

A Short Introduction to Normal-Form Games

Frederic Koriche

LIRMM, Université Montpellier II

Parcours Intelligence Artificielle
Master Informatique 2ème Année
Septembre 2011

Outline

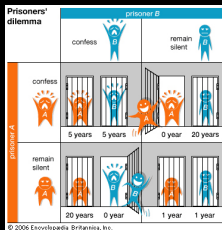
1 Normal-Form Games

2 Nash Equilibria

3 Computing Nash Equilibria

4 Learning Nash Equilibria

Normal-Form Games



		P2	
		Confess	Silent
P1	Confess	5, 5	20, 0
	Silent	0, 20	1, 1

The Prisoner's Dilemma

Two prisoners are on trial for a crime and each one faces a choice of confessing the crime or remaining silent.

- If they both remain silent, they will both serve 1 year for minor offenses.
- If only one of them confesses, his will be freed, while the other will get 20 years
- If they both confess, they will serve 5 years

Normal-Form Games

		Boy	
		Soccer	Theater
Girl	Soccer	6, 5	1, 1
	Theater	2, 2	5, 6

Battle of Sexes

A boy and a girl are deciding on how to spend their evening, by considering two possibilities: a soccer game or the theater.

- The boy prefers a soccer game.
- The girl prefers the theater.
- The both prefer to spend the evening together.

Normal-Form Games

		P2	
		Action 1	Action 2
P1	Action 1	u_{21}	u_{22}
	Action 2	u_{12}	u_{11}

$u_1(1, 2) = u_{12}$
 $u_2(1, 2) = u_{22}$

Normal Form Game

A finite n -player normal form game is a tuple $(N, \mathbf{A}, \mathbf{u})$, where

- N is a finite set of n players, indexed by i
- $\mathbf{A} = A_1 \times \cdots \times A_n$, where A_i is a finite set of actions available to player i .
Each vector $\mathbf{a} = (a_1, \cdots, a_n) \in \mathbf{A}$ is an *action profile*.
- $\mathbf{u} = (u_1, \cdots, u_n)$, where $u_i : \mathbf{A} \rightarrow \mathbb{R}$ is a real-valued *utility* (or *payoff*) function for player i .

Normal-Form Games

	1	2
1	a, a	b, c
2	c, b	d, d

Any $c > a > d > b$ defines an instance of Prisoner's Dilemma

Normal-Form Game

A finite n -player normal-form game is a tuple (\mathbf{X}, \mathbf{u}) , where

- $\mathbf{X} = X_1 \times \cdots \times X_n$, where X_i is a finite set of actions available to player i .
Each vector $\mathbf{x} = (x_1, \cdots, x_n)$ is an *action profile*.
- $\mathbf{u} = (u_1, \cdots, u_n)$, where $u_i : \mathbf{X} \rightarrow \mathbb{R}$ is a real-valued *utility* (or *payoff*) function for player i .

Normal-Form Games

	1	2
1	1, 1	0, 0
2	2, 2	1, 1

Coordination Game

An n -player game (\mathbf{X}, \mathbf{u}) is a coordination game if:

$$u_i(\mathbf{x}) = u_j(\mathbf{x}) \text{ for all } i, j \in [n], \mathbf{x} \in \mathbf{X}$$

Normal-Form Games

	Rock	Paper	Scissors
Rock	0, 0	-1, 1	1, -1
Paper	1, -1	0, 0	-1, 1
Scissors	-1, 1	1, -1	0, 0

