s) = \prod_i Pr_{s_i\sim D_i}(s_i)$.

D or $\{D_i\}$ is mixed strategy Nash equilib if $\forall i, D' : u_i(D) \geq u_i(D'_i, D_{-i})$. (Check) Eg: Matching pennies. Generalizes pure strategy equilib.

4.3.2.1 Matching pennies

$C = \begin{bmatrix} (1,-1) & (-1,1) \\ (-1,1) & (1,-1) \end{bmatrix}$. A 0 sum game; so p_i selects strategy independently. No stable s; so, p_1 randomizes s_1 to thwart 2.

4.3.3 Existance of Equilibria

(Nash) Any game with $|P|, |S_i|$ finite, $\exists$ mixed strategy Nash equilib. [**Find proof**]

Some games don't have Nash equilibrium.

4.3.4 Properties

If D_i part of Nash equilibrium,
every $t \in D_i$ is a best response to D_{-i}: $E_{D_{-i}}[t, D_i]$ is maximum: else there could've been 0 wt on t.

4.3.5 Time Complexity

Finding Nash equilibria is PPAD complete.

Games representable by digraphs with indegree, outdegree ≤ 1; problem is to find source or sink other than a 'standard source'. Like Lemke - Howson polytope.

[**Incomplete**]

4.3.6 eps Nash equilib

A special case: $\forall i, D' : u_i(D) \geq u_i(D'_i, D_{-i}) - \epsilon$

4.4 Correlated equilibrium D

(Aumann). Coordinator has distr D, samples s from D, tells each p_i its s_i. p_i not told s_j, but knows it is correlated to s_i; so knows all $Pr(s_{-i}|s_i)$. D known to every p_i. D is correlated equilib if it is not in any p_i's interest to deviate from s, assuming other p_i follow instructions:
$E_{s_{-i}\sim D|s_i}[u_i(s_i, s_{-i})] \geq E_{s_{-i}\sim D|s_i}[u_i(s'_i, s_{-i})]$.

Mixed strategy Nash equilibrium is the special case where D_i are independently randomized (with diff coins).

Not well motivated in some games.

Coordinator picks correlated equilibrium by optimizing some fn (Eg: total payoff or avg payoff).

4.4.1 Regret defn

$f_i : S_i \to S_i$, regret $r_i(s, f) = u_i(f_i(s_i), s_{-i}) - u_i(s)$:
$E_{s\sim D}[r_i(s, f_i)] \geq 0$.

4.4.2 eps correlated equilibrium

$E_{s\sim D}[r_i(s, f_i)] \leq \epsilon$.

4.4.3 Traffic light/ Chicken

$C = \begin{bmatrix} (-100,-100) & (1,0) \\ (0,1) & (0,0) \end{bmatrix}$. s = (1, 2) and (2, 1) stable; so coordinator picks one randomly. This correlation increases payoff as the low expected utility mixed strategy $D_i = (101^{-1}, 1 - 101^{-1})$ is avoided.

4.4.4 Reduction to LP

Unknowns: concatenation of vectors $\{D_i\}$. If n players, m strategies each, mn unknowns. Make inequalities: $U_i(D_i, D_{-i}) \geq U_i(D'_i, D_{-i})$; $\|D_i\|_1 = 1$; $D_i \geq 0$.

[**Incomplete**]

4.4.5 Time Complexity

Correlated equilibria form a convex set; so if game specified explicitly, can find one in polynomial time if matrix U given. But finding optimal correlated equilibrium is hard. [**Incomplete**]

5 Games: classes

5.1 2 player games

Aka bimatrix game. Row and column players: p_r, p_c; their Prob distr over S_r and S_c as vectors $:D_r, D_c$. Utility matrix wrt p_r and p_c: R, C.

5.1.1 2 - Player 0 sum games

Utility matrix $A_{i,j} = u_r(..)$. (Nash) Eg: Matching pennies.

Value paid by p_c to p_r: $v_r = D_r^* A D_c$.

Knowing D_r, p_c always selects $min(D_r^* A)$ or finds $v_c = \min_{k_1} \max_{D_r} E_{k_r \sim D_r}[u_1(k_1, k_2)] = \min_{k_1} \max_{D_r} u_1(k_1, D_r)$. So, best strategy for p_r is to maximize (Maxmin) $min(D_r^* A)$.

5.1.1.1 Solution by linear program

$v_r = max\ v; D_r > 0; \sum_i D_{r,i} = 1; (D_r^* A)_j \geq v \forall j.$

Dual of this LP finds $v_c = -v_r$ and D_c: aka Minmax / minimax theorem (Neumann).

