Algorithmic Game Theory and Applications

Nash Equilibrium and Zero-Sum Games

Solution Concept #3:
Pure Nash Equilibrium

Pure Nash Equilibrium (PNE): A pure strategy profile $(s_1, \ldots, s_n)$ such that for any player $i \in N$, *fixing the pure strategies* s_{-i} *of the other players*, player i cannot get higher utility from choosing a different pure strategy.

Mathematically: $u_i(s_i, s_{-i}) \geq u_i(s_i', s_{-i})$ for all $s_i' \in S_i$.

Equivalently: $s_i \in \arg\max_{\hat{s}_i \in S_i} u_i(\hat{s}_i, s_{-i})$

In words: s_i is a pure strategy that maximises the utility of the player, given the fixed strategies s_{-i} of the other players.

Terminology: s_i is a *pure best response* to s_{-i}.

Terminology: Player i does not have a profitable *unilateral deviation*.

Solution Concept #3*: (Mixed) Nash Equilibrium

Pure Nash Equilibrium (MNE): A mixed strategy profile $(x_1, \ldots, x_n)$ such that for any player $i \in N$, *fixing the mixed strategies x_{-i} of the other players*, player i cannot get higher utility from choosing a different mixed strategy.

Mathematically: $u_i(x_i, x_{-i}) \geq u_i(x_i', x_{-i})$ for all $x_i' \in \Delta(S_i)$.

Equivalently: $x_i \in \arg\max_{\hat{x}_i \in \Delta(S_i)} u_i(\hat{x}_i, x_{-i})$

In words: x_i is a mixed strategy that maximises the utility of the player, given the fixed strategies x_{-i} of the other players.

Terminology: x_i is a *(mixed) best response* to x_{-i}.

Terminology: Player i does not have a profitable *unilateral deviation*.

Solution Concept #3*:
(Mixed) Nash Equilibrium

Pure Nash Equilibrium (MNE): A mixed strategy profile $(x_1, \ldots, x_n)$ such that for any player $i \in N$, *fixing the mixed strategies x_{-i} of the other players*, player i cannot get higher utility from choosing a different mixed strategy.

Mathematically: $u_i(x_i, x_{-i}) \geq u_i(x_i', x_{-i})$ for all $x_i' \in \Delta(S_i)$.

Equivalently: $x_i \in \arg \max_{\hat{x}_i \in \Delta(S_i)} u_i(\hat{x}_i, x_{-i})$

Recall:
$u_i(\hat{x}_i, x_{-i}) = \mathbb{E}_{(s_i, s_{-i}) \sim (x_i, x_{-i})}[u_i(s_i, s_{-i})]$

In words: x_i is a mixed strategy that maximises the utility of the player, given the fixed strategies x_{-i} of the other players.

Terminology: x_i is a *(mixed) best response* to x_{-i}.

Terminology: Player i does not have a profitable *unilateral deviation*.

Fundamental Proposition

<u>Proposition 1:</u> A mixed strategy profile $x = (x_i, x_{-i})$ is a mixed Nash Equilibrium (MNE) if and only if, for every player $i \in N$ and every pure strategy $s_i' \in S_i$, we have

$$u_i(x_i, x_{-i}) \geq u_i(s_i', x_{-i})$$

Rock-Paper-Scissors

Consider the symmetric strategy (R, P, S) = (1/3, 1/3, 1/3) for both players. This is a MNE.

	Rock	Paper	Scissors	
Rock (R)	0, 0	-1, 1	1, -1	**1/3**
Paper (P)	1, -1	0, 0	-1, 1	**1/3**
Scissors (S)	-1, 1	1, -1	0, 0	**1/3**
	1/3	**1/3**	**1/3**	

$u_1(R, x_2) = 0$ →

$u_1(P, x_2) = 0$ →

$u_1(S, x_2) = 0$ →

$$u_1(x_1, x_2) = \frac{1}{3} \cdot 0 + \frac{1}{3} \cdot 1 + \frac{1}{3} \cdot (-1) = 0$$

Quick Recap: Efficient Algorithms

$O(\log n)$	$O(n)$	$O(n \log n)$	$O(n^2)$	$O(n^\alpha)$	$O(c^n)$
logarithmic	linear		quadratic	polynomial	exponential
The algorithm does not even read the whole input.	The algorithm accesses the input only a constant number of times.	The algorithm splits the inputs into two pieces of similar size, solves each part and merges the solutions.	The algorithm considers pairs of elements.	The algorithm performs many nested loops.	The algorithm considers many subsets of the input elements.

constant	$O(1)$	superlinear	$\omega(n)$
superconstant	$\omega(1)$	superpolynomial	$\omega(n^\alpha)$
sublinear	$o(n)$	subexponential	$o(c^n)$

Quick Recap: Efficient Algorithms

Polynomial time

$O(\log n)$	$O(n)$	$O(n \log n)$	$O(n^2)$	$O(n^\alpha)$	$O(c^n)$
logarithmic	linear		quadratic	polynomial	exponential
The algorithm does not even read the whole input.	The algorithm accesses the input only a constant number of times.	The algorithm splits the inputs into two pieces of similar size, solves each part and merges the solutions.	The algorithm considers pairs of elements.	The algorithm performs many nested loops.	The algorithm considers many subsets of the input elements.

constant	$O(1)$	superlinear	$\omega(n)$
superconstant	$\omega(1)$	superpolynomial	$\omega(n^\alpha)$
sublinear	$o(n)$	subexponential	$o(c^n)$

Quick Recap: Efficient Algorithms

Quick Recap: Efficient Algorithms

An algorithm is typically called *efficient* if it runs in polynomial time.

Quick Recap: Efficient Algorithms

An algorithm is typically called *efficient* if it runs in polynomial time.

By that, we mean in time which a polynomial function of the size of the input parameters.

Quick Recap: Efficient Algorithms

An algorithm is typically called *efficient* if it runs in polynomial time.

By that, we mean in time which a polynomial function of the size of the input parameters.

In the previous slide, that was abstractly denoted by “n”, but there might be more / more complex parameters in the input.

Quick Recap: Efficient Algorithms

An algorithm is typically called *efficient* if it runs in polynomial time.

By that, we mean in time which a polynomial function of the size of the input parameters.

In the previous slide, that was abstractly denoted by “n”, but there might be more / more complex parameters in the input.

Before we talk about efficient algorithms, we need to be sure about what our input is.

An efficient algorithm for verifying Nash equilibria

An efficient algorithm for verifying Nash equilibria

Informally:

Input: A game in normal form, a mixed strategy profile $x = (x_1, \ldots, x_n)$.

Output: Yes if x is a MNE and No if it is not.

An efficient algorithm for verifying Nash equilibria

Informally:

Input: A game in normal form, a mixed strategy profile $x = (x_1, \ldots, x_n)$.

Output: Yes if x is a MNE and No if it is not.

Formally:

Input: The number n of players, the pure strategy sets S_i, given explicitly, by listing all of their elements, the utility functions u_i given explicitly as a list of *rational* numbers, one for each pure strategy profile, e.g., $u_i(s_1, \ldots, s_n)$, the mixed strategies x_i, given as vectors $(x_{i1}, \ldots, x_{im})$ of *rational* numbers, where $m = |S_i|$.

Output: Yes if x is a MNE and No if it is not.

An efficient algorithm for verifying Nash equilibria

For every player $i \in N$ do

An efficient algorithm for verifying Nash equilibria

For every player $i \in N$ do

Compute $u_i(x_i, x_{-i})$

An efficient algorithm for verifying Nash equilibria

For every player $i \in N$ do

Compute $u_i(x_i, x_{-i})$

$$\sum_{s_1 \in S_1} \sum_{s_2 \in S_2} \cdots \sum_{s_n \in S_n} x_1(s_1) \cdot x_2(s_2) \cdot \ldots \cdot x_n(s_n) \cdot u_i(s_1, \ldots, s_n)$$

An efficient algorithm for verifying Nash equilibria

For every player $i \in N$ do

Compute $u_i(x_i, x_{-i})$

For every $s_{ij} \in S_i$ do

$$\sum_{s_1 \in S_1} \sum_{s_2 \in S_2} \cdots \sum_{s_n \in S_n} x_1(s_1) \cdot x_2(s_2) \cdot \ldots \cdot x_n(s_n) \cdot u_i(s_1, \ldots, s_n)$$

An efficient algorithm for verifying Nash equilibria

For every player $i \in N$ do

Compute $u_i(x_i, x_{-i})$

For every $s_{ij} \in S_i$ do

Compute $u_i(s_{ij}, x_{-i})$

$$\sum_{s_1 \in S_1} \sum_{s_2 \in S_2} \cdots \sum_{s_n \in S_n} x_1(s_1) \cdot x_2(s_2) \cdot \ldots \cdot x_n(s_n) \cdot u_i(s_1, \ldots, s_n)$$

An efficient algorithm for verifying Nash equilibria

For every player $i \in N$ do

Compute $u_i(x_i, x_{-i})$

For every $s_{ij} \in S_i$ do

Compute $u_i(s_{ij}, x_{-i})$

$$\sum_{s_1 \in S_1} \sum_{s_2 \in S_2} \cdots \sum_{s_n \in S_n} x_1(s_1) \cdot x_2(s_2) \cdot \ldots \cdot x_n(s_n) \cdot u_i(s_1, \ldots, s_n)$$

If $u_i(x_i, x_{-i}) < u_i(s_{ij}, x_{-i})$

An efficient algorithm for verifying Nash equilibria

For every player $i \in N$ do

Compute $u_i(x_i, x_{-i})$

For every $s_{ij} \in S_i$ do

Compute $u_i(s_{ij}, x_{-i})$

$$\sum_{s_1 \in S_1} \sum_{s_2 \in S_2} \cdots \sum_{s_n \in S_n} x_1(s_1) \cdot x_2(s_2) \cdot \ldots \cdot x_n(s_n) \cdot u_i(s_1, \ldots, s_n)$$

If $u_i(x_i, x_{-i}) < u_i(s_{ij}, x_{-i})$

Return No

An efficient algorithm for verifying Nash equilibria

For every player $i \in N$ do

Compute $u_i(x_i, x_{-i})$

For every $s_{ij} \in S_i$ do

Compute $u_i(s_{ij}, x_{-i})$

$$\sum_{s_1 \in S_1} \sum_{s_2 \in S_2} \cdots \sum_{s_n \in S_n} x_1(s_1) \cdot x_2(s_2) \cdot \ldots \cdot x_n(s_n) \cdot u_i(s_1, \ldots, s_n)$$

If $u_i(x_i, x_{-i}) < u_i(s_{ij}, x_{-i})$

Return No

Return Yes

Fundamental Proposition

Proposition 1: A mixed strategy profile $x = (x_i, x_{-i})$ is a mixed Nash Equilibrium (MNE) if and only if, for every player $i \in N$ and every pure strategy $s_i' \in S_i$, we have

$$u_i(x_i, x_{-i}) \geq u_i(s_i', x_{-i})$$

Another Fundamental Proposition

Proposition 2: A mixed strategy profile $x = (x_i, x_{-i})$ is a mixed Nash Equilibrium (MNE) if and only if, for every player $i \in N$, and for every pure strategy $s_i \in S_i$ in the support of x_i (i.e., $x_i(s_i) > 0$), we have $u_i(x_i, x_{-i}) = u_i(s_i, x_{-i})$.

