Lecture 1 Scribe Notes

Instructor: Eva Tardos *Costandino Dufort Moraites (cdm82)*

1 Lecture 1 – Monday 23 January 2012 - Mostly Intro

1.1 Overview

Special - possible extra credit for improving wikipedia articles on topics related to this course.

Motivating connecting Game Theory and Computer Science.

- The Internet. Computer Science used to be summarized by trying to make computers effective. In comparison, today, there is the Internet connecting everything and the idea that one person designing one algorithm controls everything is just wrong.
- Mechanism design. Really an engineering discipline, designing a game to get a desired outcome.

Different perspectives. Ways in which our perspective is different from the traditional approaches to game theory.

- We care about algorithms and as mentioned previously we know how to design and analyse them. Economists come from a different historical perspective when it comes to approaching these problems and there isn't the same emphasis on the complexity of finding the equilibria of a system. From the Computer Science perspective, taking up this approach brings us to interesting complexity results which we'll explore throughout the semester.
- Simplicity of mechanisms. Particularly coming from a systems background, mechanisms must be simple.
- (Approximate) optimality. Say average response or delivery time. Often times, our objective functions don't have such clear cut measures of utility. Time can be relative. Do we care about the difference between 7 and 8 seconds?

See the web page for a rough syllabus for the course.

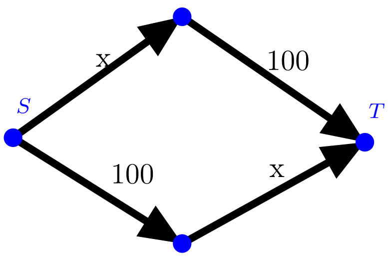
Since the book, mechanism design has changed a bit. We will be using Jason Hartline's book on this topic more than the book listed as the text for this course.

Ken Binmoore basic introduction to game theory is a good, quick intro to game theory.

1.2 An Example – Braess' paradox

We'll start by talking about Braess' paradox. Consider the following graph.

Strategy 2



Strategy 1

For every edge there is $d_e(x)$ = delay on e if x players use this edge. We will assume $d_e(x)$ is a monotone increasing (i.e. non-decreasing) function. Here strategy 1 is the bottom path and strategy 2 is the top path.

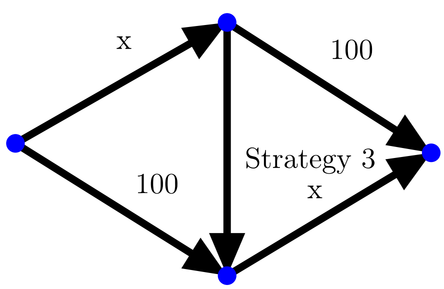
Notice the two edges with weight 100. These edges take 100 seconds to cross regardless of congestion, whereas the edges labelled x are congestion sensitive.

Our goal: predict what will happen here. The obvious prediction is that the players should split half-half. So 50 choose strategy one and another 50 choose strategy two. This prediction is a *(pure) Nash equilibirium* - an outcome so that everyone chooses a strategy such that deviating from their strategy will not improve their outcome. Total time is 150 for all players. If someone following one strategy tries to deviate, they will not improve their outcome. In this example, someone switching from strategy one to strategy two will increase their travel time by a second worsening their outcome.

Now we add an extra edge with no delay and see how this affects the strategies. Adding the extra edge provides an additional strategy: strategy 3 where a player starts out using the top edge, then uses the edge we just added and then takes the last edge to the sink.

Start with the 50-50 solution, but this is no longer an equilibrium. Now we claim that all 100 choosing strategy 3 is a Nash equilibrium with total delay 200 for all players.

Strategy 2



Strategy 1

To see this, notice that a player taking strategy 1 or strategy 2, then changes his delay time from 200 to 200 so no improvement, thus we have the desired equilibrium.

This equilibrium is not unique. 99 could choose strategy 3 and 1 could choose strategy 1. This would still be at equilibrium. The lonely guy might be jealous of his 99 friends for getting delay 199, but changing his strategy would not help his outcome from 200 so the system is still in equilibrium. There are many more similar such equilibria.

Claim that at least 98 players must choose strategy 3 and at most 1 player can choose strategy 1 and at most 1 player can choose strategy 2 for Nash equilibrium.

Contrasting the dynamics of this game with the previous, is that now total delay is 200 for most players and at best 198 for some players. In either case, making life worse for all characters. Hence Braess' paradox – adding an edge to the graph made life worse for all the players.

Questions we will look at: how do you find these equilibria? Why do we get paradoxes like this one? Are situations like this paradox really so bad? Adding the edge didn't make life that much worse for players. Can players find these equilibria?

We will look at various ways for looking at how to prevent paradoxes like this one. We will heavily rely on some sort of 'rationality' for the players. Later we'll go back and look at a more intricate definition of equilibrium. Our current definition is good enough when these equilibria are fairly stable and unique. There are however criticisms of Nash Equilibrium and next time we'll get some examples of where Nash Equilibrium doesn't capture everything we want.

Next time we'll consider the significance of various objective functions. Maybe you don't just want to make lots of money – maybe it's more important to make more money than your neighbors? Or maybe you don't just want to make as much money as possible in a game, but rather to make sure you win the game. We will explore trade offs between various measures of utility.

CS 6840 – Algorithmic Game Theory (3 pages)	Spring 2012
Lecture 2 Scribe Notes	
Instructor: Eva Tardos	*Deniz Altınbüken (da279)*

1 Lecture 2 – Wednesday 25 January 2012 - Congestion Games

1.1 Definition

Definition. Congestion games are a class of games defined as follows:

- base set of congestable elements E
- n players
- each player i has finite set of strategies S_i
- a strategy $P \in S_i$ where $P \subseteq E$
- given a strategy P_i for each player i

$$x_e = \#\{i; e \in P_i\} \text{ for } e \in E$$

- player i choosing strategy P_i experiences delay

$$\sum_{e \in P_i} d_e(x_e)$$

Remark. Strategies P_i for player i define *Pure Nash Equilibrium* iff no one player can improve the delay by changing to another strategy Q_i.

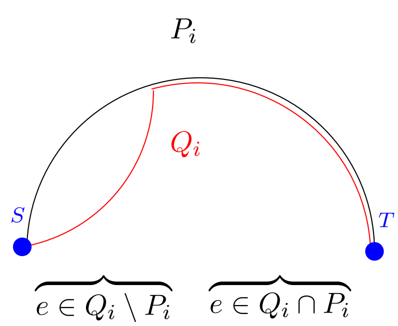


Figure 1: Strategies P_i, Q_i

Consider a player in the game shown in Figure 1. In this game, player i is switching from P_i to Q_i. As depicted, P_i and Q_i might have common parts as well as parts that differ. By switching, player i would experience the same delay in the parts that are common for P_i and Q_i and she will experience delay that results from adding one more person in Q_i in the parts that differ.

For all players i & all other $Q_i \in S_i$

$$\sum_{e \in P_i} d_e(x_e) \leq \sum_{e \in P_i \cap Q_i} d_e(x_e) + \sum_{e \in Q_i \setminus P_i} d_e(x_e + 1)$$

1.2 Equilibrium at Congestion Games

Now, lets explore the following questions:

- Does a general congestion game have a Nash Equilibrium?
- Are reasonable players able to find the Nash Equilibrium?

When we are looking for the Nash Equilibrium, the trivial approach is to change the strategy of one player and see if the resulting state is a Nash Equilibrium. In this approach, it is important to make sure that cycles do not occur to guarantee that the Nash Equilibrium is found. To see how cycles might occur, consider the following Matching Pennies Game.

Side Note If there are cycles present in the game, the equilibrium may not be found.

Example: Matching Pennies Game

- 2 players
- Strategies: H, T
- Rule:

$$m(s) = \begin{cases} \textit{player 1 wins} & \text{if strategies match} \\ \textit{player 2 wins} & \text{otherwise} \end{cases}$$

- Best response: Player starts with arbitrary strategy, switches if she loses

$$(H \mid H) \rightarrow (H \mid T) \rightarrow (T \mid T) \rightarrow (T \mid H) \rightarrow$$

1.3 Existence of Nash Equilibrium

Theorem 1. Congestion games repeated best response always finds the Nash Equilibrium.

Proof. Congestion games have a potential function Φ s.t. best response improves this function:

$$\Phi = \sum_e \sum_{k=1}^{x_e} d_e(k)$$

Consider: Player i switches from P_i to Q_i, change in Φ:

- edges $e \in P_i \setminus Q_i$ decrease by $d_e(x_e)$

- edges $e \in Q_i \setminus P_i$ increase by $d_e(x_e + 1)$

Note that, $\sum_{k=1}^{x_e} d_e(k)$ is the discrete integral of x_e, i.e. the potential function Φ is the summation of the discrete integral of x_e over all edges and the change in the potential function is equal to the change in a player's delay when she switches from strategy P_i to Q_i.

When a player changes from strategy P_i to Q_i, the change in the delay is equal to the change in the potential function Φ.

Alternate Proof. Solution minimizing Φ is the Nash Equilibrium, assuming there are a finite number of solutions and a minimum exists. Since we assumed $d_e(x)$ is a monotone increasing (i.e. non-decreasing) function, there exists only one minimum, i.e. the Nash Equilibrium.

CS 6840 – Algorithmic Game Theory (3 pages) Spring 2012

Lecture 39: The Existence of Nash Equilibrium in Finite Games

Instructor: Eva Tardos *Wenlei Xie(wx49)*

Today we will use the same methodology that we used in the last time to prove Nash Equilibrium exists in finite games.

Theorem 1 (Existence of NE)**.** Game with finite set of players and finite strategy sets has at least one (mixed) Nash Equilibrium.

Remark (Finite Game and Mixed NE)**.** This only applies to what usually called *Finite Games*. Here "finite" means two things: finite set of players and each of them has finite set of strategies. So this theorem doesn't apply to a bunch of games we studied, e.g. when the strategy is the price, the strategy set is not finite since it can be real numbers. Some other games could have infinite players. In most of these games, we actually have other arguments to prove that NE exists, usually even a better argument because we used to prove a pure strategy NE exists. And this theorem only states a mixed strategy NE exists, which is not surprising because some small 2 by 2 games, e.g. Pennies Matching game doesn't have a pure NE.

To prove this theorem, the main tool we will use is the Brouwer fixpoint theorem.

Theorem 2 (Brouwer Fixpoint Theorem)**.** If C is bounded, convex and closed, and $f : C \to C$ is continuous, there exists x s.t. $f(x) = x$.

Remark. Last time we only did it for the simplex. Generally we certainly need it to be bounded and closed. Topologically, we can make stronger statements than convex – but convex is certainly enough for today.

We will start with a natural but problematic proof. What we want to do is the same story as last time. Starting from one possible game state, which is a set of mixed strategies of all the players, we would like to know if it's an equilibrium or not. And if it's not we want a function that moves it more "closer" to the equilibrium.

Let n be the number of players and S_i to be the strategy set of player i, and Δ_i be the probability distribution space of strategies for player i, i.e.

$$\Delta_i = \{(p_s : s \in S_i) \,|\, p_s \geq 0 \text{ and } \sum_{s \in S_i} p_s = 1\} \tag{1}$$

We use C to denote the set of the mixed strategies of all the players, i.e. $C = \Delta_1 \times \Delta_2 \times \cdots \times \Delta_n$. It can be proved that C is convex, bounded and closed. Next we need a function $f : C \to C$ that the NE is a fixpoint. A natural answer is to use the best response. That is to say, given $p = (p_1, p_2, \cdots, p_n) \in C$, where $p_i \in \Delta_i$. Let q_i be the best response of player i, we could define the function as $f(p) = (q_1, q_2, \cdots, q_n)$. This can be viewed as all the players are moving to the best response state simultaneously as if others don't move.

The fundamental issue in this "proof" is that f might not be a function since the best response for the player might not be unique. A natural way to address this issue is to use lexicographic tie-breaking rule. Unfortunately the function constructed in this way might not be continuous. Let's consider the following example.

	Heads	Tails
Heads	$(+1, -1)$	$(-1, +1)$
Tails	$(-1, +1)$	$(+1, -1)$

Table 1: The payoff matrix for the matching pennies game

Example (Matching Pennies)**.** Recall the payoff matrix in the matching pennies game shown in Table 1. Suppose the mixed strategy for the first player is $(p_1, 1-p_1)$, i.e. he will turn the penny into head with probability p_1 and turn it into tail with probability $1-p_1$. Then the best response $(q_2, 1-q_2)$ for the second player is

$$\text{Best Response} = \begin{cases} q_2 = 0 & \text{if } p_1 > 1/2, \\ q_2 = 1 & \text{if } p_1 < 1/2, \\ 0 \le q_2 \le 1 & \text{if } p_1 = 1/2. \end{cases}$$

And clearly it is not continuous at $p_1 = 1/2$.

Thus, we need some better methods to fix this issue. We will discuss two options in this lecture.

Option 1 (Set Function)**.** Define $f : C \to 2^C$ as $f(p) = \{q \mid q_i \text{ is best response for } p_{-i}\}$

In this option, we hope to find the p, s.t. $p \in f(p)$. To this end we need to use a stronger fixpoint theorem by Kakutani and formally define what does "continuity" means for such set functions. We are not going to discuss the details in today's lecture.

Option 2 (More Sophisticated Objective Function)**.** Let $u_i(q, p_{-i})$ be the utility of player i playing q in response to p_{-i}. Here comes the natural best response function

$$\max_q u_i(q, p_{-i}) \to \text{ Original best response } q$$

As we have shown before, this doesn't define a function, and the natural way to make it a function breaks the continuity. Alternatively, consider

$$\max_q u_i(q, p_{-i}) - ||p_i - q||^2$$

So for player i, instead of maximizing the utility $u_i(q, p_{-i})$, it maximizing the utility minus a penalty from going away from the original p_i, i.e. $||p_i - q||^2$. Notice any positive scale for $||p_i - q||^2$ works. Suppose the maximizer for player i is q_i, we define $f(p) = (q_1, q_2, \cdots, q_n)$.

To finish the proof, we first claim that it indeed defines a function, which means the maximizer is unique.

Lemma 3. $\max_q u_i(q, p_{-i}) - ||q - p_i||^2$ is unique.

To prove this, we will use the fact that strictly concave function has unique maximization. Notice there are many definitions for strictly concave for vector functions. And we will use the following definition in our proof.

Definition (Strictly Concave)**.** $g(x)$ is strictly concave of x if

$$\forall x, x',\ \frac{1}{2}(g(x) + g(x')) > g(\frac{x + x'}{2}) \tag{2}$$

Proof. Notice that

$$u_i(q, p_{-i}) = \sum_{s \in S_i} q_s v_s(p_{-i}) \tag{3}$$

where $v_s(p_{-i})$ is the value of pure strategy s. Thus $u_i(q, p_{-i})$ is a linear function of q. And $-||q - p_i||^2$ is a strictly concave function of q, which makes $u_i(q, p_{-i}) - ||q - p_i||^2$ strictly concave, and it has unique maximization. □

To show f is continuous, we will use the following fact from convex optimization without proof.

Claim 1. If a class of optimization problems has unique optima, then the optimum is a continuous function of the coefficients in the objective function

An important part that is missing is that we want to show the fixpoint of function f is the Nash. When f is defined by the maximizer of $u_i(q, p_{-i})$, it is obvious. Now with the penalty term it is less obvious, but we can nonetheless prove it.

Lemma 4. If $f(p) = p$, then p is Nash.

If p is not Nash, there is some other best response $q = (q_1, q_2, \cdots, q_n)$. For the player i that doesn't perform best response, consider move from p_i to q_i. It will certainly increase the first part of the objective function $u_i(q_i, p_{-i})$. However the whole objective function might not be increased since the second part $||q_i - p_i||^2$ is also increased. And we will show if we just move on that direction small enough, it will be OK.

Proof. Suppose p is not Nash. Suppose one best response is $q = (q_1, q_2, \cdots, q_n)$. For player i that p_i is not best response, we have

$$u_i(q_i, p_{-i}) > u_i(p_i, p_{-i}) \tag{4}$$

Let $r_i(\epsilon) = (1 - \epsilon)p_i + \epsilon q_i$. And if player i move from p_i to $r_i(\epsilon)$, consider the change in his objective function $\delta_i(\epsilon)$

$$\begin{aligned}\delta_i(\epsilon) &= \Big(u_i(r_i(\epsilon), p_{-i}) - ||r_i(\epsilon) - p_i||^2\Big) - u_i(p_i, p_{-i}) \\ &= \epsilon\Big(u_i(q_i, p_{-i}) - u_i(p_i, p_{-i})\Big) - \epsilon^2||q_i - p_i||^2\end{aligned}$$

For small enough ϵ, we have the change $\delta_i(\epsilon) > 0$. Hence p_i does not maximize $u_i(q, p_{-i}) - ||q - p_i||^2$, which means p couldn't be a fixpoint. □

Remark (Function f)**.** The function f assumes everyone simultaneously best response, but how would people's utility change as everyone best response? We don't know. This is a proof and it wants to do this artificial yet somewhat meaningless activity, of considering a function where everyone best response as if the other people don't move. If they don't move, we reach the Nash. But if they move, it is meaningless. Notice it is not a game dynamic, in fact it is nothing but a mathematic gadget of proof.

In the following lectures, we will show if you can find the NE in a game, you can find the fixpoint of the corresponding function.

CS 683 — Learning, Games, and Electronic Markets **Spring 2007**

Notes from Week 1: Algorithms for sequential prediction

Instructor: Robert Kleinberg *22-26 Jan 2007*

1 Introduction

In this course we will be looking at online algorithms for learning and prediction. These algorithms are interesting in their own right — as a topic in theoretical computer science — but also because of their role in the design of electronic markets (e.g. as algorithms for sequential price experimentation, or for online recommendation systems) and their role in game theory (where online learning processes have been proposed as an explanation for how players learn to play an equilibrium of a game).

2 Online algorithms formalism

For general background on online algorithms, one can look at the book *Online Computation and Competitive Analysis* by Borodin and El-Yaniv, or read the notes from an online algorithms course taught by Michel Goemans at MIT, available by FTP at

`ftp://theory.csail.mit.edu/pub/classes/18.415/notes-online.ps`

In this section we give an abstract definition of online algorithms, suitable for the prediction problems we have studied in class.

Definition 1. An *online computation problem* is specified by:

1. A set of inputs $\mathscr{I} = \prod_{t=1}^{\infty} I_t$.
2. A set of outputs $\mathscr{O} = \prod_{t=1}^{\infty} O_t$.
3. A cost function $\mathsf{Cost} : \mathscr{I} \times \mathscr{O} \to \mathbb{R}$.

For a positive integer T, we will define

$$\mathscr{I}[T] = \prod_{t=1}^{T} I_t, \qquad \mathscr{O}[T] = \prod_{t=1}^{T} O_t.$$

One should interpret an element $i = (i_1, i_2, \ldots) \in \mathscr{I}$ as a sequence representing the inputs revealed to the algorithm over time, with i_t representing the part of the input revealed at time t. Similarly, one should interpret an element $o = (o_1, o_2, \ldots)$ as a sequence of outputs produced by the algorithm, with o_t being the output at time t.

Remark 1. The definition frames online computation problems in terms of an *infinite* sequence of inputs and outputs, but it is easy to incorporate problems with a finite time horizon T as a special case of the definition. Specifically, if $|I_t| = |O_t| = 1$ for all $t > T$ then this encodes an input-output sequence in which no information comes into or out of the algorithm after time T.

Definition 2. An *online algorithm* is a sequence of functions

$$F_t : \mathscr{I}[t] \to O_t.$$

An *adaptive adversary* (or, simply, *adversary*) is a sequence of functions

$$G_t : \mathscr{O}[t-1] \to I_t.$$

An adversary is called *oblivious* if each of the functions G_t is a constant function.

If F is an online algorithm and G is an adaptive adversary, the *transcript* of F and G is the unique pair $\mathsf{Trans}(F, G) = (i, o) \in \mathscr{I} \times \mathscr{O}$ such that for all $t \geq 1$,

$$\begin{aligned} i_t &= G_t(o_1, o_2, \ldots, o_{t-1}) \\ o_t &= F_t(i_1, i_2, \ldots, i_t). \end{aligned}$$

The *cost* of F and G is $\mathsf{Cost}(F, G) = \mathsf{Cost}(\mathsf{Trans}(F, G))$.

One should think of the algorithm and adversary as playing a game in which the adversary specifies a component of the input based on the algorithm's past outputs, and the algorithm responds by producing a new output. The transcript specifies the entire sequence of inputs and outputs produced when the algorithm and adversary play this game.

Remark 2. Designating an oblivious adversary is equivalent to designating a single input sequence $i = (i_1, i_2, \ldots) \in \mathscr{I}$.

Remark 3. Our definition of algorithm and adversary makes no mention of computational constraints (e.g. polynomial-time computation) for either party. In general we will want to design algorithms which are computationally efficient, but it is possible to ask meaningful and non-trivial questions about online computation without taking such constraints into account.

In defining randomized algorithms and adversaries, we think of each party as having access to infinitely many independent random bits (represented by the binary digits of a uniformly distributed element of $[0, 1]$) which are not revealed to the other party.

Definition 3. A *randomized online algorithm* is a sequence of functions

$$F_t : \mathscr{I}[t] \times [0,1] \to O_t.$$

A *randomized adaptive adversary* is a sequence of functions

$$G_t : \mathscr{O}[t-1] \times [0,1] \to I_t.$$

A randomized adversary is called *oblivious* if the output of each function $G_t(o,y)$ depends only on the parameter y.

If F and G are a randomized algorithm and randomized adaptive adversary, respectively, then the *transcript* of F and G is the function $\mathsf{Trans}(F,G) : [0,1] \times [0,1] \to \mathscr{I} \times \mathscr{O}$ which maps a pair (x,y) to the unique input-output pair (i,o) satisfying:

$$\begin{aligned} i_t &= G_t(o_1, o_2, \ldots, o_{t-1}, y) \\ o_t &= F_t(i_1, i_2, \ldots, i_t, x) \end{aligned}$$

for all $t \geq 1$. The *cost* of F and G is $\mathsf{Cost}(F,G) = \mathbf{E}[\mathsf{Cost}(\mathsf{Trans}(F,G)(x,y))]$, when the pair (x,y) is sampled from the uniform distribution on $[0,1]^2$.

Remark 4. A randomized oblivious adversary is equivalent to a probability distribution over input sequences $i = (i_1, i_2, \ldots) \in \mathscr{I}$.

Remark 5. In class I defined a randomized algorithm using an infinite sequence of independent random variables $(x_1, x_2, \ldots) \in [0,1]^\infty$, and similarly for a randomized adversary. Consequently the transcript $\mathsf{Trans}(F,G)$ was described as a function from $[0,1]^\infty \times [0,1]^\infty$ to $\mathscr{I} \times \mathscr{O}$. This was unnecessarily complicated: a single random number $x \in [0,1]$ contains infinitely many independent random binary digits, so it already contains as much randomness as the algorithm would need for an entire infinite sequence of input-output pairs. Accordingly, in these notes I have simplified the definition by assuming that the algorithm's and adversary's random bits are contained in a single pair of independent random real numbers (x,y), with x representing the algorithm's supply of random bits and y representing the adversary's supply.

3 Binary prediction with one perfect expert

As a warm-up for the algorithms to be presented below, let's consider the following "toy problem." The algorithm's goal is to predict the bits of an infinite binary sequence $\vec{B} = (B_1, B_2, \ldots)$, whose bits are revealed one at a time. Just before the t-th bit is revealed, a set of n experts make predictions $b_{1t}, b_{2t}, \ldots, b_{nt} \in \{0,1\}$. The algorithm is allowed to observe all of these predictions, then it makes a guess denoted by $a_t \in \{0,1\}$, and then the truth, B_t, is revealed. We are given a promise that there is at least one expert whose predictions are always accurate, i.e. we are promised that $\exists i \, \forall t \; b_{it} = B_t$.

This prediction problem is a special case of the framework described above. Here, $I_t = \{0,1\} \times \{0,1\}^n$ and $O_t = \{0,1\}$. The input i_t contains all the information revealed to the algorithm after it makes its $(t-1)$-th guess and before it makes its t-th guess: thus i_t consists of the value of B_{t-1} together with all the predictions $b_{1t}, \ldots, b_{nt}$. The output o_t is simply the algorithm's guess a_t. The cost $\mathsf{Cost}(i, o)$ is the number of times t such that $a_t \neq B_t$.

Consider the following algorithm, which we will call the "Majority algorithm": at each time t, it consults the predictions of all experts who did not make a mistake during one of the first $t-1$ steps. (In other words, it considers all experts i such that $b_{is} = B_s$ for all $s < t$.) If more of these experts predict 1 than 0, then $a_t = 1$; otherwise $a_t = 0$.

Theorem 1. *Assuming there is at least one expert i such that $b_{it} = B_t$ for all t, the Majority algorithm makes at most $\lfloor \log_2(n) \rfloor$ mistakes.*

Proof. Let S_t denote the set of experts who make no mistakes before time t. Let $W_t = |S_t|$. If the Majority algorithm makes a mistake at time t, it means that at least half of the experts in S_t made a mistake at that time, so $W_{t+1} \leq \lfloor W_t/2 \rfloor$. On the other hand, by assumption we have $|W_t| \geq 1$ for all t. Thus the number of mistakes made by the algorithm is bounded above by the number of iterations of the function $x \mapsto \lfloor x/2 \rfloor$ required to get from n down to 1. This is $\lfloor \log_2(n) \rfloor$. □

Remark 6. The bound of $\lfloor \log_2(n) \rfloor$ in Theorem 1 is information-theoretically optimal, i.e. one can prove that no deterministic algorithm makes strictly fewer than $\lfloor \log_2(n) \rfloor$ mistakes on every input.

Remark 7. Although the proof of Theorem 1 is very easy, it contains the two essential ingredients which will reappear in the analysis of the Weighted Majority and Hedge algorithms below. Namely, we define a number W_t which measures the "remaining amount of credibility" of the set of experts at time t, and we exploit two key properties of W_t:

- When the algorithm makes a mistake, there is a corresponding multiplicative decrease in W_t.
- The assumption that there is an expert whose predictions are close to the truth implies a lower bound on the value of W_t for all t.

The second property says that W_t can't shrink too much starting from its initial value of n; the first property says that if W_t doesn't shrink too much then the algorithm can't make too many mistakes. Putting these two observations together results in the stated mistake bound. Each of the remaining proofs in these notes also hinges on these two observations, although the manipulations required to justify the two observations become more sophisticated as the algorithms we are analyzing become more sophisticated.

Algorithm $\mathsf{WMA}(\varepsilon)$

/* Initialization */
$w_i \leftarrow 1$ **for** $i = 1, 2, \ldots, n$.

/* Main loop */
for $t = 1, 2, \ldots$
    /* Make prediction by taking weighted majority vote */
    if $\sum_{i\,:\,b_{it}=0} w_i > \sum_{i\,:\,b_{it}=1} w_i$
        output $a_t = 0$;
    else
        output $a_t = 1$.

    Observe the value of B_t.

    /* Update weights multiplicatively */
    $E_t \leftarrow \{\text{experts who predicted incorrectly}\}$
    $w_i \leftarrow (1-\varepsilon) \cdot w_i$ **for all** $i \in E_t$.
end

Figure 1: The weighted majority algorithm

4 Deterministic binary prediction: the Weighted Majority Algorithm

We now present an algorithm for the same binary prediction problem discussed in Section 3. This new algorithm, the Weighted Majority algorithm, satisfies a provable mistake bound even when we don't assume that there is an expert who never makes a mistake. The algorithm is shown in Figure 1. It is actually a one-parameter family of algorithms $\mathsf{WMA}(\varepsilon)$, each with a preconfigured parameter $\varepsilon \in (0, 1)$.