Zero-Sum Game

An two-player game (X, u) is a zero-sum game if:

$$u_1(x) + u_2(x) = 0 \text{ for all } x \in X_1 \times X_2$$

Normal-Form Games

	1	2
1	2, 1	0, 0
2	0, 0	1, 2

$$\mathbf{p}_1 = (0, 1)$$

$$\mathbf{p}_2 = \left(\frac{1}{2}, \frac{1}{2}\right)$$

$$u_2(\mathbf{s}) = \frac{1}{2}(0) + \frac{1}{2}(2) = 1$$

Strategies

Let $\mathbb{S}(X)$ be the set of probability distributions over a finite set X

- A *mixed strategy* for player i is a probability distribution s_i over X_i .
- A *mixed strategy profile* is a vector $\mathbf{s} = (s_1, \dots, s_n)$ in $\mathbf{S} = \mathbb{S}(X_1) \times \dots \times \mathbb{S}(X_n)$.
- The *expected utility* of player i in the mixed strategy profile \mathbf{s} is

$$u_i(\mathbf{s}) = \sum_{\mathbf{x} \in \mathbf{X}} u_i(\mathbf{x}) \prod_{j=1}^n s_j(x_j)$$

Outline

1 Normal-Form Games

2 Nash Equilibria

3 Computing Nash Equilibria

4 Learning Nash Equilibria

Nash Equilibria

	1	2
1	2, 1	0, 0
2	0, 0	1, 2

Nash Equilibrium

Let $\mathbf{s}_{-i} = (s_1, \dots, s_{i-1}, s_{i+1}, \dots, s_n)$ denote the strategy profile without player i 's strategy. Player i 's *best response* to the strategy profile \mathbf{s}_{-i} is a mixed strategy s_i^* such that

$$u_i(s_i^*, \mathbf{s}_{-i}) \geq u_i(s_i, \mathbf{s}_{-i}) \text{ for all } s_i \in \mathbb{S}(X_i)$$

Nash Equilibria

	1	2
1	2, 1	0, 0
2	0, 0	1, 2

Nash Equilibrium

A strategy profile $s = (s_1, \dots, s_n)$ is a *Nash Equilibrium* if, for all players i , s_i is a best response to s_{-i} .

In a Nash Equilibrium s no player can change his strategy, and thereby improve his payoff, assuming that other players stick to the strategies they have chosen in s .

Nash Equilibria

	1	2
1	2, 1	0, 0
2	0, 0	1, 2

Nash's Theorem (1951)

Every game with a finite number of players and finite number of actions per player has at least one Nash equilibrium

Nash Equilibria

	1	2	3
1	2, 1	0, 1	0, 0
2	1, 1	1, 0	5, 0
3	2, 1	4, 2	0, 0

Domination

Let s_i and s'_i be two strategies of player i and S_{-i} be the set of all strategy profiles of the remaining players.

- s_i *strictly dominates* s'_i if for all $\mathbf{s}_{-i} \in S_{-i}$, we have $u_i(s_i, \mathbf{s}_{-i}) > u_i(s'_i, \mathbf{s}_{-i})$
- s_i *weakly dominates* s'_i if for all $\mathbf{s}_{-i} \in S_{-i}$, we have $u_i(s_i, \mathbf{s}_{-i}) \geq u_i(s'_i, \mathbf{s}_{-i})$

Nash Equilibria

	1	2	3
1	2, 1	0, 1	0, 0
2	1, 1	1, 0	5, 0
3	2, 1	4, 2	0, 0

Dominated Strategy

A strategy s_i is strictly (weakly) dominated if some other strategy s'_i strictly (weakly) dominates s_i

Nash Equilibria

	1	2
1	2, 1	0, 1
2	1, 1	1, 0
3	2, 1	4, 2

Dominated Strategy

A strategy s_i is strictly (weakly) dominated if some other strategy s'_i strictly (weakly) dominates s_i

Nash Equilibria

	1	2
3	2, 1	4, 2

Iterated Elimination

By iteratively applying dominated elimination, the final game has a Nash equilibrium. If the solution is strictly dominating, then the Nash equilibrium is unique.

Nash Equilibria

	2
3	4, 2

Iterated Elimination

By iteratively applying dominated elimination, the final game has a Nash equilibrium. If the solution is strictly dominating, then the Nash equilibrium is unique.