A quick proof of $\Leftarrow$

Proposition 2: A mixed strategy profile $x = (x_i, x_{-i})$ is a mixed Nash Equilibrium (MNE) if and only if, for every player $i \in N$, and for every pure strategy $s_i \in S_i$ in the support of x_i (i.e., $x_i(s_i) > 0$), we have $u_i(x_i, x_{-i}) = u_i(s_i, x_{-i})$.

A quick proof of $\Leftarrow$

Proposition 2: A mixed strategy profile $x = (x_i, x_{-i})$ is a mixed Nash Equilibrium (MNE) if and only if, for every player $i \in N$, and for every pure strategy $s_i \in S_i$ in the support of x_i (i.e., $x_i(s_i) > 0$), we have $u_i(x_i, x_{-i}) = u_i(s_i, x_{-i})$.

Let $x = (x_i, x_{-i})$ be a MNE. This immediately implies $u_i(s_i, x_{-i}) \leq u_i(x_i, x_{-i})$ for all $s_i \in S_i$.

A quick proof of $\Leftarrow$

Proposition 2: A mixed strategy profile $x = (x_i, x_{-i})$ is a mixed Nash Equilibrium (MNE) if and only if, for every player $i \in N$, and for every pure strategy $s_i \in S_i$ in the support of x_i (i.e., $x_i(s_i) > 0$), we have $u_i(x_i, x_{-i}) = u_i(s_i, x_{-i})$.

Let $x = (x_i, x_{-i})$ be a MNE. This immediately implies $u_i(s_i, x_{-i}) \leq u_i(x_i, x_{-i})$ for all $s_i \in S_i$.

Assume by contradiction that for some strategy $s_i' \in \text{supp}(x_i)$, we had $u_i(s_i, x_{-i}) < u_i(x_i, x_{-i})$

A quick proof of $\Leftarrow$

Proposition 2: A mixed strategy profile $x = (x_i, x_{-i})$ is a mixed Nash Equilibrium (MNE) if and only if, for every player $i \in N$, and for every pure strategy $s_i \in S_i$ in the support of x_i (i.e., $x_i(s_i) > 0$), we have $u_i(x_i, x_{-i}) = u_i(s_i, x_{-i})$.

Let $x = (x_i, x_{-i})$ be a MNE. This immediately implies $u_i(s_i, x_{-i}) \leq u_i(x_i, x_{-i})$ for all $s_i \in S_i$.

Assume by contradiction that for some strategy $s_i' \in \text{supp}(x_i)$, we had $u_i(s_i, x_{-i}) < u_i(x_i, x_{-i})$

Let $s_i^* \in \arg\max_{s_i \in \text{supp}(x_i)} u_i(s_i, x_{-i})$. By definition this is $\geq u_i(x_i, x_{-i})$

A quick proof of $\Leftarrow$

Proposition 2: A mixed strategy profile $x = (x_i, x_{-i})$ is a mixed Nash Equilibrium (MNE) if and only if, for every player $i \in N$, and for every pure strategy $s_i \in S_i$ in the support of x_i (i.e., $x_i(s_i) > 0$), we have $u_i(x_i, x_{-i}) = u_i(s_i, x_{-i})$.

Let $x = (x_i, x_{-i})$ be a MNE. This immediately implies $u_i(s_i, x_{-i}) \leq u_i(x_i, x_{-i})$ for all $s_i \in S_i$.

Assume by contradiction that for some strategy $s_i' \in \text{supp}(x_i)$, we had $u_i(s_i, x_{-i}) < u_i(x_i, x_{-i})$

Let $s_i^* \in \arg\max_{s_i \in \text{supp}(x_i)} u_i(s_i, x_{-i})$. By definition this is $\geq u_i(x_i, x_{-i})$

Consider the alternative mixed strategy x_i' such that $x_i'(s_i) = x_i(s_i)$ for all pure strategies $s_i \neq s_i', s_i^*$ and

$x_i(s_i') = 0$

$x_i'(s_i^*) = x_i(s_i^*) + x_i(s_i')$

A quick proof of $\Leftarrow$

Proposition 2: A mixed strategy profile $x = (x_i, x_{-i})$ is a mixed Nash Equilibrium (MNE) if and only if, for every player $i \in N$, and for every pure strategy $s_i \in S_i$ in the support of x_i (i.e., $x_i(s_i) > 0$), we have $u_i(x_i, x_{-i}) = u_i(s_i, x_{-i})$.

Let $x = (x_i, x_{-i})$ be a MNE. This immediately implies $u_i(s_i, x_{-i}) \leq u_i(x_i, x_{-i})$ for all $s_i \in S_i$.

Assume by contradiction that for some strategy $s_i' \in \text{supp}(x_i)$, we had $u_i(s_i, x_{-i}) < u_i(x_i, x_{-i})$

Let $s_i^* \in \arg\max_{s_i \in \text{supp}(x_i)} u_i(s_i, x_{-i})$. By definition this is $\geq u_i(x_i, x_{-i})$

Consider the alternative mixed strategy x_i' such that $x_i'(s_i) = x_i(s_i)$ for all pure strategies $s_i \neq s_i', s_i^*$ and

$x_i(s_i') = 0$

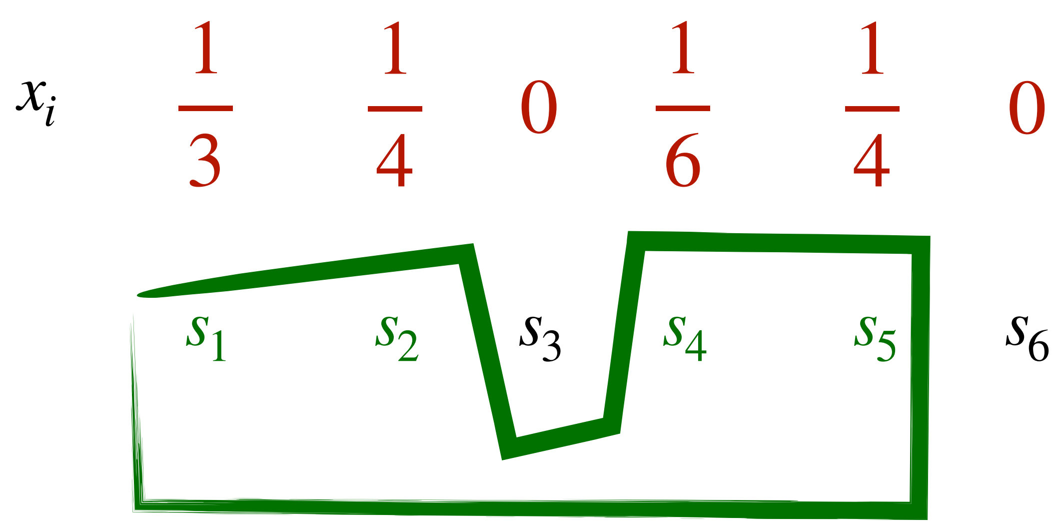
$x_i'(s_i^*) = x_i(s_i^*) + x_i(s_i')$

x_i' results in higher utility, a contradiction!

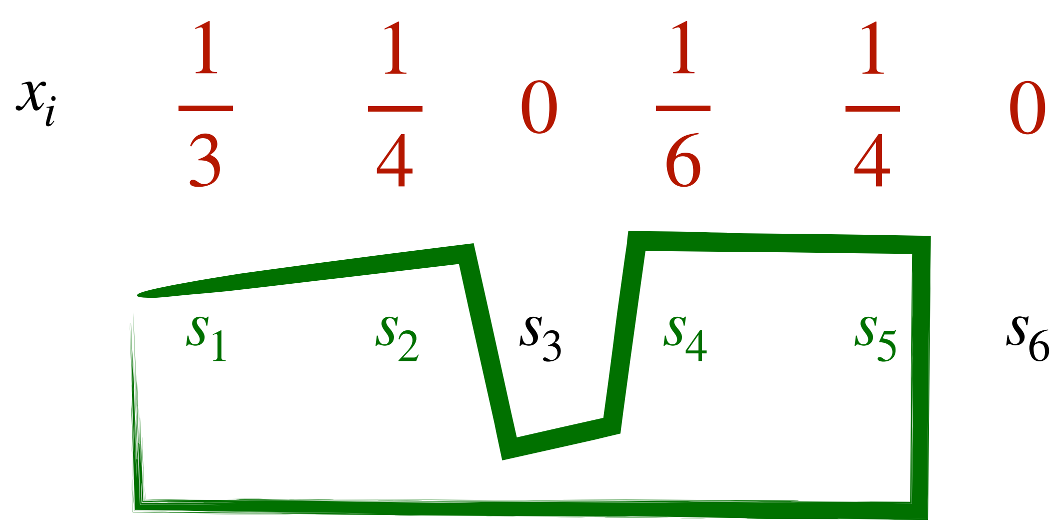
Via example:

x_i	$\frac{1}{3}$	$\frac{1}{4}$	0	$\frac{1}{6}$	$\frac{1}{4}$	0
	s_1	s_2	s_3	s_4	s_5	s_6

Via example:

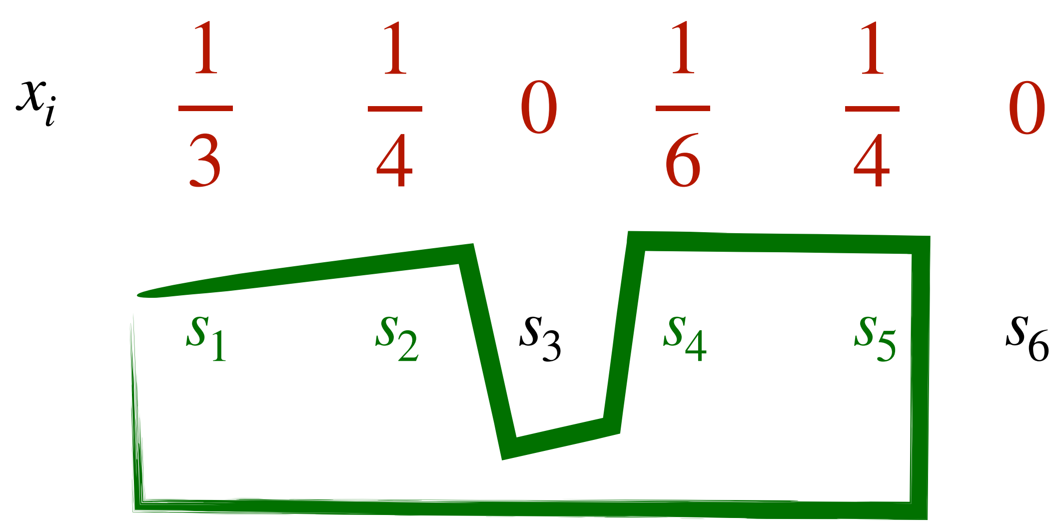


Via example:



Claim: The utility $u_i(s_i, x_{-i})$ for every strategy in the support is the same.