Theorem 2. *Let M denote the number of mistakes made by the algorithm $\mathsf{WMA}(\varepsilon)$. For every integer m, if there exists an expert i which makes at most m mistakes, then*

$$M < \left(\frac{2}{1-\varepsilon}\right) m + \left(\frac{2}{\varepsilon}\right) \ln(n).$$

Proof. Let w_{it} denote the value of w_i at the beginning of the t-th iteration of the main loop, and let $W_t = \sum_{i=1}^{n} w_{it}$. The hypothesis implies that there is an expert i such that $w_{iT} \geq (1-\varepsilon)^m$ for all T, so

$$W_T > w_{iT} \geq (1-\varepsilon)^m \tag{1}$$

for all T. On the other hand, if the algorithm makes a mistake at time t, it implies that

$$\sum_{i \in E_t} w_{it} \geq \frac{W_t}{2},$$

hence

$$\begin{aligned} W_{t+1} &= \sum_{i \in E_t} (1-\varepsilon) \cdot w_{it} + \sum_{i \notin E_t} w_{it} \\ &= \sum_{i=1}^{n} w_{it} - \varepsilon \sum_{i \in E_t} w_{it} \\ &\leq W_t \left(1 - \frac{\varepsilon}{2}\right). \end{aligned}$$

For any $T > 0$, we find that

$$\frac{W_T}{W_0} = \prod_{t=0}^{T-1} \frac{W_{t+1}}{W_t} \leq \left(1 - \frac{\varepsilon}{2}\right)^M \tag{2}$$

where M is the total number of mistakes made by the algorithm $\mathsf{WMA}(\varepsilon)$. Combining (1) with (2) and recalling that $W_0 = \sum_{i=1}^n w_{i0} = \sum_{i=1}^n 1 = n$, we obtain

$$\frac{(1-\varepsilon)^m}{n} < \frac{W_T}{W_0} \leq \left(1 - \frac{\varepsilon}{2}\right)^M.$$

Now we take the natural logarithm of both sides.

$$\begin{aligned} \ln(1-\varepsilon)m - \ln(n) &< \ln\left(1 - \frac{\varepsilon}{2}\right) M && (3) \\ \ln(1-\varepsilon)m - \ln(n) &< -(\varepsilon/2)M && (4) \\ \ln\left(\frac{1}{1-\varepsilon}\right) m + \ln(n) &> (\varepsilon/2)M && (5) \\ \left(\frac{2}{\varepsilon}\right) \ln\left(\frac{1}{1-\varepsilon}\right) m + \left(\frac{2}{\varepsilon}\right) \ln(n) &> M && (6) \\ \left(\frac{2}{1-\varepsilon}\right) m + \left(\frac{2}{\varepsilon}\right) \ln(n) &> M && (7) \end{aligned}$$

where (4) was derived from (3) using identity (21) from the appendix of these notes, and (7) was derived from (6) using identity (22) from the appendix. □

5 Randomized prediction: the Hedge Algorithm

We now turn to a generalization of the binary prediction problem: the "best expert" problem. In this problem, there is again a set of n experts, which we will identify

Algorithm $\mathsf{Hedge}(\varepsilon)$

```
/* Initialization */
w_x ← 1 for x ∈ [n]

/* Main loop */
for t = 1, 2, ...
    /* Define distribution for sampling random strategy */
    for x ∈ [n]
        p_t(x) ← w_x / (Σ_{y=1}^n w_y)
    end
    Choose x_t ∈ [n] at random according to distribution p_t.
    Observe cost function c_t.

    /* Update score for each strategy */
    for x ∈ [n]
        w_x ← w_x · (1 − ε)^{c_t(x)}
    end
end
```

Figure 2: The algorithm $\mathsf{Hedge}(\varepsilon)$.

with the set $[n] = \{1, 2, \ldots, n\}$. In each time step t, the adversary designates a cost function c_t from $[n]$ to $[0, 1]$, and the algorithm chooses an expert $x_t \in [n]$. The cost function C_t is revealed to the algorithm only after it has chosen x_t. The algorithm's objective is to minimize the sum of the costs of the chosen experts, i.e. to minimize $\sum_{t=1}^{\infty} c_t(x_t)$.

Observe that this problem formulation fits into the formalism specified in Section 2; the input sequence $(i_1, i_2, \ldots)$ is given by $i_t = c_{t-1}$, the output sequence $(o_1, o_2, \ldots)$ is given by $o_t = x_t$, and the cost function is

$$\mathsf{Cost}(i, o) = \sum_{t=1}^{\infty} i_{t+1}(o_t) = \sum_{t=1}^{\infty} c_t(x_t).$$

Also observe that the binary prediction problem is a special case of the best expert problem, in which we define $c_t(x) = 1$ if $b_{xt} \neq B_t$, 0 otherwise.

Figure 2 presents a randomized online algorithm for the best expert problem. As before, it is actually a one-parameter family of algorithms $\mathsf{Hedge}(\varepsilon)$ with a preconfigured parameter $\varepsilon \in (0, 1)$. Note the algorithm's similarity to $\mathsf{WMA}(\varepsilon)$: it maintains a vector of weights, one for each expert, and it updates these weights multiplicatively using a straightforward generalization of the multiplicative update rule in WMA. The main difference is that WMA makes its decisions by taking a weighted majority vote of the experts, while Hedge makes its decisions by performing a weighted random selection of a single expert.

Theorem 3. *For every randomized adaptive adversary, for every $T > 0$, the expected cost suffered by* Hedge(ε) *satisfies*

$$\mathbf{E}\left[\sum_{t=1}^{T} c_t(x_t)\right] < \left(\frac{1}{1-\varepsilon}\right)\mathbf{E}\left[\min_{x\in[n]}\sum_{t=1}^{T} c_t(x)\right] + \left(\frac{1}{\varepsilon}\right)\ln(n). \tag{8}$$

Proof. Let w_{xt} denote the value of w_x at the beginning of the t-th iteration of the main loop, and let $W_t = \sum_{x=1}^{n} w_{xt}$. Note that w_{xt}, W_t are random variables, since they depend on the adversary's choices which in turn depend on the algorithm's random choices in previous steps. For an expert $x \in [n]$, let $c_{1..T}(x)$ denote the total cost

$$c_{1..T}(x) = \sum_{t=1}^{T} c_t(x).$$

Let $x^* = \arg\min_{x\in[n]} c_{1..T}(x)$. We have

$$W_T > w_{x^*t} = (1-\varepsilon)^{c_{1..T}(x^*)}$$

and after taking logarithms of both sides this becomes

$$\ln(W_T) > \ln(1-\varepsilon)c_{1..T}(x^*) \tag{9}$$

On the other hand, we can bound the expected value of $\ln(W_T)$ from above, using an inductive argument. Let w_{*t} denote the vector of weights $(w_{1t}, \ldots, w_{nt})$.

$$\begin{aligned}
\mathbf{E}(W_{t+1} \,|\, w_{*t}) &= \sum_{x=1}^{n} \mathbf{E}\left((1-\varepsilon)^{c_t(x)} w_{xt} \,|\, w_{*t}\right) && (10)\\
&\le \sum_{x=1}^{n} \mathbf{E}\left((1-\varepsilon c_t(x)) w_{xt} \,|\, w_{*t}\right) && (11)\\
&= \sum_{x=1}^{n} w_{xt} - \varepsilon \mathbf{E}\left(\sum_{x=1}^{n} c_t(x) w_{xt} \,|\, w_{*t}\right) && (12)\\
&= W_t \cdot \left(1 - \varepsilon \mathbf{E}\left(\sum_{x=1}^{n} c_t(x) p_t(x) \,|\, w_{*t}\right)\right) && (13)\\
&= W_t \cdot (1 - \varepsilon \mathbf{E}(c_t(x_t) \,|\, w_{*t})) && (14)\\
\mathbf{E}(\ln(W_{t+1}) \,|\, w_{*t}) &\le \ln(W_t) + \ln(1 - \varepsilon \mathbf{E}(c_t(x_t) \,|\, w_{*t})) && (15)\\
&\le \ln(W_t) - \varepsilon \mathbf{E}(c_t(x_t) \,|\, w_{*t}) && (16)
\end{aligned}$$

$$\begin{aligned}
\varepsilon \mathbf{E}(c_t(x_t) \,|\, w_{*t}) &\leq \ln(W_t) - \mathbf{E}(\ln(W_{t+1}) \,|\, w_{*t}) && (17)\\
\varepsilon \mathbf{E}(c_t(x_t)) &\leq \mathbf{E}(\ln(W_t)) - \mathbf{E}(\ln(W_{t+1})) && (18)\\
\varepsilon \mathbf{E}\left(\sum_{t=1}^{T} c_t(x_t)\right) &\leq \ln(n) - \mathbf{E}(\ln(W_T)). && (19)
\end{aligned}$$

Here, (11) is derived using identity (23) from the appendix, (13) is derived using the fact that $p_t(x) = w_{xt}/W_t$, (14) is derived using the observation that x_t is a random element sampled from the probability distribution $p_t(\cdot)$ on $[n]$, (15) and (16) are derived using the identities (24) and (21) respectively, (18) is derived by taking the unconditional expectation of both sides of the inequality, and (19) is derived by summing over t and recalling that $W_0 = n$.

Combining (9) and (19) we obtain

$$\begin{aligned}
\varepsilon \mathbf{E}\left(\sum_{t=1}^{T} c_t(x_t)\right) &< \ln(n) - \ln(1-\varepsilon)\mathbf{E}(c_{1..T}(x^*))\\
\mathbf{E}\left(\sum_{t=1}^{T} c_t(x_t)\right) &< \left(\frac{1}{\varepsilon}\right)\ln(n) + \frac{1}{\varepsilon}\ln\left(\frac{1}{1-\varepsilon}\right)\mathbf{E}(c_{1..T}(x^*))\\
\mathbf{E}\left(\sum_{t=1}^{T} c_t(x_t)\right) &< \left(\frac{1}{\varepsilon}\right)\ln(n) + \left(\frac{1}{1-\varepsilon}\right)\mathbf{E}(c_{1..T}(x^*))
\end{aligned}$$

where the last line is derived using identity (22) from the appendix. □

6 Appendix: Some useful inequalities for logarithms and exponential functions

In various steps of the proofs given above, we applied some useful inequalities that follow from the convexity of exponential functions or the concavity of logarithms. In this section we collect together all of these inequalities and indicate their proofs.

Lemma 4. *For all real numbers x,*

$$1 + x \leq e^x \tag{20}$$

with equality if and only if $x = 0$.

Proof. The function e^x is strictly convex, and $y = 1 + x$ is the tangent line to $y = e^x$ at $(0, 1)$. □

Lemma 5. *For all real numbers $x > -1$,*

$$\ln(1 + x) \leq x \tag{21}$$

with equality if and only if $x = 0$.

Proof. Take the natural logarithm of both sides of (20). □

Lemma 6. *For all real numbers $y \in (0,1)$,*

$$\frac{1}{y}\ln\left(\frac{1}{1-y}\right) < \frac{1}{1-y}. \tag{22}$$

Proof. Apply (21) with $x = \frac{y}{1-y}$, then divide both sides by y. □

Lemma 7. *For every pair of real numbers $x \in [0,1], \varepsilon \in (0,1)$,*

$$(1-\varepsilon)^x \le 1 - \varepsilon x \tag{23}$$

with equality if and only if $x = 0$ or $x = 1$.

Proof. The function $y = (1-\varepsilon)^x$ is strictly convex and the line $y = 1 - \varepsilon x$ intersects it at the points $(0,1)$ and $(1, 1-\varepsilon)$. □

Lemma 8. *For every random variable X, we have*

$$\mathbf{E}(\ln(X)) \le \ln(\mathbf{E}(X)) \tag{24}$$

with equality if and only if there is a constant c such that $\Pr(X = c) = 1$.

Proof. Jensen's inequality for convex functions says that if f is a convex function and X is a random variable,

$$\mathbf{E}(f(X)) \ge f(\mathbf{E}(X)),$$

and that if f is strictly convex, then equality holds if and only if there is a constant c such that $\Pr(X = c) = 1$. The lemma follows by applying Jensen's inequality to the strictly convex function $f(x) = -\ln(x)$. □

CS 683 — Learning, Games, and Electronic Markets **Spring 2007**

Notes from Week 2: Prediction algorithms and zero-sum games

Instructor: Robert Kleinberg *29 Jan – 2 Feb 2007*

1 Summary of Week 1

Here are some closing observations about sequential prediction algorithms.

- For the problem of predicting a sequence using expert advice, we saw two algorithms: a deterministic algorithm which satisfies a mistake bound

$$M \leq \left(\frac{2}{1-\varepsilon}\right) m + \left(\frac{2}{\varepsilon}\right) \ln n$$

and a randomized algorithm which satisfies a mistake bound

$$M \leq \left(\frac{1}{1-\varepsilon}\right) m + \left(\frac{1}{\varepsilon}\right) \ln n,$$

where n is the number of experts, m is the number of mistakes made by the best expert, and $\varepsilon > 0$ is a parameter which is pre-configured by the algorithm designer.

- You should think of $1/(1-\varepsilon)$ as being equivalent to $1 + O(\varepsilon)$. (For example, when $\varepsilon < 1/2$ we have $1/(1-\varepsilon) < 1 + 2\varepsilon$.) Hence when $m \gg \ln(n)$, the randomized prediction algorithm comes *very* close to making the same number of mistakes as the best expert.

- Neither algorithm needs to know the value of m in order to achieve these mistake bounds.

- The best randomized algorithm makes half as many mistakes as the best deterministic algorithm. This seems to be a recurring theme in sequential prediction problems. (You saw another example of it on your homework.) The factor of 2 is not because it's a binary prediction problem; randomization also saves a factor of 2 when predicting the data in a k-ary sequence for $k > 2$.

- The randomized prediction algorithm actually applies to a more general problem — the "best expert" problem — in which there is a cost associated with each expert in each time step, and the cost of every expert (a real number between 0 and 1) is revealed only after the algorithm picks one expert.

- The analysis of both algorithms follows the same rough outline. It's important to remember this rough outline: every time the algorithm accrues one unit of cost, there is a corresponding multiplicative decrease in the total "weight" of the experts. Since the total weight can never sink below the weight of the best expert — and it starts out only n times larger than the best expert's weight — the algorithm can only accrue $O(\log n)$ more units of cost than the best expert.

2 Regret

We've seen a randomized algorithm Hedge for the best-expert problem whose expected cost relative to the best expert x^* satisfies

$$\mathbf{E}[\mathsf{Cost}(\mathsf{Hedge})] \leq (1+2\varepsilon)\mathsf{Cost}(x^*) + \frac{\ln(n)}{\varepsilon}, \tag{1}$$

for an arbitrary constant $\varepsilon \in (0, \frac{1}{2})$. Whenever you see a bound that says that an algorithm computes a solution whose cost is within a $1+2\varepsilon$ factor of the optimum, for an arbitrarily small $\varepsilon > 0$, a natural follow-up question is whether one can actually get the cost to be $1 + o(1)$ times the optimum, and if so how small can we make the $o(1)$ term?

Let's rewrite (1) as

$$\mathbf{E}[\mathsf{Cost}(\mathsf{Hedge}) - \mathsf{Cost}(x^*)] \leq 2\varepsilon\, \mathsf{Cost}(x^*) + \frac{\ln(n)}{\varepsilon}. \tag{2}$$

Definition 1 (Informal definition of regret.). The *regret* of an online learning algorithm ALG is the maximum (over all input instances) of the expected difference in cost between the algorithm's choices and the best choice.

Definition 2 (Formal definition of regret, valid for this lecture.). The *regret* of an online learning algorithm ALG relative to a class of adversaries $\mathcal{G}$ is

$$R(\mathsf{ALG}, \mathsf{ADV}) = \sup_{G \in \mathcal{G}} \mathbf{E}\left[\max_{x \in [n]} \sum_{t=1}^{\infty} c_t(x_t) - c_t(x)\right],$$

where c_t denotes the (random) cost function chosen at time t by adversary G playing against algorithm ALG (i.e. the component i_{t+1} in the transcript $\mathsf{Trans}(\mathsf{ALG}, G)$), and x_t denotes the expert chosen at time t by ALG playing against G (i.e. the component o_t in the transcript $\mathsf{Trans}(\mathsf{ALG}, G)$).

Inequality (2) gives a useful upper bound on regret in cases where it is possible to bound, *a priori*, the cost of the best expert x^*. Two such cases are the following.

Finite time horizon Say that an adversary G has *time horizon* T if $c_t(x) = 0$ for all $t > T$ and $x \in [n]$, regardless of the choices made by the algorithm. Denote the set of all such adversaries by $\mathcal{G}[T]$. For an adversary in $\mathcal{G}[T]$, $\mathsf{Cost}(x^*) \leq T$.

Geometric discounting Say that an adversary G has *discount factor* $\delta = 1 - r$ if r is a positive constant and there is another adversary $hatG$ — with cost functions $\hat{c}_t$ taking values between 0 and 1 — such that $c_t = \delta^t \hat{c}_t$. Denote the set of all such adversaries by $\mathcal{G}\langle 1-r \rangle$. For an adversary in $\mathcal{G}\langle 1-r \rangle$, $\mathsf{Cost}(x^*) \leq 1/r$.

By choosing $\varepsilon = \sqrt{\ln(n)/2T}$ in the case of a finite time horizon T, or $\varepsilon = \sqrt{r\ln(n)/2}$ in the case of discount factor $1-r$, we obtain upper bounds on the regret of Hedge against the adversary sets $\mathcal{G}[T]$ and $\mathcal{G}\langle 1-r \rangle$.

$$R\left(\mathsf{Hedge}\left(\sqrt{\ln(n)/2T}\right), \mathcal{G}[T]\right) \leq 2\sqrt{2T\ln(n)} \tag{3}$$

$$R\left(\mathsf{Hedge}\left(\sqrt{r\ln(n)/2}\right), \mathcal{G}\langle 1-r \rangle\right) \leq 2\sqrt{2\ln(n)/r} \tag{4}$$

2.1 The doubling trick

To achieve the regret bound (3), the algorithm designer must know the time horizon T in advance, to specify the appropriate value of ε when the algorithm is initialized. We can avoid this assumption that T is known in advance, at the expense of a constant factor, using the *doubling trick*. Whenever we reach a time step t such that t is a power of 2, we restart the algorithm (forgetting all of the information gained in the past) setting ε to $\sqrt{\ln(n)/2t}$. Let us denote this algorithm by Hedge^*. If $2^k \leq T < 2^{k+1}$, the algorithm satisfies the following upper bound on its regret:

$$\begin{aligned}
R(\mathsf{Hedge}^*, \mathcal{G}[T]) &= \sup_{G \in \mathcal{G}[T]} \mathbf{E}\left[\sup_{x\in[n]} \sum_{t=1}^{T} c_t(x_t) - c_t(x)\right] \\
&= \sup_{G \in \mathcal{G}[T]} \mathbf{E}\left[\sup_{x\in[n]} \sum_{j=0}^{k} \sum_{t=2^j}^{2^{j+1}-1} c_t(x_t) - c_t(x)\right] \\
&\leq \sum_{j=0}^{k} \sup_{G \in \mathcal{G}[2^j]} \mathbf{E}\left[\sum_{x\in[n]} \sum_{t=1}^{2^j} c_t(x_t) - c_t(x)\right] \\
&= \sum_{j=0}^{k} R\left(\mathsf{Hedge}\left(\sqrt{ln(n)/2^{j+1}}\right), \mathcal{G}[2^j]\right) \\
&\leq \sum_{j=0}^{k} 2\sqrt{2^{j+1}\ln(n)} \\
&< 2\sqrt{2^{k+1}\ln(n)} \sum_{i=0}^{\infty} 2^{-i/2} \\
&< 7\sqrt{T\ln(n)}.
\end{aligned}$$

We can use this doubling trick whenever we have an algorithm with known time horizon T, whose regret is $O(T^\alpha)$ for some $\alpha > 0$, to obtain another algorithm with unknown time horizon, whose regret is $O(T^\alpha)$ for all time horizons T.

2.2 A lower bound for regret

The regret bound (3) is information-theoretically optimal up to a constant factor: a matching lower bound of $\Omega\left(\sqrt{T\ln(n)}\right)$ arises by considering an input in which the costs $\{c_t(x) : 1 \leq t \leq T, x \in [n]\}$ costitute a set of Tn independent uniformly distributed random samples from $\{0, 1\}$. The central limit theorem tells us that with high probability, there is an expert whose total cost is $\frac{T}{2} - \Omega\left(\sqrt{T\ln(n)}\right)$. On the other hand, it is obvious that any randomized algorithm will have expected cost $T/2$.

3 Normal-form games, mixed strategies, and Nash equilibria

3.1 Definitions

Definition 3. A normal-form game is specified by:

- A set $\mathcal{I}$ of *players*.
- For each player $i \in \mathcal{I}$, a set A_i of *strategies*.
- For each player $i \in \mathcal{I}$, a *payoff function*

$$u_i : \prod_{i \in \mathcal{I}} A_i \to \mathbb{R}.$$

When a normal-form game has two players, we will generally refer to them as the *row player* (player 1) and the *column player* (player 2) and we will write the payoff functions in a matrix whose rows and columns are indexed by elements of A_1 and A_2, respectively. The entry in row r and column c of the matrix is the ordered pair $(u_1(r,c), u_2(r,c))$.

3.2 Examples of two-player normal-form games

Example 1. (Bach or Stravinsky) Two players have to decide whether to go to a concert of Bach or of Stravinsky. One prefers Bach, the other prefers Stravinsky, but both of them prefer going to a concert together over going alone.

	B	S
B	(2,1)	(0,0)
S	(0,0)	(1,2)

Example 2. (Bach and Stravinsky) Two players live in neighboring rooms. Each must decide whether to play their music at low volume or at high volume. Each one would prefer to play their own music at high volume and would prefer their neighbor's music to be played at low volume.

	Q	L
Q	(3,3)	(1,4)
L	(4,1)	(2,2)

This game is a form of the famous *prisoner's dilemma* game. Each player is better off playing "L" no matter what the opponent's strategy is. Yet the outcome (L,L) is worse for both players than the outcome (Q,Q).

Example 3. (Penalty kick) There are two players: striker and goalie. Each must choose whether to go left or right. If both choose the same direction, the goalie wins. If both choose opposite directions, the striker wins.

	L	R
L	(-1,1)	(1,-1)
R	(1,-1)	(-1,1)

This game is sometimes called *matching pennies*.

3.3 Mixed strategies

A *mixed strategy* for a player of a normal-form game is a rule for picking a random strategy from its strategy set. More formally, for a finite set A, let $\boldsymbol{\Delta}(A)$ denote the set of all probability distributions on A, i.e.

$$\boldsymbol{\Delta}(A) = \left\{ p : A \to [0,1] \;\middle|\; \sum_{a \in A} p(a) = 1 \right\}.$$

(This definition can be extended to infinite sets using measure theory, but the formalism required to deal with this extension is outside the scope of this course.) The elements of $\boldsymbol{\Delta}(A_i)$ are called *mixed strategies* of player i.

Elements of $\prod_{i \in \mathcal{I}} A_i$ are called *pure strategy profiles*. Elements of $\prod_{i \in \mathcal{I}} \boldsymbol{\Delta}(A_i)$ are called *mixed strategy profiles*. The payoff function of a game can be extended from pure strategy profiles to mixed strategy profiles by defining the payoff of a mixed strategy profile to be the expected payoff when every player samples their random strategy independently:

$$u_i(p_1, p_2, \ldots, p_{|\mathcal{I}|}) = \sum_{\vec{a}} u_i(\vec{a}) p_1(a_1) p_2(a_2) \ldots p_{|\mathcal{I}|}(a_{|\mathcal{I}|}),$$

where the sum runs over all pure strategy profiles $\vec{a} = (a_1, a_2, \ldots, a_{|\mathcal{I}|})$.

3.4 Nash equilibrium

Let $k = |\mathcal{I}|$. For a strategy profile $\vec{a} = (a_1, a_2, \ldots, a_k)$, and an element $a'_i \in A_i$, we introduce the notation (a'_i, a_{-i}) to denote:

$$(a'_i, a_{-i}) = (a_1, a_2, \ldots, a_{i-1}, a'_i, a_{i+1}, \ldots, a_k).$$

In words, (a'_i, a_{-i}) is the strategy profile obtained by changing player i's strategy from a_i to a'_i.

Definition 4 (Nash equilibrium). A mixed strategy profile $\vec{p} = (p_1, p_2, \ldots, p_k)$ is a *mixed Nash equilibrium* (or, simply, Nash equilibrium) if it is the case that for all $i \in \mathcal{I}$ and all $q_i \in \mathbf{\Delta}(A_i)$,

$$u_i(q_i, p_{-i}) \leq u_i(p_i, p_{-i}).$$

If each p_i is a pure strategy (i.e. a mixed strategy which assigns probability 1 to a single element of A_i) then $\vec{p}$ is a *pure Nash equilibrium.*

Example 4. There are two pure Nash equilibria of "Bach or Stravinsky", namely (B,B) and (S,S). There is also another mixed Nash equilibrium in which player 1 chooses B with probability 2/3, S with probability 1/3; and player 2 chooses B with probability 1/3, S with probability 2/3. Interestingly, in the mixed equilibrium the payoff for both players is 2/3, so *both* of the pure equilibria are better for *both* players. (In game-theoretic terms, the mixed Nash equilibrium is *Pareto dominated* by the pure equilibria. If x and y are two outcomes of a game, we say that x Pareto dominates y when at least one player strictly prefers x to y, and no player strictly prefers y to x.) This example illustrates three of the critiques of the Nash equilibrium concept.

1. In situations like this where there are multiple Nash equilibria, we can't predict which equilibrium the players will choose (if any).

2. Moreover, the different equilibria imply different payoffs for the two players, so we can't even predict their payoffs.

3. The contention that players will select a Nash equilibrium of the game seems to rely on circular reasoning. Player 1 wants to play its side of the equilibrium strategy profile because it believes that player 2 will play its own side of the same strategy profile, and vice-versa. In situations where there are multiple equilibria, why should we assume that both players will be able to coordinate their beliefs in this way?

These critiques of Nash equilibrium are not the end of the story; they are the beginning of a very interesting story in which game theorists have tried to enhance the theory in various ways to address these critiques and improve the predictive power of game-theoretic equilibrium concepts. In particular, some game theorists have tried addressing critique (3) using models in which players arrive at an equilibrium by playing the game repeatedly and using learning rules to adapt to their opponent's past behavior. This theory of learning in games will be one of the main topics we address in the coming weeks.

Example 5. The "Bach and Stravinsky" game (i.e. the prisoner's dilemma) has only one Nash equilibrium, the pure strategy profile (L,L). This is a *dominant strategy equilibrium*, meaning that each player is better off playing L than Q, *no matter what the other player does.*

Example 6. The "penalty kick" game has no pure Nash equilibrium. In the unique mixed Nash equilibrium, each player assigns probability 1/2 to both strategies.

4 Two-player zero-sum games and von Neumann's minimax theorem

Definition 5. A *two-player zero-sum game* is one in which $\mathcal{I} = \{1, 2\}$ and $u_2(a_1, a_2) = -u_1(a_1, a_2)$ for all pure strategy profiles (a_1, a_2).

A famous theorem of von Neumann illustrates that the equilibria of two-player zero-sum games are much simpler than the equilibria of general two-player games.

Theorem 1 (von Neumann's minimax theorem). *For every two-player zero-sum game with finite strategy sets A_1, A_2, there is a number $v \in \mathbb{R}$, called the* game value*, such that:*

1.
$$v = \max_{p\in\mathbf{\Delta}(A_1)} \min_{q\in\mathbf{\Delta}(A_2)} u_1(p, q) = \min_{q\in\mathbf{\Delta}(A_2)} \max_{p\in\mathbf{\Delta}(A_1)} u_1(p, q)$$

2. *The set of mixed Nash equilibria is nonempty. A mixed strategy profile (p, q) is a Nash equilibrium if and only if*
$$\begin{aligned} p &\in \arg\max_p \min_q u_1(p, q) \\ q &\in \arg\min_q \max_p u_1(p, q) \end{aligned}$$

3. *For all mixed Nash equilibria (p, q), $u_1(p, q) = v$.*

Remark 1. Among other things, the theorem implies that two-player zero-sum games don't suffer from critiques (2) and (3) discussed above. Although there can be multiple equilibria, part (3) of the theorem says that all equilibria result in the same payoffs for both players. Moreover, the players don't need to coordinate with each other in order to play an equilibrium: by part (2) it is sufficient for each of them to choose a mixed strategy in their respective arg max or arg min set, without knowing which equilibrium mixed strategy the opponent is going to choose.

4.1 Some extra observations about Hedge

Before giving the proof of Theorem 1 we must point out two simple properties of the Hedge algorithm which were not derived in previous lectures.

4.1.1 Using Hedge for maximization problems.

Although we defined and analyzed Hedge in the context of online cost-minimization problems, it is easy to adapt the algorithm to the setting of online payoff maximization. The simplest way to do this is to just transform payoff functions into cost functions. If $g_t : [n] \to [0,1]$ is the payoff function at time t, then define $c_t(x) = 1 - g_t(x)$ and use Hedge to compute a sequence of experts x_t which approximately minimize $\sum_{t=1}^{\infty} c_t(x_t)$, which is the same as approximately maximizing $\sum_{t=1}^{\infty} g_t(x_t)$. This suffices for the purpose of bounding the additive regret when using Hedge for maximization problems, since the additive difference between Hedge and the best expert is unaffected by the transformation $c_t(x) = 1 - g_t(x)$. (In other words, if x^* is the best expert, then $c_t(x_t) - c_t(x^*)$ is equal to $g_t(x^*) - g_t(x_t)$; summing over t we find that the algorithm's regret is the same in both the maximization and minimization contexts.)

For future reference, Appendix A presents a slightly different version of Hedge, denoted by MaxHedge, which is suitable for maximization problems. In the appendix we prove the following multiplicative bound which is analogous to Theorem 3 from last week's notes.

Theorem 2. *For every randomized adaptive adversary, for every $T > 0$, the expected payoff gained by* MaxHedge(ε) *satisfies*

$$\mathbf{E}\left[\sum_{t=1}^{T} g_t(x_t)\right] > (1-\varepsilon)\,\mathbf{E}\left[\max_{x\in[n]}\sum_{t=1}^{T} g_t(x)\right] - \left(\frac{1}{\varepsilon}\right)\ln(n). \tag{5}$$

Corollary 3. *For every $T > 0$, if $\varepsilon = \sqrt{\ln(n)/T}$, the expected payoff gained by* MaxHedge(ε) *against any adaptive adversary satisfies*

$$\mathbf{E}\left[\sum_{t=1}^{T} g_t(x_t)\right] > \mathbf{E}\left[\max_{x\in[n]}\sum_{t=1}^{T} g_t(x)\right] - 2\sqrt{T\ln(n)}. \tag{6}$$

In the rest of these notes, we will not distinguish between the algorithms Hedge and MaxHedge.