5.1.1.2 Reduction from constant (k) sum 2 player games

Use a equivalent utility matrix $A_{i,j} = u_r(..)$. So, any constant sum game has well defined value: $(v_r, k - v_r)$.

5.1.2 Symmetric two player games

R, C are same. D_c is the best response to itself. $D_c = D_r = D$.

5.1.2.1 Finding Nash Equilibrium

(Lemke - Howson). Consider inequalities $Ax \leq 1$, $x \geq 0$; visualize 2d case like an LP: intersecting halfspaces in a plane with axes $\{x_i\}$. $\binom{2n}{n}$ verteces in n-d polytope.

Solution pt must lie in some vertex where payoff is maximum; at solution pt, $\forall i$: $A_i x = 1$ and $i \in D$ by prop of Nash equilib, or $x_i = 0$ and $i \notin D$. To get the final strategy, take x, and scale it so that $\sum x_i = 1$. Move from vertex to vertex by relaxing constraints and moving along edges.

Almost always runs in poly time. (Smoothed complexity.)

5.1.2.2 Reduction from general 2 player games

R and C are $m \times n$. Make symmetric game $\left[\begin{smallmatrix} 0 & R \\ C^T & 0 \end{smallmatrix}\right]$; Find solution distribution $\left[\begin{smallmatrix} x \\ y \end{smallmatrix}\right]$: now, x and y best responses to each other, so solution to original game.

5.2 Games with n turns

5.2.1 Casting as a simultaneous move game

Each p_i picks full strategy from S_i^n. But, $|S_i^n| = |S_i|^n$; so games with turns are a compact representation. Extensive form: Game tree with payoffs at leaves.

5.2.2 Subgame Perfect Nash Equilibria

Nash Equilib with notion of order of moves: Strategy should be Nash even if any prefix of the game is already played.

5.2.2.1 Ultimatum game

p_1, p_2 split money m; 1 turn each: p_1 offers n; p_2 accepts or reject. p_2's interest to accept whatever offered. Cast to a simultaneous move game. Many Nash equilibria: If p_2 rejects if $n < o$, p_1 must offer o. 1 subgame perfect nash equilib.

5.3 Games with partial info about utilities

Work with beliefs about others' properties and preferences.

5.3.1 Bayesian Games

Eg: Card game:
Only distribution of others' cards known.

5.3.1.1 Bayesian first price auction

$\{p_i\}$ have values $\{v_i\}$ for item. If all info available; best strategy for p_i: choose $s_i = v_j$ next lower to v_i. If only distribution of v_k for other players known, p_i bids second E[v_j next lowest to $v_i | v_i$ is max]. [**Find proof**]

5.4 Cooperative games

Groups (G ..) can change strategies jointly.

5.4.1 Strong Nash Equilibrium

In s, $G \subset P$ has **joint deviation** if $\exists s'_G | u_i(s) \leq u_i(s'_G, s_{-A}) \forall p_i \in G$, and for some $p_j \in G, u_i(s) < u_i(s'_G, s_{-A})$.

s is strong Nash if no $G \subset P$ has joint deviation. Similarly, mixed strategy Nash equilibria. Few games have this. Eg: Stable marriage problem.

5.4.2 Stable Marriage problem

[**Incomplete**]

5.4.3 Transferable utility

[**Incomplete**]

5.5 Market equilibria

5.5.1 Pricing and Allocation game with linear utility

Aka Fisher's linear case. Bipartite graph G = (I, B) of goods and buyers: edges indicate interest of $b \in B$ in $i \in I$. Quantity of i scaled to 1; price vector for I: p; money vector for B: m. Utility of i for b: $u_{b,i}$.

Want to find optimal p (pricing) and partition items among B: allocation x. Equilibrium properties: all money, goods exhausted.

Best bang per buck for b: $a_b = max_i \frac{u_{b,i}}{p_i}$: a linear function: so 'linear case'.

Primal dual approach: Start with arbit p = 1; find x; find $\{b\}$ with excess money; adjust price; repeat.

Finding x by reduction to network flow problem: add source s and sink t; connect s to all I and t to all B; set capacities of edges in original graph to be ∞ and on new edges to match a(i) and m(b); thence find c.

5.5.2 Find best allocation

(Arrow Debreu) Agents come in with goods portfolio, utilities for various goods, leave with goods: money only inbetween. Generalizes Fisher's linear case: .

Auction based approx algorithm solves it: Market clears approximately.

5.6 Repeated games with partial info about utilities

p_1 in uncertain environment (p_{-1}); utilities of p_{-1} not known. Eg: Choosing a route to go to school.