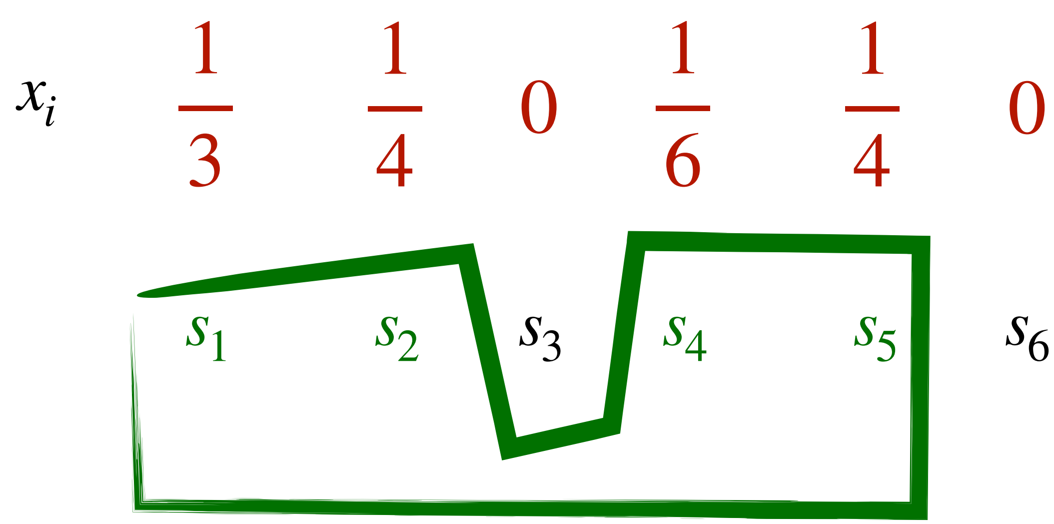
Via example:



Claim: The utility $u_i(s_i, x_{-i})$ for every strategy in the support is the same.

By contradiction: Assume that this is not the case.

Via example:

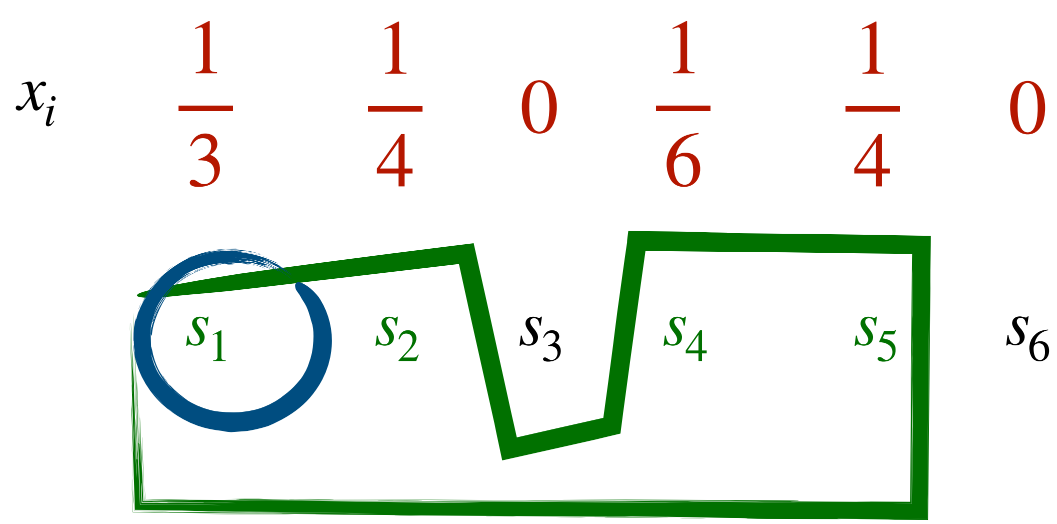


Claim: The utility $u_i(s_i, x_{-i})$ for every strategy in the support is the same.

By contradiction: Assume that this is not the case.

Then there are two pure strategies s_i, s_j such that s_i gives less utility than s_j.

Via example:

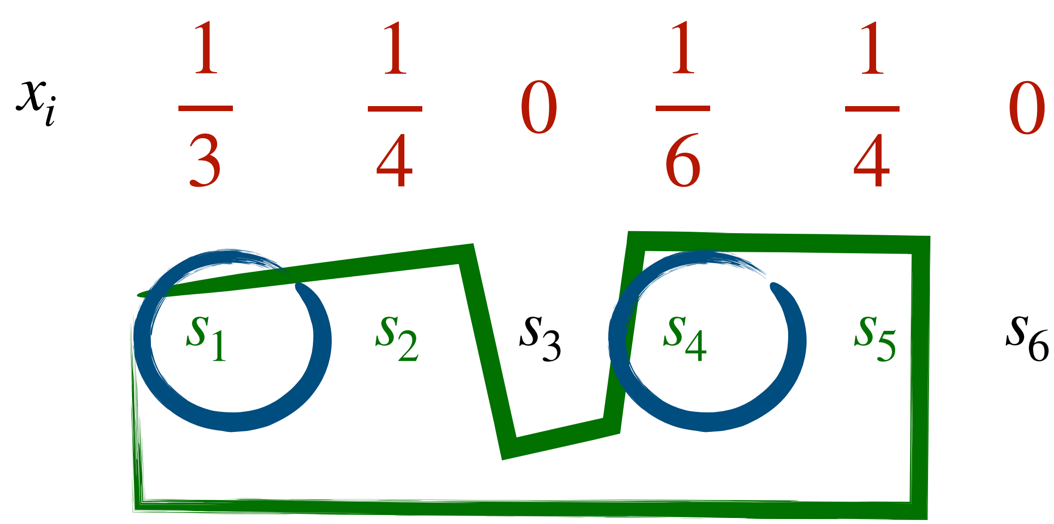


Claim: The utility $u_i(s_i, x_{-i})$ for every strategy in the support is the same.

By contradiction: Assume that this is not the case.

Then there are two pure strategies s_i, s_j such that s_i gives less utility than s_j.

Via example:

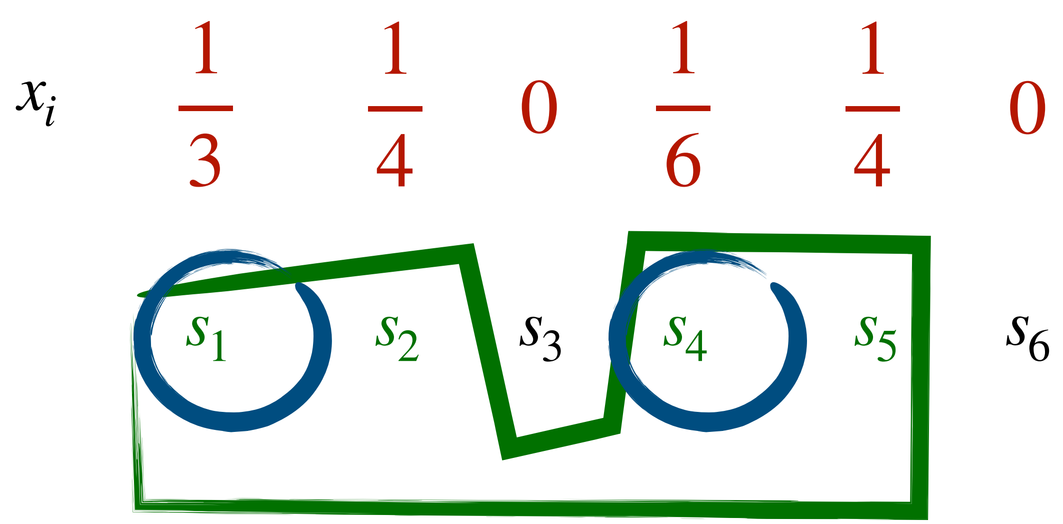


Claim: The utility $u_i(s_i, x_{-i})$ for every strategy in the support is the same.

By contradiction: Assume that this is not the case.

Then there are two pure strategies s_i, s_j such that s_i gives less utility than s_j.

Via example:



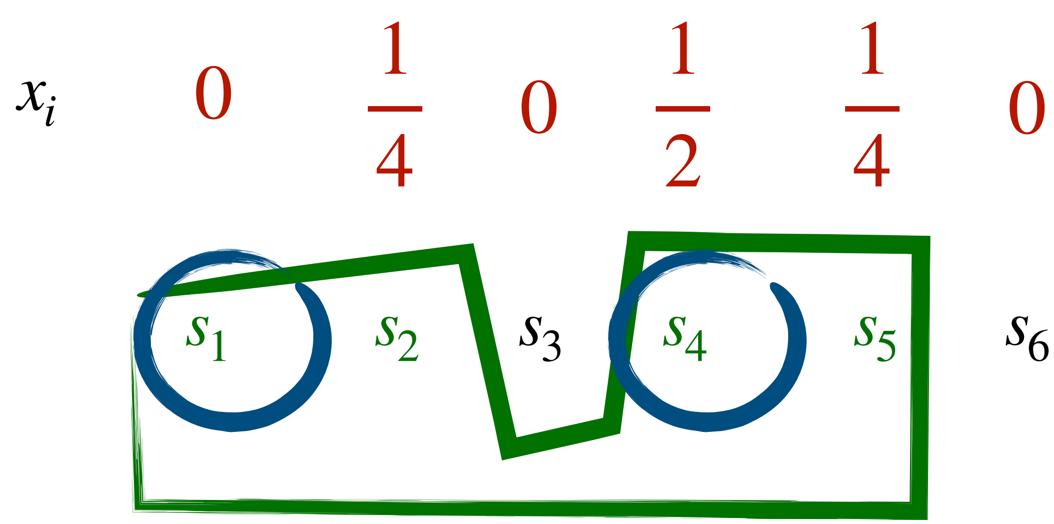
Claim: The utility $u_i(s_i, x_{-i})$ for every strategy in the support is the same.

By contradiction: Assume that this is not the case.

Then there are two pure strategies s_i, s_j such that s_i gives less utility than s_j.

Take the probability from s_i and move it to s_j.