4.1.2 Hedge tracks the best mixture of experts

For the best-expert problem, we can extend each payoff function g_t from a function on $[n]$ to a function on $\mathbf{\Delta}([0,1])$ by averaging:

$$g_t(p) = \sum_{x\in[n]} p(x) g_t(x).$$

Observe that for any $p \in \mathbf{\Delta}([0,1])$,

$$\sum_{t=1}^{T} g_t(p) \le \max_{x \in [n]} \sum_{t=1}^{T} g_t(x),$$

since the left side is a weighted average of the values of $\sum_{t=1}^{T} g_t(x)$ as x runs over all elements of $[n]$. Using this fact, we see that the bounds (5) and (6) extend to distributions:

$$\mathbf{E}\left[\sum_{t=1}^{T} g_t(x_t)\right] > (1-\varepsilon)\,\mathbf{E}\left[\max_{p \in \mathbf{\Delta}([n])} \sum_{t=1}^{T} g_t(p)\right] - \left(\frac{1}{\varepsilon}\right)\ln(n). \tag{7}$$

$$\mathbf{E}\left[\sum_{t=1}^{T} g_t(x_t)\right] > \mathbf{E}\left[\max_{p \in \mathbf{\Delta}([n])} \sum_{t=1}^{T} g_t(p)\right] - 2\sqrt{T\ln(n)}. \tag{8}$$

4.2 The main lemma

The hardest step in the proof of von Neumann's minimax theorem is to prove that

$$\max_p \min_q u_1(p,q) \ge \min_q \max_p u_1(p,q).$$

We will prove this fact using online learning algorithms. The basic idea of the proof is that if the players are allowed to play the game repeatedly, using Hedge to adapt to the other player's moves, then low-regret property of Hedge guarantees that the time-average of each player's mixed strategy is nearly a best response to the time-average of the other player's mixed strategy.

Lemma 4. *For any two-player zero-sum game,*

$$\max_{p \in \mathbf{\Delta}(A_1)} \min_{q \in \mathbf{\Delta}(A_2)} u_1(p,q) \ge \min_{q \in \mathbf{\Delta}(A_2)} \max_{p \in \mathbf{\Delta}(A_1)} u_1(p,q).$$

Proof. Without loss of generality, assume that $0 \le u_1(a_1,a_2) \le 1$ for all $a_1 \in A_1, a_2 \in A_2$. (If the original payoff function u_1 doesn't satisfy these bounds, we can replace u_1 with the function $bu_1 + c$ for suitable constants $b, c > 0$.)

Let $n = \max\{|A_1|, |A_2|\}$ and for any positive $\delta > 0$ let

$$\begin{aligned} T &= \lceil 4\ln(n)/\delta^2 \rceil \\ \varepsilon &= \sqrt{\ln(n)/T}. \end{aligned}$$

Suppose that each player uses Hedge(ε) (with "experts" corresponding to elements of the player's strategy set) to define a T-step sequence of mixed strategies in response to the other player's sequence of mixed strategies. More precisely, player 1 defines a sequence of mixed strategies $p_1, p_2, \dots, p_T$ and player 2 defines a sequence of mixed strategies $q_1, q_2, \ldots, q_T$, according to the following prescription. Player 1 runs the payoff-maximizaton version of $\mathsf{Hedge}(\varepsilon)$, defining the payoff function at time t by $g_t(x) = u_1(x, q_t)$. The mixed strategy p_t is taken to be the distribution from which the algorithm samples at time t, i.e. $p_t(x) = w_{xt} \Big/ \left(\sum_{y \in A_1} w_{yt}\right)$, where w_{xt} is the weight which $\mathsf{Hedge}(\varepsilon)$ assigns to strategy x at time t. Similarly, player 2 runs the payoff-maximization version of $\mathsf{Hedge}(\varepsilon)$, defining the payoff function at time t by $g_t(x) = 1 - u_1(p_t, x)$, and q_t is taken to be the distribution from which the algorithm samples at time t.

Our choice of T and ε guarantees that each player's regret is at most δT, using Corollary 3. Hence we have

$$\min_q \frac{1}{T} \sum_{t=1}^{T} u_1(p_t, q) + \delta \;\; \geq \;\; \frac{1}{T} \sum_{t=1}^{T} u_1(p_t, q_t) \;\; \geq \;\; \max_p \frac{1}{T} \sum_{t=1}^{T} u_1(p, q_t) - \delta \tag{9}$$

where the first inequality follows from considering Player 2's regret, and the second inequality follows from considering Player 1's regret. Letting

$$\overline{p} = \frac{1}{T} \sum_{t=1}^{T} p_t, \qquad \overline{q} = \frac{1}{T} \sum_{t=1}^{T} q_t,$$

we can rewrite (9) as

$$\min_q u_1(\overline{p}, q) + \delta \;\; \geq \;\; \max_p u_1(p, \overline{q}) - \delta. \tag{10}$$

Trivially, we have

$$\max_p \min_q u_1(p, q) + \delta \;\; \geq \;\; \min_q u_1(\overline{p}, q) + \delta \tag{11}$$

$$\max_p u_1(p, \overline{q}) - \delta \;\; \geq \;\; \min_q \max_p u_1(p, q) - \delta \tag{12}$$

and combining (10)-(12) we find that

$$\max_p \min_q u_1(p, q) + \delta \geq \min_q \max_p u_1(p, q) - \delta. \tag{13}$$

The lemma follows because δ can be made arbitrarily close to zero. □

It will be useful to considering the following alternate proof of Lemma 4, which is almost identical to the first proof.

Alternate proof of Lemma 4. As before, assume without loss of generality that $0 \leq u_1(a_1, a_2) \leq 1$ for all strategy profiles (a_1, a_2). Let $\delta > 0$ be an arbitrarily small positive number, and define n, T, ε as above. Player 1 still uses $\mathsf{Hedge}(\varepsilon)$ to define a sequence of mixed strategies $p_1, p_2, \ldots, p_T$ in response to the payoff function induced by the opponent's sequence of strategies. But player 2 now chooses its strategies adversarially, according to the prescription

$$q_t \in \arg\min_q u_1(p_t, q). \tag{14}$$

Note that the set of mixed strategies minimizing player 1's payoff always contains a pure strategy, so we may assume q_t is a pure strategy if desired.

Define $\overline{p}, \overline{q}$ as above. We find that

$$\begin{aligned}
\max_p \min_q u_1(p,q) &\geq \min_q u_1(\overline{p}, q) \\
&= \min_q \frac{1}{T} \sum_{t=1}^{T} u_1(p_t, q) \\
&\geq \frac{1}{T} \sum_{t=1}^{T} \min_q u_1(p_t, q) \\
&= \frac{1}{T} \sum_{t=1}^{T} u_1(p_t, q_t) \\
&\geq \max_p \frac{1}{T} \sum_{t=1}^{T} u_1(p, q_t) - \delta \\
&= \max_p u_1(p, \overline{q}) - \delta \\
&\geq \min_q \max_p u_1(p,q) - \delta.
\end{aligned}$$

□

4.3 Proof of Theorem 1

In this section we complete the proof of von Neumann's minimax theorem.

Proof of Theorem 1. For any mixed strategy profile $(\hat{p}, \hat{q})$ we have

$$u_1(\hat{p}, \hat{q}) \leq \max_p u_1(p, \hat{q}).$$

Taking the minimum of both sides as $\hat{q}$ ranges over $\mathbf{\Delta}(A_2)$ we find that

$$\min_q u_1(\hat{p}, q) \leq \min_q \max_p u_1(p,q).$$

Taking the maximum of both sides as $\hat{p}$ ranges over $\mathbf{\Delta}(A_1)$ we find that

$$\max_p \min_q u_1(p,q) \leq \min_q \max_p u_1(p,q).$$

The reverse inequality was proven in Lemma 4. Thus we have established part (1) of Theorem 1.

Note that the sets $B_1 = \arg\max_p \min_q u_1(p,q)$ and $B_2 = \arg\min_q \max_p u_1(p,q)$ are both nonempty. (This follows from the compactness of $\boldsymbol{\Delta}(A_1)$ and $\boldsymbol{\Delta}(A_2)$, the contintuity of u_1, and the finiteness of A_1 and A_2.) If $p \in B_1$ and $q \in B_2$ then

$$v = \min_q u_1(p,q) \leq u_1(p,q) \leq \max_p u_1(p,q) = v$$

hence $u_1(p,q) = v$. Moreover, since $q \in B_2$, player 1 can't achieve a payoff greater than v against q by changing its mixed strategy. Similarly, since $p \in B_1$, player 2 can't force player 1's payoff to be less than v by changing its own mixed strategy. Hence (p,q) is a Nash equilibrium. Conversely, if (p,q) is a Nash equilibrium, then

$$u_1(p,q) = \max_p u_1(p,q) \geq v \tag{15}$$

$$u_1(p,q) = \min_q u_1(p,q) \leq v \tag{16}$$

and this implies that in each of (15), (16), the inequality on the right side is actually an *equality*, which in turn implies that $p \in B_1$ and $q \in B_2$. This completes the proof of (2) and (3). □

The proof of the minimax theorem given here, using online learning, differs from the standard proof which uses ideas from the theory of linear programming. The learning-theoretic proof has a few advantages, some of which are spelled out in the following remarks.

Remark 2. The procedure of using Hedge to approximately solve a zero-sum game is remarkably fast: it converges to within δ of the optimum using only $O(\log(n)/\delta^2)$ steps, provided the payoffs are between 0 and 1. (By "converges to within δ", we mean that it outputs a pair of mixed strategies, $(\overline{p}, \overline{q})$ such that $\min_q u_1(\overline{p}, q) \geq v - \delta$ and $\max_p u_1(p, \overline{q}) \leq v + \delta$, where v is the game value.) This is especially important when one of the players has a strategy set whose size is exponentially larger than the size of the natural representation of the game. See Example 7 below for an example of this.

Recall that in the second proof of Lemma 4 we remarked that player 2's strategies q_t could be taken to be pure strategies. Note also that the Hedge algorithm used by player 1 only needs to look at the scores in a column of the payoff matrix if the corresponding strategy has been used by player 2 some time in the past. Thus, as long as we have an oracle for finding player 2's best response to any mixed strategy, we need only look at a very sparse subset of the payoff matrix — a set of $O(\log(n)/\delta^2)$ columns — to compute a mixed strategy for player 1 which obtains an additive δ-approximation to the game value. Again, see Example 7 for an example in which it is reasonable to assume that we don't have an explicit representation of the payoff matrix, but we can examine any desired column and we have an oracle for finding player 2's best response to any mixed strategy.

Example 7 (The VPN eavesdropping game). Let $G = (V, E)$ be an undirected graph. In the "VPN eavesdropping game", player 1 chooses an edge of G, and player 2 chooses a spanning tree of G. For any edge e and spanning tree T, the payoff of player 1 is

$$u_1(e, T) = \begin{cases} 1 & \text{if } e \in T \\ 0 & \text{otherwise.} \end{cases}$$

(We can think of player 1 as an eavesdropper who can listen on any single edge of G, and player 2 as someone who is setting up a virtual private network on the edges of T, to join together all the nodes of G. The game is a win for player 1 if he or she eavesdrops on an edge which is part of the VPN.)

Note that, in general, the cardinality of player 2's strategy set is exponential in the size of G. Thus the parameter n appearing in the proof of Lemma 4 will be exponential in the size of the game's natural representation. However, it is easy to examine any particular column of the payoff matrix u_1: the column corresponding to a spanning tree T will be a vector of 0's and 1's, with 1's in the rows corresponding to the edges of T. Moreover, it is easy to compute player 2's best response to any mixed strategy of player 1: one simply computes a minimum spanning tree of G, where the weight of each edge is equal to the probability of player 1 picking that edge.

Consequently, there is an algorithm for approximately solving the game (up to an additive error of δ) which requires only $O(\log(M)/\delta^2)$ minimum spanning tree computations, where M is the total number of minimum spanning trees of G. (If G has V vertices, then by Cayley's formula $M \leq V^{V-2}$. Hence $\log(M)$ is always polynomial — in fact, nearly linear — in the number of vertices of G.)

Remark 3. Another consequence of the second proof of Lemma 4 is that player 2 has a mixed strategy which has *sparse support* — i.e. at most $O(\log(n)/\delta^2)$ strategies have positive probability — yet it achieves an additive δ-approximation to the game value. By symmetry, player 1 also has a mixed strategy with sparse support which achieves an additive δ-approximation to the game value. Hence the game has an approximate Nash equilibrium in which both players use sparsely-supported mixed strategies.

Remark 4. If player 2 is not playing rationally, by using Hedge player 1 comes close to achieving the best possible payoff against whatever distribution of strategies player 2 happens to be using. This property would not be ensured in repeated play if player 1 instead solved the game offline, picked a strategy in $\arg\max_p \min_q u_1(p, q)$, and always used this strategy.

Remark 5. If we think about our intuition of how human beings learn to play games against each other, the process is probably more similar to a learning algorithm such as Hedge than to a linear programming algorithm such as the simplex method. Hence another benefit of the learning-theoretic proof is that it gives an intuitive justification for why human beings are able to find the equilibria of zero-sum games.

4.4 Yao's lemma

The von Neumann minimax theorem has an important consequence in computer science. Suppose we have a computational problem with a finite set of possible inputs I, and we are considering a finite set of possible algorithms A. For example, I might be the set of all n-bit binary strings, and A might be the set of all Boolean circuits of size at most n^3 which accept an n-bit input and return a valid output for the problem under consideration. Suppose we have a parameter $t(i,a)$ which corresponds to the cost of running algorithm a on input i. For example, $t(i,a)$ could denote the algorithm's running time, or the cost of the solution it computes.

We may interpret this scenario as a two-player zero-sum game in which player 1 specifies an input, player 2 specifies an algorithm, and $t(i,a)$ is the payoff for player 1. Let $\mathcal{D} = \boldsymbol{\Delta}(I)$ denote the set of all probability distributions on inputs, and let $\mathcal{R} = \boldsymbol{\Delta}(A)$ denote the set of all probability distributions on algorithms, i.e. the set of all randomized algorithms. We can extend the function t to mixed strategy profiles in the usual way, i.e.

$$t(d,r) = \sum_{i \in I} \sum_{a \in A} t(i,a) d(i) r(a).$$

Lemma 5 (Yao's Lemma).

$$\max_{d \in \mathcal{D}} \min_{a \in A} t(d,a) = \max_{d \in \mathcal{D}} \min_{r \in \mathcal{R}} t(d,r) = \min_{r \in \mathcal{R}} \max_{d \in \mathcal{D}} t(d,r) = \min_{r \in \mathcal{R}} \max_{i \in I} t(i,r).$$

Proof. The second equality is a restatement of von Neumann's minimax theorem. The first and third equalities follow from the fact that for any mixed strategy of one player, the other player always has a best response which is a pure strategy, i.e.

$$\begin{aligned} \forall d \in \mathcal{D} \ \min_{r \in \mathcal{R}} t(d,r) &= \min_{a \in A} t(d,a) \\ \forall r \in \mathcal{R} \ \max_{d \in \mathcal{D}} t(d,r) &= \max_{i \in I} t(i,r). \end{aligned}$$

□

5 Learning equilibria in non-zero sum games

Here we present a short example to illustrate that the dynamics of learning processes in non-zero-sum games can be much more complicated. We will consider a variant of the well-known "rock, paper, scissors" game, with the following payoff matrix.

	R	P	S
R	(-5,-5)	(-1,1)	(1,-1)
P	(1,-1)	(-5,-5)	(-1,1)
S	(-1,1)	(1,-1)	(-5,-5)

Consider the dynamics studied in the second proof of Lemma 4, i.e. player 1 uses Hedge(ε) to choose mixed strategies p_t, and player 2 responds adversarially with a pure strategy $q_t \in \arg\min_q u_1(p_t, q)$. If one looks at the strategies of both players during the course of an infinite time history, the timeline is divided into epochs during which player 2 is always picking the same strategy in {R,P,S}, and player 1 is adjusting its mixed strategy accordingly. The lengths of these epochs are approximated by a geometric progression. For example, in an epoch when player 2 is always picking R, player 1 is increasing the probability of P, decreasing the probability of S, and *rapidly* decreasing the probability of R. At some point when the probability of S is small enough and the probability of P is large enough, player 2 will shift to playing S. Player 2 will continue playing S until player 1 increases the probability of R enough to make P more attractive than S for player 2. However, this shift from S to P takes longer (by a constant factor) than the previous shift from R to S, because player 1 decreased the weight assigned to R very rapidly during the period when player 2 was playing R, and player 1 increased the weight assigned to R much more slowly during the period when player 2 was playing S.

As a consequence of these observations, we see that the average of the strategies chosen by player 2 (the mixed strategy denoted by $\overline{q}$ in the proof of Lemma 4) never converges! Similarly, the average of the strategies chosen by player 1 never converges. So the description of the type of equilibrium achieved by this process (if any) must be significantly more complicated than Nash equilibrium. During the next few weeks we will be discussing equilibrium concepts for non-zero-sum games and analyzing the types of equilibria which arise as the limiting outcomes of different learning processes.

A Appendix: The MaxHedge algorithm

The algorithm MaxHedge(ε) — a version of Hedge suited for payoff maximization rather than cost minimization — is presented in Figure 1. In this section we analyze the algorithm, proving Theorem 2 and Corollary 3.

Lemma 6. *For $x > 0$,*

$$\frac{1}{x}\ln(1+x) > 1 - x. \tag{17}$$

Proof. We have

$$\ln\left(\frac{1}{1+x}\right) = \ln\left(1 - \frac{x}{1+x}\right) < -\left(\frac{x}{1+x}\right).$$

Multiplying both sides by $-1/x$,

$$\frac{1}{x}\ln(1+x) > \frac{1}{1+x}.$$

Finally, the inequality $1 > (1-x)(1+x)$ implies that $\frac{1}{1+x} > 1 - x$, which concludes the proof of the lemma. □

Algorithm $\mathsf{MaxHedge}(\varepsilon)$

/* Initialization */
$w_x \leftarrow 1$ **for** $x \in [n]$

/* Main loop */
for $t = 1, 2, \ldots$
    /* Define distribution for sampling random strategy */
    for $x \in [n]$
        $p_t(x) \leftarrow w_x \Big/ \left(\sum_{y=1}^{n} w_y \right)$
    end
    Choose $x_t \in [n]$ at random according to distribution p_t.
    Observe payoff function g_t.

    /* Update score for each strategy */
    for $x \in [n]$
        $w_x \leftarrow w_x \cdot (1+\varepsilon)^{g_t(x)}$
    end
end

Figure 1: The algorithm $\mathsf{MaxHedge}(\varepsilon)$.

Proof of Theorem 2. Let w_{xt} denote the value of w_x at the beginning of the t-th iteration of the main loop, and let $W_t = \sum_{x=1}^{n} w_{xt}$. Note that w_{xt}, W_t are random variables, since they depend on the adversary's choices which in turn depend on the algorithm's random choices in previous steps. For an expert $x \in [n]$, let $g_{1..T}(x)$ denote the total payoff

$$g_{1..T}(x) = \sum_{t=1}^{T} g_t(x).$$

Let $x^* = \arg\max_{x \in [n]} g_{1..T}(x)$. We have

$$W_T > w_{x^*t} = (1+\varepsilon)^{g_{1..T}(x^*)}$$

and after taking logarithms of both sides this becomes

$$\ln(W_T) > \ln(1+\varepsilon) g_{1..T}(x^*) \tag{18}$$

On the other hand, we can bound the expected value of $\ln(W_T)$ from above, using an inductive argument. Let w_{*t} denote the vector of weights $(w_{1t}, \ldots, w_{nt})$.

$$\mathbf{E}(W_{t+1} \,|\, w_{*t}) \;=\; \sum_{x=1}^{n} \mathbf{E}\left((1+\varepsilon)^{g_t(x)} w_{xt} \,|\, w_{*t} \right) \tag{19}$$

$$
\begin{aligned}
&\leq \sum_{x=1}^{n} \mathbf{E}\left((1+\varepsilon g_t(x)) w_{xt} \,|\, w_{*t}\right) && (20)\\
&= \sum_{x=1}^{n} w_{xt} + \varepsilon \mathbf{E}\left(\sum_{x=1}^{n} g_t(x) w_{xt} \,|\, w_{*t}\right) && (21)\\
&= W_t \cdot \left(1 + \varepsilon \mathbf{E}\left(\sum_{x=1}^{n} g_t(x) p_t(x) \,|\, w_{*t}\right)\right) && (22)\\
&= W_t \cdot (1 + \varepsilon \mathbf{E}(g_t(x_t) \,|\, w_{*t})) && (23)\\
\mathbf{E}(\ln(W_{t+1}) \,|\, w_{*t}) &\leq \ln(W_t) + \ln(1 + \varepsilon \mathbf{E}(g_t(x_t) \,|\, w_{*t})) && (24)\\
&\leq \ln(W_t) + \varepsilon \mathbf{E}(g_t(x_t) \,|\, w_{*t}) && (25)\\
\mathbf{E}(\ln(W_{t+1}) \,|\, w_{*t}) - \ln(W_t) &\leq \varepsilon \mathbf{E}(g_t(x_t) \,|\, w_{*t}) && (26)\\
\mathbf{E}(\ln(W_{t+1})) - \mathbf{E}(\ln(W_t)) &\leq \varepsilon \mathbf{E}(g_t(x_t)) && (27)\\
\mathbf{E}(\ln(W_T)) - \ln(n) &\leq \varepsilon \mathbf{E}\left(\sum_{t=1}^{T} g_t(x_t)\right) && (28)
\end{aligned}
$$

Here, (20) is derived using the identity $(1+\varepsilon)^x \leq 1+\varepsilon x$, which is valid for $\varepsilon > 0$ and $0 \leq x \leq 1$. Step (22) is derived using the fact that $p_t(x) = w_{xt}/W_t$, (23) is derived using the observation that x_t is a random element sampled from the probability distribution $p_t(\cdot)$ on $[n]$, (24) is derived using Jensen's inequality, (27) is derived by taking the unconditional expectation of both sides of the inequality, and (28) is derived by summing over t and recalling that $W_0 = n$.

Combining (18) and (28) we obtain

$$
\begin{aligned}
\varepsilon \mathbf{E}\left(\sum_{t=1}^{T} g_t(x_t)\right) &> \ln(1+\varepsilon)\mathbf{E}(g_{1..T}(x^*)) - \ln(n)\\
\mathbf{E}\left(\sum_{t=1}^{T} g_t(x_t)\right) &> \frac{1}{\varepsilon}\ln\left(\frac{1}{1+\varepsilon}\right)\mathbf{E}(g_{1..T}(x^*)) - \left(\frac{1}{\varepsilon}\right)\ln(n)\\
\mathbf{E}\left(\sum_{t=1}^{T} g_t(x_t)\right) &> (1-\varepsilon)\,\mathbf{E}(g_{1..T}(x^*)) - \left(\frac{1}{\varepsilon}\right)\ln(n) && (29)
\end{aligned}
$$

where the last line is derived using identity (17) from the Lemma above. □

Proof of Corollary 3. The corollary follows by combining (29) above with the trivial bound $\mathbf{E}(g_{1..T}(x^*)) \leq T$. □

CS 6840 – Algorithmic Game Theory	Spring 2012

Lecture 14 – Wednesday 22 February 2012 – Scribe Notes

Instructor: Eva Tardos — *Boris Burkov (bb393)*

Coarse correlated equilibria as a convex set

Last time - Looked at algorithm that guarantees no regret Last last time - Defined coarse correlated equilibrium as a probability distribution on strategy vectors

Definition. $p(s)$ s.t. $E(u_i(s)) \geq E(u_i(x, s_{-i}))\ \forall i, \forall x$.

which lead to the corollary:

Corollary 1. All players using small regret strategies gives an outcome that is close to a coarse correlated equilibrium

The next natural question to ask is: Does there exist a coarse correlated equilibrium? We consider finite player and strategy sets.

Theorem 2. With finite player and strategy sets, a coarse correlated equilibrium exists.

Proof 1. We know that a Nash equilibrium exists. Then let $p_1, \ldots p_n$ be probability distributions that form a Nash equilibrium. Observe that $p(s) = \Pi_i p_i(s_i)$ is a coarse correlated equilibrium. □

Proof 2. (doesn't depend on Nash's theorem). Idea: Algorithm from last lecture finds it with small error. Consider

$$\min_p[\max_i[\max_x[E_p(u_i(x_i, s_{-i})) - E_p(u_i(s))]]]$$

The quantity inside the innermost max is the regret of players i about strategy x. If this minimum is ≤ 0, then p is a coarse correlated equilibrium. The minimum cannot equal $\epsilon > 0$ as we know by the algorithm that we can find a p with arbitrarily small regret. In this instance, $\frac{\epsilon}{2}$ would be sufficient to reach a contradiction. Hence, we know that the infimum must be less than or equal to 0 but does the minimum exist? Since we have a continuous function over p, the compact space of probability distributions, we must attain the infimum, so the minimum is in fact ≤ 0, so a coarse correlated equilibrium exists. □

Remark. This minimum can be calculated as the solution of a linear program satisfying $\sum p(s) = 1, p(s) \geq 0$ and the no regret inequality for each (i, x) pair.

2-person 0-sum games

The game is defined by a matrix a with the first players strategies labelling the rows and the second players strategies labelling the columns. a_{ij} is the amount Player 1 pays to Player 2 if strategy vector (i, j) plays.

Theorem 3. Coarse correlated equilibrium in these games is (essentially) the same as the Nash equilibrium.

To be a bit more precise, let $p(i,j)$ be at coarse correlated equilibrium. When considering Player 1, we care about q Player 2's marginal distribution. $q(j) = \sum_i p(i,j)$. Since Player 1 has no regret, we have that

$$\sum_{ij} a_{ij} p(i,j) \leq \min_i \sum_j a_{ij} q_j$$

Likewise, let $r(i) = \sum_j p(i,j)$ be Player 1's marginal distribution, so Player 2's lack of regret tells that:

$$\sum_{ij} a_{ij} p(i,j) \geq \max_j \sum_i a_{ij} r_i$$

Theorem 4. q, r from above are Nash equilibria.

Proof. The best response to q is

$$\min_i \sum_j a_{ij} q_j \leq \sum_{ij} r(i) q(j) \leq \max_j \sum_i a_{ij} r_i$$

the last of which is the best response to r. Thus, we also have

$$\sum_{ij} a_{ij} p(i,j) \leq \min_i \sum_j a_{ij} q_j \leq \sum_{ij} r(i) q(j) \leq \max_j \sum_i a_{ij} r_i \leq \sum_{ij} a_{ij} p(i,j)$$

Which implies the result, since they must all be equal. $\square$

CS 6840 – Algorithmic Game Theory (3 pages)	Spring 2012
Lecture Scribe Notes	
Instructor: Eva Tardos	*Sidharth Telang (sdt45)*

1 Lecture – Friday 17 February 2012 - Other equilibria

The following notation is used. $[n] = \{1, 2, \ldots, n\}$ is used to denote the set of players. Player i has strategy set S_i. $\bar{s}$ denotes a strategy vector. $\bar{s}_i$ denotes the i^{th} entry of $\bar{s}$ and $\bar{s}_{-i}$ denotes $\bar{s}$ without the i^{th} entry. $c_i(\bar{s})$ denotes the cost player i incurs when the players play $\bar{s}$.

We say a sequence of plays $(\bar{s}_1, \bar{s}^2, \ldots \bar{s}^T)$ is no regret for player i if and only if

$$\Sigma_{t=1}^T c_i(\bar{s}^t) \leq min_{x \in S_i} \Sigma_{t=1}^T c_i(x, \bar{s}_{-i}^t)$$

which means that player i does at least as well as he would have had he chosen any fixed strategy in hindsight.

Recall that a mixed Nash equilibrium is defined as a probability distribution p_i for every player i over S_i such that for every player i and every $x \in S_i$

$$\mathbb{E}(c_i(\bar{s})) \leq \mathbb{E}(c_i(x, \bar{s}_{-i}))$$

where $\bar{s}$ is now a random variable. That is, the probability $\bar{s}$ is played is $\Pi_i p_i(\bar{s}_i)$. Let this be denoted by $p(\bar{s})$.

Here we note that the more natural definition of enforcing the expected cost of any player i under p_i to be no more than that when i switches to any other probability distribution p'_i is equivalent to the above definition. This is because the expected cost of player i on switching to a probability distribution will be a convex combination of his expected cost on switching to fixed strategies.

A sequence of plays defines a probability distribution on the set of strategy vectors. We set $p(\bar{s})$ to be the frequency of $\bar{s}$, that is the number of times $\bar{s}$ was played divided by the total number of plays.

If a sequence of plays are no regret for all players we have for every player i

$$\Sigma_{t=1}^T c_i(\bar{s}^t) \leq min_{x \in S_i} \Sigma_{t=1}^T c_i(x, \bar{s}_{-i}^t)$$

which is equivalent to the condition that for every player i

$$\Sigma_{\bar{s}} p(\bar{s}) c_i(\bar{s}) \leq min_{x \in S_i} \Sigma_{\bar{s}} p(\bar{s}) c_i(x, \bar{s}_{-i})$$

Such a probability distribution is defined as a coarse correlated equilibrium.