5.6.1 Model

Same game repeated T times; At time t: p_1 uses online algorithm H to pick distr $D_H^{(t)}$ over S_1. p_1 picks action $k_1^{(t)}$ from $D_H^{(t)}$. Loss/ cost function for p_1: $c_1 : \times_i S_i \to [0,1]$. $c_1^{(t)}(k_1^{(t)}) := c_1(k_1^{(t)}, D_{-1}^{(t)})$, $c_1(D) := E_{x \sim D}[c_1(x)]$.

5.6.1.1 Model with full info about costs

H gets cost vector $c_1^{(t)} \in [0,1]^{|S_1|}$, pays cost
$c_1(D_H^{(t)}, D_{-1}^{(t)}) = E_{k_1^{(t)} \sim D_H^{(t)}}[c_1(k_1^{(t)}, D_{-1}^{(t)})] = E_{k_1^{(t)} \sim D_H^{(t)}}[c_1^{(t)}(k_1^{(t)})]$.

Total loss for H: $L_H^{(T)} = \sum c_1(D_H^{(t)}, D_{-1}^{(t)})$.

5.6.1.2 Model with partial info about costs

Aka Multi Armed Bandit (MAB) model.
p_1 (or H) pays cost for $k_1^{(t)}$: $c_1(k_1^{(t)}, D_{-1}^{(t)}) = c_1^{(t)}(k_1^{(t)})$.

Total loss for H: $L_H^{(T)} = \sum c_1(k_1^{(t)}, D_{-1}^{(t)})$.

5.6.1.3 Goal

Minimize $\frac{L_?^{(T)}}{T}$. Maybe other p_i do the same. $D_{-1}^{(t)}$ and $c_1^{(t)}$ can vary arbitrarily over time; so, model is adversarial.

5.6.2 Best response algorithm

For every i: Start with s; suppose s_{-i} fixed, do hill climbing by varying s_i.

5.6.3 Regret analysis

H incurs loss $L_H^{(T)}$; p_1 sees simple policy π would have had much lower loss. Comparison class of orithms G. π best algorithm in G: $L_\pi^{(T)} = min_{g\in G} L_g^{(T)}$. Regret $R_G = L_H^{(T)} - L_\pi^{(T)} = max_{g\in G}(L_H^{(T)} - L_g^{(T)})$.

5.6.3.1 Goal

Minimize R_G.

5.6.3.2 Regret wrt all policies: Lower bound

$G_{all} = \{g : T \to S_1\}$: $\exists$ sequence of loss vectors $c_1^{(t)}$: $R_{G_{all}} \geq T(1 - |S_1|^{-1})$:

For $k' = argmin_{k_1^{(t)}} Pr_{D_H^{(t)}}(k_1^{(t)})$, $c_1^{(t)}(k') = 0$, for others,
$c_1^{(t)}(k_1^{(t)}) = 1$; $\min_{k_1^{(t)}} Pr_{D_H^{(t)}}(k_1^{(t)}) \leq |S_1|^{-1}$.

So, must restrict G.

5.6.4 External regret

Aka Combining Expert Advice. $G = \{i^T : i \in S_1\}$, policies where all $k_1^{(t)}$ are the same; π is best single action. $L_\pi^{(T)} = \sum c_1(\pi, D_{-1}^{(t)})$.

If H has low external regret bound: H matches performance of offline algorithm. [**Find proof**]H comparable to optimal prediction rule from some large hyp class H. [**Find proof**]

5.6.4.1 Deterministic Greedy (DG) algorithm

$S_1^{(t-1)} = \left\{i : argmin_{i \in S_1} L_i^{(t-1)}\right\}$,
$k_1^{(t)} = \min_{i \in S_1^{(t-1)}} i$. $L_{DG}^{(T)} \leq |S_1| min_i(L_i^{(T)}) + (|S_1| - 1)$: Suppose $c_1^{(t)} \in \{0,1\}^{|S_1|}$. For every increase in $\min_i L_i^{(t)}$, max loss $|S_1|$: For $L_{DG}^{(t)} = L_{DG}^{(t-1)} + 1$ but $\min_i L_i^{(t)} = \min_i L_i^{(t-1)}$: $S_1^{(t)} \subseteq S_1^{(t-1)}$; so count num of times $S_1^{(t)}$ can shrink by 1.