Via example:



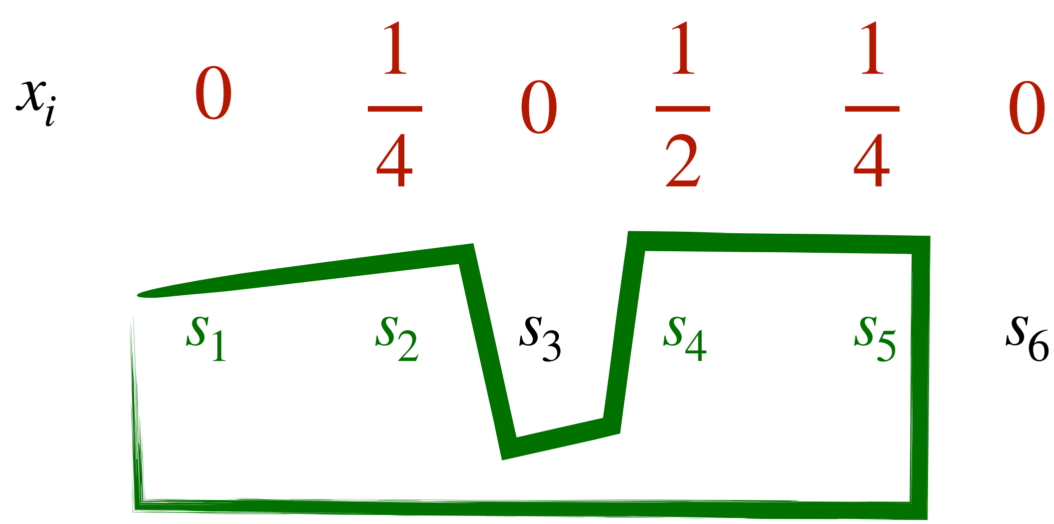
Claim: The utility $u_i(s_i, x_{-i})$ for every strategy in the support is the same.

By contradiction: Assume that this is not the case.

Then there are two pure strategies s_i, s_j such that s_i gives less utility than s_j.

Take the probability from s_i and move it to s_j.

Via example:



Claim: The utility $u_i(s_i, x_{-i})$ for every strategy in the support is the same.

By contradiction: Assume that this is not the case.

Then there are two pure strategies s_i, s_j such that s_i gives less utility than s_j.

Take the probability from s_i and move it to s_j.

Via example:

x_i	0	$\frac{1}{4}$	0	$\frac{1}{2}$	$\frac{1}{4}$	0
	s_1	s_2	s_3	s_4	s_5	s_6

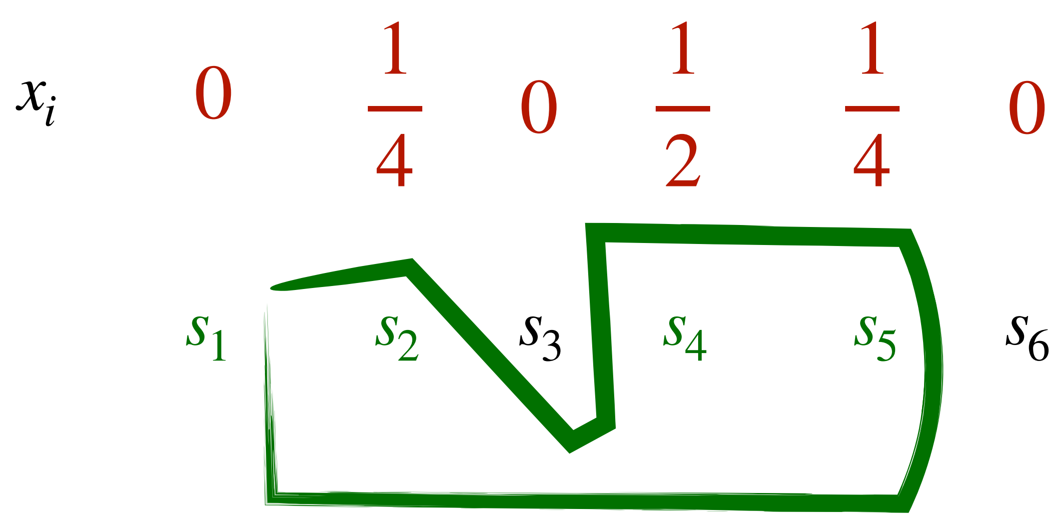
Claim: The utility $u_i(s_i, x_{-i})$ for every strategy in the support is the same.

By contradiction: Assume that this is not the case.

Then there are two pure strategies s_i, s_j such that s_i gives less utility than s_j.

Take the probability from s_i and move it to s_j.

Via example:



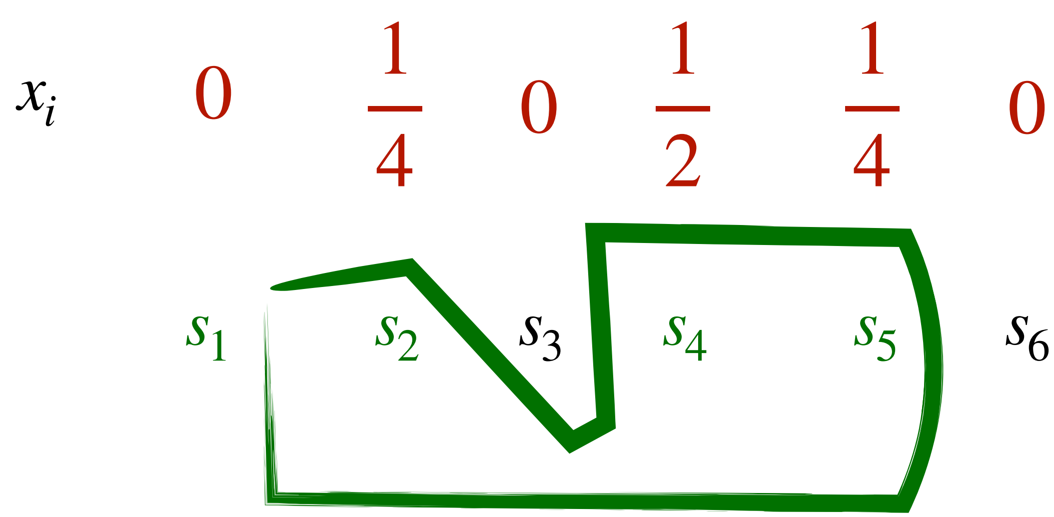
Claim: The utility $u_i(s_i, x_{-i})$ for every strategy in the support is the same.

By contradiction: Assume that this is not the case.

Then there are two pure strategies s_i, s_j such that s_i gives less utility than s_j.

Take the probability from s_i and move it to s_j.

Via example:



Claim: The utility $u_i(s_i, x_{-i})$ for every strategy in the support is the same.

By contradiction: Assume that this is not the case.

Then there are two pure strategies s_i, s_j such that s_i gives less utility than s_j.

Take the probability from s_i and move it to s_j.

We have created a better (i.e., with higher expected utility) mixed strategy x_i'.

Back to Choosing the TV show

Both
(Peep Show, Peep Show)
and
(FOTC, FOTC),
are PNE!

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

Both
(Peep Show, Peep Show)
and
(FOTC, FOTC),
are PNE!

What about mixed equilibria?

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

Both
(Peep Show, Peep Show)
and
(FOTC, FOTC),
are PNE!

What about mixed equilibria?

Remember: By definition, PNE are MNE, so we already have found two!

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

Both
(Peep Show, Peep Show)
and
(FOTC, FOTC),
are PNE!

What about mixed equilibria?

Remember: By definition, PNE are MNE, so we already have found two!

We know here that if there are any others, they have to have full support.

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

Both
(Peep Show, Peep Show)
and
(FOTC, FOTC),
are PNE!

What about mixed equilibria?

Remember: By definition, PNE are MNE, so we already have found two!

We know here that if there are any others, they have to have full support.

We call those *fully mixed*.

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

Assume that we have a mixed equilibrium (x, y)

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

Assume that we have a mixed equilibrium (x, y)

<u>Note:</u> We use x and y here instead of x_1 and x_2 because we only have two players. We will therefore use x_i, y_i to denote probabilities.

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

Assume that we have a mixed equilibrium (x, y)

Note: We use x and y here instead of x_1 and x_2 because we only have two players. We will therefore use x_i, y_i to denote probabilities.

Let (x_1, x_2) be the mixed strategy of Player 1 and (y_1, y_2) be the mixed strategy of Player 2.

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

By Proposition 2, we know that

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

By Proposition 2, we know that

$$u_1(x_1, y) = u_1(x_2, y)$$

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

By Proposition 2, we know that

$$u_1(x_1, y) = u_1(x_2, y)$$

In other words,

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

By Proposition 2, we know that

$$u_1(x_1, y) = u_1(x_2, y)$$

In other words,

$$10y_1 + 5y_2 = y_1 + 7y_2$$

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

By Proposition 2, we know that

$$u_1(x_1, y) = u_1(x_2, y)$$

In other words,

$$10y_1 + 5y_2 = y_1 + 7y_2$$

$$\Rightarrow 9y_1 - 2y_2 = 0 \text{ (1)}$$

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

By Proposition 2, we know that

$$u_1(x_1, y) = u_1(x_2, y)$$

In other words,

$$10y_1 + 5y_2 = y_1 + 7y_2$$

$$\Rightarrow 9y_1 - 2y_2 = 0 \text{ (1)}$$

We also have:

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

By Proposition 2, we know that

$$u_1(x_1, y) = u_1(x_2, y)$$

In other words,

$$10y_1 + 5y_2 = y_1 + 7y_2$$

$$\Rightarrow 9y_1 - 2y_2 = 0 \text{ (1)}$$

We also have:

$$y_1 + y_2 = 1 \text{ (2)}$$

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

By Proposition 2, we know that

$$u_1(x_1, y) = u_1(x_2, y)$$

In other words,

$$10y_1 + 5y_2 = y_1 + 7y_2$$

$$\Rightarrow 9y_1 - 2y_2 = 0 \text{ (1)}$$

We also have:

$$y_1 + y_2 = 1 \text{ (2)}$$

Combining (1) and (2) we get

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

By Proposition 2, we know that

$$u_1(x_1, y) = u_1(x_2, y)$$

In other words,

$$10y_1 + 5y_2 = y_1 + 7y_2$$

$$\Rightarrow 9y_1 - 2y_2 = 0 \text{ (1)}$$

We also have:

$$y_1 + y_2 = 1 \text{ (2)}$$

Combining (1) and (2) we get

$$y_1 = 2/11, \quad y_2 = 9/11$$

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

Similarly we can calculate

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Back to Choosing the TV show

Similarly we can calculate

$$x_1 = 9/11, \quad x_2 = 2/11$$

	Peep Show	FOTC
Peep Show	10, 7	5, 5
FOTC	1, 1	7, 10

Another Fundamental Proposition

Proposition 2: A mixed strategy profile $x = (x_i, x_{-i})$ is a mixed Nash Equilibrium (MNE) if and only if, for every player $i \in N$, and for every pure strategy $s_i \in S_i$ in the support of x_i (i.e., $x_i(s_i) > 0$), we have $u_i(x_i, x_{-i}) = u_i(s_i, x_{-i})$.

Question: *Can you translate the idea we just used into an algorithm, which takes advantage of the proposition above?*

An algorithm for computing Nash equilibria in 2-player games

Assume that we have *magical access* to the supports for all mixed strategies in the MNE.