Definition. A coarse correlated equilibrium is defined as a probability distribution p over strategy vectors such that for every player i

$$\Sigma_{\bar{s}} p(\bar{s}) c_i(\bar{s}) \leq min_{x \in S_i} \Sigma_{\bar{s}} p(\bar{s}) c_i(x, \bar{s}_{-i})$$

We have seen that the distribution induced by a sequence of plays that are no regret for every player is a coarse correlated equilibrium.

It's easy to see that every Nash is a coarse correlated equilibrium. But a coarse correlated equilibrium p induces a Nash equilibrium if there exists a probability distribution p_i for every player i such that for every $\bar{s}$, $p(\bar{s})$ can be expressed as $\Pi_i p_i(\bar{s}_i)$.

We look at the example of Rock-paper-scissors to find a coarse correlated equilibrium. The following table describes the payoff where (x, y) denotes that the payoff to the row player is x and that column player is y.

	R	P	S
R	(0,0)	(-1,1)	(1,-1)
P	(1,-1)	(0,0)	(-1,1)
S	(-1,1)	(1,-1)	(0,0)

This game admits a unique Nash equilibrium, the mixed Nash of choosing one of the three strategies at random.

A uniform distribution on $(R, P), (R, S), (P, R), (P, S), (S, R), (S, P)$, that is, the non-tie strategy vectors, is a coarse correlated equilibrium. We can see that if any player chooses a fixed strategy, his expected payoff will stay the same i.e. 0.

If we change the payoff table to the following

	R	P	S
R	(-2,-2)	(-1,1)	(1,-1)
P	(1,-1)	(-2,-2)	(-1,1)
S	(-1,1)	(1,-1)	(-2,-2)

then the same is a coarse correlated equilibrium, where the expected payoff per player is 0. Here choosing a fixed strategy will decrease any player's payoff to -2/3.

This modified game too has a unique Nash which is choosing each strategy uniformly at random, giving each player a negative payoff of -2/3.

Here we note that in this example the coarse correlated equilibrium is uniform over a set of strategy vectors that form a best response cycle.

We now define a correlated equilibrium.

Definition. A correlated equilibrium is defined as a probability distribution p over strategy vectors such that for every player i, and every strategy $s_i \in S_i$

$$\Sigma_{\bar{s}} p(\bar{s}|\bar{s}_i = s_i) c_i(\bar{s}) \leq min_{x \in S_i} \Sigma_{\bar{s}} p(\bar{s}|\bar{s}_i = s_i) c_i(x, \bar{s}_{-i})$$

Intuitively, it means that in such an equilibrium, every player is better off staying in the equilibirum than choose a fixed strategy, when all the other players assume that this player stays in equilibrium. Staying in equilibrium hence can be thought of as following the advice of some coordinator. In other words, when other players assume you follow your advice, you are better off following the advice than deviating from it.

We consider as an example the game of Chicken. Two players play this game, in which each either Dares to move forward or Chickens out. If both Dare, they will crash, if one Dares then he wins and the other loses and if none Dare then no one wins. The payoffs are as follows.

	D	C
D	(-10,-10)	(1,0)
C	(0,1)	(0,0)

This game has three Nash equilibra, two are pure and one is mixed. The pure equilibra are (D, C) and (C, D). The mixed Nash is choosing to Dare(D) with probability large enough to drive down the other player's expected payoff if he chose to just Dare, and small enough to ensure that just Chickening is not a better option.

A correlated equilibrium would be the uniform distribution over $(D, C), (C, C), (C, D)$. We can think of the co-ordinator as a traffic light to each player. A player can view his light but not the other player's. If a player is told to Chicken, it's possible (with probability $1/2$) the other has been told to Dare, hence it's better to Chicken. If a player is told to Dare, the other player has been told to Chicken, and hence it's best to Dare.

COM S 6840 – Algorithmic Game Theory	Jan 27, 2012
Lecture 3: Continuous Congestion Games	
Instructor: Eva Tardos	*Scribe: Karn Seth*

1 Review: Atomic Congestion Games

Recall the definition of an Atomic Congestion Game from last lecture, which consisted of the following:

- E, a finite set of congestible elements.
- Players $i \in \{1, \ldots, n\}$, each with a strategy set S_i, where each strategy $P \in S_i$ is a subset of E. (Each strategy choice "congests" some of the congestible elements.)
- Delay functions $d_e \geq 0$ for each $e \in E$.

Further, given a set of strategy choices $P_i \in S_i$ for each player i, we defined the following:

- The *congestion* on an element e, given by $x_e = |\{i : e \in P_i\}|$, the number of players congesting that element.
- The *delay* on each element e, given by $d_e(x_e)$.
- The *cost* for each player i, equal to $\sum_{e \in P_i} d_e(x_e)$, the sum of delays for all elements used by that player.

We also defined a set of strategies to be a *Nash Equilibrium* if no single player could improve their cost by swapping only his/her own strategy. More formally,

$$\forall i, \forall Q_i \in S_i, \sum_{e \in P_i} d_e(x_e) \leq \sum_{e \in P_i \cap Q_i} d_e(x_e) + \sum_{e \in Q_i - P_i} d_e(x_e + 1)$$

We also showed that each Atomic Congestion Game has a Nash Equilibrium, and in fact this Nash Equilibrium can be found quite naturally, by performing *iterative best response.* Our proof of used the following potential function:

$$\Phi = \sum_{e \in E} \sum_{1}^{x_e} d_e(x_e)$$

We showed that each step of the iterative best response algorithm strictly reduced the value of this potential function, with the decrease in Φ being exactly the decrease in the cost of the user changing his/her strategy in that iteration. Further, we showed that any local minimum corresponds to a Nash Equilibrium.

We noted an inelegance of the atomic version of congestion games was that the expression for Nash Equilibria contains a "+1". When the number of players is very large, this 1 player should have a very tiny effect. With this in mind, we defined a non-atomic version of Congestion Games.

2 Non-Atomic Congestion Games

Our definition of non-atomic congestion games uses the fact that players are now infinitesimally small. We have the following components:

- The finite set of congestible elements E, which remains the same.
- Instead of n players, we have n *types* of players, with a number r_i reflecting the "amount" of players of type i. Each type i selects from strategy set S_i, and we assume for simplicity that the S_i are mutually disjoint. (The r_i can be thought of as "rate" of traffic between a particular source and sink, for example).
- The delay functions, d_e for every $e \in E$, are now assumed to be continuous.
- We allow each type of players to distribute fractionally over their strategy set. We let $f_P \geq 0$ represent the amount of players using strategy P. Then we have the constraint $\sum_{P \in S_i} f_P = r_i$, that is, all the players of type i have some strategy.
- The congestion on e is defined similarly to the atomic case: $x_e = \sum_{P:e \in P} f_P$.

A choice of strategies, f_P, is now said to be a Nash Equilibrium if the following holds:

$$\forall i, \forall P \in S_i \text{ s.t. } f_P > 0, \forall Q \in S_i, \sum_{e \in P} d_e x_e \leq \sum_{e \in Q} d_e(x_e)$$

The equation reflects the fact that not even changing the strategy of a tiny amount of players of a single type can decrease the cost experienced.

We now wish to show that such a Nash Equilibrium exists.

3 Existence of a Nash Equilibrium

We will utilize the non-atomic analogue of the potential function from atomic games:

$$\Phi = \sum_{e \in E} \int_o^{x_e} d_e(z) dz$$

We claim that the minimum of this function is a Nash Equilibrium. But how do we know that such a minimum exists?

Observe first that Φ is continuous. This follows from the fact that the inner terms are integrals of a continuous function (d_e) with a continuous upper limit, and are thus continous. Further, the sum of continuous functions is also continuous. It follows that Φ is continuous.

Also notice that the set we are optimizing over is *compact*, and continuous functions have minima over compact sets.

[*Note* : A compact set is one that is bounded and contains the limit of every convergent sequence of elements the set. For example, $[0, \infty)$ is not compact because it is not bounded, and $[0, 2)$ is not compact because we can construct an infinite sequence of numbers converging to 2, but 2 is not in the set.

Note also that any decreasing function, for example $f(x) = 7 - 2x$, does not have a minimum over these sets, because we can always find an element with smaller value.

Further, the set we are optimizing over is bounded from above and below, because we have the restrictions that $f_P \geq 0$ and $\sum_{P \in S_i} f_P = r_i$, and this set is also "closed", that is, it contains the limits of all sequences in it. Hence it is a compact set.]

It follows that there exists a set of f_P minimizing Φ. It remains to show that a minimum of Φ is actually a Nash Equilibrium.

Claim 1 *A minimum of Φ is a Nash Equilibrium.*

Proof. We give a somewhat informal proof of this claim.

Suppose that we have a set of f_P minimizing Φ that is not a Nash equilibrium. Then $\exists i, P \in S_i$ with $f_p > 0$ and $\exists Q \in S_i$ such that

$$\sum_{e \in P} d_e(x_e) > \sum_{e \in Q} d_e(x_e)$$

The idea is to take a tiny amount $\delta < f_P$ of players using strategy P and change them to strategy Q, that is, change the strategies to $f_P - \delta$ and $f_Q - \delta$.

Notice that increasing f_Q by δ increases x_e for $e \in Q$ by the same δ. This has the effect of increasing changing the x_e term in Φ to $\int_0^{x_e+\delta} d_e(x_e)$. Since d_e is continuous, the change is approximately $\delta \cdot d_e(x_e)$ (with error that is proportional to δ^2, using Taylor bounds from calculus) . A similar argument holds when we decrease f_P by δ.

Then, as long as δ is sufficiently small and the error is sufficiently low (as it is proportional to δ^2), the change in Φ from changing a δ amount of players from P to Q is given by approximately

$$\delta \cdot (\sum_{e \in Q} d_e(x_e) - \sum_{e \in P} d_e(x_e)) < 0$$

Which contradicts the fact that our original set of strategies minimized Φ. It follows that any minimum of Φ must also be a Nash Equilibrium.

Lecture 4 Scribe Notes

Instructor: Eva Tardos *Patrick Steele (prs233)*

1 Price of anarchy in non-atomic congestion games

The price of anarchy is a measure of the quality of Nash solutions compared to a centrally designed optimal solution. We consider non-atomic congestion games with:

- congestible elements E
- user types $i = 1, \dots, n$
- strategy sets S_i for all i
- congestion $d_e(x)$ along element e given x users
- f_p users choosing strategy p

Define the congestion along each element e as

$$x_e = \sum_{p|e \in p} f_p,$$

and recall that equilibrium is attained if for all $f_p > 0$, user types i, and $p, q \in S_i$ we have that $\sum_{e \in p} d_e(x_e) \leq \sum_{e \in q} d_e(x_e)$; alternatively, equilibrium is attained when

$$\phi = \sum_e \int_0^{x_e} d_e(\xi)\, d\xi$$

is minimized.

1.1 Measuring the quality of solutions

- Sum of delays / average delay
- Maximum delay
- Pareto optimal – doesn't require a shared objective.

Note that minimizing average delay implies Pareto optimality. We consider minimizing average delay, or minimizing

$$\sum_p f_p \sum_{e \in p} d_e(x_e) = \sum_e d_e(x_e) \sum_{p \in e} f_p = \sum_e d_e(x_e) x_e.$$

Definition. Delay is (λ, μ)-smooth if for all $x, y > 0$

$$yd(x) \leq \lambda y d(y) + \mu x d(x).$$

We choose x as a Nash solution and y as an optimal solution.

Lemma. The linear delay function $d(x) = ax + b$ is $(1, 1/4)$-smooth for $a, b \geq 0$.

Proof. We want to show that $y(ax+b) \leq y(ay+b)+\frac{1}{4}x(ax+b)$. Let $x, y \geq 0$ be given, and suppose first that $x \leq y$. Then

$$y(ax + b) \leq y(ay + b)$$
$$y(ax + b) \leq y(ay + b) + \frac{1}{4}x(ax + b)$$

since each term is non-negative. Now consider the case when $y < x$. We want to show that $yd(x) \leq yd(y) + \frac{1}{4}xd(x)$, or

$$yd(x) - yd(y) \leq \frac{1}{4}xd(x)$$
$$y(ax + b) - y(ay + b) \leq \frac{1}{4}x(ax + b)$$
$$ayx - ay^2 \leq \frac{1}{4}ax^2 + \frac{1}{4}xb.$$

Since $b \geq 0$ it is sufficient to show that

$$ayx - ay^2 \leq \frac{1}{4}ax^2.$$

If $a = 0$, we are done. If $a > 0$, we are interested in upper-bounding $ayx - ay^2$ with respect to y. Using elementary calculus we can see that the function $f(y) = axy - ay^2$ attains a maximum value when $y = x/2$, and so we have that

$$ayx - ay^2 \leq ax \cdot \frac{x}{2} - a\frac{x^2}{4} = \frac{1}{4}ax^2,$$

as required. □

Theorem 1. Suppose the delay function is (λ, μ)-smooth. If a flow f is a Nash equilibrium and a flow f^* is optimal (with respect to the sum of delays) then

$$\sum_e x_e d_e(x_e) \leq \frac{\lambda}{1-\mu} \sum_e x_e^* d_e(x_e^*).$$

Proof. Let p_j and p_j^* be paths between the same source and sink at Nash equilibrium and optimality, respectively, and let δ_j flow along p_j at Nash and along p_j^* at optimality. Since p_j is at Nash, we have that

$$\sum_{e \in p_j} d_e(x_e) \leq \sum_{e \in p_j^*} d_e(x_e)$$
$$\sum_j \delta_j \sum_{e \in p_j} d_e(x_e) \leq \sum_j \delta_j \sum_{e \in p_j^*} d_e(x_e)$$
$$\sum_e d_e(x_e) \sum_{p_j | e \in p_j} \delta_j \leq \sum_e d_e(x_e) \sum_{p_j^* | e \in p_j^*} \delta_j$$
$$\sum_e d_e(x_e) x_e \leq \sum_e d_e(x_e) x_e^*.$$

Since d_e is (λ, μ)-smooth, we have

$$\begin{aligned}
\sum_e d_e(x_e)x_e &\leq \sum_e d_e(x_e)x_e^* \\
\sum_e d_e(x_e)x_e &\leq \lambda \sum_e x_e^* d_e(x_e^*) + \mu \sum_e x_e d_e(x_e) \\
\sum_e d_e(x_e)x_e - \mu \sum_e x_e d_e(x_e) &\leq \lambda \sum_e x_e^* d_e(x_e^*) \\
(1-\mu) \sum_e d_e(x_e)x_e &\leq \lambda \sum_e x_e^* d_e(x_e^*) \\
\sum_e d_e(x_e)x_e &\leq \frac{\lambda}{1-\mu} \sum_e x_e^* d_e(x_e^*),
\end{aligned}$$

as required. □

CS 6840 – Algorithmic Game Theory (2 pages)	Spring 2012
Lecture 5 Scribe Notes	
Instructor: Eva Tardos	*Lior Seeman*

1 Price of Anarchy for non-atomic congestion game

Theorem 1. If the delay functions are (λ, μ)-smooth (for all x, y $yd(x) \leq \lambda y d(y) + \mu x d(x)$), then the total delay in a Nash equilibrium $\leq \frac{\lambda}{1-\mu}$ total delay in optimum, where total delay is equal to $\sum_P f_P(\sum_{e \in P} d_e(x_e)) = \sum_e x_e d_e(x_e)$.

Proof. Let f be the flow at a Nash equilibrium and X be the congestion it creates, and let f^* be the flow at optimum and X^* be the congestion it creates. Let $\delta_1, \ldots, \delta_N$ be disjoint groups of $r_1, \ldots, r_n$, such that all members of δ_i are of the same type and all use P_i in f and P_i^* in f^*.

We know that for each member of δ_i

$$\sum_{e \in P_i} d_e(x_e) \leq \sum_{e \in P_i^*} d_e(x_e)$$

We can multiply this by δ_i and sum for all i and we get

$$\sum_i \delta_i \sum_{e \in P_i} d_e(x_e) \leq \sum_i \delta_i \sum_{e \in P_i^*} d_e(x_e)$$

Changing the order of summation we get

$$\sum_e d_e(x_e) \sum_{i: e \in P_i} \delta_i \leq \sum_e d_e(x_e) \sum_{i: e \in P_i^*} \delta_i$$

We now notice that $\sum_{i: e \in P_i} \delta_i = x_e$ and $\sum_{i: e \in P_i^*} \delta_i = x_e^*$. So by using smoothness we get

$$\sum_e d_e(x_e) x_e \leq \sum_e d_e(x_e) x_e^* \leq \lambda \sum_e d_e(x_e^*) x_e^* + \mu \sum_e d_e(x_e) x_e$$

Rearranging terms we get what we wanted to prove. □

2 Price of Anarchy for the discrete version

We use a more general game formalization:

- n players, numbered $1 \ldots n$
- each player has a strategy set S_i
- Given a strategy $s_i \in S_i$ for each player, each player has a cost function, $C_i(S)$, which is a function of the strategy vector $S = (s_1 \ldots s_n)$
- we say that $S = (s_1 \ldots s_n)$ is a Nash equilibrium if for every player i and for every strategy $s_i' \in S_i$, $C_i(S) \leq C_i(s_i', S_{-i})$ (S_{-i} is the vector where all coordinates except for i are the same as in S).

- We say that such a game is (λ, μ)-smooth if for all strategy vectors S, S^* $\sum_i C_i(S_i^*, S_{-i}) \leq \lambda \sum_i C_i(S^*) + \mu \sum_i C_i(S)$.

Theorem 2. (Roughgarden '09) If a game is (λ, μ)-smooth for $\mu < 1$ then the total cost at a Nash equilibrium is $\leq \frac{\lambda}{1-\mu}$ minimum possible total cost.

Proof. Let S be the strategy vector in a Nash equilibrium and S^* be a strategy vector in a minimum cost solution. From Nash we know that

$$C_i(S) \leq C_i(S_i^*, S_{-i})$$

We can sum this for all i's, apply smoothness and get

$$\sum_i C_i(S) \leq \sum_i C_i(S_i^*, S_{-i}) \leq \lambda \sum_i C_i(S^*) + \mu \sum_i C_i(S)$$

Rearranging terms we get what we wanted to prove. □

This gives a general framework for Price of Anarchy proofs, and it was shown that many of the proofs were actually reproving this theorem with specific parameters that matched their settings.

2.1 Smoothness for discrete congestion games

Let $p_1, \ldots, p_n$ and $p_1^*, \ldots, p_n^*$ be two series of paths chosen by the players that result in congestions X and X^*.

We say that a discrete congestion game is (λ, μ)-smooth if for all such P and P^*, $\sum_i (\sum_{e \in p_i^* \cap p_i} d_e(x_e) + \sum_{e \in p_i^* \setminus p_i} d_e(x_e + 1)) \leq \lambda \sum_e x_e^* d_e(x_e^*) + \mu \sum_e x_e d_e(x_e)$.

CS 6840 – Algorithmic Game Theory (3 pages) 2/3/2012

Lecture 6: Utility Games

Instructor: Eva Tardos *Scribe: Jane Park (jp624)*

1 Announcements

- Scribe duty: try to get it done within a week while it's fresh.
- Final Projects
  1. Two types
     (a) Choose a favorite subarea from what we cover, "realize what we are missing" and "further the literature."
     (b) Incorporate game theoretic thinking into something else you're working on.
  2. Length: absolute max 10 pages. Min 5 or 6 pages.
  3. Partners: Try to work in pairs for the final project. Triples for the project is okay, if you can make it work.

2 Review: Price of Anarchy Bounds for Congestion Games

We derived Price of Anarchy bounds from the (λ, μ)-smooth inequality. A lot of you didn't buy this proof, so I'm doing two things to convince you: 1) We assume things we need. I'll provide examples where these things are actually true. 2) convince you that the bounds are sharp (we can't do any better).

3 Utility Games

Today we switch to utility games, another example of a game where the smoothness inequality results in games that have *Price of Anarchy* bounds. So far we've covered cost-minimizing games, but in other games like utility games, you derive benefits from participating in the game.

Definition. Utility game

- Players $1, \ldots, n$; player i has strategy set S_i
- Strategies give vector $s = s_1, \ldots, s_n$
- Player i's **utility** $u_i(s) \geq 0$: depends on strategy vector (what you're doing and what everyone else is doing)
- **Goal of player**: maximize utility
- s is Nash if $\forall i, s_i' \in S_i$:
$$u_i(s_i', s_{-i}) \leq u_i(s)$$

- A game is $(\lambda - \mu)$-smooth if:

$$\sum u_i(s_i^*, s_{-i}) \geq \lambda \sum_i u_i(s^*) - \mu \sum_i \mu_i(s)$$

Each $\sum$-term is utility (negative cost). We would like to place an upperbound on the cost, or lower bound on utilit at Nash. Intuition: if the optimal solution is better than the current solution, we hope someone discovers it via his/her $u_i(s_i^*, s_{-i})$ utility being high.

Note: for now we always assume that everyone knows everything, that is, we consider "full information games".

Theorem 1. If s is Nash equilibrium and s^* maximizes sum (s is at least as good as optimal):

$$\sum u_i(s) \geq \frac{\lambda}{\mu + 1} \sum_i u_i(s^*)$$

Proof.

$$\sum_i u_i(s) \geq \sum_i u_i(s_i^*, s_{-i}) \geq \lambda \sum_i u_i(s^*) - \mu \sum_i u_i(s)$$

$$(1 + \mu) \sum_i u_i(s) \geq \lambda \sum_i u_i(s^*)$$

□

Remark. This is a very different kind of game–no congestion/congestible elements.

Example. *Location Game*

Clients desire some sort of service, and k service **providers** position themselves to provide as much of this service as possible. Service providers(i) offer different **prices** p_{ij} to different **clients**(j). There is a service **cost** $c_{ij} > 0$ (not fixed, determined by location). Each client has associated **value** Π_i.

Client j selects the min price p_{ij}, and only if $\Pi_j \geq p_{ij}$. A client's benefit is $\Pi_j - p_{ij}$, and a client reacts to price only. We allow customers to make harmless changes (0-benefit) for mathematical simplicity.

Service provider i has customers A_i and benefit $\sum_{j \in A_i} p_{ij} - c_{ij}$. Provider locations are set, but prices can change often. Providers undercut each other to a point, then stop at equilibrium.

The natural outcome for given locations is that clients choose $\min_i c_{ij}$ (the nearest location) and the price offered is:

$$p_{ij} = \max(c_{ij}, \min(\Pi_j, \min_{k \neq i'} c_{kj}))$$

Explanation: $\min_{k \neq i'} c_{kj}$: the second cheapest location (the provider that you're afraid will undercut you). If the value of the customer's lower, charge that, but never charge less than your cost to serve the customer.

Note: This must be thought of as a multi-stage game where locations are fixed first, then prices are chosen.

Technical assumption: Cost never exceeds benefits.

$$\Pi_j \geq c_{ij}, \forall i, j$$

This assumption allows setting a much simpler rule for prices:

$$p_{ij} = \begin{cases} min_{i \neq k} c_{kj} & \text{by cheapest provider } k \\ c_{kj} & \text{by everyone else} \end{cases}$$

This assumption is helpful, and also harmless. There is no loss of generality since if cost does exceed value, we replace c_{ij} by Π_j. If this edge ever gets used for service, the price will be $p_{ij} = c_{ij}$ as all other edges have $c_{kj} \leq \Pi_j$ by assumption. This means that no one has any benefit from the edge: the user's benefit is $\Pi_j - p_{ij} = 0$ and the edge contributes to the provider's benefit by $p_{ij} - c_{ij} = 0$. So while our proposed solution can have such an edge with changed cost, we can drop the edge from the solution without affecting the anyone's solution quality.

Theorem 2. This game is also a potential game. Service providers are players, and their change in benefit is exactly matched by the change in potential function. We claim that the potential function that works here is social welfare–the sum of everyone's "happiness", and set Φ = social welfare.

Proof. Social welfare is the sum of all client and user benefits. Note: money (prices) does *not* contribute to benefit/welfare, but makes the economy run. Money cancels out due to its negative contribution on the client side and the positive contribution on the provider side. i_j is the location serving j.

$$\Phi = \sum_{j:\text{client served}} \Pi_j - c_{i_j j}$$

Change in Φ if provider i stops participating: let A_i be the set of users served by i. Each of them has to switch to the second closest provider now, so service cost increases; $\Delta\Phi = \sum_{i \in A_i} -c_{ij} + min_{k \neq i} c_{kj}$. The second closest provider was setting the price for $j \in A_i$, so $\Delta\Phi = \sum_{i \in A_i} -c_{ij} + p_{ij}$, exactly the value i had in the game.

To evaluate the change when i changes location, think of a two step process for provider switching location: 1) Go home (lose benefit) 2) Come back (gain benefit).

Note that the social welfare is the potential function. Will come back to it. □

CS 6840 – Algorithmic Game Theory (3 pages)	Feb. 6, 2012
Lecture 7: Generalized Utility Games	
Instructor: Eva Tardos	*Scribe: Danfeng Zhang (dz94)*

1 Review: Facility Location Game

Recall the facility location problem discussed in last class. In this problem, there are a set of clients that need a service and a set of service providers. Each service provider i selects a location from possible locations A_i and offer price $p_{i,j}$ to client j. Note that providers may offer different prices to different clients.

Moreover, each location is associated with a cost $c_{i,j}$ for serving customer j from location i. We assume that client j has a value π_j for service. A strategy vector S is simply a vector of locations selected by each service provider. That is, $S = \{s_1, \ldots, s_k\}$ for k providers.

We may consider this problem as a three-stage game as follows.

Stage 1. Providers select locations.

Stage 2. Providers setup prices for clients. Each provider i provides second cheapest cost among all other providers. That is $p_{i,j} = \min_{i \neq k} c_{k,j}$ by cheapest provider, and $c_{k,j}$ by everyone else.

Stage 3. Each user selects a provider for a service, and pay the specified price.

Last time, we also showed that this is a potential game with social welfare as a potential function:

$$\Phi = \sum_j (\pi_j - c_{i_j,j})$$

, where i_j is the min cost location to user j.

2 Generalized Frame for Utility Games

In this lecture, we would provide several desired properties on the utility games and derive useful results from these properties. Later, we will see that the facility location game is just one example of such games. The contents of this lecture came from the paper [Vetta2002] and Chapter 19 of the textbook.

Recall that in a utility game, each player i choose a location s_i. Social welfare is a function $U(S)$, where S is the vector of locations.

Here are the properties we require on the games:

Property 1. If player i selects location s_i and others select S_{-i} other locations. Player i gets utility $u_i(s_i, S_{-i})$. We assume

$$\sum_i u_i(s_i, S_{-i}) \leq U(S)$$

Property 2. $U(S) \geq 0$ and U is monotone on S. Also, $U(S)$ has decreasing marginal utility (same as submodular): for provider sets $X \subseteq Y$ and some extra service provider s, we have

$$U(X + s) - U(X) \geq U(Y + s) - U(Y)$$

Remark. While other requirements of this property are reasonable, the monotonicity requirement is questionable. This assumption ignores the cost of providing new service.

Lemma. The potential function of facility location game $\Phi = \sum_j (\pi_j - c_{i_j,j})$ has decreasing marginal utility property.

Proof. With X, more clients switch to s when added. Moreover, for each client j switching to s, its previous cost in $Y \leq$ cost in X. □

Property 3. $u_i(s_i, S_{-i}) \geq U(S) - U(S_{-i})$

Remark. Notice that in the facility location game, we have a stronger condition $u_i(s_i, S_{-i}) = U(S) - U(S_{-i})$.

3 Price of Anarchy

With the properties defined above, we next prove the main theorem of this lecture. We first recall the definition of (λ, μ)-smooth games.

Definition. A game is (λ, μ)-smooth if for all strategy vectors S, S^* we have

$$\sum_i u_i(s_i^*, S_{-i}) \geq \lambda \sum_i u_i(S^*) - \mu \sum_i u_i(S)$$

As shown in previous lectures, if the social welfare function is (λ, μ)-smooth, then we have the result that (social welfare at Nash) $\geq \frac{\lambda}{1+\mu}$ (optimal social welfare).

Next, we prove the main theorem of this class.

Theorem 1. Service location games are $(1, 1)$-smooth. (Hence, Nash $\geq \frac{1}{2}$ Optimal)

Proof. We first define $P_i^* = \{s_1^*, s_2^*, \ldots, s_i^*\}$, which is the first i prefix of S^*.

$$\begin{aligned}
\sum_i u_i(s_i^*, S_{-i}) &\geq \sum_i U(S_{-i} + s_i^*) - U(S_{-i}) && \text{(By Property 3)} \\
&\geq \sum_i U(S_{-i} + s_i^* + P_{i-1}^* + s_i) - U(S_{-i} + P_{i-1}^* + s_i) && \text{(By Property. 2)} \\
&= \sum_i U(S + P_i^*) - U(S + P_{i-1}^*) \\
&= U(S + S^*) - U(S) && \text{(telescoping sum)} \\
&\geq U(S^*) - U(S) && \text{(monotonicity)}
\end{aligned}$$

□

Remark. This property is not "strictly" (1,1)-smooth by definition, but very close. Note that the proof of Nash $\geq \frac{1}{2}$ Opt is not finished yet. The rest of proof will be shown in next class, using Property 1.