5.6.4.2 Deterministic algorithm Lower bound

For any deterministic online algorithm H', $\exists$ loss seq where $L_{H'}^{(T)} = T, min_{i \in S_1}(L_i^{(t)}) \leq \lfloor T/|S_1| \rfloor$: $c_1^{(t)}(k_1^{(t)}) = 1$, for other i, $c_1^{(t)}(i) = 0$; so $L_{H'}^{(T)} = T$; some action used by H' $\leq \lfloor T/|S_1| \rfloor$ times; so $min_{i \in S_1}(L_i^{(t)}) \leq \lfloor T/|S_1| \rfloor$.

So find rand algorithm.

5.6.4.3 Rand Weighted majority algorithm (RWM)

Suppose $c_1^{(t)} \in \{0,1\}^{|S_1|}$. Treat S_1 as a bunch of experts: Want to put as much wt as possible on best expert. Let $|S_1| = N$. Init weights $w_i^{(1)} = 1$, total wt $W^{(1)} = N$, $Pr_{D_H^{(1)}}(i) = N^{-1}$.

If $c_1^{(t-1)}(i) = 1$, $w_i^{(t)} = w_i^{(t)}(1-\eta)$, $Pr_{D_1^{(t)}}(i) = \frac{w_i^{(t)}}{W^{(t)}}$. [**Find proof**]Like analysis of mistake bound of panel of k experts in colt ref.

For $\eta < 2^{-1}$, $L_H^{(T)} \leq (1+\eta)\min_{i \in S_1} L_i^{(t)} + \frac{\ln N}{\eta}$. Any time H sees significant expected loss, big drop in W. $W^{(T+1)} \geq max_i w_i^{(T+1)} = (1-\eta)^{\min_i L_i^{(T)}}$. [**Incomplete**]

For $\eta = \min\left\{\sqrt{\ln N/T}, 2^{-1}\right\}$: $L_H^{(T)} \leq \min_i L_i^{(T)} + 2\sqrt{T \ln N}$. If T unknown, use 'guess and double' with const loss in regret. [**Find proof**]

5.6.4.4 Polynomial weights algorithm

Extension of RWM to $c_1^{(t)} \in [0,1]^{|S_1|}$. Wt update is $w_i^{(t)} = w_i^{(t)}(1 - \eta c^{(t-1)}(i))$. $L_H^{(T)} \leq \min_i L_i^{(T)} + 2\sqrt{T \ln N}$. [**Find proof**]

5.6.4.5 Rand algorithm Lower bounds

If $T < \log_2 N$: For any online algorithm H, $\exists$ stochastic generation of losses: $E[L_H^{(T)}] = T/2$, but $\min_i L_i^{(t)} = 0$: at t=1 let N/2 actions get loss 1; at time t: half the actions which had a loss 0 at time t-1 get loss 1; so, probability mass on actions with $0 = 2^{-1}$.

If N=2, $\exists$ stochastic generation of losses: $E[L_H^{(T)} - \min_i L_i^{(T)}] = \Omega(\sqrt{T})$. [**Find proof**]

5.6.4.6 Convergence to equilibrium: 2 player constant sum repeated game

All p_i use algorithm H with external regret R; Value of game: (v_i). Avg loss: $\frac{L_H^{(T)}}{T} \leq v_i$. [**Find proof**]If $R_G = O(\sqrt{T})$, convergence to v_i.

5.6.5 Low external regret algorithm in partial cost info model

Exploration vs exploitation tradeoff in algorithms.

algorithm MAB: Divide time T into K blocks; in each time block $\tau + 1$: explore and get cost vector: execute action i at random time to get vector of RV's: $\hat{c}^{(\tau)}$, also exploit: use distr $D^{(\tau)}$ as strategy; pass $\hat{c}^{(\tau)}$ to full info external regret algorithm F with ext regret $R^{(K)}$ over K time steps; get distr $D^{(\tau+1)}$ from F.

Max Loss during exploration steps: NK. RV for total loss of F over K time blocks: $\hat{L}_F^{(T)} = \frac{T}{K}\sum_\tau p^\tau c^\tau \leq \frac{T}{K}(min_i \hat{L}_i^{(K)} + R^{(K)}$. Taking expectation, $L_{MAB}^{(T)} = E[\hat{L}_{MAB}^{(T)}] = E[\hat{L}_F^{(T)} + NK] \leq \frac{T}{K}(E[min_i\hat{L}_i^{(K)}] + R^{(K)}) + NK \leq \frac{T}{K}(min_i E[\hat{L}_i^{(K)}] + R^{(K)}) + NK \leq min_i L_i^{(T)} + \frac{T}{K}R^{(K)} + NK$.