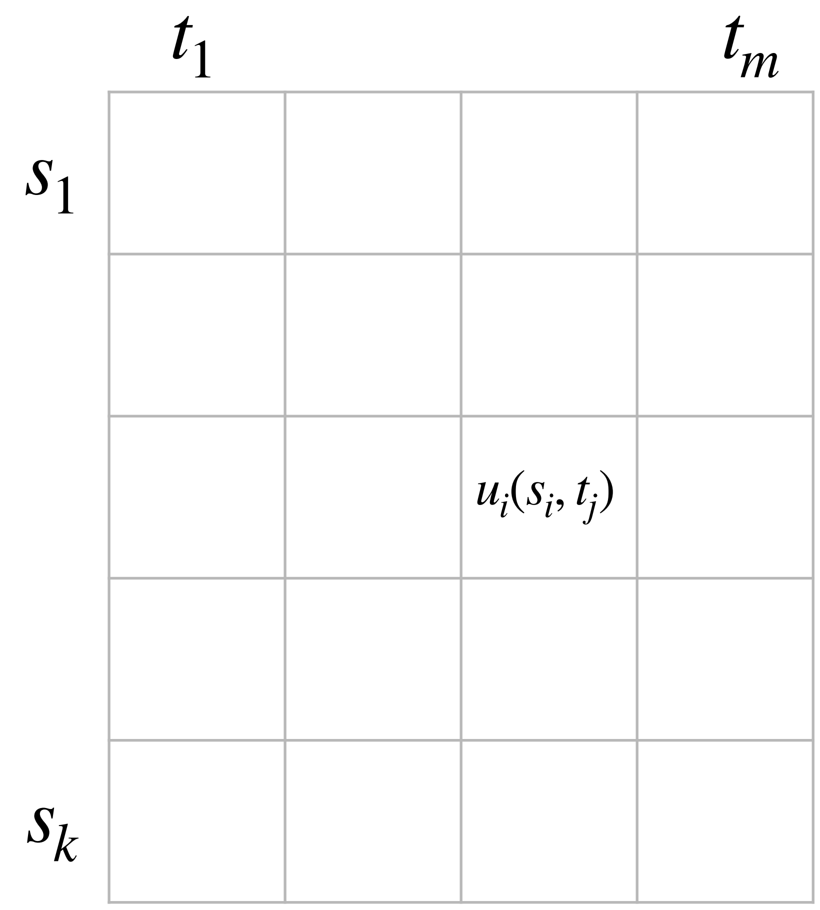
In algorithms, we often call this *oracle access*.

We can then write a set of inequalities:

$$\sum_{y_j \in \text{supp}(y)} y(t_j) \cdot u_i(s_i, t_j) \quad =$$

$$\sum_{y_j \in \text{supp}(y)} y(t_j) \cdot u_i(s_i', t_j) \text{ for all } s_i, s_i \in \text{supp}(x)$$

$$\sum_{y_j \in \text{supp}(y)} y(t_j) = 1$$



$y(t_j) = \Pr[\, y \text{ chooses } t_j \,]$

 An algorithm for computing Nash equilibria in 2-player games

Assume that we have *magical access* to the supports for all mixed strategies in the MNE.

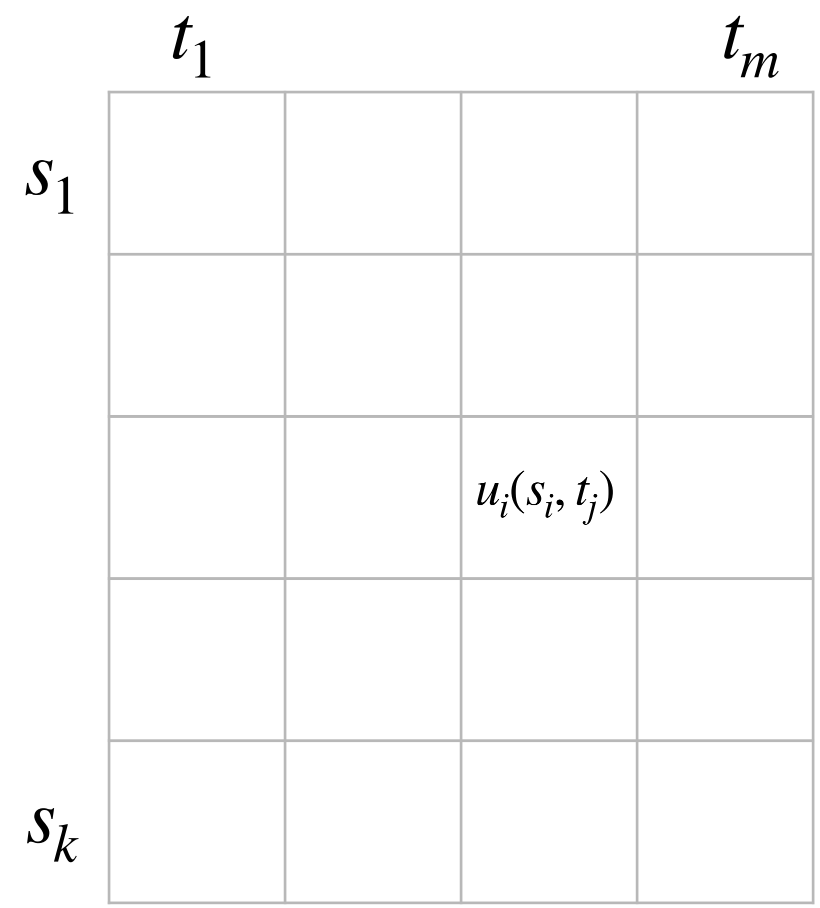
In algorithms, we often call this *oracle access*.

We can then write a set of inequalities:

$$\sum_{y_j \in \text{supp}(y)} y(t_j) \cdot u_i(s_i, t_j) \quad =$$

$$\sum_{y_j \in \text{supp}(y)} y(t_j) \cdot u_i(s_i', t_j) \text{ for all } s_i, s_i \in \text{supp}(x)$$

$$\sum_{y_j \in \text{supp}(y)} y(t_j) = 1$$



$y(t_j) = \text{Pr}[\, y \text{ chooses } t_j \,]$

This computes the equilibrium strategy of Player 2, based on Player 1

An algorithm for computing Nash equilibria in 2-player games

Assume that we have *magical access* to the supports for all mixed strategies in the MNE.

In algorithms, we often call this *oracle access*.

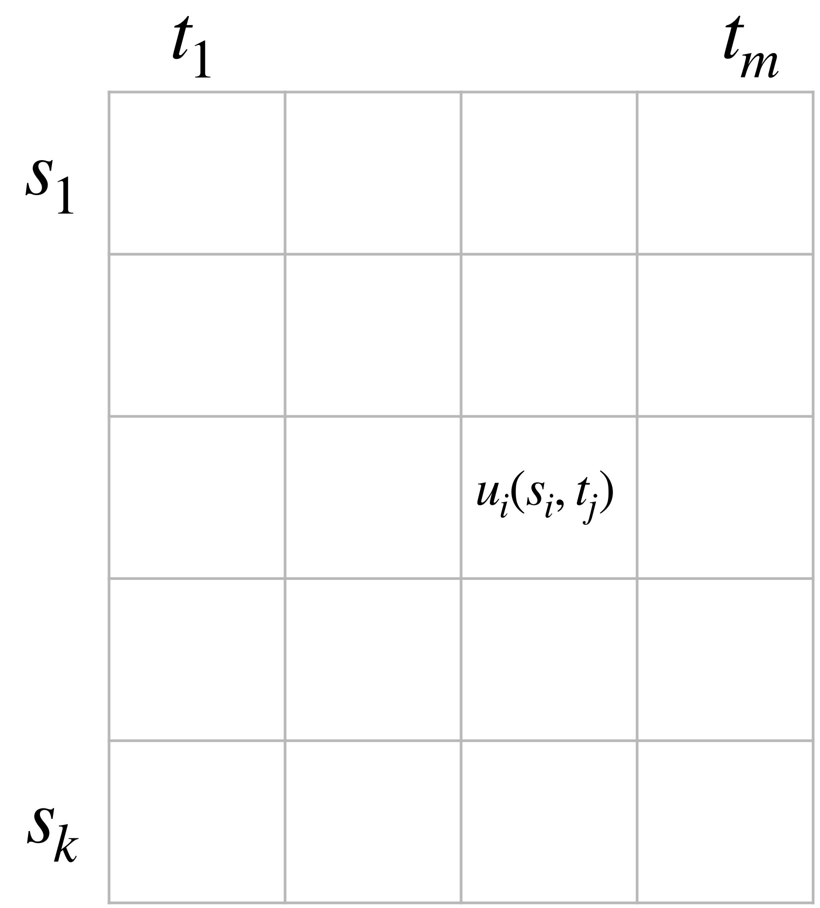
We can then write a set of inequalities:

$$\sum_{y_j \in \text{supp}(y)} y(t_j) \cdot u_i(s_i, t_j) \quad = $$

$$\sum_{y_j \in \text{supp}(y)} y(t_j) \cdot u_i(s_i', t_j) \text{ for all } s_i, s_i \in \text{supp}(x)$$

$$\sum_{y_j \in \text{supp}(y)} y(t_j) = 1$$

This computes the equilibrium strategy of Player 2, based on Player 1



$y(t_j) = \Pr[\, y \text{ chooses } t_j \,]$

Similarly we can compute the equilibrium strategy x of Player 1, based on Player 2.

An algorithm for computing Nash equilibria in 2-player games

Assume that we have *magical access* to the supports for all mixed strategies in the MNE.

In algorithms, we often call this *oracle access*.

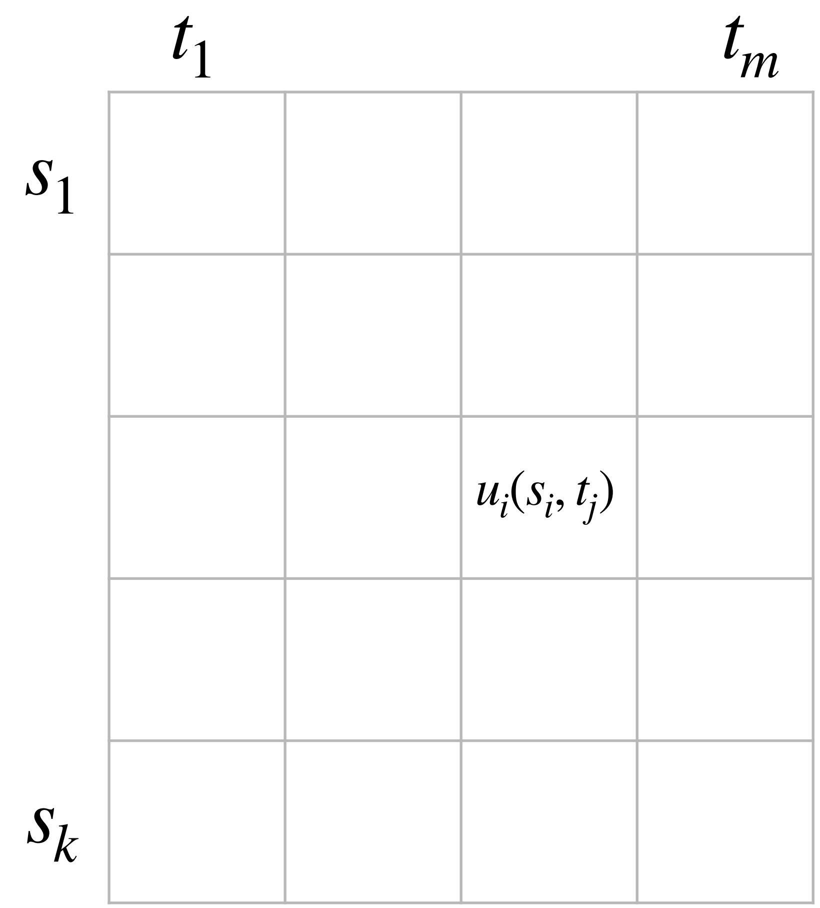
We can then write a set of inequalities:

$$\sum_{y_j \in \text{supp}(y)} y(t_j) \cdot u_i(s_i, t_j) \quad = \quad \sum_{y_j \in \text{supp}(y)} y(t_j) \cdot u_i(s_i', t_j) \text{ for all } s_i, s_i \in \text{supp}(x)$$

Can we solve this in polynomial time?

$$\sum_{y_j \in \text{supp}(y)} y(t_j) = 1$$

This computes the equilibrium strategy of Player 2, based on Player 1



$y(t_j) = \Pr[\, y \text{ chooses } t_j \,]$

Similarly we can compute the equilibrium strategy x of Player 1, based on Player 2.