4 Extra Example

In this part, we show another example of utility games that satisfy the requirements.

In this example, there are still a set of clients and a set of service providers. We model life as a graph, as shown in Figure 1.

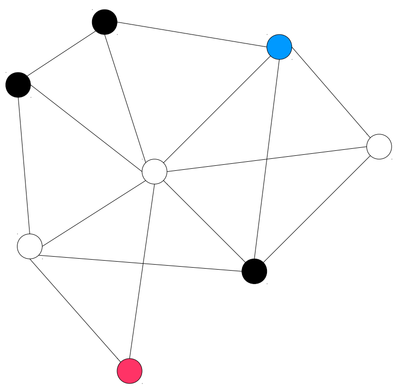


Figure 1: Example

In this graph, each node corresponds to a client. Service providers select among same nodes in the graph as their locations. For instance, service providers select the black nodes in Figure 1. There is an edge between a client and a location if the client is interested in the service on that node.

All users select providers in the following way:

- Each client selects the provider on the same node if there is one. For instance, the clients on black nodes select the providers on the same node.
- If there are providers on neighboring nodes, then client evenly share the services from those providers. For instance, the blue node shares the services provided by the two neighboring black nodes.

The unfortunate red node in Figure 1 gets no service since no neighbor is a provider.

The main difference between the facility location game and this example is that we assume the providers have to undercut other competitors' price in the previous one since only one provider could win the client; but that is not true in this example since services are shared among the competitors.

Due to the lack of time, we would not provide detailed utility function for this example. However, the point is that this example also satisfies all the required properties we used in this lecture.

CS 6840 – Algorithmic Game Theory (4 pages)	Spring 2012
Lecture 9 Scribe Notes	
Instructor: Vasilis Syrgkanis	*Sin–Shuen Cheung*

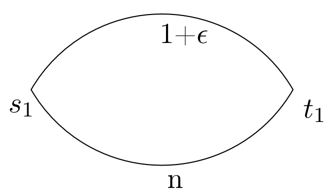
Today's topic: book Sec. 19.3. Reference: Anshelevich et al. *The Price of Stability for Network Design with Fair Cost Allocation.* FOCS 2004.

1 Network Design Games

Description:

- n players;
- each player i: connect (s_i, t_i) on a directed network $G = (V, E)$;
- strategy for player i: $P_i \in \mathcal{P}_i$;
- each $e \in E$ has a cost c_e;
- fair cost allocation: $d_e(n_e) = \frac{c_e}{n_e}$, where n_e is the number of players choosing e;
- player cost: $C_i(S) = \sum_{e \in P_i} \frac{c_e}{n_e}$;
- social cost: $SC(S) = \sum_i C_i(S) = \sum_{e \in S} n_e \frac{c_e}{n_e} = \sum_{e \in S} c_e$

Example 1: consider the following network: n players can choose either edge to connect s_1 and t_1.



One Nash is that all players choose the edge with cost $1 + \epsilon$. In this case, player's cost $C_i = \frac{1+\epsilon}{n}$. The other Nash is that everyone chooses the edge with cost n, where the player's cost $C_i = n/n = 1$.

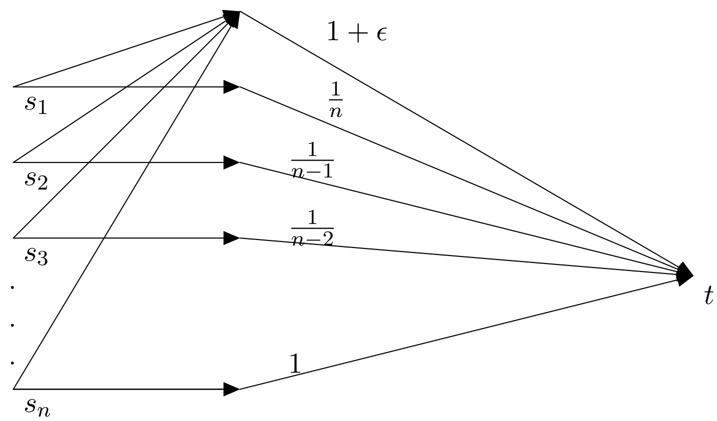
From the analysis above, PoA $\geq n$ in this class of games. On the other hand, PoA is at most n, since in an NE, a player's worse–case cost is at most $\sum_{e \in P_i^*} c_e \leq \sum_{e \in \text{OPT}} c_e$. Therefore the summation of all players' costs is upper bounded by n times of the optimal cost.

More naturally, we are interested in relation between the *best* Nash and the optimal.

1.1 Price od Stability

Definition: Price of Stability (PoS) $= \dfrac{SC(\text{Best–NE})}{SC(\text{OPT})}$

Example 2: consider the following network: Each player i wants to connect from s_i to t. The costs on edges are shown in the figure, if they have costs.



Obviously the optimal strategy has $SC(\text{OPT}) = 1 + \epsilon$, where everyone chooses the route with the $(1 + \epsilon)$ edge. There is a unique Nash for this game – that is player i chooses the route with the $\frac{1}{n+1-i}$ edge. $SC(\text{U–NE}) = 1 + \frac{1}{2} + \frac{1}{3} + \cdots + \frac{1}{n} = H_n$. Thus, here PoS$\geq H_n = O(\log n)$. Comparing with PoA, which is n, PoS is still exponentially better.

Now we are interested in upper bounding PoS for network design games.

Notice that by definition, network design games are congestion games. Thus they are potential games with the poetential function as follows:

- $\Phi(S) = \sum_e \sum_{i=1}^{n_e} d_e(i) = \sum_e \sum_{i=1}^{n_e} \frac{c_e}{i}$.

A special Nash among all is the global minimizer of the potential. However, min potential $\neq$ min SC. In the main theorem, we present that min potential is an approximate of min SC in some sense.

Theorem 1. Let us consider a congestion game with potential function $\Phi(\cdot)$. Suppose that for any strategy S,

$$A \cdot SC(S) \leq \Phi(S) \leq B \cdot SC(S),$$

then PoS$\leq B/A$.

Proof. Let NE denote the global minimizer of the potential, which is a Nash.

$$SC(\text{NE}) \leq 1/A \cdot \Phi(\text{NE}) \leq 1/A \cdot \Phi(\text{OPT}) \leq B/A \cdot SC(\text{OPT}).$$

□

For the class of network design games, we have the following corollary.

Corollary 2. PoS of network design games is $\leq H_n$.

Proof. SC is the sum of the costs of all edges:

$$SC(S) = \sum_{e \in S} c_e.$$

The potential is by definition

$$\Phi(S) = \sum_{e \in S} \sum_{i=1}^{n_e} \frac{c_e}{i} = \sum_{c \in S} c_e H_{n_e}.$$

Therefore,

$$SC(S) \leq \Phi(S) \leq H_n \cdot SC(S).$$

Then apply Theorem 1 to prove the corollary. □

For congestion games with *linear delays*:

- $d_e(n_e) = a_e n_e + b_e$, where $a_e, b_e \geq 0$,

we have the following theorem

Theorem 3. For congestion games with linear delays as defined above, PoS ≤ 2.

Proof. The social cost is

$$SC(S) = \sum_e n_e d_e(n_e) = \sum_e a_e n_e^2 + b_e n_e.$$

The potential is

$$\Phi(S) = \sum_e \sum_{i=1}^{n} (a_e i + b_e) = \sum_e (a_e \frac{n(n+1)}{2} + b_e n_e).$$

Therefore,

$$\frac{1}{2} SC(S) \leq \Phi(S) \leq SC(S).$$

Again, apply Theorem 1 to complete the proof. □

More generally, for the class of network design games, we consider the case where the cost c_e is no longer a constant. Suppose that

- $c_e(i)$ is a concave and monotone increasing function of i, and thus that $\frac{c_e(i)}{i}$ is a decreasing function of i.

Then we have the following theorem.

Theorem 4. For the class of network design games, we assume that the building cost c_e is a concave and increasing function of n_e. Then PoS $\leq H_n$.

Proof. The social cost is

$$SC(S) = \sum_e c_e(n_e).$$

The potential is

$$\Phi(S) = \sum_e \sum_{i=1}^{n_e} \frac{c_e(i)}{i}.$$

Thus,

$$\Phi(S) \leq \sum_e \sum_{i=1}^{n_e} \frac{c_e(n_e)}{i} = \sum_e c_e(n_e) H_{n_e} \leq H_n \cdot SC(S),$$

where the first inequality follows from our assumption that $c_e(\cdot)$ is increasing. On the other hand, noticing that $\frac{c_e(i)}{i}$ is a decreasing function of i, we have that

$$SC(S) = \sum_e c_e(n_e) = \sum_e \sum_{i=1}^{n_e} \frac{c_e(n_e)}{n_e} \leq \sum_e \sum_{i=1}^{n_e} \frac{c_e(i)}{i} = \Phi(S).$$

By applying Theorem 1 we complete the proof. □

For Example 2, if we remove the directedness, the Nash would be that everyone goes through the cheapest edge, which is also the optimal. Then the H_n bound is no longer tight. In fact, when the underlying graph is undirected, it is an open question that whether there is a constant PoS instead of H_n. The best lower bound is ≈ 2.24.

Can we compute the best Nash? Unfortunately computing the best Nash is NP–hard. Computing the Nash that minimizes the potential is also NP-hard.

CS 6840 – Algorithmic Game Theory (2 pages)	Spring 2012
Lecture 15 Scribe Notes	
Instructor: Eva Tardos	*Jesseon Chang (jsc282)*

1 Lecture 15 – Friday 24 February 2012 - Single Item Auctions

1.1 Single Item Auctions

- n players
- Player i has value v_i for the item.
- If player i wins the item then the social value is v_i.

1.2 Second Price Auction

- Each player bids a value/willingness to pay b_i.
- Select i such that $\max_i b_i$ and make him/her pay $p_i = \max_{j \neq i} b_j$.

Property 1: A second price auction is truthful. For each player i, bidding $b_i = v_i$ dominates all other bids.

If a player i bids $b_i < v_i$ and $b_i < \max_{j \neq i} b_j < v_i$, i will want to deviate. If a player i bids $b_i > v_i$ and $v_i < \max_{j \neq i} b_j < b_i$, i will want to deviate.

Nash Equilibria?

- $b_i = v_i$ for all i is a Nash and maximizes social welfare.
- There exists other equilibria where player i with the maximum v_i makes a bid greater than the second largest value and smaller than v_i. $\max_{j \neq i} b_j < b_i < v_i$.
- Yes, there exist Nash equilibria that are not socially optimal. For example, for two players: $v_1 < v_2$, $b_1 > v_2$ and $b_2 = 0$.

All equilibria where $b_i \leq v_i$ for all i are socially optimal.

Proof: If winner i has $b_i < v_i$ and $\exists j : v_j > v_i$, the solution is not a Nash equilibrium, as j wants to deviate and outbid i. Thus there cannot exist a Nash equilibrium where the player with the highest value does not win.

1.3 English Auction

- Raise price of item slowly.
- Once there is only one player still interested, that player wins.
- Players are no longer interested once the price equals their value of the item.
- Similarly to second price auction, winner pays an amount equal to the second highest value.

1.4 Posted Price Auction

- Post a price p.
- Sell to anyone at price p if $b_i > p$.
- If a full information game, $p = \max_i v_i - \epsilon$.

Full information game is unrealistic. We consider a Bayesian game.
Bayesian Game:

- Players draw values v_i from a known probability distribution.
- Each v_i is independent and drawn from the distribution.
- An example: $v_i \in [0, 1]$ uniform
- In a second price auction, for any i $\Pr(i \text{ wins}) = \frac{1}{n}$.
- Set the sell price p such that $\Pr(v > p) = \frac{1}{n}$. If $v_i \in [0, 1]$ uniform, then $p = 1 - \frac{1}{n}$.

Theorem: Assuming values are drawn independently from identical distributions, this fixed price auction results in: $\mathrm{E}(\text{value for winner}) \geq \frac{e-1}{e} \mathrm{Ex}(\max_i v_i)$.

Expected value of our auction:

- The probability that the first player doesn't win is $1 - \frac{1}{n}$ by our choice of price.
- The probability there is no winner is $(1 - \frac{1}{n})^n$.
- The expected value for the winner is $\mathrm{Ex}(v|v \geq p)$.
- Expected value of our auction is $(1 - (1 - \frac{1}{n})^n) \mathrm{Ex}(v|v \geq p) \approx (1 - \frac{1}{e}) \mathrm{Ex}(v|v \geq p)$.

Fact: We can bound the expected value of the auction above by the value of the optimal auction. We consider an auction where the seller has an unlimited number of items to sell and a player has a $\frac{1}{n}$ chance of winning. We call this auction the unlimited auction.

value of optimal auction $\geq$ max value in unlimited auction $= n(\frac{1}{n}) \mathrm{Ex}(v|v \geq p) = \mathrm{Ex}(v|v \geq p)$. Thus the value of our auction above is bounded by $\mathrm{Ex}(v|v \geq p)$.

CS 6840 Notes

Eva Tardos

February 28, 2012

Bayesian Auctions

Last time - single-item auction

- User's value drawn independently from distribution $\mathcal{F}$.
- There are n users, what we know about them is the all the same.

Two types of auctions

- Second price auction (select $\max_i b_i$)
- Fixed price p such that $Pr(v > p) = \frac{1}{n}$

Notation

- User has value v_i (drawn from $\mathcal{F}$)
- Social welfare is v_i for i that receives the item.
  This could be in expectation if we randomize who gets the item.

Today - First Price Auction

This is a traditional game, in contrast to the second price auction where it is optimal to bid truthfully.

- Distribution $\mathcal{F}$ is known, player values v_i are drawn independently, and all players know this distribution
- Player bids b_i
- Select $\max_i b_i$, and maximum bidder gets item and pays b_i
- Benefit is $v_i - p_i$ if player i wins

Bidding $b_i = v_i$ guarantees no gain If you don't win, you gain nothing. If you win, then the net value of what you gain is still 0.

Theorem

The following bid is a Nash equilibrium:

$$b(v) = \text{bid if value is } v = Ex(\max_{j \neq i} v_j \mid v_j \leq v \ \forall\ j)$$

Conditioning: v is highest. Expectation: expected value of 2nd highest.

In other words, what's the expected value of the second highest bid, supposing that you have the highest bid?

Assuming all players use this bidding strategy, does the player with the highest value win? Deterministically yes! Each $b_i(v)$ is the same function (independent of i). Also, $b(v)$ is monotone in v, so the highest value wins. This results in the same outcome as the second price auction (=outcome-equivalent to the second price auction).

Is this also revenue-equivalent to the second price auction? Yes! Suppose you are a player i with value v. You are winning with the same probability as in a second-price auction. In fact, for each player, price is same as the expected price in a second-price auction.

Proof of Theorem

Suppose player 1 feels like deviating. Suppose that his bid $b_1(v)$ has a range from 0 to some unknown number.

Is it better to bid above any other player's range? No. Since you're guaranteed to win by bidding just at the top of someone's range anyway.

So consider a plausible alternate bid $b(z) < v$. This is effectively bluffing that your value is z rather than v.

Goal: solve calculus problem of what the best z is. If $z = v$, then $b()$ is Nash.

Player with $b(z)$ if $z \geq v_i \ \forall \ i > 1$ then the probability is $Pr(\max_{i>1} v_i < z)$.

What you pay is $b(z) = Ex(\max_{i>1} v_i | v_i < z \ \forall \ i)$

Thus, your expected value is

$$Pr(\max_{i>1} v_i < z) \left(v - Ex(\max_{i>1} v_i | v_i < z \ \forall \ i) \right)$$

Let $\max_{i>1} v_i = X$, a random variable.

Rewritten, the expected value is

$$Pr(X < z) \cdot (v - E(X | X \leq z))$$

The expected value can be written as

$$\begin{aligned} & Pr(X < z)v - \int_0^\infty (Pr(X < z) - Pr(X < \xi)) d\xi \\ = & Pr(X < z)v - Pr(X < z)z + \int_0^z Pr(X < \xi) d\xi \end{aligned}$$

Taking derivatives w.r.t z, we get

$$-Pr(X < z) + Pr(X < z) + Pr(X < z)'(v - z)$$

by the product rule and because the derivative of an integral is the value inside, and simplifying,

$$Pr(X < z)'(v - z)$$

To maximize our expectation, set $v = z$, and we need to verify that this is a maximum by checking that $Pr(X < z)$ is monotone and hence its derivative is positive.

Side Note about Expectations and Probabilities

X is any random variable ≥ 0.

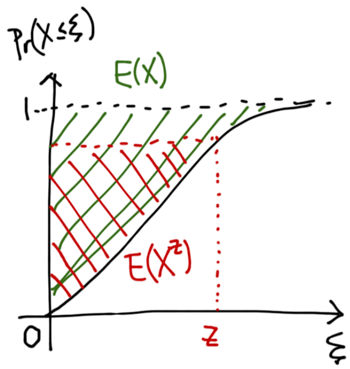
$Ex(X) = \int_0^\infty (1 - Pr(X < z))dz = \int_0^\infty Pr(X \geq z)dz$

Why? If X is discrete, taking values $1, ..., u$, $E(X) = \sum_i i \cdot Pr(X = i) = \sum_i Pr(X \geq i)$. The continuous version of this follows immediately.

Also,

$Ex(X|X < z) \cdot Pr(X < z) = Ex(X^z)$ where $X^z = \begin{cases} X \text{ if } X < z \\ 0 \text{ otherwise} \end{cases}$

$= \int_0^\infty Pr(X < z) - Pr(X < \xi)d\xi = \int_0^\infty Pr(z > X > \xi)d\xi$



CS 6840 – Algorithmic Game Theory (3 pages)	Spring 2012
Lecture 18 Scribe Notes	
Instructor: Eva Tardos	*Daniel Fleischman (df288)*

Lecture 18 – Monday 23 January 2012 - The VCG Mechanism

Overview/Review

The purpose of this lecture is to show the **VCG mechanism**, which generalizes the Vickery Auction to much more general settings. It does this by giving a pricing mechanism so that to make the auction **truthful** (i.e., each player's best strategy is to bid his true value).

Remember our definition of a single item auction:

- n players
- a value v_i for each player i
- goal: pick a winner i^* maximizing the social welfare $SW(i^*) = v_{i^*}$
- each player maximizes each own utility: $u_i = v_i - p$ if $i = i^*$ and $u_i = 0$ otherwise.

In this setting, we have a truthful auction: the Second Price Auction (or Vickery Auction):

- each player i submits a bid b_i to the auctioneer
- the player with the highest bid wins the auction ($i^* = \operatorname{Arg\,max} b_i$)
- player i^* pays the second highest bid

This is interesting because we had an optimization problem (find the maximum v_i) over unknown input. So we defined a game to solve it.

Vickery-Clarke Groves mechanism

We are interested in solving the following optimization problem:

- n players
- a set of *alternatives* A that we can perform
- player i has a value $v_i(a)$ for each $a \in A$
- If alternative a^* is selected, player i's utility is $u_i = v_i(a^*) - p_i$
- goal is to select the alternative A^* that maximizes the social welfare: $\sum_i v_i(a^*)$

If the values are again, unknown, we can define a game such that:

- a strategy for each player is a function $b_i : A \to \mathbb{R}$

- player i reports $b_i(.)$
- picks a^* that maximizes $\sum_i b_i(a^*)$ (in other words, treat the bids as if they were the values)
- charge player i with

$$p_i = \left[\max_{a \in A} \sum_{j \neq i} b_j(a)\right] - \sum_{j \neq i} b_j(a^*)$$

as we will see, the previous game solves the problem we are trying to solve, because it is truthful (and therefore, each player will report bid $b_i(a) = v_i(a)$).

VCG Mechanism is truthful

Theorem 1. VCG is truthful (in other words: $u_i(v_i, b_{-i}) \geq u_i(b_i, b_{-i})$ for all b_i).

Proof.

$$\begin{aligned} u_i(b_i, b_{-i}) &= v_i(a^*) - p_i \\ &= v_i(a^*) - \left[\left[\max_{a \in A} \sum_{j \neq i} b_j(a)\right] - \sum_{j \neq i} b_j(a^*)\right] \\ &= \underbrace{\left[v_i(a^*) + \sum_{j \neq i} b_j(a^*)\right]}_{\text{depends on } b_i \text{ through } a^*} - \underbrace{\left[\max_{a \in A} \sum_{j \neq i} b_j(a)\right]}_{\text{doesn't depend on } b_i} \end{aligned}$$

Remember that a^* maximizes $b_i(a^*) + \sum_{j \neq i} b_j(a^*)$, and player i wants to maximize $v_i(a^*) + \sum_{j \neq i} b_j(a^*)$. So its best strategy is to bid his actual value.

Properties of the VCG Mechanism

There are (at least) two interesting properties of the VCG mechanism.

The first one is that $p_i \geq 0$ (in other words, the auctioneer never pays to the players). This is clear by the definition of p_i. This property is called **no-positive transfers**.

The second property is that (if $v_i \geq 0$ then) $u_i \geq 0$ (i.e., the players "play because they want"). To see this:

$$u_i(v_i, b_{-i}) = \max_{a^* \in A} \left[v_i(a^*) + \sum_{j \neq i} b_j(a^*)\right] - \max_{a \in A} \sum_{j \neq i} b_j(a) \geq 0$$

Example: Single Item Auction

Here the alternatives are $A = \{1, 2, \ldots, n\}$, the player we choose to win. If $\tilde{v}_i \in \mathbb{R}$ is the value of that item for each player then $v_i : A \to \mathbb{R}$ is $v_i(i) = \tilde{v}_i$ and $v_i(j) = 0$.

The alternative selected $i^* \in A$ is the one which maximizes $\sum_i b_i(i^*)$ ($= \max_i \tilde{v}_i$ if truthful).

The player each player pays is $p_i = \left[\max_{a\in A} \sum_{j\neq i} b_j(a)\right] - \sum_{j\neq i} b_j(i^*)$. If $i = i^*$ then the second term is 0 and the first term is the second highest bid. If $i \neq i^*$ then both the first and the second term are $\tilde{v}_{i^*}$ and $p_i = 0$.

Example: Multiple Item Auction

Here we have the following setting:

- n players
- n houses
- player i has a value $\tilde{v}_{ij}$ for house J

The set of alternatives is $A = \{\text{all matchings from players to houses}\}$.

If we had all the values, maximizing the social welfare is the problem of finding a matching of maximum total value, which is known as the weighted bipartite matching problem.

In this case, we ask for bids $\tilde{b}_{ik}$ (we will think of this as the function $b_i(a) = \tilde{b}_{ik}$ if in the matching a, the house k goes to player i. To select the alternative (matching) a^* we solve a maximum weighted bipartite matching problem using the bids as weights. Let a_i be the house given to player i under alternative a. The price will be:

$$p_i = \left[\max_{a\in A} \sum_{j\neq i} b_j(a)\right] - \sum_{j\neq i} b_j(a^*) = \left[\max_{a\in A} \sum_{j\neq i} \tilde{b}_{ja_j}\right] - \sum_{j\neq i} \tilde{b}_{ja^*_j}$$

The second part of the equation above is a simple computation, and the first part is simply a maximum weighted bipartite matching where we set to 0 all the weights of player i to all houses.

Another interpretation to this is that player i should pay “the harm” it causes to the other players (the difference from the benefit they would have without him and how much they have with him).

CS6840 - Algorithmic Game Theory (3 pages) Spring 2014

March 14 - Smoothness in Auction Games

Instructor:Eva Tardos Chris Liu(cl587)

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: $\text{argmax}_a \sum_i b_i(a)$.

2. First price auction - outcome: $\text{argmax}_i\, b_i$. payment: $p_i = b_i$ if $i = \text{argmax}_i\, b_i$, 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 (λ, μ) smooth if $\exists s^*$ s.t $\forall s$ $\sum_i u_i(s_i^*, s_{-i}) \geq \lambda \operatorname{OPT} - \mu \operatorname{SW}(s)$.

Remarks:

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

$$\sum_i u_i(s_i^*, s_{-i}) + \underbrace{\left(\sum_i p_i(s)\right)}_{\text{auctioneer "deviating"}} \geq \lambda \operatorname{OPT} - \mu \underbrace{\left(\sum_i u_i(s) + \sum_i p_i(s)\right)}_{\operatorname{SW}(s)}$$

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 (λ, μ) smooth if $\exists s^*$ s.t $\forall s$,

$$\sum_i u_i(s_i^*, s_{-i}) \geq \lambda \operatorname{OPT} - \mu \sum_i p_i(s)$$

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 (λ, μ) 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 (λ, μ) smooth implies a Nash equilibrium strategy profile s satisfies $\operatorname{SW}(s) \geq \frac{\lambda}{\max\{1,\mu\}} \operatorname{OPT}$

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:

$$\operatorname{SW}(s) \geq \sum_i u_i(s_i^*, s_{-i}) + \sum_i p_i(s)$$
$$\sum_i (u_i(s) + p_i(s)) \geq \sum_i u_i(s_i^*, s_{-i}) + \sum_i p_i(s)$$

$$\sum_i (u_i(s) + p_i(s)) \geq \lambda \, \text{OPT} - \mu \sum_i p_i(s) + \sum_i p_i(s) \qquad \text{by auction smoothness}$$
$$\sum_i u_i(s) + \mu \sum_i p_i(s) \geq \lambda \, \text{OPT}$$
$$\max\{\mu, 1\} \left(\sum_i u_i(s) + \sum_i p_i(s) \right) \geq \lambda \, \text{OPT}$$
$$\text{SW}(s) \geq \frac{\lambda}{\max\{1, \mu\}} \text{OPT} \quad \square$$

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 (λ, μ) smooth with an s^* 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:

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

$$\mathbb{E}_v\left[\sum_i u_i(s)\right] \geq \mathbb{E}_v\left[\lambda \, \text{OPT} - \mu \sum_i p_i(s)\right] \qquad \text{by smoothness}$$
$$\mathbb{E}_v\left[\sum_i u_i(s)\right] + \mathbb{E}_v\left[\mu \sum_i p_i(s)\right] \geq \mathbb{E}_v\left[\lambda \, \text{OPT}\right]$$
$$\mathbb{E}_v[\text{SW}(s)] \geq \frac{\lambda}{\max\{1, \mu\}} \mathbb{E}_v[\text{OPT}] \quad \square$$

Next time: Examples of auctions that satisfy (λ, μ) smoothness in this framework.

Examples of Smooth Auctions (Part 1)

Scribe: Jiayang Gao

Mar.17, 2014
Course: CS 6840
Instructor: Eva Tardos

Last lecture, we defined smoothness of auctions as following:

Definition 1. *An auction game is* (λ, μ) *smooth if* $\exists s^*$*, s.t,* $\sum_i u_i(s_i^*, s_{-i}) \geq \lambda OPT - \mu \sum_i p_i(s)$*. Where* $o(s)$ *is the outcome at strategy vector* s*,* $V_i(o(s))$ *is the value of player* i *at outcome* $o(s)$*,* $p_i(s)$ *is the payment of player* i *given strategy vector* s*, and* $u_i(s) = V_i(o(s)) - p_i(s)$*,* $OPT = \max_o \sum_i V_i(o)$*.*

Using smoothness, we also had the following two theorems on PoA bounds for full info. game and Bayesian game (respectively).

Theorem 1. *For a full information game,* (λ, μ) *smooth implies for any Nash* s*,* $SW(s) \geq \frac{\lambda}{\max(1,\mu)} OPT$*.*

Theorem 2. *For a Bayesian game,* (λ, μ) *smooth with* s_i^* *depends only on* v_i *for all* i*, implies for any Nash* s*,* $E[SW(s)] \geq \frac{\lambda}{\max(1,\mu)} E[OPT]$*.*

In this lecture and next lecture, we will look at examples of smooth games.

Example 1: First Price Auction of a single item

- Players $1, \ldots, n$.
- Values of getting the item $(v_1, \ldots, v_n)$, and value = 0 if not getting it.
- Bids $(b_1, \ldots, b_n)$.

We use the following simple argument to show that the game is $(\frac{1}{2}, 1)$ smooth if we let $s_i^* = \frac{v_i}{2}$ for all i.

Proof. If $j = \arg\max_i v_i$, then $u_j(s_j^*, s_{-j}) \geq \frac{1}{2} v_j - \sum_i p_i(s)$ because

- If j wins, $u_j = v_j - s_j^*(v_j) = \frac{v_j}{2} \geq \frac{1}{2} v_j - \sum_i p_i(s)$.
- If j loses, $u_j = 0$, and $\max_i b_i > \frac{1}{2} v_j$. Notice that $\sum_i p_i(s) = \max_i b_i$ because the maximum bid person pays his bid, and others pays 0. Therefore, $u_j = 0 > \frac{1}{2} v_j - \sum_i p_i(s)$.

If $i \neq \arg\max_i v_i$, then $u_i(s_i^*, s_{-i}) \geq 0$ because if wins, utility is half of his value which is positive, and if loses, utility is 0.