Using the $O(\sqrt{K \log N})$ algorithm, with $K = (\frac{T}{K}R_K)$, we get $L_{MAB}^{(T)} \leq min_i L_i^{(T)} + O(T^{2/3}N^{1/3}\log N)$.

5.6.6 Swap regret

Comparison algorithm (H,g) is H with some swap fn $g: S_1 \to S_1$.

5.6.6.1 Internal regret

A special case: Swap every occurance of action b_1 with action b_2. Modification fn: $switch_i(k_i, b_1, b_2) = k_i$ except $switch_i(b_1, b_1, b_2) = b_1$.

5.6.6.2 Low Internal regret algorithm using external regret minimization algorithms

Let $N = |S_i|$; $(A_1, .., A_N)$ copies of algorithm with external regret bound R. Master algorithm H gets from A_i distr $q_i^{(t)}$ over S_i; makes matrix $Q^{(t)}$ with $q_i^{(t)}$ as rows; finds stationary distr vector $p^{(t)} = p^{(t)}Q^{(t)}$: Picking $k_i \in S_i$ same as picking A_j first, then picking $k_i \in S_i$; gets loss vector $c^{(t)}$; gives A_i loss vector $p_i^{(t)}c^{(t)}$.

$\forall j: L_{A_i} = \sum_t p_i^{(t)}\langle c^{(t)}, q_i^{(t)}\rangle \leq \sum_t p_i^{(t)}c_j^{(t)} + R$. Also, Sum of percieved losses = actual loss. So, for any swap fn g, $L_H^T \leq \sum_i\sum_t p_i^{(t)}c_{g(i)}^{(t)} + NR = L_{F,g}^{(T)} + NR$.

Thence, using polynomial weights algorithm, swap regret bound $O(\sqrt{|S_1|T\log|S_1|})$.

5.6.6.3 Convergence to Correlated equilibrium

Every p_i uses strategy with swap regret $\leq R$: then empirical distr Q over $\times_i S_i$ is an $\frac{R}{T}$ correlated equilibrium. $R = L_H^{(T)} - L_{H,g}^{(T)} = \sum_t E_{s^{(t)} \sim D^{(t)}}[r_i(s,g)] = TE_{s \sim Q}[r_i(s,g)]$.

Convergence if all players have sublinear swap regret.

5.6.6.4 Frequency of dominated strategies

p_1 uses algorithm with swap regret R over time T; w: avg over T of prob weight on ϵ dominated strategies; so $\epsilon wT \leq R$; so $w \leq \frac{R}{T\epsilon}$.

If algorithm minimizes external regret using polynomial weights algorithm, freq of doing dominated actions tends to 0.

6 Mechanism design

6.1 Mechanism design

Engineer / implement social choice function. Mech design in making protocols for computer network problems: algorithmorithmic mechanism design. Electronic market design: mech design in electronic markets.

Set of alternatives A. L: set of all linear orders over A. Preference order of $p_i :>_i$.

6.1.1 Social welfare function

$F : L^n \to L$.

6.1.1.1 Properties

Unanimity: $F(<^n) =<$. Dictatorship. Independence of irrelevant attributes: $F((<_i))$ between alternatives a_1 and a_2 depends only on $\{a_1 <_i a_2\}$.

(Arrow): Every social welfare fn over A with $|A| > 2$ that satisfies unanimity and independence of irrelevant alternatives is a dictatorship.

6.1.2 Social choice function

$f : L^n \to A$. Eg: Elections; markets: allocation of goods and money; auctions; government policy; runs of network protocols.

6.1.2.1 Voting methods

Ways of finding outcome of multicandidate (> 2) elections.

Majority vote won't work: Condorcet paradox. Strategic voting: p_i lies about his preference in order to get some one to win.

6.1.2.2 Strategic manipulation

Incentive compatible mechanism: No p_i can be better off by lying about his prefs.

6.1.3 VCG mechanism

Maximizes social welfare: $\sum_i u_i(a)$. A general scheme: fix specific functions to suit need.

6.2 Auctions

1st price auction. 2nd price auction. Generalized 2nd price auction: winner pays a price between 1st price and 2nd price.

6.2.1 Combinatorial auctions

Each p_i has a utility for every subset of goods: $u_i(S) : S \subseteq A$.

6.2.2 Search auctions

How to order the list of ads? Payment per click u_i. Clicks p_i get: n_i. Find $argmax_i u_i$. Or find $argmax_i n_i u_i$.

[**Incomplete**]

6.3 Prediction markets

Markets whose purpose is to find a probability. People who buy low and sell high are rewarded for improving the prediction, those who buy high and sell low are punished for degrading it.

Bibliography