A bit more precisely

By using the notation of utilities, we want a solution to the following system of inequalities:

1. $\forall i \in N, \forall s_j \in \text{ supp}(x_i), \quad u_i(s_j, x_{-i}) = w_i$ (Proposition 2)
2. $\forall i \in N, \forall s_j \notin \text{supp}(x_i), \quad u_i(s_j, x_{-i}) \leq w_i$ (MNE condition)
3. $\forall i \in N, \quad \sum_{s_j \in S_j} x_i(s_j) = 1$ (probabilities)
4. $\forall i \in N, \forall s_j \in \text{ supp}(x_i) \;\; x_i(s_j) \geq 0$ (in the support)
5. $\forall i \in N, \forall s_j \notin \text{ supp}(x_i) \;\; x_i(s_j) = 0$ (not in the support)

A bit more precisely

By using the notation of utilities, we want a solution to the following system of inequalities:

1. $\forall i \in N, \forall s_j \in$ supp(x_i), $\;u_i(s_j, x_{-i}) = w_i$ (Proposition 2)

2. $\forall i \in N, \forall s_j \notin$ supp(x_i), $\;u_i(s_j, x_{-i}) \leq w_i$ (MNE condition)

3. $\forall i \in N, \quad \sum_{s_j \in S_j} x_i(s_j) = 1$ (probabilities)

4. $\forall i \in N, \forall s_j \in$ supp(x_i) $\;x_i(s_j) \geq 0$ (in the support)

5. $\forall i \in N, \forall s_j \notin$ supp(x_i) $\;x_i(s_j) = 0$ (not in the support)

This actually holds for any number of players, but the inequalities are linear only for two players. Why?

 An algorithm for computing Nash equilibria in 2-player games

Assume that we have *magical access* to the supports for all mixed strategies in the MNE.

In algorithms, we often call this *oracle access*.

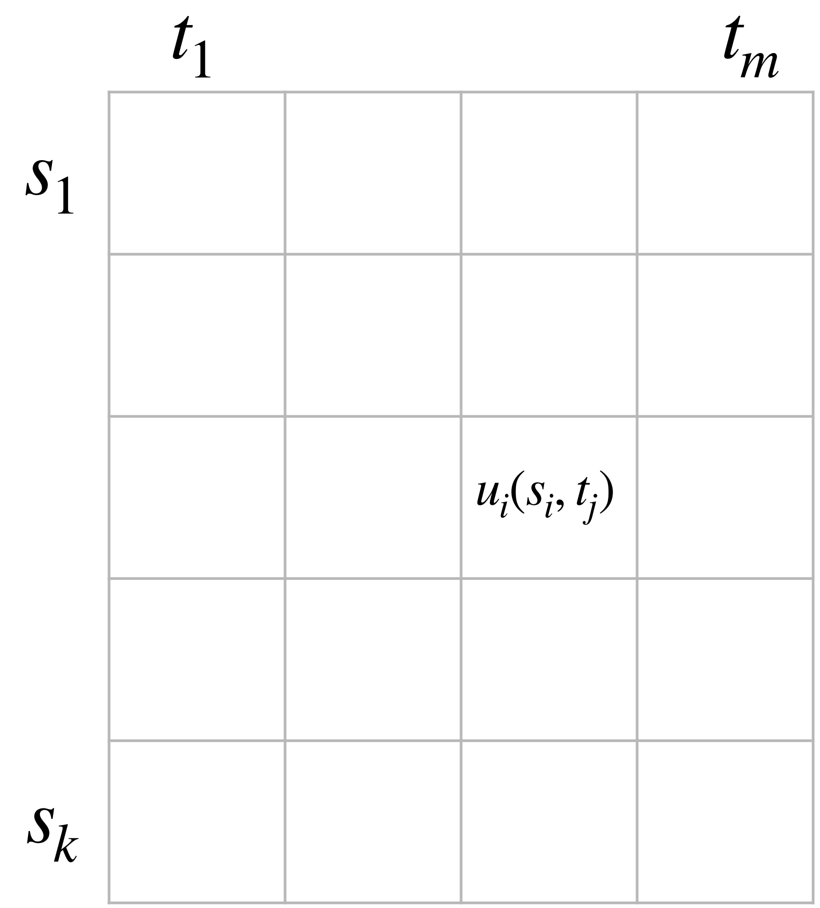
We can then write a set of inequalities:

$$\sum_{y_j \in \text{supp}(y)} y(t_j) \cdot u_i(s_i, t_j) \quad =$$

We can solve this in polynomial time!

$$\sum_{y_j \in \text{supp}(y)} y(t_j) \cdot u_i(s_i', t_j) \text{ for all } s_i, s_i \in \text{supp}(x)$$

$$\sum_{y_j \in \text{supp}(y)} y(t_j) = 1$$



$y(t_j) = \Pr[\, y \text{ chooses } t_j \,]$

An algorithm for computing Nash equilibria in 2-player games

Assume that we have *magical access* to the supports for all mixed strategies in the MNE.

In algorithms, we often call this *oracle access*.

We can then write a set of inequalities:

$$\sum_{y_j \in \text{supp}(y)} y(t_j) \cdot u_i(s_i, t_j) \quad = \quad \sum_{y_j \in \text{supp}(y)} y(t_j) \cdot u_i(s_i', t_j) \text{ for all } s_i, s_i \in \text{supp}(x)$$

We can solve this in polynomial time!

$$\sum_{y_j \in \text{supp}(y)} y(t_j) = 1$$

What about this?

	t_1			t_m
s_1				
			$u_i(s_i, t_j)$	
s_k				

$y(t_j) = \Pr[\, y \text{ chooses } t_j \,]$

Support Enumeration

1. Guess the support for the each mixed equilibrium strategy.

Support Enumeration

1. Guess the support for the each mixed equilibrium strategy.

   That means enumerate (i.e., try out all) the different possible supports.

Support Enumeration

1. Guess the support for the each mixed equilibrium strategy.

   That means enumerate (i.e., try out all) the different possible supports.

2. Write and solve the system of linear equations.

Support Enumeration

1. Guess the support for the each mixed equilibrium strategy.

   That means enumerate (i.e., try out all) the different possible supports.

2. Write and solve the system of linear equations.

3. If the system does not return “infeasible”, use the previous algorithm to verify that what you have computed is indeed a MNE.

Support Enumeration

1. Guess the support for the each mixed equilibrium strategy.

   That means enumerate (i.e., try out all) the different possible supports.

2. Write and solve the system of linear equations.

   3. If the system does not return “infeasible”, use the previous algorithm to verify that what you have computed is indeed a MNE.

   4. If it is, break and terminate. Otherwise continue.

Support Enumeration

1. Guess the support for the each mixed equilibrium strategy.

   That means enumerate (i.e., try out all) the different possible supports.

2. Write and solve the system of linear equations.

3. If the system does not return “infeasible”, use the previous algorithm to verify that what you have computed is indeed a MNE.

4. If it is, break and terminate. Otherwise continue.

Question 1: How do we know that one of the supports will indeed give us a MNE?

Support Enumeration

1. Guess the support for the each mixed equilibrium strategy.

   That means enumerate (i.e., try out all) the different possible supports.

2. Write and solve the system of linear equations.

3. If the system does not return “infeasible”, use the previous algorithm to verify that what you have computed is indeed a MNE.

4. If it is, break and terminate. Otherwise continue.

Question 1: How do we know that one of the supports will indeed give us a MNE?

Question 2: How fast is this algorithm? How many possible supports are there?

2-player Zero-Sum Games

2-player Zero-Sum Games

2 players with pure strategy sets

$S = \{s_1, \ldots, s_{m_1}\}$ and $T = \{t_1, \ldots, t_{m_2}\}$

2-player Zero-Sum Games

2 players with pure strategy sets
$S = \{s_1, \ldots, s_{m_1}\}$ and $T = \{t_1, \ldots, t_{m_2}\}$

The utility functions are such that for any $s_i \in S$ and $t_i \in T$, we have $u_1(s_i, t_i) = -u_2(s_i, t_i)$

2-player Zero-Sum Games

2 players with pure strategy sets
$S = \{s_1, \ldots, s_{m_1}\}$ and $T = \{t_1, \ldots, t_{m_2}\}$

The utility functions are such that for any $s_i \in S$ and $t_i \in T$, we have $u_1(s_i, t_i) = -u_2(s_i, t_i)$

We can therefore drop the subscripts and write $u(i, j)$ as a shorthand for $u_1(s_i, t_j)$ and $-u(i, j)$ as a shorthand for $u_1(s_i, t_j)$.

2-player Zero-Sum Games

2 players with pure strategy sets
$S = \{s_1, \ldots, s_{m_1}\}$ and $T = \{t_1, \ldots, t_{m_2}\}$

The utility functions are such that for any $s_i \in S$ and $t_i \in T$, we have $u_1(s_i, t_i) = -u_2(s_i, t_i)$

We can therefore drop the subscripts and write $u(i, j)$ as a shorthand for $u_1(s_i, t_j)$ and $-u(i, j)$ as a shorthand for $u_1(s_i, t_j)$.

Player 1 is trying to maximise the utility (maximiser) and Player 2 is trying to minimise it (minimiser).

Rock-Paper-Scissors

	Rock	Paper	Scissors
Rock (R)	0, 0	-1, 1	1, -1
Paper (P)	1, -1	0, 0	-1, 1
Scissors (S)	-1, 1	1, -1	0, 0

Rock-Paper-Scissors

	Rock	Paper	Scissors
Rock (R)	0	-1	1
Paper (P)	1	0	-1
Scissors (S)	-1	1	0

Quick Detour: Linear Algebra Refresher

Quick Detour: Linear Algebra Refresher

For matrix operations, we will assume column-vector notation.