Sum up over all players we get

$$\sum_i u_i(s_i^*, s_{-i}) \geq \frac{1}{2}v_j - \sum_i p_i(s) = \frac{1}{2}OPT - \sum_i p_i(s)$$

Thus the game is $(\frac{1}{2}, 1)$ smooth. □

Thus, according to Theorem 1 and Theorem 2, (notice Theorem 2 applies because here s_i^* only depends on v_i), we have $SW(s) \geq \frac{1}{2}OPT$ for full info game and $E[SW(s)] \geq \frac{1}{2}E[OPT]$ for Bayesian game.

In fact, we can get a tighter bound on PoA as follows.

Theorem 3. *For the single item first price auction defined above, the game is* $(1 - \frac{1}{e}, 1)$ *smooth.*

Proof. Let b_i be randomly chosen according to probability distribution $f(x) = \frac{1}{v_i - x}$ from the interval $[0, (1 - \frac{1}{e}v_i)]$. This probability distribution is well defined because $\int_0^{v_i(1-\frac{1}{e})} \frac{1}{v_i - x}dx = [-\ln(v_i - x)]_0^{v_i(1-\frac{1}{e})} = -\ln(\frac{v_i}{e}) + \ln(v_i) = \ln(\frac{v_i}{v_i/e}) = 1$.

We use the similar technique as above, that

- If $i \neq \arg\max_i v_i$, then $u_i(s_i^*, s_{-i}) \geq 0$.
- If $i = \arg\max_i v_i$. Then $v_i = OPT$. Let $p = \max_{j \neq i} b_j$, then $u_j(s_j^*, s_{-j}) = \int_p^{v_i(1-\frac{1}{e})} f(x)(v_i - x)dx = v(1 - \frac{1}{e}) - p = v_i(1 - \frac{1}{e}) - \max_{j \neq i} b_j \geq v_i(1 - \frac{1}{e}) - \max_j b_j = (1 - \frac{1}{e})OPT - \sum_j p_j$.

Sum up over all i we get

$$\sum_i u_i(s_i^*, s_{-i}) \geq (1 - \frac{1}{e})OPT - \sum_i p_i(s)$$

Therefore the game is $(1 - \frac{1}{e}, 1)$ smooth. □

Similarly, according to Theorem 1 and Theorem 2, we have $SW(s) \geq \frac{e-1}{e}OPT$ for full info game and $E[SW(s)] \geq \frac{e-1}{e}E[OPT]$ for Bayesian game.

Comments:

1. For $s_i^* = \frac{v_i}{2}, o(s^*) = OPT$ because bid is monotone in value, so the maximum value player is always getting the item.
2. For s_i^* random in interval $[0, (1 - \frac{1}{e}v_i)]$, it is possible that $o(s^*) \neq OPT$, because there's possibility even for the max value player to bid close to 0. So in this case the max value person not always get the item.
3. So far we analyzed single item auction. We will talk about how to generalize to multiple item auction next time.

CS6840 - Algorithmic Game Theory (3 pages) Spring 2014

March 19 - Smoothness in Multiple Items Auction Games

Instructor:Eva Tardos Cathy Fan

1 Review:

Definition. An auction is (λ, μ)- smooth if $\exists s^*$, s.t. for all s:

$$\Sigma_i u_i(s_i^*, s_{-i}) \geq \lambda OPT - \mu \Sigma_i p_i(s).$$

Smooth auctions: Set up:

- $o(s)$: outcome
- $v_i(o)$: value of player i. $OPT = \max_o \Sigma v_i(o)$
- $u_i(s) = v_i(o(s)) - p_i(s)$
- $p_i(s) =$ ith payment

Last Time: Smoothness for single item 1st price auction.

Theorem 1. All pay single item auction is $(\frac{1}{2}, 1)$-smooth for any distribution of values.

Proof. : Let $i^* = \arg\max_i V_i$. Let $s_j^* = 0$ for $j \neq i^*$ and s_i^*: randomly chosen according to uniform distribution in $[0, v_i]$. For $j \neq i^*$:

$$u_j(s_j^*, s_{-j}) \geq 0;$$

for $j = i^*$, let $p = \max_{j \neq i^*} s_j$, then:

$$\begin{aligned} u_{i^*}(s_{i^*}^*, s_{-i^*}) &\geq -E(s_i^*) + v_{i^*} Pr(i^* \text{ wins}) \\ &= -\frac{v_{i^*}}{2} + v_{i^*}(\frac{v_{i^*} - p}{v_{i^*}}) \\ &= 0.5 v_{i^*}^* - p \\ &\geq 0.5 v_{i^*}^* - \Sigma_j p_j(s) \end{aligned}$$

Sum up over all i, we get:

$$\Sigma_i u_i(s_i^*, s_{-i}) \geq \frac{1}{2} OPT - \Sigma_i p_i(s)$$

2 Multiple Items:

2.1 Set up for today:

- Unit demand bidders

- Items on sale: Ω
- Players: $1, ..., n$
- Player i has value $v_{ij} \geq 0$ for item j
- $A \subset \Omega$, player i's value for set $A \neq \emptyset$ is $\max_{j \in A} v_{ij}$ (there is free disposal).

2.2 Smoothness

Today: each item is sold on first price.

VCG Mechanism: uses OPT assignment. First price auction uses opt assignment in analysis, but not on mechanism.
Max value matching (optimal matching): $\max_M \Sigma_{(i,j)\in M} v_{ij}$, M represents a Matching.

Theorem 2. 1st price multiple items auction is $(\frac{1}{2}, 1)$-smooth (also $(1 - \frac{1}{e}, 1)$-smooth).

Proof. Take optimal matching M^*. If $(i, j) \in M$ (player i,item j), then bid $s^*_{i^*} = \frac{v_{ij}}{2}$ for item j and bid 0 for all other items. If i is unmatched in M, bid 0 on all items.

If i unmatched,

$$u_i(s^*_i, s_{-i}) \geq 0;$$

Else, $(i, j) \in M$,

$$u_i(s^*_i, s_{-i}) \geq \frac{v_{ij}}{2} - p_j(s).$$

$p_j(s)$ is price for item j on bids s. (This is because if player i wins item j, $u_i(s^*_i, s_{-i}) = \frac{v_{ij}}{2}$; if player i loses item j, item j's price $p_j(s)$ is $\geq \frac{v_{ij}}{2}$.) Sum over i:

$$\Sigma_i u_i(s^*_i, s_{-i}) \geq \frac{1}{2}\Sigma_{(i,j)\in M} v_{ij} - \Sigma_{j\in A} p_j(s) = \frac{1}{2}OPT - \Sigma_j p_j(s)$$

($p_j = 0$ if item j not in assigned).

Corollary 3. Nash equilibrium s for full information game satisfies:

$$SW(s) \geq \frac{\lambda}{\max\{1, \mu\}} OPT.$$

Want Bayesian version:

Option 1: s^*_i depends only on v_i (ith valuation). We used it in single item 1st price auction. Doesn't apply to either "all-pay" of auctions with multiple items.

3 Next Time:

Theorem: smooth game $\rightarrow$ Bayesian PoA small

2nd price auction

CS6840 - Algorithmic Game Theory (3 pages) Spring 2014

March 21 - Bayesian Price of Anarchy in Smooth Auction

Instructor:Eva Tardos Xiaodong Wang(xw285)

1 Administrative

- PS3 deadline is extended to March 24/25
- Project proposal is 1-4 pages

2 Smoothness ⇒ Bayesian Price of Anarchy

Auction game is (λ, μ) smooth if for fixed v, $\exists\ s^*(v)$, s.t $\forall s$ (any),

$$\sum_i u_i(s_i^*(v), s_{-i}) \geq \lambda\, \mathrm{OPT}(v) - \mu \sum_i p_i(s)$$

- Bayesian values $\in$ distribution
- $u_i^{v_i}(s)$ = utility of i when value is v_i; v_i can be a vector
- OPT(v) = max SW when values are v
- $u_i^{v_i}(s_i^*, s_{-i})$ depends on v_i
- s^* depends on values v: $s^*(v)$

Theorem 1. If $\exists\ s^*(v)$, and auction is (λ, μ) smooth and s_i^* depends only on v_i (and not on v_{-i}, then

$$\mathbb{E}(\underbrace{SW(Nash)}_{a\ Bayesian\ Nash}) \geq \frac{\lambda}{\max\{1, \mu\}}\, \mathbb{E}_v(\mathrm{OPT}(v))$$

Example smooth games:

- $s_i^*(v_i)$: first price single item
- $s_i^*(v) : \begin{cases} all\ pay \\ price\ with\ multiple\ item\ and\ unit\ demand \end{cases}$

Today:

Theorem 2. if an auction is (λ, μ) smooth (even if s_i^* depends on all coordinates of v), and the distribution of values for different players is independent, then:

$$\mathbb{E}(SW(BayesianNash)) \geq \frac{\lambda}{\max\{1, \mu\}} \mathbb{E}_v(\mathrm{OPT}(v))$$

- values to different items of a single bidder can be correlated
- values to items of different bidders cannot be correlated
- common knowledge: the distribution of values, as well as the strategies used at Bayesian Nash $s_i(v_i)$, i.e., s_i as a function of v_i, is common knowledge.
- if s is Bayesian Nash, then for all i and s_i' and all v_i,

$$\mathbb{E}_{v_{-i}}(u_i^{v_i}(s_i(v_i), s_{-i}(v_{-i}))|v_i) \geq \mathbb{E}_{v_{-i}}(u_i^{v_i}(s_i', s_{-i}(v_{-i}))|v_i)$$

  An example of Bayesian Nash: 2 bidders, uniform [0,1] distribution, and first price auction, $b_i(v_i) = v_i/2$.

Proof. of the Theorem.
take w_{-i} from value distribution of v_{-i}; take $s_i^*(v_i, w_{-i})$, and use this as s_i'. At a Bayesian Nash equilibrium

$$\mathbb{E}_{v_{-i}}(u_i^{v_i}(s)|v_i) \geq \mathbb{E}_{v_{-i},w_{-i}}(u_i^{v_i}(s_i^*(v_i, w_{-i}), s_{-i}(v_{-i}))|v_i)$$

Taking also expectation over v_i we get:

$$\mathbb{E}_v(u_i^{v_i}(s(v))) \geq \mathbb{E}_{v,w_{-i}}(u_i^{v_i}(s_i^*(v_i, w_{-i}), s_{-i}(v_{-i})))$$

sum up,

$$\mathbb{E}_v(\sum_i u_i^{v_i}(s(v))) = \sum_i \mathbb{E}_v(u_i^{v_i}(s)) \underbrace{\geq}_{Nash} \sum_i \mathbb{E}_{v,w_{-i}}(u_i^{v_i}(s_i^*(v_i, w_{-i}), s_{-i}(v_i)))$$

(v_i, w_{-i}) is of random draw of the type v, because the different coordinates are independent. Define a new variable $t = (v_i, w_{-i})$ as a phantom player, or simply as renaming of the variables (v_i, w_{-i}), and let $z = (w_i, v_{-i})$ using a new random variable w_i. Using the new variables t and z we can rewrite our sum as follows.

$$\sum_i \mathbb{E}_{v,w_{-i}}(u_i^{v_i}(s_i^*(v_i, w_{-i}), s_{-i}(v_i))) = \sum_i \mathbb{E}_{t,z}(u_i^{t_i}(s^*(t), s_{-i}(z))) \underbrace{\geq}_{smoothness} \mathbb{E}_{z,t}(\lambda\, \mathrm{OPT}(t) - \mu \sum_i p_i(s(z)))$$
$$= \lambda\, \mathbb{E}_t(\mathrm{OPT}(t)) - \mu\, \mathbb{E}_z(\sum_i p_i(s(z)))$$

$$\Rightarrow \; \mathbb{E}_v(\sum_i u_i^{v_i}(s(v))) \geq \lambda \, \mathbb{E}_t(\mathrm{OPT}(t)) - \mu \, \mathbb{E}_z(\sum_i p_i(s(z)))$$

$$\mathbb{E}_v(SW(Nash)) = \mathbb{E}_v(\sum_i u_i^{v_i}(s(v))) + \mathbb{E}_v(\sum_i p_i(s(v))) \leq \frac{\lambda}{\max(1,\mu)} \, \mathbb{E}_z(SW(s(z))) \quad \square$$

March 24 -Generalized second prize I

Instructor:Eva Tardos Daniel Freund (df365)

"Second prize" with one item was truthful and thus too simple. An application of Generalized Second Prize auctions is in found in selling ads next to search.

Simple model: advertisers bid on ads

$b_i \rightarrow$ willingness of advertiser i to pay for a click (bidding language allows dependence on lots of info)
[Budget$B_i = \max$ total "over a day"] we ignore today $\rightarrow$ think of it as so big that we won't reach it.

model advertiser's value: v_i as value per click (depends on search term, time of day, location of search etc...), 0 for no click
Questionable assumption: is the value really 0 if the advertiser's ad was displayed?

Probability of getting a click

position j for ads $\rightarrow$ has probability α_j to get a click
ad i itself has probability γ_i for getting a click (depends like v_i on everything)
Questionable assumption: ad i in position j gets click with probability $\alpha_j \gamma_i$

Optimal assignment

The value of advertisement i in position j is $v_{ij} = v_i \gamma_i \alpha_j = v_i \mathbb{P}[i \text{ gets clicked on in position } j]$

We may assume, after renumbering, that $\alpha_1 \geq \alpha_2 \geq ... \geq \alpha_n$ and $v_1\gamma_1 \geq v_2\gamma_2 \geq ... \geq v_n\gamma_n$. The optimal assignment is then given by assigning ad i to α_i (this can be seen with a simple exchange-argument: if an assignment is not sorted like this, then there is some pair $i, i+1$ sorted in the wrong order. Swapping them will increase $\sum_i v_i \mathbb{P}[i \text{ gets clicked on}]$).

This gives rise to the following algorithm:
<u>ALG:</u>
ask bidders for b_i
compute γ_i
sort by $b_i \gamma_i$
assign slots in this order.

Pricing

Historically speaking there have been the following versions:
Version 1 (First Price): Pay b_i if clicked. Problem: consider two players bidding for two advertisement locations. for a while they keep outbidding each other for the better advertisement location until eventually, one decides to take the worse one for very little - but then the other one can take the better one for just a little more and the outbidding starts all over again $\rightarrow$ unstable.
Version 2: set p_i to be the minimum needed for i to keep her slot, i.e.: $p_i = \min\{p : p\gamma_i \geq b_{i+1}\gamma_{i+1}\} = \frac{b_{i+1}\gamma_{i+1}}{\gamma_i}$.

Observation: $p_i \leq b_i$. Is this truthful?

Consider two players, $v_1 = 8, v_2 = 5, \alpha_1 = 1, \alpha_2 = .6, \gamma_1 = \gamma_2 = 1$. If both players bid truthfully, player 2 pays 0, but player has value $(v_1 - p_1)\alpha_1 = 3$ (her expected utility), but with an alternate bid - say 4 - $(v_1 - 0) = 8 \cdot .6 = 4.8 > 3$, so the mechanism is not truthful!

Next class:smoothness-style analysis of a Price of Anarchy result for generalized 2nd-price (assumption: $b_i \leq v_i \forall i$ - How bad is this assumption?)

March 14 - Price of Anarchy for GSP

Instructor:Eva Tardos Jonathan DiLorenzo (jd753)

Administrative details:

The fourth problem set will be out after break.
That will be the last one before the final.

Generalized Second Price (GSP)

Def:
n is the number of slots.
$\forall i.$ we have some α_i corresponding to the click-through rate at slot i.
m is the number of ads (or advertisers).
v_j is the value per click for ad j.
γ_j is the quality factor for ad j.
The probability of someone clicking ad j in slot i is $\alpha_i \times \gamma_j$.

For today, we will assume that $\forall j.\ \gamma_j = 1$. This is a common assumption, which mostly serves to simplify notation.

In GSP, we ask advertisers for some bid b_j and sort by $b_j \times \gamma_j$ (i.e. sort b_j given our assumption). Note that bids are given at a per-click rate, not a total.

We can safely assume $b_1 \geq ... \geq b_m$ (on account of the sorting noted above), which means that $\forall i.\ p_i = b_{i+1}$ because it's a second price auction (well, excluding that last i, where we say $p_i = 0$).

The slots have a total ordering based on α, so also assume WLOG that $\alpha_1 \geq ... \geq \alpha_n$.

We can set $m = n$ by adding phantom slots (if $n < m$, where they have an $\alpha = 0$), or by adding phantom bidders (if $m < n$, where they have a $b = 0$), so for simplicity, we'll consider this situation.

On a side note, apparently Google invented the γ and Yahoo did not initially use it. The γ helps the search company get more money.

Price of Anarchy

Theorem 1. *Price of Stability of GSP is* 1 *(i.e. in a full info game there exists a Nash Eq. that is optimal).*

We will come back to this theorem a bit after break as it needs a topic that hasn't yet been covered.

So now to the part we're actually dealing with:

Theorem 2. *Price of Anarchy: all Bayesian Nash of GSP have* $SW(NE) \geq \frac{1}{4}SW(Opt)$ *assuming* $\forall i.\ b_i \leq v_i$ *where* i *are advertisers.*

In fact, it turns out that $SW(NE) \geq \frac{1}{2(1-\frac{1}{e})}SW(Opt)$, but we won't prove this today. Also, note that the second condition, $\forall i.\ b_i \leq v_i$ tends to be accurate since you don't want to bid more than your value as bidding above your value is dominated by bidding the value itself (see further explanation at the end).

It's best to think of the value/click as not being random. You can sort of figure this out if you're an advertiser. In actuality, the real randomness comes from γ which turns out to be super random, but in our case we're assuming it to be 1.

We prove Theorem 2:

Proof. Recall that $u_i = (v_i - p_i) * \alpha_{k_i}$ where k_i is the slot that i gets with bid b_i.

Firstly, we choose some $b_i^* = \frac{v_i}{2}$ because this happens to be convenient for our proof.

If b is the Bayesian Nash vector and b^* is the bid vector from above, then:

$E_{v_{-i}}(u_i(b_i^*, b_{-i})|v_i) \leq E_{v_{-i}}(u_i(b_i))|v_i)$

by the definition of a Nash. We take the expectation over v_i and sum over i:

$\sum_i E_v(u_i(b_i^*, b_{-i})) \leq \sum_i E_v(u_i(b_i))$

And so we get our standard Bayesian Nash.

Now, suppose in Opt, ad i goes to slot j_i. In that case, i contributes $v_i \times \alpha_{j_i}$ to $SW(OPT)$. Note that this is the value times the number of clicks, since $\gamma = 1$.

Let β_j be the bid that actually wins slot j in GSP. Note that this is a random variable. Also, recall that $b_i^* = \frac{v_i}{2}$. Then: $u_i(b_i^*, b_{-i}) \geq \frac{1}{2}v_i\alpha_{j_i} - \beta_{j_i}\alpha_{j_i}$.

Here is the intuition for why this is true: We want to claim that i 'wins' if he gets slot j_i (the slot he gets in optimum) or better. He loses if he gets a slot lower than j_i. If he wins, then the price $p_i \le b_i^* = \frac{v_i}{2}$. Then, $v_i - p_i \ge \frac{v_i}{2}$ and the number of clicks is greater than or equal to α_{j_i} (since the slots are ordered by α and he did at least as well as slot j_i). Thus, the above inequality holds (since u_i must be greater than the first term on the right side of the inequality).

If he loses, then it's still true because $\frac{v_i}{2} \le \beta_{j_i}$ so it just says that $u_i \ge 0$ (or some negative number).

Now, we sum over all players (explanations of some steps below the equations):

$$\begin{aligned}
\sum_i u_i(b_i^*, b_{-i}) &\ge \frac{1}{2}\sum_i v_i \alpha_{j_i} - \sum_i \alpha_{j_i}\beta_{j_i} && (1)\\
&= \frac{1}{2}OPT(v) - \sum_i \alpha_{j_i}\beta_{j_i} && (2)\\
&= \frac{1}{2}OPT(v) - \sum_i \alpha_{k_i} b_i && (3)\\
&\ge \frac{1}{2}OPT(v) - \sum_i \alpha_{k_i} v_i && (4)\\
&= \frac{1}{2}OPT(v) - SW(b(v)) && (5)
\end{aligned}$$

Of note:
k_i in steps (3) and (4) are meant to denote the slot that player i gets with bid b_i.
The equation in (3) is true because if we sum over i, we cover all the values whether we use the j_i notation or not.
The inequality in (4) is true because $\forall i. v_i \ge b_i$.

Thus, we now know both of these things:
$\sum_i E_v(u_i(b_i^*, b_{-i})) \ge \frac{1}{2}E_v(OPT(v)) - E_v(SW(b(v)))$
$\sum_i E_v(u_i(b_i^*, b_{-i})) \le \sum_i E_v(u_i(b_i)) \le E_v(SW(b(v)))$

and so we get:

$2E_v(SW(b(v))) \ge \frac{1}{2}E_v(OPT(v))$

And so we've proven what we want.

Final claim: Bidding above your value is a dominated strategy. bid $b_i > v_i$ is dominated by $b_i = v_i$. If you're bidding above your value either you pay more and you're hosed or you pay less than your value and then you may as well have bid the same as your value.

Thus, the assumption made that $b_i \le v_i$ is a decent assumption. Of course, in the real world we may want to drive neighbors out of business or make sure that they don't get business at least, in which case bidding above our value is perhaps worth it. Though arguably you could include that in your value.

□

CS6840 - Algorithmic Game Theory (4 pages) Spring 2014

March 28 - Greedy Algorithm as a Mechanism

Instructor:Eva Tardos Thodoris Lykouris (tl586)

Main result

This lecture is based on a result by Brendan Lucier and Alan Borodin [1]. The main result is the following:

Theorem 1. If a greedy algorithm is a c-approximation in the optimization version of the problem then, in the game-theoretic version of the problem, it derives a Price of Anarchy of at most c with first price and $(c+1)$ with second price.

Before proving the result, we need to first understand what we really mean with this theorem.

Framework of the optimization version

- Set S of items on sale.
- Each bidder $i \in [n]$ has value $v_i(A)$ for subset $A \subseteq S$.

The goal of the greedy algorithm is to maximize the social welfare:

$$\max_{\text{disjoint } A_1,\ldots,A_k \subseteq S} \sum v_i(A_i)$$

Mechanism for the game-theoretic version

- All users $i \in [n]$ declare a bid $b_i(A)$ for every subset $A \subseteq S$.
- We then run the previous algorithm to determine the allocation.
- For the pricing, we could have:
  1. If i gets A_i, charge her $b_i(A_i)$ (first price)
  2. If i gets A_i, charge her $\Theta_i(A_i)$ (second price), where $\Theta_i(A)$ will be defined later. Note that, in this case, we need an extra no overbidding assumption: $\forall i, A : b_i(A) \leq v_i(A)$.

Greedy algorithm We will consider the case that the greedy algorithm uses some function $f(i, A, v) \to \mathbb{R}$ to determine its next step in the allocation. This function f should be monotone non-decreasing in the value v for fixed i, A and satisfy the property $\forall i, v, A \subseteq A' : f(i, A, v) \geq f(i, A', v)$.

The algorithm is the following: In decreasing order of $f(i, A, v_i(A))$ give A to i and remove i from the game.

The latter (removing part) gives a unit-demand feature in the players and captures the fact that the valuation that some player has on the union of two sets is not the sum of their valuations. Hence, we are not allowed to assign him another set, once something is assigned to him as then the valuations are no more valid.

Possible catches

1. There is no assumption on the valuation function (monotonicity/submodularity) in the theorem. The reason why this is not a problem is hidden in the "if" statement. These assumptions guarantee the existence of a greedy algorithm in most settings. However, the theorem just takes care in transforming an approximation algorithm for the optimization version of the problem to a mechanism with decent Price of Anarchy to the game-theoretic version of the problem.

2. There is exponential amount of information. This is, as well, related to the greedy algorithm and not with the theorem. In fact, there exist greedy algorithms that behave well and fit in our framework. We will give some examples of this form.

Examples

1. The problem of finding a matching of maximum value has a very simple 2-approximation greedy algorithm (sorting edges by value and iteratively adding the edge with the maximum value among the edges that have unassigned adjacent vertices). This case behaves well as the number of items is small.

2. A case more close to our problem is when every player i is interested in just one set A_i. By sorting them by v_i or $\frac{v_i}{|A_i|}$, we get a n-approximation, which gets better if we sort by $\frac{v_i}{\sqrt{|A_i|}}$. This case behaves well as just few items have non-zero value.

3. The routing problem where there is a graph G and some $\{s_i, t_i\}$ and we have value v_i for any $(s_i - t_i)$ path. Although we might have an exponential numbers of possible paths/items, their values are given implicitly.

c-approximation algorithm

An algorithm is called a c-approximation for a maximization problem if the value of its solution is at least $\frac{1}{c}$ the value of the optimal solution.

Second price

Last but not least, we need to define what $\Theta_i(A)$ (used in second price auction) is. This corresponds to the critical price related to player i and set A, i.e. the smallest price which would allow him to still win the set.

More formally, $\Theta_i(A)$ equals to the minimum bid that gets set A to player i when the algorithm favors i in all ties. The latter is to avoid the need of bidding slightly above to strictly win the auction. The number depends on b_{-i} but not in b_i.

Proof of Theorem

Suppose that b is the bids' trajectory in Nash, which results in solution $A_1, \ldots, A_n$ and Opt is the solution of disjoint sets $O_1, \ldots O_n$ that maximizes $\sum_i v_i(O_i)$.

Suppose that $X_1, \ldots, X_n$ is the allocation that maximizes $\sum_i b_i(X_i)$ (different from Opt as we are not maximizing on the real valuations but on the bids). It holds that $\sum_i b_i(O_i) \leq \sum_i b_i(X_i)$ (as Opt was among the possible allocations).

In addition, as the algorithm is c-approximation, it holds that $\sum_i b_i(X_i) \leq c \sum_i b_i(A_i)$.

Hence, we have the following inequality to which we will refer as (*):

$$\sum_i b_i(O_i) \leq c \sum_i b_i(A_i)$$

Claim 2.

$$\sum_i \Theta_i(O_i) \leq c \sum_i b_i(A_i)$$

Proof. Let the following bids:

$$b'_i(A) = \begin{cases} b_i(A) & \text{if } A \neq O_i \\ \Theta_i(A) - \epsilon & \text{else} \end{cases}$$

We define $b^*_i(A) = \max(b_i, b'_i)$. As a result, the outcome is not affected as, either:

- A is in the winning set in which case it doesn't alter
- it keeps its value without being in the winning set
- it increases to slightly less than its critical value thus not getting in the winning set.

Applying (*) on b^*, using that $b'_i(A) \leq b^*_i(A)$ and taking $\epsilon \to 0$, the claim follows. □

We will continue the proof for the case of the second price (the case of the first price is similar).

$$b_i^*(A) = \begin{cases} v_i(A) & \text{if } A = O_i \\ 0 & \text{else} \end{cases}$$

As b is Nash, we have $\forall i : u_i(b) \geq u_i(b_i^*, b_{-i})$. Furthermore, $u_i(b_i^*, b_{-i}) \geq v_i(O_i) - \Theta_i(O_i)$ as the right hand is negative in the case that i has 0 utility and the inequality holds with equality from the definition of utility otherwise.

Combining the two inequalities and summing over all i, we have:

$$\sum_i u_i(b) \geq \sum_i (b_i^*, b_{-i}) \geq \sum_i v_i(O_i) - \sum_i \Theta_i(O_i) = OPT - \sum_i \Theta_i(O_i)$$

By the Claim, we have $\sum_i \Theta_i(O_i) \leq c \sum_i b_i(A_i)$ and, by the no overbidding assumption, $b_i(A_i) \leq v_i(A_i)$. Hence, it holds

$$\sum_i u_i(b) \geq OPT - \sum_i \Theta_i(O_i) \geq OPT - c \sum_i b_i(A_i) \geq OPT - c \sum_i v_i(A_i)$$

This inequality $\sum_i u_i(b) \geq OPT - c \sum_i v_i(A_i)$ is smoothness-like. Adding the prices on the left hand, we have:

$$\sum_i v_i(A_i) \geq OPT - c \sum_i v_i(A_i)$$

which results in a Price of Anarchy of at most $(c + 1)$.

Open Questions An interesting open question is to what extent the above technique can be extended to other (non-greedy) approximations. That is, when turned into games, can they generate good Price of Anarchy results?

References

[1] B. Lucier and A. Borodin. Price of anarchy for greedy auctions. In *Proceedings of the Twenty-first Annual ACM-SIAM Symposium on Discrete Algorithms*, SODA '10, pages 537–553, Philadelphia, PA, USA, 2010. Society for Industrial and Applied Mathematics.

CS6840 - Algorithmic Game Theory (4 pages)	Spring 2014
April 7 - Auction, Smoothness, and Second Price	
Instructor:Eva Tardos	Sung Min Park(sp765)

1 Outline

In the last few lectures we looked at smoothness analysis (but not quite exactly the "Roughgarden smoothness" for utility games) in the following examples of auctions:

- 2nd price item auction
- generalized second price
- greedy algorithms as mechanism

Today, we look at the general smoothness for an auction on many items, all sold on second price. Player i's value for item j is v_{ij}. All players have unit demands, so there is free disposal; if player i gets a set of items S, the value for that player is just the maximum valued item in that set, $v_i(S) = \max_{j \in S} v_{ij}$.

On Wednesday, we will look at a more general class of valuation, and that will tie up our study of auctions.