Nash Equilibria

Correlated Equilibrium

For an n -player game (\mathbf{X}, \mathbf{u}) , let $\mathbb{S}(\mathbf{X})$ be the space of all probability distributions over the profile space \mathbf{X} . Then, a probability distribution p in $\mathbb{S}(\mathbf{X})$ is a *correlated equilibrium* if for all players i and actions $x_j, x'_j \in X_i$, we have

$$\sum_{\mathbf{x}_{-i}} p(x_j, \mathbf{x}_{-i}) u_i(x_j, \mathbf{x}_{-i}) \geq \sum_{\mathbf{x}_{-i}} p(x'_j, \mathbf{x}_{-i}) u_i(x'_j, \mathbf{x}_{-i})$$

Theorem

Every Nash equilibrium is a correlated equilibrium

Outline

1 Normal-Form Games

2 Nash Equilibria

3 Computing Nash Equilibria

4 Learning Nash Equilibria

Computing Nash Equilibria

Nash Equilibrium Problem

Given a finite game $G = (\mathbf{X}, \mathbf{u})$, find a Nash equilibrium for G .

Computing Nash Equilibria

Nash Equilibrium Problem

Given a finite game $G = (\mathbf{X}, \mathbf{u})$, find a Nash equilibrium for G .

Papadimitriou's Theorem (2006)

The Nash Equilibrium Problem is PPAD complete (PPAD is a subclass of NP)

Computing Nash Equilibria

Nash Equilibrium Problem for Two-Player Zero-Sum Games

Given a finite, two-player zero-sum game $G = (\mathbf{X}, \mathbf{u})$, find a Nash equilibrium for G .

Computing Nash Equilibria

Nash Equilibrium Problem for Two-Player Zero-Sum Games

Given a finite, two-player zero-sum game $G = (\mathbf{X}, \mathbf{u})$, find a Nash equilibrium for G .

von Neumann MinMax Theorem

In any finite two-player game, the expected payoff for each player is identical for all Nash equilibria.

Computing Nash Equilibria

Nash Equilibrium Problem for Two-Player Zero-Sum Games

Given a finite, two-player zero-sum game $G = (\mathbf{X}, \mathbf{u})$, find a Nash equilibrium for G .

von Neumann MinMax Theorem

In any finite two-player game, the expected payoff for each player is identical for all Nash equilibria.

LP Reduction

Let A be the matrix of payoffs, representing the winnings of row player and losses of column player.

Let s_r^* and s_c^* be the mixed strategies for row and column players that form a Nash equilibrium.

If s_r is a known strategy used by the row player, then $s_r^\top A$ is the vector of expected payoffs for the column player.

maximize v

subject to $s_r \geq 0$

$$\sum_i s_r(i) = 1$$

$$(s_r^\top A)_j \geq v \text{ for all } j$$

Outline

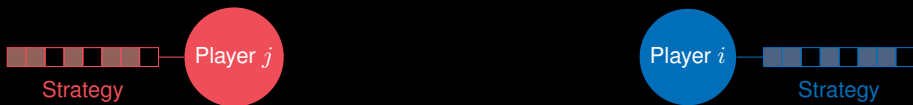
1 Normal-Form Games

2 Nash Equilibria

3 Computing Nash Equilibria

4 Learning Nash Equilibria

Learning Nash Equilibria



Initialization

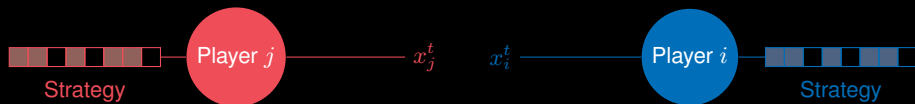
We assume a finite, two-player zero-sum game. The utilities are *unknown*.