Quick Detour: Linear Algebra Refresher

For matrix operations, we will assume column-vector notation.

For example, for $z = (z_1, z_2, \ldots, z_m) \in \mathbb{R}^m$, we have:

Quick Detour: Linear Algebra Refresher

For matrix operations, we will assume column-vector notation.

For example, for $z = (z_1, z_2, \ldots, z_m) \in \mathbb{R}^m$, we have:

$$z = \begin{bmatrix} z_1 \\ z_2 \\ \vdots \\ z_m \end{bmatrix} \quad \text{and} \quad z^\top = [z_1 \quad z_2 \quad \cdots \quad z_m].$$

Quick Detour: Linear Algebra Refresher

Quick Detour: Linear Algebra Refresher

Consider a matrix $A = (a_{i,j}) \in \mathbb{R}^{m \times n}$. Then:

Quick Detour: Linear Algebra Refresher

Consider a matrix $A = (a_{i,j}) \in \mathbb{R}^{m \times n}$. Then:

$$A\boldsymbol{y} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} a_{1,1}y_1 + a_{1,2}y_2 + \cdots + a_{1,n}y_n \\ a_{2,1}y_1 + a_{2,2}y_2 + \cdots + a_{2,n}y_n \\ \vdots \\ a_{m,1}y_1 + a_{m,2}y_2 + \cdots + a_{m,n}y_n \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{n} a_{1,j}y_j \\ \sum_{j=1}^{n} a_{2,j}y_j \\ \vdots \\ \sum_{j=1}^{n} a_{m,j}y_j \end{bmatrix}$$

Quick Detour: Linear Algebra Refresher

Consider a matrix $A = (a_{i,j}) \in \mathbb{R}^{m \times n}$. Then:

$$A\boldsymbol{y} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} a_{1,1}y_1 + a_{1,2}y_2 + \cdots + a_{1,n}y_n \\ a_{2,1}y_1 + a_{2,2}y_2 + \cdots + a_{2,n}y_n \\ \vdots \\ a_{m,1}y_1 + a_{m,2}y_2 + \cdots + a_{m,n}y_n \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{n} a_{1,j}y_j \\ \sum_{j=1}^{n} a_{2,j}y_j \\ \vdots \\ \sum_{j=1}^{n} a_{m,j}y_j \end{bmatrix}$$

$$\boldsymbol{x}^\top A = \begin{bmatrix} x_1 & x_2 & \cdots & x_m \end{bmatrix} \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{m} x_i a_{i,1} & \sum_{i=1}^{m} x_i a_{i,2} & \cdots & \sum_{i=1}^{m} x_i a_{i,n} \end{bmatrix}$$

Quick Detour: Linear Algebra Refresher

Consider a matrix $A = (a_{i,j}) \in \mathbb{R}^{m \times n}$. Then:

$$A\boldsymbol{y} = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix} \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{bmatrix} = \begin{bmatrix} a_{1,1}y_1 + a_{1,2}y_2 + \cdots + a_{1,n}y_n \\ a_{2,1}y_1 + a_{2,2}y_2 + \cdots + a_{2,n}y_n \\ \vdots \\ a_{m,1}y_1 + a_{m,2}y_2 + \cdots + a_{m,n}y_n \end{bmatrix} = \begin{bmatrix} \sum_{j=1}^{n} a_{1,j}y_j \\ \sum_{j=1}^{n} a_{2,j}y_j \\ \vdots \\ \sum_{j=1}^{n} a_{m,j}y_j \end{bmatrix}$$

$$\boldsymbol{x}^\top A = [x_1 \quad x_2 \quad \cdots \quad x_m] \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & a_{2,n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m,1} & a_{m,2} & \cdots & a_{m,n} \end{bmatrix} = \begin{bmatrix} \sum_{i=1}^{m} x_i a_{i,1} & \sum_{i=1}^{m} x_i a_{i,2} & \cdots & \sum_{i=1}^{m} x_i a_{i,n} \end{bmatrix}$$

$$\boldsymbol{x}^\top A\boldsymbol{y} = \sum_{i=1}^{m} \sum_{j=1}^{n} x_i a_{i,j} y_j$$

Back to 2-player Zero-Sum Games

Back to 2-player Zero-Sum Games

Let $A = (a_{i,j}) \in \mathbb{R}^{m \times n}$ be the payoff matrix of the game.

Back to 2-player Zero-Sum Games

Let $A = (a_{i,j}) \in \mathbb{R}^{m \times n}$ be the payoff matrix of the game.

Let x and y be the strategies of the maximiser and the minimiser respectively.

Back to 2-player Zero-Sum Games

Let $A = (a_{i,j}) \in \mathbb{R}^{m \times n}$ be the payoff matrix of the game.

Let x and y be the strategies of the maximiser and the minimiser respectively.

The (expected) utility is $u(x, y) = x^\top A y = \sum_{i=1}^{m_1} \sum_{j=1}^{m_2} (x_i \cdot a_{ij} \cdot y_j)$

Solution Concept #4:
Minimax (Optimal) Strategies

Solution Concept #4:
Minimax (Optimal) Strategies

Due to von Neumman (1928).

Solution Concept #4:
Minimax (Optimal) Strategies

Due to von Neumman (1928).

Choose the strategy that is the best possible against any choice of your opponent.

Solution Concept #4: Minimax (Optimal) Strategies

Due to von Neumman (1928).

Choose the strategy that is the best possible against any choice of your opponent.

In other words, assuming that the opponent is trying to make sure you get as little utility as possible, choose the strategy that maximises your utility.

Solution Concept #4:
Minimax (Optimal) Strategies

Due to von Neumman (1928).

Choose the strategy that is the best possible against any choice of your opponent.

In other words, assuming that the opponent is trying to make sure you get as little utility as possible, choose the strategy that maximises your utility.

Maximise your minimum possible utility (hence “minimax”).

Solution Concept #4:
Minimax (Optimal) Strategies

Due to von Neumman (1928).

Choose the strategy that is the best possible against any choice of your opponent.

In other words, assuming that the opponent is trying to make sure you get as little utility as possible, choose the strategy that maximises your utility.

Maximise your minimum possible utility (hence "minimax").

Why is this the rational thing to do in Zero-Sum games?

Back to 2-player Zero-Sum Games

Let $A = (a_{i,j}) \in \mathbb{R}^{m \times n}$ be the payoff matrix of the game.

Let x and y be the strategies of the maximiser and the minimiser respectively.

The (expected) utility is $u(x, y) = x^\top A y = \sum_{i=1}^{m_1} \sum_{j=1}^{m_2} (x_i \cdot a_{ij} \cdot y_j)$

Back to 2-player Zero-Sum Games

Let $A = (a_{i,j}) \in \mathbb{R}^{m \times n}$ be the payoff matrix of the game.

Let x and y be the strategies of the maximiser and the minimiser respectively.

The (expected) utility is $u(x, y) = x^\top A y = \sum_{i=1}^{m_1} \sum_{j=1}^{m_2} (x_i \cdot a_{ij} \cdot y_j)$

Maximiser chooses x^* to maximise $\min_{y \in \Delta(Y)} (x^*)^\top A y$,

i.e., $x^* \in \arg\max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$

Back to 2-player Zero-Sum Games

Let $A = (a_{i,j}) \in \mathbb{R}^{m \times n}$ be the payoff matrix of the game.

Let x and y be the strategies of the maximiser and the minimiser respectively.

The (expected) utility is $u(x, y) = x^\top A y = \sum_{i=1}^{m_1} \sum_{j=1}^{m_2} (x_i \cdot a_{ij} \cdot y_j)$

Maximiser chooses x^* to maximise $\min_{y \in \Delta(Y)} (x^*)^\top A y$,

i.e., $x^* \in \arg \max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$

Minimiser chooses y^* to minimise $\max_{x \in \Delta(X)} x^\top A y^*$

i.e., $y^* \in \arg \min_{y \in \Delta(Y)} \max_{x \in \Delta(X)} x^\top A y$

Back to 2-player Zero-Sum Games

Maximiser chooses x^* to maximise $\min_{y \in \Delta(Y)} (x^*)^\top A y$,

i.e., $x^* \in \arg\max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$

Back to 2-player Zero-Sum Games

Maximiser chooses x^* to maximise $\min_{y \in \Delta(Y)} (x^*)^\top A y$,

i.e., $x^* \in \arg\max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$

Minimiser chooses y^* to minimise $\max_{x \in \Delta(X)} x^\top A y^*$

i.e., $y^* \in \arg\min_{y \in \Delta(Y)} \max_{x \in \Delta(X)} x^\top A y$

Back to 2-player Zero-Sum Games

Maximiser chooses x^* to maximise $\min_{y \in \Delta(Y)} (x^*)^\top A y$,

i.e., $x^* \in \arg\max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$

Minimiser chooses y^* to minimise $\max_{x \in \Delta(X)} x^\top A y^*$

i.e., $y^* \in \arg\min_{y \in \Delta(Y)} \max_{x \in \Delta(X)} x^\top A y$

Let $v_x = \max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$ be

the *payoff* of the maximiser from the optimal strategy x^*.

Back to 2-player Zero-Sum Games

Maximiser chooses x^* to maximise $\min_{y \in \Delta(Y)} (x^*)^\top A y$,

i.e., $x^* \in \arg \max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$

Minimiser chooses y^* to minimise $\max_{x \in \Delta(X)} x^\top A y^*$

i.e., $y^* \in \arg \min_{y \in \Delta(Y)} \max_{x \in \Delta(X)} x^\top A y$

Let $v_x = \max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$ be

the *payoff* of the maximiser from the optimal strategy x^*.

Let $v_y = \min_{y \in \Delta(Y)} \max_{x \in \Delta(X)} x^\top A y$ be

the *minus the payoff* of the minimiser from the optimal strategy y^*.

Back to 2-player Zero-Sum Games

Let $v_x = \max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$ be

the *payoff* of the maximiser from the optimal strategy x^*.

Back to 2-player Zero-Sum Games

Let $v_x = \max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$ be

the *payoff* of the maximiser from the optimal strategy x^*.

Let $v_y = \min_{y \in \Delta(Y)} \max_{x \in \Delta(X)} x^\top A y$ be

the *minus the payoff* of the minimiser from the optimal strategy y^*.

Back to 2-player Zero-Sum Games

Let $v_x = \max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$ be

the *payoff* of the maximiser from the optimal strategy x^*.

Let $v_y = \min_{y \in \Delta(Y)} \max_{x \in \Delta(X)} x^\top A y$ be

the *minus the payoff* of the minimiser from the optimal strategy y^*.

Observation: $v_x \leq v_y$.

Back to 2-player Zero-Sum Games

Let $v_x = \max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$ be

the *payoff* of the maximiser from the optimal strategy x^*.

Let $v_y = \min_{y \in \Delta(Y)} \max_{x \in \Delta(X)} x^\top A y$ be

the *minus the payoff* of the minimiser from the optimal strategy y^*.