2 Smoothness and PoA

In the proofs of price of anarchy for both Generalized Second Price and mechanisms based on greedy algorithms, we made use of the following smoothness property ($\star$):

$$\exists \text{ bid } b_i^* \ \forall i \text{ s.t. } \forall b$$
$$\sum_i u_i(b_i^*, b_{-i}) \geq \lambda \cdot \text{OPT} - \mu \cdot \sum_i b_i(A_i)$$

Here, $\text{OPT} = \max_{\mathcal{O}} \sum_i v_i(\mathcal{O}_i)$ where $\mathcal{O} = (\mathcal{O}_1, ..., \mathcal{O}_n)$ is an allocation of items to players. Similarly, $A = (A_1, ..., A_n)$ is the allocation made by the mechanism on bids b.

For GSP, $\lambda = \frac{1}{2}, \mu = 1$; for mechanisms based on greedy algorithms, $\lambda = 1$, $\mu = c$ (the approximation factor).

We have shown this multiple times in different contexts, but as a review, the above lemma implies the following bound on PoA given some additional assumptions:

Claim 1. In a full information game, if ($\star$) holds and $b_i(X) \leq v_i(X)$ $\forall i$ $\forall X$ (i.e. the bidders are *conservative*), then SW(CCE) $\geq \frac{\lambda}{\mu+1} \cdot$OPT

Proof. Recall that for a CCE (or a learning outcome) that is some distribution on bid b, we have

$$\mathbb{E}_b(u_i(b_i', b_{-i})) \leq \mathbb{E}_b(u_i(b)) \quad \forall \text{ player } i, \quad \forall \text{ alternate bid } b_i'$$

Hence, we have

$$\begin{aligned}
\text{SW}(b) &\geq \mathbb{E}_b\left[\sum_i u_i(b)\right] && (1)\\
&\geq \mathbb{E}_b\left[\sum_i u_i(b_i^*, b_{-i})\right] && (2)\\
&\geq \lambda \cdot \text{OPT} - \mu \cdot \mathbb{E}_b\left[\sum_i b_i(A_i)\right] && (3)\\
&\geq \lambda \cdot \text{OPT} - \mu \cdot \mathbb{E}_b\left[\sum_i v_i(A_i)\right] && (4)\\
& && (5)
\end{aligned}$$

(1) holds since social welfare is the sum of utilities of all players plus the auctioneer utility
(2) holds because the distribution on b is a CCE
(3) is due to smoothness and linearity of expectations
(4) uses the conservative assumption

The right most term is just SW(b), so after rearranging we get the desired PoA bound. □

Observation With the conservative assumption, this is exactly Roughgarden's smoothness for utility games.

3 Auction example

Now, we look at the case of many items, second price auction with unit demand bidders. Recall that if we instead used first price auction, then the cost of the optimal social welfare is given by the maximum matching between players and items, i.e.

$$\text{OPT} = \max_{\text{matching } \mathcal{M}} \sum_{(i,j)\in\mathcal{M}} v_{ij}$$

Claim 2. second price item auction is (1,1) smooth in the sense of ($\star$).

Proof. We need to come up with some special bids b^*. Suppose j_i^* is the item player i gets in the optimal allocation. Then, let

$$\text{b}_i^* = \begin{cases} v_{ij} & \text{if } j = j_i^* \\ 0 & \text{otherwise} \end{cases}$$

Of course, the players don't know what j_i^* is so they can't bid like above practically. We will come back to address this issue.

We can lower bound the utility of a player i bidding b_i^* as

$$\begin{aligned} u_i(b_i^*, b_{-i}) &\geq v_{ij_i^*} - \max_{k\neq i} b_{kj_i^*} \\ &\geq v_{ij_i^*} - \max_{k} b_{kj_i^*} \end{aligned}$$

Summing over all players,

$$\begin{aligned} \sum_i (b_i^*, b_{-i}) &\geq \sum_i v_{ij_i^*} - \sum_i \max_k b_{kj_i^*} \\ &= \text{OPT} - \sum_i b_i(A_i) \end{aligned}$$

The last equality follows from observing that since $\max_k b_{kj_i^*}$ is the maximum bid in b for item j_i^*, if we sum over all players, we are effectively summing the highest bid over all items, which is equal to $\sum_i b_i(A_i)$.

□

4 Learning and PoA bound

We pointed out above that the players do not actually know j_i^*, so though we were able to prove the claim we may wonder what the claim actually means practically. The idea is that we let players use learning, where their options are between the n items; then they bid v_{ij} for the chosen item j and 0 for all others.

The corollary of Claim 2 is that if the players use learning, social welfare in expectation is at least $\frac{1}{2}$OPT in the above setting.

Now, is conservativeness a reasonable assumption? This is not necessarily so, as the following example shows: consider a game with two items A and B. Player 1 has value 1 for both items, and player 2 has value $\frac{1}{2}$ for both items. We may assume there are other players with lower values. Player 1 bids 1 for one item and 0 for the other. Player 2 bids $\frac{1}{2}$ for each item, since this is a full information and he knows that he's going to lose one item to player A. Now, player 2 is *not* conservative as $b_2(A, B) = \frac{1}{2} + \frac{1}{2} = 1 > v_2(A, B) = \frac{1}{2}$.

April 9 - Complement Free Valuations

Instructor:Eva Tardos Bryce Evans (bae43)

Auctions with More Complex Valuations

So far we studied second price style auctions for the following valuations.

$$\begin{aligned} & v_1 \in \mathbb{R}, \text{Single Item} \\ & GSP \\ \boldsymbol{(a)} & \text{Unit Demand} && u_i(A) = \max_j v_{ij} \\ \boldsymbol{(b)} & \text{Additive} && u_i(A) = \sum_{j \in A} v_{ij} \end{aligned}$$

For the additive valuations the optimal solution is $\sum_i \max_i v_{ij}$, where each auction is separate, and no collection between the items.

Today we will consider a General Class of Valuations. – Generalizing $\boldsymbol{(a)}$ and $\boldsymbol{(b)}$ – Each i possible ways to use items v_{ij}^k

$\boldsymbol{(i)}$ $v_i(A) = \max_k \sum_{j \in A} v_{ij}^k$

Claim. This class of valuations contains Unit Demand

$$v_{ij}^k = \begin{cases} v_{ij} & \text{if } k = j \\ 0 & \text{otherwise} \end{cases} \qquad (0, \ldots, 0, v_{ij}, 0, \ldots, 0)$$

Theorem. Item Auctions on Second Price each sold separately, bidders conservative,$\sum_{j \in A} b_{ij} \leq v_i(A)$ for all i and all subset of the items, then Social Welfare Nash (or CCE) $\geq \frac{1}{2}$OPT

Assuming Valuations of $\boldsymbol{(i)}$ form, $b_{ij} = i^{th}$ bid for item i, let the winning bid for item j be $b(j) = \max_i(b_{ij})$.

Proof. Consider OPT location. $O_1, \ldots, O_n$ set items going to bidders $1, \ldots, n$. $V_i(O_i) = \max_k(\sum_{j \in O_i} v_{ij}^k)$, and let k_i be the vector on which the maximum is achieved.

Now define $b_{ij}^* = v_{ij}^{k_i}$, and we claim that this bid satisfies the usual smoothness style inequality.

we have $u_i(b_i^*, b_{-i}) \geq \sum_{j \in O_i}(v_{ij}^{k_i} - b(j))$

(To See why, assume with this bid, person i wins a set A. Now

$$\begin{aligned} u_i(b_i^*, b_{-i}) &= V_i(A) - \sum_{j\in A} b(j) \geq \sum_{j\in A}(v_{ij}^{k_i} - b(j)) \\ &\geq \sum_{j\in(A\cap O_i)} (v_{ij}^{k_i} - b(j)) \\ &\geq \sum_{j\in(A\cap O_)} (v_{ij}^{k_i} - b(j)) \end{aligned}$$

Where the inequality in the top line follows from the definition of V_i, the inequality in teh second line follows as winning additional items $A \setminus O_i$ only make the value higher, and the last inequality follows as the added terms are negative.

Sum Over all players, and using that the bids b form an equilibrium (and hence deviating to b^* doesn't improve player utility), we get:

$$\begin{aligned} \sum u_i(b) &\geq \sum_i \sum_{j\in O_i} v_{ij}^{k_i} - \sum_i \sum_{j\in O_i} b(j) = SW(\text{OPT}) - \sum_j b(j) \\ &\geq SW(\text{OPT}) - \sum_i \sum_{j\in A_i} \geq SW(\text{OPT}) + \sum_i v_i(A_i) = \geq SW(\text{OPT}) + SW(\text{NASH}) \end{aligned}$$

where A_i is the set of items won by player i in Nash, and that last inequality used the assumption of no overbidding.

Now rearranging terms, and using the fact that $\sum u_i(b) \leq SW(\text{NASH})$ we get

$$\sum u_i(b) + \sum v_i(A_i) \geq SW(\text{OPT})$$

□

Next class we will talk about what valuations can be written in the form used in this proof.

April 11 - Complement Free Valuations

Instructor:Eva Tardos

Classes of valuations

We started to consider three classes of valuations last time. For a set A, we will use $v(A)$ to be the value of set A to a user. We will not index valuations with users this class, as we will only consider one user. For all classes we consider today, we will assume that $v(\emptyset) = 0$, value is monotone, that is $A \subset B$ implies that $v(A) \leq v(B)$ (there is free disposal). Note that this also implies that $v(A) \geq 0$ for all A.

1. subadditive valuations, requiring that for any pair of disjoint sets X and Y we have $v(X) + v(Y) \geq v(X \cup Y)$.
2. diminishing marginal value, requiring that for any element j and any pair of sets $S \subset S'$ we have $v(S + j) - v(S) \geq v(S' + j) - v(S')$
3. fractionally subadditive: defined as a function v obtained from a set of vectors v^k with coordinates v^k_j for some $k = 1, ..$ with $v(A) = \max_k \sum_{j \in A} v^k_j$.

First we want to show that diminishing marginal value has the following alternate definition called submodular. A function is submodular, if for any two sets A and B the following holds.

$$v(A) + v(B) \geq v(A \cap B) + v(A \cup B).$$

Claim 1. A function v that is nonnegative, monotone, and $v(\emptyset) = 0$, it is submodular if and only if it satisfies the diminishing marginal value property.

Proof. First, we show by induction that for a pair of sets $S \subset S'$, and a any set A the following diminishing marginal value property holds $v(S \cup A) - v(S) \geq v(S' \cup A) - v(S')$. We show this by induction on $|A|$. When $|A| = 1$ this is the diminishing marginal value property. When $A = A' + j$, by the induction hypothesis $v(S \cup A') - v(S) \geq v(S' \cup A') - v(S')$, by the diminishing marginal value property applied to $S \cup A' \subset S' \cup A'$, we get $v(S \cup A' + j) - v(S \cup A') \geq v(S' \cup A' + j) - v(S' \cup A')$. Adding the two we get $v(S \cup A) - v(S) \geq v(S' \cup A) - v(S')$ as claimed.

For sets $S \subset S'$ a set A disjoint from S', let $X = S \cup A$, and $Y = S'$ then the diminishing marginal value property is exactly the submodular property with X and Y, and vice versa, the submodular property for sets X and Y is this diminishing marginal value property with $S' = Y$, $S = X \cap Y$ and $A = X \setminus Y$. □

Next we show that all fractionally subadditive functions are subadditive.

Claim 2. A fractionally subadditive function is subadditive.

Proof. Let A and B two disjoint sets. The value $v(A \cap B) = \max_k \sum_{j \in A \cup B} v_j^k$. Let k^* be the value that takes the maximum. Now we have

$$v(A \cup B) = \sum_{j \in A \cup B} v_j^{k^*} = \max_k \sum_{j \in A} v_j^{k^*} + \max_{k^*} \sum_{j \in B} v_j^k \leq v(A) + v(B).$$

□

Claim 3. Any submodular function is fractionally subadditive.

Proof. For a submodular function v, we define vectors v_j^k that define v as a required for a fractionally subadditive function. For any order k of the elements, let B_j^k denote the set of first j elements of the order k. For ℓ's element in this order, $\{æ\} = B_\ell^k - B_{\ell-1}^k$, we define $v_j^k = v(B_\ell^k) - v(B_{\ell-1}^k)$. We claim that this defines v.

For a set A, and any order k that starts with A, clearly $v(A) = sum_{j \in A} v_j^k$.

We need to show that for all orders k we have $v(A) \leq \sum_{j \in A} v_j^k$. For this order k define the related order k' that is the same as k in ordering A, but has elements not in A after all elements of A. By the above $v(A) = \sum_{j \in A} v_j^{k'}$, and by the diminishing marginal value property $v_j^{k'} \leq v_j^k$ for all $j \in A$. □

Finally, we wonder about how many functions needed in defining a fractionally subadditive function, and which functions can be defined this way. For a vector v^k to be useable in the definition, it must satisfy $v_j^k \geq 0$ and $\sum_{j \in A} v_j^k \leq v(A)$ for all sets A. To be able to define a function v as fractionally subadditive, for all sets X we need such a vector v^k that also has $\sum_{j \in X} v_j^k = v(X)$. Looking for such a v^k can be written this as a linear program as follows:

$$x_j \geq 0 \text{for all } j \quad (1)$$

$$\sum_{j \in A} x_j \leq v(A) \text{for all sets } A \quad (2)$$

$$\sum_{j \in X} x_j = v(X) \quad (3)$$

A valuation v is fractionally subadditive, if and only if this linear program has a solution for all sets X. Note that this also shows that it suffices to have 2^n vectors v^k in the definition. To see the condition required for a function to be fractionally subadditive, one takes linear programming dual (or Farkas lemma) to get the condition needed to make the above linear program solvable.

Lecture 34 Scribe Notes

Scribe: Ben Perlmutter (bsp52)

April 11, 2012

1 Bandwidth Sharing

Many users want to use a limited resource, how do we allocate this resource as efficiently as possible?
We define the problem:

- user i has utility $U_i(x)$ for x amount of bandwidth
- user i pays w_i for x amount of bandwidth and recieves net utility of $U_i(x) - w_i$
- the total amount of bandwidth available is B

We will also use assumptions regarding $U_i(x)$

- $U_i(x) \geq 0$
- $U_i(x)$ is increasing and concave
- $U(x)$ is continuous and differentiable (not neccessary, but useful for convenience)

The assumptions of $U_i(x)$ being increasing and concave implies that more of x is better, but two times of x doesn't imply twice the utility.

2 How to we allocate the resource optimally?

2.1 First idea: set a price p and let everyone individually optimize their own welfare

e.g. each player individually finds
$\arg\max_x U_i(x) - px$
Solving for this maximum yields
$U_i^{'}(x) - p = 0$
which simplifies to
$U_i^{'}(x) = p$
This function is monotone decreasing.
Of course, if every player maximizes this function individually, it could result in players cumulatively asking for more of the resource than available. Thus, we define a market clearing price p:

2.1.1 Definition of Market Clearing

p is market clearing if there exists amounts $x_i ... x_n$ such that x_i maximizes $U_i(x) - px$ and $\sum_i x_i = B$ where B is the total amount of the resource
If $U_i(x)$ is only non-decreasing, then some users will not want any of the resource, even at price $p = 0$.
In this case, $\sum_i x_i \leq B$ by the property of free disposal; if property is not valuable, one can dispose of it for free.

2.1.2 Market Clearing Price Lemma

If a market clearing price exists, division of B into $x_i ... x_n$ is socially optimal.
$\sum_i U_i(x)$ is the max among all ways to divide B.

2.1.3 Proof of Market Clearing Price Lemma

Let $x_1^*...x_n^*$be the optimal amounts
We know $U_i(x_i) - px_i \geq U_i(x_i^*) - px_i^*$ because $U_i(x_i) - px_i$ was defined to be the maximum.
$\Rightarrow \sum_i U_i(x_i) - p\sum_i x_i \geq \sum_i U_i(x_i^*) - p\sum_i x_i^*$
We know that $p\sum_i x_i = pB$ since $\sum_i x_i = B$
$\Rightarrow \sum_i U_i(x_i) \geq \sum_i U_i(x_i^*) + p(B - \sum_i x_i^*)$
$B - \sum_i x_i^*$ is zero at optimal because all of B is allocated to users

2.1.4 Does such a p exist?

We can always find a market clearing p using the following algorithm:

- set $p = 0$ and if $\sum_i x_i(p) \leq B$, then we are done
- else, raise p until $\sum_i x_i(p) = B$

Will the price rise forever?
This will not happen because of the bounded derivative (marginal utility). If the price was ever raised higher than $U_i^{'}(\frac{B}{n})$, then users will only want to buy at most $\frac{B}{n}$ amount of bandwidth and the total amount of bandwidth requested would be less than or equal to the total amount of bandwidth.
$p = \arg\max_i U_i^{'}(\frac{B}{n})$ results in $x_i \leq \frac{B}{n}$ for all i

2.2 Another idea of optimal allocation: Fair Sharing

The game:

1. Ask every user for how much money they are willing to pay for the resource w_i
2. Collect the money
3. Distribute the resource in x_i amounts by the formula $x_i = (\frac{w_i}{\sum_j w_j}) * (B)$

This yields an effective price of $p = \frac{\sum_j w_j}{B}$

2.2.1 Is the distribution $w_1...w_n$ optimal?

Assume w_j for all $j \neq i$
Users will individually optimize. Imagine a game where every user bids their value w_i, and get some bandwidth in return. At a Nash equilibrium, each w_i is optimal given w_j for all $j \neq i$ are fixed. Thus, the following arg max is a Nash equilibrium.
$\arg\max_{w_i} U_i[(\frac{w_i}{\sum_j w_j}) * (B)] - w_i$
$U_i^{'}(\frac{w_i}{\sum_j w_j} * B)(\frac{1}{\sum_j w_j} * B - \frac{w_i}{(\sum_j w_j)^2} * B) - 1 = 0$
By the effective price and #3, we conclude
$\Rightarrow U_i^{'}(x_i)(\frac{1}{p} - \frac{1}{p} * \frac{x_i}{B}) - 1 = 0$
$\Rightarrow U_i^{'}(x_i)(1 - \frac{x_i}{B}) = p$
Comparing this to price equlibrium $U_i^{'}(x_i) = p$ we see that when $\frac{x_i}{B}$ is close to zero, the two conditions are almost the same. In the context of the internet, users usually do not have a significantly large share of the bandwidth and this pricing scheme is thus approximately optimal.

CS 6840 – Algorithmic Game Theory — Friday, April 13, 2012

Lecture 35 Scribe Notes

Instructor: Eva Tardos — *Anirvan Mukherjee (am767)*

1 Overview

1.1 Summary

In this lecture, we:

- Analyze the Price of Anarchy of the fair-sharing model for bandwidth sharing along a single network edge (as introduced in the previous lecture). It turns out to be at most $\frac{4}{3}$.
- Introduce a new approach to bounding the PoA of a set of problems – we create a many-to-one mapping f from our set of problems into a more restricted subset of problems, such that f can only increase the PoA. We do this strategically so that it is easier to calculate the PoA of the subset. This is different from approaches focused on agent behavior.

2 Context

2.1 Recap of Bandwidth Fair-Sharing

Last lecture, we introduced a bandwidth sharing problem:

- n users want to share a *single edge* of a network.
- Users have utility $U_i(x)$ for bandwidth x, which is non-negative, monotone nondecreasing, concave, and differentiable.
- Users receive allocations x_i.
- The edge has total capacity $B = \sum_i x_i$.

We came up with the following fair-sharing allocation scheme:

- Each user comes up with a bid w_i representing his/her willingness to pay.
- Users pay their w_i.
- Users receive a fraction of the bandwidth proportional to their bids: $x_i = \left(\frac{w_i}{\sum_j w_j}\right) B$.
- If a user increases his bid, he will get more bandwidth, but also will increase $p_{eff} = \frac{\sum_j w_j}{B}$.

2.2 Key Results from Last Lecture

Last lecture, we found that:

- Price equilibrium: $U_i'(x_i) = p$, e.g. when i is allocated bandwidth until marginal payoff becomes zero.
- Proof that price equilibrium exists in the first place.
- Fair-sharing Nash Equilibrium: $U_i'(x_i)\left(1 - \frac{x_i}{B}\right) = p_{eff}$.
- In a case with many users, $\frac{x_i}{B} \approx 0$, so this mechanism is approximately optimal.

3 Price of Anarchy Analysis

3.1 Overview

As mentioned in the Summary, we will perform two steps which will map a given problem (specified by a set of U_i functions) to one whose PoA is strictly not worse. We will then be able to reason algebraically about the PoA of a simpler, restricted set of problems, and upper-bound the PoA in the general case.

The three steps are detailed in the following subsections. We'll use the notation: x_i is the allocation to i at Nash Equilibrium, x_i^* is the optimal (in the maximum sum-of-utilities sense) allocation, and use p to refer to p_{eff}.

3.2 Step 1: Map into the set of linear functions, $U_i(x) = a_i x_i + b_i$

Consider a corresponding problem in which each U_i is mapped to a new utility function V_i, which is the **tangent** to $U_i(x)$ at $x = x_i$, the Nash allocation. Explicitly, $V_i(x) = U_i'(x_i)(x - x_i) + U(x_i)$.

We see that the allocation $\vec{x}$ is still at Nash Equilibrium:

$$V'(x_i) = U'(x_i) \implies V_i'(x_i)\left(1 - \frac{x_i}{B}\right) = p_{eff}$$

Note that the optimal social value didn't get worse: Because U_i are concave, $V_i(x_i^*) \geq U_i(x_i^*)\ \forall i$, and thus there exists an allocation at least as socially optimal as the optimal allocation in the U problem.

Because the Nash value didn't change and the optimum didn't decrease, the PoA did not decrease.

3.3 Step 2: Map into the space of linear functions through $(0, 0)$, $Y_i(x) = a_i x_i$

Consider a correspondign problem in which each $V_i(x) = a_i x + b_i$ is mapped to a new utility function $Y_i(x) = a_i x$, which is $V_i(x)$ shifted to cross the origin. We'll show that the PoA in this restricted subset of problems is not improved.

First, observe that $b_i \geq 0$ must have been true, since $U_i(0) \geq 0$ by stipulation, and $b = V_i(0) \geq U_i(0)$ due to V_i never being less than U_i (a consequence of concavity). It follows that each of the V_i was shifted **down** to get W_i.

Now note that $Y'(x_i) = V'(x_i) = U'(x_i)$, so as before, the Nash allocation doesn't change.

From these, we see that the Nash social value decreases by $b_\Sigma = \sum_i b_i \geq 0$, and the optimal allocation must have decreased by the same amount (a vertical shift does not introduce a chance to improve the allocation).

Letting O and N be the respective total social values from utility functions V_i, we have:

$$\begin{aligned} O &\geq N \\ O \times N - N \times b_\Sigma &\geq O \times N - O \times b_\Sigma \\ N(O - b_\Sigma) &\geq O(N - b_\Sigma) \\ \frac{O - b_\Sigma}{N - b_\Sigma} &\geq \frac{O}{N} \end{aligned}$$

Thus, we see that the PoA has not decreased under this mapping.

3.4 Step 3: Bound the worst PoA in the restricted problem space

At this point, we note that the socially optimal allocation awards the entire bandwith to the user with the highest a_i. For convenience, we'll sort all users by a_i, so that the optimal allocation gives B to a_1, for a total optimal utility of $O' = Ba_1$.

The sum of utilities at Nash, on the other hand, is $N' = \sum_i Y_i(x_i) = \sum_i a_i x_i = a_1 x_1 + \sum_{i>1} a_i x_i$.

Note that if $a_i \leq p$, then in the Nash allocation, $x_i = 0$, so only people with $a_i > 0$ contribute to decreased social welfare. We'll use this fact to, holding the optimal value constant now (instead of the Nash), make the Nash value as poor as possible.

Recall that, at equilibrium, $Y'(x_i) = a_i \left(1 - \frac{x_i}{B}\right) = p$. Rearranging this, we see that i's utility is $Y(x_i) = a_i x_i = B(a_i - p)$. To conceive a worst-case bound, we want to make this value as low as possible *while still allocating to* i, i.e. be as wasteful as possible of this capacity x_i, which was allocated to i rather than 1. So, the worst case bound comes from choosing a_i very close to p, that is, $a_i = p + \varepsilon$ for very small ε. It follows that px_i is a lower bound on $Y_i(x_i) = a_i x_i$, the Nash utility.

$$
\begin{aligned}
PoA &\leq O'/N' \\
&= \frac{Ba_1}{a_1x_1 + \sum_{i>1} a_ix_i} \\
&\leq \frac{Ba_1}{a_1x_1 + \sum_{i>1} px_i} \\
&= \frac{Ba_1}{a_1x_1 + p\left(\sum_{i>1} x_i\right)} \\
&= \frac{Ba_1}{a_1x_1 + p(B - x_1)} \\
&= \frac{Ba_1}{a_1x_1 + a_1\left(1 - \frac{x_1}{B}\right)(B - x_1)} \\
&= \frac{Ba_1}{a_1x_1 + a_1\left(1 - \frac{x_1}{B}\right)(B - x_1)} \\
&= \frac{B}{x_1 + (1 - \frac{x_1}{B})(B - x_1)} \\
&= \frac{1}{\frac{x_1}{B} + \left(1 - \frac{x_1}{B}\right)^2}
\end{aligned}
$$

Differentiating with respect to the ratio $\frac{x_1}{B}$, we find that our PoA upper bound occurs at $\frac{x_1}{B} = \frac{1}{2}$ via calculus, so that worst case PoA is:

$$
\boxed{PoA \leq \frac{1}{\frac{1}{2} + \left(1 - \frac{1}{2}\right)^2} = \frac{4}{3}}
$$

Which is our final result.

4 Existence of Nash Equilibrium

Last lecture, we saw that there was necessarily a price equilibrium. As it turns out, an almost identical proof works to show that there exists a Nash Equilibrium. We can even reduce the proof of existence of a Nash Equilibrium to the same proof used for a price equilibrium:

- We seek to establish the existence of an allocation such that $U_i'(x_i)\left(1 - \frac{x_i}{B}\right) = p_{eff}$.
- Define 'effective' utility function whose derivative is $U_{i,eff}'(x_i) = U_i'(x_i)\left(1 - \frac{x_i}{B}\right)$. This can be found by integrating by parts.
- Note that $U_{i,eff}'$ is decreasing if $U_i'(x)$ were decreasing, since the multiplicative factor is decreasing in x_i, so our property of concavity is maintained.
- Since the multiplicative factor is > 0 for all $x_i < B$, the multiplicative factor is positive, and thus $U_{i,eff}'$ is positive.

- Thus, this 'effective' utility function has the properties we required of the actual utility function in our proof of the existence of a price equilibrium.

5 Overview of Next Lecture

- We'll introduce a network version of the problem, in which each user has a desired path through the network, bids for each edge $e \in$ his path, and receives bandwidth equal to the minimum of his bandwidth along any edge in the path.
- We'll analyze a mechanism in which we run fair-sharing on each edge individually.
- We'll show that the PoA of fair-sharing in the network game is also $\frac{4}{3}$

CS 6840 – Algorithmic Game Theory (5 pages)	Spring 2012

Lecture 36 Scribe Notes: Fair sharing and networks

Instructor: Eva Tardos — *Scribe: Jean Rougé (jer329)*

NOTE: This lecture is based on a paper by R. Johari and J.N. Tsitsiklis published in 2004 that you can find on the course's website.

1 Review of previous lectures

We've investigated fair sharing on a single link during the last two lectures. Let us recall the setting:

* n users compete for bandwidth on a single link of total capacity B
* each user i has his own utility function $U_i(x)$ for x amount of bandwidth
* for all i, we assume that U_i is concave, continously differentiable, monotone increasing, and positive
* user i pays an amount w_i to get an amount x_i of bandwidth, and receives net utility $U_i(x_i) - w_i$

We've seen two disctinct ways of allocating the resource :

1. **Price equilibirum:** we post a fixed price p per amount and we let every user choose his own amount x_i by solving the optmization problem

$$x_i = \arg\max_x (U_i(x) - px)$$

then we showed that this is an equilibrium if either $p = 0$ and $\sum_i x_i \leq B$, or $p > 0$ and $\sum_i x_i = B$; and we also showed that the solution for price equilibirum is socially optimum, i.e. that it maximizes $\sum_i U_i(x_i)$.

2. **Fair sharing as a game:** now each user offers an amount of money w_i, and as a result gets his *fair share*

$$x_i = \frac{x_i}{\sum_j w_j} B$$

We proved that any Nash equilibrium for this game satisfies that for every user i, either

$$U_i'(0) \leq p \text{ and } w_i = 0$$

or

$$U_i'(x_i)\left(1 - \frac{x_i}{B}\right) = p$$

where $p = \frac{\sum_i w_i}{B}$ is the "implicit" price at which the bandwidth gets sold.
Finally we've also seen that the price of anarchy in that game is bounded by $\frac{4}{3}$.

2 And now to networks

Now we want to extend these results to a network comprising a number of links.