- Player i receives action space X_i
- Player j receives action space X_j

Trials (Rounds)

- *Action i* : player i take action x_i^t drawn at random according to s_i^t
- *Action j* : player j take action x_j^t drawn at random according to s_j^t
- *Response i* : player i receives utility $u_i(x_i^t, x_j^t)$
- *Response j* : player j receives utility $u_j(x_i^t, x_j^t)$

Learning Nash Equilibria



Initialization

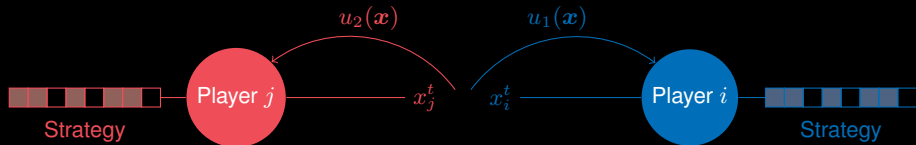
We assume a finite, two-player zero-sum game. The utilities are *unknown*.

- Player i receives action space X_i
- Player j receives action space X_j

Trials (Rounds)

- *Action i* : player i take action x_i^t drawn at random according to s_i^t
- *Action j* : player j take action x_j^t drawn at random according to s_j^t
- *Response i* : player i receives utility $u_i(x_i^t, x_j^t)$
- *Response j* : player j receives utility $u_j(x_i^t, x_j^t)$

Learning Nash Equilibria



Initialization

We assume a finite, two-player zero-sum game. The utilities are *unknown*.

- Player i receives action space X_i
- Player j receives action space X_j

Trials (Rounds)

- *Action i*: player i take action x_i^t drawn at random according to s_i^t
- *Action j*: player j take action x_j^t drawn at random according to s_j^t
- *Response i*: player i receives utility $u_i(x_i^t, x_j^t)$
- *Response j*: player j receives utility $u_j(x_i^t, x_j^t)$

Learning Nash Equilibria

Regret

The regret of player i after T rounds is

$$R_i(T) = \sum_{t=1}^T u_i(x_i^t, x_j^t) - u_i(x_i^*, x_j^t)$$

Learning Nash Equilibria

Regret

The regret of player i after T rounds is

$$R_i(T) = \sum_{t=1}^T u_i(x_i^t, x_j^t) - u_i(x_i^*, x_j^t)$$

Full Information Setting

Here, i always receives the vector of utilities $u_i(\cdot, x_j^t)$ associated to each possible action. If v_i is the value of the game for i , and i is playing with an Hannan consistent algorithm (e.g. weighted majority), then its mean expected regret converge to v_i .

Learning Nash Equilibria

Regret

The regret of player i after T rounds is

$$R_i(T) = \sum_{t=1}^T u_i(x_i^t, x_j^t) - u_i(x_i^*, x_j^t)$$

Full Information Setting

Here, i always receives the vector of utilities $u_i(\cdot, x_j^t)$ associated to each possible action. If v_i is the value of the game for i , and i is playing with an Hannan consistent algorithm (e.g. weighted majority), then its mean expected regret converge to v_i .

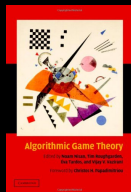
Hedge (A Hannan Consistent Algorithm with Low Regret)

Initialization: set $w_k^1 = 1$ for each action k .

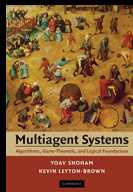
Trials: for each $t = 1, 2, \dots$

- (1) Strategy: set $s_k^t = \frac{1}{Z^t} w_k^t$.
- (2) Action: play action k with probability s_k^t
- (3) Response: receive $u^t(k)$ for each action k .
- (4) Update: set $w_k^{t+1} = w_k^t \exp(\eta_t u^t(k))$ for each action k

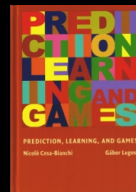
A Short Bibliography



Algorithmic
Game Theory
(Nisan et. al.)



Multiagent
Systems
(Shoham &
Leyton-Brown)



Prediction,
Learning
and Games
(Cesa-Bianchi
& Lugosi)