Observation: $v_x \leq v_y$.

Because:

$$\max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y = \min_{y \in \Delta(Y)} (x^*)^\top A y \leq (x^*)^\top A y^* \leq \max_{x \in \Delta(X)} x^\top A y^* = \min_{y \in \Delta(Y)} \max_{x \in \Delta(X)} x^\top A y$$

Back to 2-player Zero-Sum Games

Let $v_x = \max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$ be

the *payoff* of the maximiser from the optimal strategy x^*.

Let $v_y = \min_{y \in \Delta(Y)} \max_{x \in \Delta(X)} x^\top A y$ be

the *minus the payoff* of the minimiser from the optimal strategy y^*.

Observation: $v_x \leq v_y$.

Back to 2-player Zero-Sum Games

Let $v_x = \max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$ be

the *payoff* of the maximiser from the optimal strategy x^*.

Let $v_y = \min_{y \in \Delta(Y)} \max_{x \in \Delta(X)} x^\top A y$ be

the *minus the payoff* of the minimiser from the optimal strategy y^*.

Observation: $v_x \leq v_y$.

Back to 2-player Zero-Sum Games

Let $v_x = \max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y$ be

the *payoff* of the maximiser from the optimal strategy x^*.

Let $v_y = \min_{y \in \Delta(Y)} \max_{x \in \Delta(X)} x^\top A y$ be

the *minus the payoff* of the minimiser from the optimal strategy y^*.

Observation: $v_x \leq v_y$.

von Neumann's Minimax Theorem (1928, 1944): $v_x = v_y$

von Neumann’s Theorem

von Neumann's Theorem

$v_x = v_y = v$ (value of the game)

von Neumann's Theorem

$v_x = v_y = v$ (value of the game)

The theorem can be interpreted as follows:

von Neumann's Theorem

$v_x = v_y = v$ (value of the game)

The theorem can be interpreted as follows:

- For the maximiser, there is a mixed strategy, which, regardless of what the minimiser does, guarantees the maximiser a payoff of at least v.

von Neumann's Theorem

$v_x = v_y = v$ (value of the game)

The theorem can be interpreted as follows:

- For the maximiser, there is a mixed strategy, which, regardless of what the minimiser does, guarantees the maximiser a payoff of at least v.

- For the minimiser, there is a mixed strategy, which, regardless of what the maximiser does, guarantees the minimiser a "loss" or "cost" of at most v.

von Neumann's Theorem

$v_x = v_y = v$ (value of the game)

The theorem can be interpreted as follows:

- For the maximiser, there is a mixed strategy, which, regardless of what the minimiser does, guarantees the maximiser a payoff of at least v.

- For the minimiser, there is a mixed strategy, which, regardless of what the maximiser does, guarantees the minimiser a "loss" or "cost" of at most v.

- If the maximiser played a strategy that could only achieve a smaller payoff, or the minimiser played a strategy that incurred a higher loss, they could switch to the minimax/maximin (optimal) strategies.

von Neumann's Theorem

$v_x = v_y = v$ (value of the game)

The theorem can be interpreted as follows:

- For the maximiser, there is a mixed strategy, which, regardless of what the minimiser does, guarantees the maximiser a payoff of at least v.

- For the minimiser, there is a mixed strategy, which, regardless of what the maximiser does, guarantees the minimiser a "loss" or "cost" of at most v.

- If the maximiser played a strategy that could only achieve a smaller payoff, or the minimiser played a strategy that incurred a higher loss, they could switch to the minimax/maximin (optimal) strategies.

- So these strategies are the only reasonable/rational outcomes of the game.

But wait a second…

But wait a second…

Aren’t MNE reasonable outcomes of games?

But wait a second…

Aren’t MNE reasonable outcomes of games?

We in fact computed an MNE is RPS and it looks pretty reasonable!

Rock-Paper-Scissors

Consider the symmetric strategy (R, P, S) = (1/3, 1/3, 1/3) for both players. These are **optimal strategies.**

	Rock	Paper	Scissors	
Rock (R) $u_1(R, x_2) = 0$ →	0, 0	-1, 1	1, -1	**1/3**
Paper (P) $u_1(P, x_2) = 0$ →	1, -1	0, 0	-1, 1	**1/3**
Scissors (S) $u_1(S, x_2) = 0$ →	-1, 1	1, -1	0, 0	**1/3**
	1/3	**1/3**	**1/3**	

$$u_1(x_1, x_2) = \frac{1}{3} \cdot 0 + \frac{1}{3} \cdot 1 + \frac{1}{3} \cdot (-1) = 0$$

But wait a second…

Aren’t MNE reasonable outcomes of games?

We in fact computed an MNE is RPS and it looks pretty reasonable!

But wait a second…

Aren't MNE reasonable outcomes of games?

We in fact computed an MNE is RPS and it looks pretty reasonable!

Theorem: Let (x^*, y^*) be a pair of mixed strategies of a 2-player Zero-Sum game. Then x^* and y^* are both optimal strategies if and only if (x^*, y^*) is a MNE.

Proof of the Theorem

Proof of the Theorem

Assume that (x^*, y^*) is a pair of optimal strategies.

Proof of the Theorem

Assume that (x^*, y^*) is a pair of optimal strategies.

By the minimax theorem, we know that

$$\max_{x \in \Delta(X)} x^\top A y^* = \min_{y \in \Delta(Y)} (x^*)^\top A y = (x^*)^\top A y^*$$

Proof of the Theorem

Assume that (x^*, y^*) is a pair of optimal strategies.

By the minimax theorem, we know that

$$\max_{x \in \Delta(X)} x^\top A y^* = \min_{y \in \Delta(Y)} (x^*)^\top A y = (x^*)^\top A y^*$$

Consider a deviation of the maximiser to x'. From the above, it has to hold that $(x')^\top A y^* \leq (x^*)^\top A y^*$, i.e., the utility of the maximiser cannot increase.

Proof of the Theorem

Assume that (x^*, y^*) is a pair of optimal strategies.

By the minimax theorem, we know that

$$\max_{x \in \Delta(X)} x^\top A y^* = \min_{y \in \Delta(Y)} (x^*)^\top A y = (x^*)^\top A y^*$$

Consider a deviation of the maximiser to x'. From the above, it has to hold that $(x')^\top A y^* \leq (x^*)^\top A y^*$, i.e., the utility of the maximiser cannot increase.

The argument for the minimiser is similar.

Proof of the Theorem

Proof of the Theorem

Assume that (x^*, y^*) is a MNE.

Proof of the Theorem

Assume that (x^*, y^*) is a MNE.

By definition, that means that for the maximiser, we have

$$(x^*)^\top A y^* = \max_{x \in \Delta(X)} x^\top A y^* \geq \min_{y \in \Delta(Y)} \max_{x \in \Delta(x)} x^\top A y \quad (1)$$

Proof of the Theorem

Assume that (x^*, y^*) is a MNE.

By definition, that means that for the maximiser, we have

$$(x^*)^\top A y^* = \max_{x \in \Delta(X)} x^\top A y^* \geq \min_{y \in \Delta(Y)} \max_{x \in \Delta(x)} x^\top A y \quad (1)$$

and for the minimiser we have

Proof of the Theorem

Assume that (x^*, y^*) is a MNE.

By definition, that means that for the maximiser, we have

$$(x^*)^\top A y^* = \max_{x \in \Delta(X)} x^\top A y^* \geq \min_{y \in \Delta(Y)} \max_{x \in \Delta(x)} x^\top A y \quad (1)$$

and for the minimiser we have

$$(x^*)^\top A y^* = \min_{y \in \Delta(Y)} (x^*)^\top A y \leq \max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y \quad (2)$$

Proof of the Theorem

Assume that (x^*, y^*) is a MNE.

By definition, that means that for the maximiser, we have

$$(x^*)^\top A y^* = \max_{x \in \Delta(X)} x^\top A y^* \geq \min_{y \in \Delta(Y)} \max_{x \in \Delta(x)} x^\top A y \quad (1)$$

and for the minimiser we have

$$(x^*)^\top A y^* = \min_{y \in \Delta(Y)} (x^*)^\top A y \leq \max_{x \in \Delta(X)} \min_{y \in \Delta(Y)} x^\top A y \quad (2)$$

By the minimax theorem, we know that the RHS of both (1) and (2) are equal. This is only possible if the two inequalities are satisfied with equality ⇒ both strategies are optimal.

In 2-player Zero-Sum Games

Minimax Strategies = MNE

In 2-player Zero-Sum Games
Minimax Strategies = MNE

Theorem: Let (x^*, y^*) be a pair of mixed strategies of a 2-player Zero-Sum game. Then x^* and y^* are both optimal strategies if and only if (x^*, y^*) is a MNE.

In 2-player Zero-Sum Games
Minimax Strategies = MNE

Theorem: Let (x^*, y^*) be a pair of mixed strategies of a 2-player Zero-Sum game. Then x^* and y^* are both optimal strategies if and only if (x^*, y^*) is a MNE.

This provides a proof of the minimax theorem. How?

A few loose ends from today…

A few loose ends from today…

How do we solve those systems of linear equations to run the support enumeration algorithms for computing MNE?

A few loose ends from today…

How do we solve those systems of linear equations to run the support enumeration algorithms for computing MNE?

How do we prove von Neumann's minimax theorem?

A few loose ends from today…

How do we solve those systems of linear equations to run the support enumeration algorithms for computing MNE?

How do we prove von Neumann's minimax theorem?

- Yes, it follows from Nash, but we haven't proven that either.

A few loose ends from today…

How do we solve those systems of linear equations to run the support enumeration algorithms for computing MNE?

How do we prove von Neumann's minimax theorem?

- Yes, it follows from Nash, but we haven't proven that either.

- Actually, it was proven before Nash's Theorem, and has an easier proof.

A few loose ends from today…

How do we solve those systems of linear equations to run the support enumeration algorithms for computing MNE?

How do we prove von Neumann's minimax theorem?

- Yes, it follows from Nash, but we haven't proven that either.

- Actually, it was proven before Nash's Theorem, and has an easier proof.

How can we compute optimal strategies in 2-player Zero-Sum games? Algorithms?

A few loose ends from today…

How do we solve those systems of linear equations to run the support enumeration algorithms for computing MNE?

How do we prove von Neumann's minimax theorem?

- Yes, it follows from Nash, but we haven't proven that either.

- Actually, it was proven before Nash's Theorem, and has an easier proof.

How can we compute optimal strategies in 2-player Zero-Sum games? Algorithms?

Can we have efficient algorithms for computing MNE in (general sum) games, even for 2 players?

A few loose ends from today…

How do we solve those systems of linear equations to run the support enumeration algorithms for computing MNE?

How do we prove von Neumann’s minimax theorem?

- [illegible]

- [illegible] easier proof.

LINEAR PROGRAMMING

How can we compute optimal strategies in 2-player Zero-Sum games? Algorithms?

Can we have efficient algorithms for computing MNE in (general sum) games, even for 2 players?