Let us define this new setting:

* we consider a graph $G = (V, E)$, where each edge $e \in E$ has a bandwidth capacity $b_e \geq 0$
* user i wants to use links along a fixed path P_i in G
* each user i offers an amount of money $w_{i,e}$ for every edge $e \in P_i$ along his path
* and as a result, player i gets allocated an amount $x_{i,e}$ for each edge edge $e \in P_i$; and he actually enjoys bandwidth[1]

$$x_i = \min_{e \in P_i} x_{i,e}$$

* thus the net utility for user i is

$$U_i(x_i) - \sum_{e \in P_i} w_{i,e}$$

To be able to say something on the Nash equilibria in this game, we'll need to have a result comparable to the price equilibrium theorem we proved for the single-link setting, so that we'll be able to compare a Nash equilibrium to the price equilibrium.

2.1 Price equilibirum theorem

Like we did last time for a single link, let us define a price p_e for every edge $e \in E$; then each user i maximizes his utility by solving the optimization problem

$$x_i = \arg\max_x \left(U_i(x) - x \sum_{e \in P_i} p_e \right)$$

Definition. $(p_e)_{e \in E}$ defines a *price equilibrium* if for every edge $e \in E$, either

$$\sum_{i | e \in P_i} x_i = b_e$$

or

$$\sum_{i | e \in P_i} x_i \leq b_e \text{ and } p_e = 0$$

And it turns out that we get the same result as in the single-link case:

Theorem 1. A price equilibrium is socially optimal, i.e. it maximizes $\sum_i U_i(x_i)$.

[1] The paper by Johari and Tsitsiklis mentioned above is more general than this, in particular the following analysis can be extended to other definitions of x_i as a function of $(x_{i,e})_e$; what's important is that the utility depends on a global variable which in turn depends on the local allocations of the elemtary resources.

Proof. Let us consider a price equilibrium $(x_1, x_2, \ldots, x_n)$, and let us compare it to a socially optimum allocation $(x_1^*, x_2^*, \ldots, x_n^*)$. By definition of the x_i, for all user i,

$$U_i(x_i) - x_i \sum_{e \in P_i} p_e \geq U_i(x_i^*) - x_i^* \sum_{e \in P_i} p_e$$

Now by summing over all users

$$\sum_i U_i(x_i) \geq U_i(x_i^*) + \sum_i x_i \sum_{e \in P_i} p_e - \sum_i x_i^* \sum_{e \in P_i} p_e \tag{1}$$

Yet by reversing the order of summation

$$\sum_i x_i \sum_{e \in P_i} p_e = \sum_e p_e \sum_{i|e \in P_i} x_i = \sum_e p_e b_e$$

where the last equality comes from the definition of a price equilibirum : either $\sum_{i|e \in P_i} x_i = b_e$ or $p_e = 0$, hence in either case $p_e \sum_{i|e \in P_i} x_i = p_e b_e$. Besides, by reversing the order of summation again,

$$\sum_i x_i^* \sum_{e \in P_i} p_e = \sum_e p_e \sum_{i|e \in P_i} x_i^* \leq \sum_e p_e b_e$$

where the last inequality comes from the fact that we can't allow users to exceed an edge's capacity. Hence,

$$\sum_i x_i \sum_{e \in P_i} p_e - \sum_i x_i^* \sum_{e \in P_i} p_e \geq 0$$

and so (1) becomes

$$U_i(x_i) \geq U_i(x_i^*)$$

□

Note that this theorem does not state that a price equilibrium exists. Using convex optimization, one can prove that price equilibrium exists, when utilities are concave. However, we did not prove this here.

2.2 Network sharing as a game

Now let's get back to considering the game outlined at the beginning of this section : each user i offers an amount of money $w_{i,e}$ for every edge $e \in P_i$ along his path, and as a result gets allocated the amount of bandwidth $x_{i,e}$ according to fair-sharing:

$$x_{i,e} = \frac{w_{i,e}}{\sum_j w_{j,e}} b_e$$

Then the actual bandwitdh he actually enjoys is the minimum along his path, i.e.

$$x_i = \min_{e \in P_i} x_{i,e}$$

which results in the net utility

$$U_i(x_i) - \sum_{e \in P_i} w_{i,e}$$

Unfortunately, this natural definition for this game cannot have an equilibrium : indeed, if an user i is the only one competing for a given edge e, then any offer $w_{i,e} > 0$ will ensure that he'll have the whole link for himself alone; but if he offers $w_{i,e} = 0$ he won't get anything.

So we're going to consider a slightly modified version of the game to account for this:

1. each user i, for any edge along his path, either makes an offer $w_{i,e} > 0$ or asks for a free bandwidth $f_{i,e}$ over that edge
2. now for any edge $e \in E$:
   * if anyone offered money for e, we share e according to the fair-share rule
   * if no one offered money for e and if we can accomodate all the requests, i.e. $\sum_{i|e\in P_i} f_{i,e} \leq b_e$ then we give away the bandwidth for free, i.e. $x_{i,e} = f_{i,e}$
   * if no one offered money for e but we can't accomodate all the requests, i.e. $\sum_{i|e\in P_i} f_{i,e} > b_e$, then nobody gets anything, i.e. $x_{i,e} = 0$ (the idea being that this is an over-demanded resource, so we're not willing to give it away for free)

We are going to show for this game a similar result to the one we've seen for the single-link setting:

Theorem 2. The price of anarchy in this game is at most $\frac{4}{3}$. That is, if $(x_1, x_2, \ldots, x_n)$ and $(x_1^*, x_2^*, \ldots, x_n^*)$ respectively are the allocation at a Nash equilibrium and a socially optimal allocation, then

$$\sum_i U_i(x_i) \geq \frac{3}{4} \sum_i U_i(x_i^*)$$

The proof of this theorem wasn't completed during that lecture, by lack of time. We are only going to establish a caracterization of Nash equilibria here, and the rest of the proof will be derived in lecture 37 scribe notes.

This characterization of Nash equilibria we're looking for would be an analog of the result we've recalled at the beginning on Nash equilibria for the single-link setting. We had obtained this characterization as the result of a single-variate optimization problem in the player's offer w. Here, by contrast, each player makes a number of offers $w_{i,e}$, and we don't want to try and solve a multi-variate optimization problem. Let us see how we can translate this problem into a single-variate problem.

We only have to notice that at equilibrium, for every user i, and for all edge $e \in P_i$, $x_{i,e} = x_i$ (indeed, user i has no interest in having more bandwidth on one edge than on another, since he only enjoys the minimum of all them).

Then we can express all the $w_{i,e}$ variables as functions of x_i, since by definition

$$x_{i,e} = \frac{w_{i,e}}{\sum_j w_{j,e}} b_e$$

Re-arranging the terms, and using that at equilibrium $x_i = x_{i,e}$, we get that at equilibrium

$$w_{i,e} = x_i \frac{\sum_{j\neq i} w_{j,e}}{b_e - x_i}$$

And now we're down to a single-variate optimization problem: we're looking for

$$x_i = \arg\max_x \left(U_i(x) - \sum_{e \in P_i} \frac{\sum_{j \neq i} w_{j,e}}{b_e - x} \right)$$

Setting the derivative of the expression above to 0, we get the following characterization: for every user i, at equilibirum, either

$$U_i'(0) \leq \sum_{e \in P_i} p_e \text{ and } x_i = 0$$

or else

$$U_i'(x_i) = \sum_{e \in P_i} p_e \frac{1}{1 - \frac{x_i}{b_e}}$$

where p_e is the unit price at which edge e gets sold, namely $p_e = \frac{\sum_{i | e \in P_i} w_{i,e}}{b_e}$.

See lecture 37 scribe notes for the end of this proof.

CS 6840 – Algorithmic Game Theory (3 pages)	Spring 2012
Lecture 38 Notes	
Instructor: Eva Tardos	*Scirbe: Marcus Lim (mkl65)*

1 Lecture 38 – Friday 20 April 2012 - Price Equilibrium in Arrow-Debreu Model

1.1 Setup

- Goods $\{1, ..., k\}$.
- Players $\{1, ..., n\}$.
- Player i brings $\bar{w}_i = (w_1, ..., w_k)$ amount of goods to the market, and has utility $U_i(\bar{x}_i)$, where $\bar{x}_i = (x_{i1}, ..., x_{ik})$, where x_{ij} = amount of good j that player i gets.
- Assume utilities $U_i(\cdot)$ strictly monotone increasing, strictly concave, continuously differentiable.

1.2 Price Equilibrium

Let $p = (p_1, ..., p_k)$ be the prices for each good. Each player i sells $\bar{w}_i$ to get $p \cdot \bar{w}_i$ amount of money that is used for trading. Given prices, each player finds

$$\bar{x}_i = \arg\max_{\bar{x}} \{U_i(\bar{x}) : p \cdot \bar{x} \leq p \cdot \bar{w}_i, \bar{x} \geq 0\}$$

Note that since $U_i(\cdot)$ is strictly concave, $\bar{x}_i$ is unique. Also, since $U_i(\cdot)$ is strictly monotone increasing (in every dimension), $p \cdot \bar{x}_i = p \cdot \bar{w}_i$.

Definition. Prices $p = (p_1, ..., p_k), p_j > 0$ is a price equilibrium if the resulting $\bar{x}_1, ..., \bar{x}_n$ optima satisfy:

$$\forall j \qquad \sum_i x_{ij} \leq \sum_i w_{ij}$$

Note that by strict monotonicity of utilities, if $p_j = 0$ then all users want $x_{ij} = \infty$, so that cannot be an equilibrium.

Lemma (Market clearing)**.** For all goods j, $\sum_i x_{ij} = \sum_i w_{ij}$.

Proof. As noted earlier, we have

$$\begin{aligned} p \cdot \bar{x}_i &= p \cdot \bar{w}_i \\ \sum_i p \cdot \bar{x}_i &= \sum_i p \cdot \bar{w}_i \\ \sum_j p_j \sum_i x_{ij} &= \sum_j p_j \sum_i w_{ij} \end{aligned}$$

The only way for this to be equal is that they are term-by-term equal, so

$$\begin{aligned} p_j \sum_i x_{ij} &= p_j \sum_i w_{ij} \\ \sum_i x_{ij} &= \sum_i w_{ij} \end{aligned}$$

More generally, $p_j(\sum_i x_{ij}) - \sum_i w_{ij}) = 0$ even if $U_i(\cdot)$ is only monotone increasing.

Definition (Simplex)**.** $\Delta_n := \{x \in \mathbb{R}^n : x_i \geq 0, \sum_i x_i = 1\}$.

Theorem 1 (Brouwer Fixed Point Theorem)**.** If function $f : \Delta_n \to \Delta_n$ is continuous, then there exists x such that $f(x) = x$.

Theorem 2. Equilibrium prices exist.

Proof. Note that if p is a price equilibrium, then αp is also a price equilibrium for any $\alpha > 0$. WLOG, restrict to prices such that $p \in \Delta_n$. Let $\bar{x}_1, ..., \bar{x}_n$ be user optima, and let

$$\begin{aligned} e_j &= \left[\sum_j (x_{ij} - w_{ij})\right]^+ \\ f(p) &= \bar{p} \\ \forall j \qquad \bar{p}_j &= \frac{p_j + e_j}{\sum_i (p_i + e_i)} \end{aligned}$$

Lemma 3. p is price equilibrium $\iff f(p) = p$.

Proof. Clearly, p is price equilibrium $\implies f(p) = p$. Thus, we only need to show that if p is not a price equilibrium, then p is not a fixed point of f. Note that price changes unless e_j/p_j is fixed for all j. We claim that there exist a good j such that $\sum_i x_{ij} \leq \sum_i w_{ij}$. Recall,

$$\sum_j p_j \sum_i x_{ij} = \sum_j p_j \sum_i w_{ij}$$

Hence, it cannot be the case that $e_j > 0$ and for all goods j, $\sum_i x_{ij} > \sum_i w_{ij}$. Thus, if p is not a price equilibrium, then there is some good j such that $e_j > 0$ and hence, there must be some good that will have its price reduced under f, so p is not a fixed point of f.

Lemma 4. f is continuous.

Proof. $\bar{p}$ is continuous, and e_j is continous, so we only need $\bar{x}_i$ to be continous for all players i. Using a fact from continuous optimization, optimizer $\bar{x}_i$ (unique) is a continuous function of p, so f is continuous.

Lemma 5. $f : \Delta_n \to \Delta_n$ is a function. If prices are zero, then $x_{ij} = \infty$ and e_j is unbounded. Hence, we need $\bar{x}_i$'s to be bounded to make e_j's bounded. To do this, we modify the user optimization to include an extra condition.

$$\bar{x}_i = \arg\max_{\bar{x}} \left\{ U_i(\bar{x}) : p\bar{x} \leq p\bar{w}, \quad \forall j, x_j \geq 0, \quad \forall j, x_j \leq \sum_i w_{ij} + 1 \right\}$$

Note that the last condition cannot be tight at the fixed point as it violates price equilibirum conditions. Hence, this does not change the problem, but ensures that $\bar{x}_i$'s are bounded, and $f : \Delta_n \to \Delta_n$ is indeed a function.

Applying Brouwer's fixed point theorem to f shows that price equilibrium p exists.

CS 6840 – Algorithmic Game Theory (3 pages) Spring 2012

Lecture 19 – A Game with Private Information

Instructor: Eva Tardos *Scribe: Kenneth Chong*

Note: For administrative details regarding Homework 3 and the project, please view the first 7 minutes of the VideoNote lecture.

1 Introduction

In Lecture 17, we discussed two types of single-item auctions: first-price and second-price. For the former, we applied a Bayesian framework, in which we assumed that players independently draw values from a publicly known distribution, and use a single bidding function that is monotone in their values. Under this framework, we observed that the two auctions are:

- Outcome equivalent: the player with the highest value wins.
- Revenue equivalent: the payment collected from the winner is equal in expectation.

In general, two forms of auctions may not be outcome equivalent. However, even under more sophisticated Bayesian frameworks, outcome equivalence implies revenue equivalence (the revenue equivalence theorem). We consider a game under which the latter holds.

2 The Game

Suppose now that:

- Player i has a private value v_i drawn from distribution $\mathcal{F}_i$
- Player values are independent of another.
- Distributions $\mathcal{F}_i$ are public knowledge, but values are not.
- Players have individual bidding functions $b_i(v_i)$.

We construct the following mechanism, which converts bids into outcomes and payments:

1. Players submit bids $b_i(v_i)$.
2. Mechanism gives each player an amount $X_i \geq 0$ (possibly a random variable), and charges each player price p_i. Net value for each player is $v_i X_i - p_i$.

Remark: Letting $X_i \in \{0, 1\}, \sum_i X_i = 1, p_i = b_i(v_i)$ for the "high bidder" (and zero for everyone else), we recover a single-item auction. Letting $X_i \in [0, 1]$, we obtain a lottery where each player pays p_i for a probability X_i of "winning" the item.

3 Revenue Equivalence

The Nash equilibrium strategies for this game can be neatly characterized:

Theorem: The bid functions form a Nash equilibrium if and only if:

1. For each i,
$$x_i(v_i) = \mathbb{E}[X_i|v_i = v]$$
is nondecreasing in v

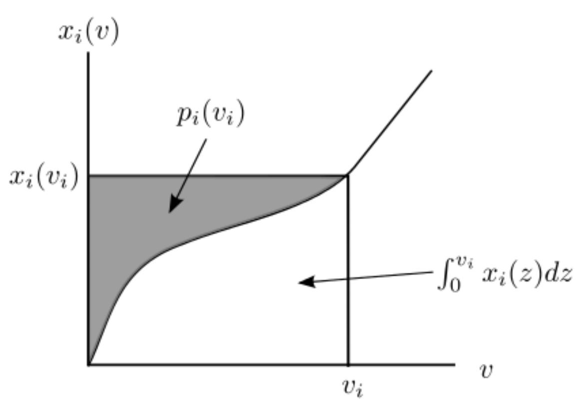
2. Prices $p_i(v_i) = \mathbb{E}[p_i|v_i = v]$ satisfy
$$p_i(v_i) = x_i(v_i)v_i - \int_0^{v_i} x_i(z)\,\mathrm{d}z + p_i(0)$$

Where the expectation is taken over other players' value distributions $\mathcal{F}_i$.

Remark: By statement 2, because player payments depend only on $x_i(v_i)$ (the outcomes), revenue equivalence follows by corollary (with the additional assumption that $p_i(0) = 0$).

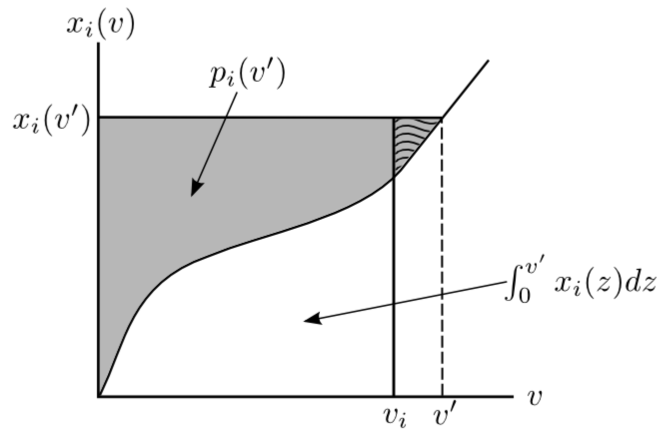
Proof: (NE $\implies$ 1) Suppose there exists a player i and values $v < v'$ such that $x_i(v) > x_i(v')$. By definition of Nash equilibrium, if player i's value is v, he prefers placing a bid using his true value to bluffing a value v':
$$x_i(v)v - p_i(v) \geq x_i(v')v - p_i(v')$$
Similarly, if player i has value v', he prefers not to bluff value v:
$$x_i(v')v' - p_i(v') \geq x_i(v)v' - p_i(v)$$
Summing the two equations, canceling, and regrouping terms, we get
$$x_i(v)v + x_i(v')v' \geq x_i(v')v + x_i(v)'$$
$$[x_i(v) - x_i(v')]v \geq [x_i(v) - x_i(v')]v'$$
$$[x_i(v) - x_i(v')](v - v') \geq 0$$
Since we assumed $v < v'$ and $x_i(v) > x_i(v')$, contradiction.

(1 & 2 $\implies$ NE) By picture. For convenience, assume that the bid functions are onto (the theorem still holds if this is relaxed). Consider the following plot of $x_i(v_i)$ versus v:



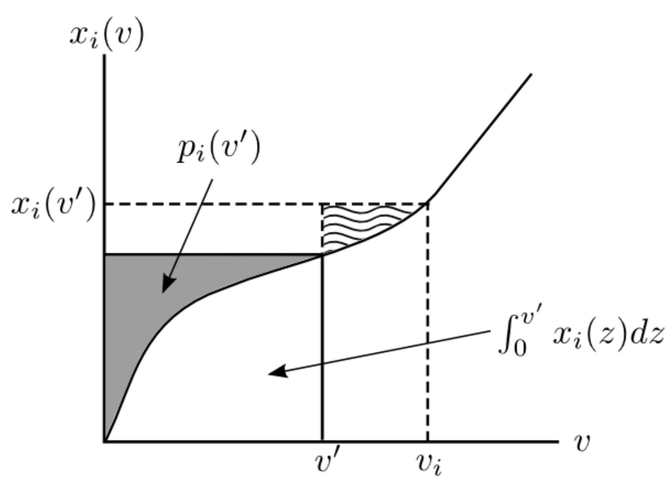
(Because we assumed that bid functions are onto, we can ignore jumps in the graph.) The area bounded by the rectangle represents the value of the item that player i receives. The shaded area is player i's payment, the white area his net payoff.

If player i bluffs a value $v' > v_i$, consider the following plot:



Although he increases the amount he receives to $x_i(v')$, player i values the item at $x_i(v')v_i$, the area of the solid rectangle. However, his payment increases to $p_i(v')$, resulting in a net loss represented by the wavy area in the graph.

If player i bluffs a value $v' < v_i$, a similar phenomenon occurs:



Player i thus could have increased his net value by the wavy area if he bid according to his true value v_i. This implies that player i has no incentive to place a bid different from that corresponding to his true value. We conclude that we are at a Nash equilibrium.

To be continued...

CS 6840 – Algorithmic Game Theory (5 pages) Spring 2012

Scribe Notes - Lecture 20

Instructor: Eva Tardos *Isabel Mette Kloumann (imk36)*

Lecture 20 – Monday March 3 2012 - Game with Private Information - continued

Please see sections 2.4 and 2.5 in Hartline's *Approximation and Economic Design* for a discussion similar to these notes.

1 Summary of lecture.

Nash equilibria - Review and discuss conditions for Nash equilibrium in Bayes payment/outcome games.

Revenue equivalence - Find that if 0-valued agents have zero cost, mechanisms are revenue equivalent, i.e. that if mechanisms produce the same outcomes in Nash equilibrium, they have the same expected revenue.

Importance of revenue equivalence - An auctioneer is free to choose outcomes, and thus can choose them to maximize revenue (this means that auctioneers just have to solve an optimization problem to optimize their revenue).

From value space to probability space - We will introduce a change of variables that takes us from value space to probability space. This change of variables will ultimately simplify the mathematics required to solve the above optimization problem, as well as provide valuable insight.

2 Review of Bayes payment/outcome game set up and conditions for Nash equilibrium.

Review of the game:

- player i has value v_i, where v_i is drawn independently from a distribution F_i
- player i's outcome is $X_i \geq 0$ and their payment is P_i
- utility for player i is $v_iX_i - P_i$. Observe that v_iX_i is the value player i enjoys given the outcome of the game, and P_i is how much they have to pay for it.

Comment: we have been discussing the Nash equilibria of these games solely by thinking about outcomes rather than bidding structures.

IMPORTANT: There was a major error in the definitions of x_i and p_i from the previous lecture. The error is fixed in the posted lecture notes. Note that the proof of the erroneous theorem was actually a valid proof of the correct theorem. We will state the correct theorem here, and discuss why the proof from last time is applicable to this theorem and not the erroneous one.

2.1 Correcting the definitions from last lecture.

We define player i's expected outcome and expected payment as a function of their value v_i:

expected outcome - $x_i(v) = Exp(X_i | v_i = v)$

expected payment - $p_i(v) = Exp(P_i | v_i = v)$

The expectations are conditioned on player i's value v_i being equal to v and integrated over all possible values v_j taken from F for all players $j \neq i$. These statements encode the qualitative idea that this is a game with private information.

[The error from last lecture: we defined x_i and p_i to be conditioned on the OTHER players' values - this doesn't make sense, given that this is a private information game. Rather, it makes sense to **condition on our value**, and consider the **expectation of other players' values**.]

2.2 Theorem giving the conditions for Nash equilibrium.

Theorem 1. A game with outcomes and payments is in a Nash equilibrium iff it has the following properties:

monotonicity the expected outcome, $x_i(v_i)$, is monotone in v_i.

payment identity the expected payment is given by

$$p_i(v) = vx_i(v) + p_i(0) - \int_0^{x_i(v)} x_i(z)dz. \tag{1}$$

$p_i(0)$ is the required payment given you don't value the item - this term is typically zero, as one usually wants a game with free participation.

2.3 Discussion of proof from last lecture.

Monotonicity: Also see Hartline p 32. Let $v' \geq v$. If you are a player with value v and you consider bluffing and naming another value v', then you will experience the outcome

$$\underbrace{vx_i(v) - p_i(v)}_{\text{outcome if you name true value}} \geq \underbrace{vx_i(v') - p_i(v')}_{\text{outcome if you bluff}}. \tag{2}$$

If you are a player with value v' and you consider bluffing and naming another value v, then you will experience the outcome

$$\underbrace{v'x_i(v') - p_i(v')}_{\text{outcome if you name true value}} \geq \underbrace{v'x_i(v) - p_i(v)}_{\text{outcome if you bluff}}. \tag{3}$$

Adding these two inequalities we get

$$(v' - v)(x_i(v') - x_i(v)) \geq 0. \tag{4}$$

This implies that $v' \geq v$ implies $x_i(v') \geq x_i(v)$.

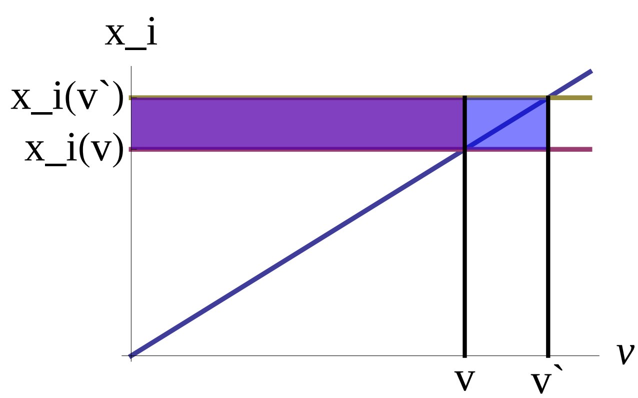
[Comment: This is true in the current game, but couldn't have been true last time. We were conditioning on other people's values: how could we bluff if we were talking about expectations of our value and other players' fixed values? It doesn't make sense to condition on other people's values, because we don't know them - this is a game with private information! We can only condition on our own value, and change our behavior with respect to that value. We can only consider the other players' expected values (i.e. behaviors).]

Payment identity: We can rearrange inequalities 2 and 3 to observe:

$$v(x_i(v') - x_i(v)) \leq p_i(v') - p_i(v) \leq v'(x_i(v') - x_i(v)). \quad (5)$$

This gives us lower and upper bounds on the price difference between v and v'.

Figure 1:



Say v' is a little larger than v, i.e. $v' = v + \delta$. Graphically this means the difference in $p(v') - p(v)$ must be at least the purple area in figure 1, and at most the blue + purple area. This means that as we increase v' the price goes as the blue area. This is a pictorial proof of the theorem's forward statement of the payment identity, that the price is

$$p(i) = \underbrace{vx_i(v)}_{\text{whole box}} - \underbrace{\int_0^v x_i(z)dz}_{\text{area under curve}} . \quad (6)$$

So far we have the forward direction of the theorem: if the game is in a Nash equilibrium, the outcomes and payments of players must be given by statements (1) and (2) in the theorem. Last time we give a proof of both directions. For our purposes today, it's satisfactory to focus only on the forward direction.

3 Revenue equivalence and its implications.

If we add the condition that $p_i(0) = 0$ for all i, then the payment identity shows **revenue equivalence**: the outcomes determine the payments.

Why do we care about revenue equivalence? Given you are an auctioneer who controls the outcomes, if you have revenue equivalence you also control the revenue! Thus, to maximize your revenue, you simply need to choose outcomes x_i that make p_i as high as possible: your task becomes a simple optimization problem!

3.1 How to use revenue equivalence in auction design.

To maximize the expectation, evaluate the expected v subject to the distribution F. This is, **find x_i that are monotone in v such that the expected payments are maximal**:

$$max \sum_i Exp_{F_i}(p_i(v)) \tag{7}$$

This problem could become awkward: we are maximizing a function that is a double integral over the distribution F_i. (The expectation involves an integral of $p_i(v)$ over F_i, and $p_i(v)$ is itself an integral over F_i.)

3.1.1 Simplify the integrals: change from value space to probability space.

Cumulative distribution maps values to probabilities - $F_i(v) = Pr(z \leq v) = 1 - q$ is the cumulative distribution, that is, it's the probability that we would sample a value less than v.

Inverse cumulative distribution maps probabilities to values - With probability q a player will have a value above $v = F^{-1}(1-q)$.

A one-to-one correspondence - F gives a one-to-one correspondence between q and v. Probability that you're less than some very small value is 0, probability that you're less than some very large value is 1. Probability that you're less than v is $1 - q$. Probability you're greater than v is q.

The buying probability as an interpretation of q - q can be interpreted as the **buying probability**: a player will value the item above v with probability $q = Pr(v > z) = 1 - Pr(v \leq z) = 1 - F(v)$.

THE PLAN: Instead of sampling and integrating over v in $F(v)$, we will sample and integrate over $q \in [0,1]$. Sampling over q will be a nicer process to think about: v comes from F_i which is a 'weird' distribution, whereas q is a probability and thus comes from the 'friendly' interval $[0,1]$.

- $v_i(q)$ is the value that corresponds to probability q, i.e. $v_i(q) = F^{-1}(1-q)$.
- When we sampled a value v from F, we were thinking about the expected payments and outcomes given the price of the item is v: $x_i(v)$ and $p_i(v)$
- Now that we're sampling a probability q from [0,1] we will be thinking about the expected payments and outcomes given the item will be sold with probability q: $x_i(v(q)) = x_i(q)$ and $p_i(v(q)) = p_i(q)$.

Next time we will develop a theorem that gives the conditions for Bayes-Nash equilibrium in probability space.

Summary: sampling q instead of v turns awkward integrals into friendly integrals.

4 Thinking about a transaction with a single item and a single user.

How would you sell a single item to a single user if you're revenue maximizing?

Buyers - There is one buyer with distribution F_i.

Products - You have one item to sell.

What do you do? There is no competition between the buyers, your sole action is to set the price p. So, you ought to set p to maximize revenue.

Revenue curve expressed in terms of the price you set p - You will make $p * Pr(v > p) = p(1 - F(p))$. Expected revenue with price v: $v(1 - F(v))$.

Revenue curve expressed with probability of buying q - The probability a player will buy is q and the associated price is $v_i(q)$: you will make revenue $q * v_i(q)$.

Preview: Ultimately our goal will be to implement a change of variables from value to virtual value. This will both simplify the calculus and give insight into where the formula for revenue comes from.