Topic 2: Normal-form Games

Syllabus - Spring 2023

Maël van Beek (vanbeek.3@osu.edu)

Last updated: 2023-02-05

Prep work: Read Algorithmic Complexity, then watch P vs NP and the Computational Complexity Zoo. The following questions will be discussed in class:

- How does game theory relate to P and NP?
- What are the practical implications of algorithmic complexity for game theory?

1 Solving the Prisonner's Dilemma

Create a function, "`findPNE()`", that accepts 2 payoff matrices and returns:

1. The Nash Equilibrum (D,d)
2. The utility of player 1 and 2

1.1 Inputs: payoff matrices

```
pd1 <- matrix(c(1,5,
                0,3), nrow = 2, byrow = TRUE)
pd2 <- matrix(c(1,0,
                5,3), nrow = 2, byrow = TRUE)
rownames(pd1) <- c("D", "C")
colnames(pd1) <- c("d", "c")
rownames(pd2) <- c("D", "C")
colnames(pd2) <- c("d", "c")
```

Listing 1: Payoff matrices for player 1 and 2

Listing 1 produces two matrices, `pd1` and `pd2`, visible in table 1 which represent the payoffs of player 1 and 2 respectively.

	d	c
D	1	5
C	0	3

(a) Player 1

	d	c
D	1	0
C	5	3

(b) Player 2

Table 1: PD payoffs

1.2 Expected Output

The results should look something like this:

```
resultsPD <- findPNE(pd1, pd2)
```

Listing 2: Calling `findPNE()`

```
+++ Number of Pure N.E.: 1
+++ Equilibrium 1: (D, d) with utilities: (1, 1)
```

2 Solving the Stag Hunt

Now, use your function `findPNE()` on the following Stag Hunt payoff matrices:

```
sh1 <- matrix(c(1,1,
                0,5), nrow = 2, byrow = TRUE)
sh2 <- matrix(c(1,0,
                1,5), nrow = 2, byrow = TRUE)
rownames(sh1) <- c("D", "C")
colnames(sh1) <- c("d", "c")
rownames(sh2) <- c("D", "C")
colnames(sh2) <- c("d", "c")
```

Listing 3: Stag Hunt payoff matrices for player 1 and 2

Feel free to adjust your code as needed.

	d	c
D	1	1
C	0	5

(a) Player 1

	d	c
D	1	0
C	1	5

(b) Player 2

Table 2: Stag Hunt payoffs

```
resultsSH <- findPNE(sh1, sh2)
```

Listing 4: Calling `findPNE()` on the Stag Hunt matrices

```
+++ Number of Pure N.E.: 2
+++ Equilibrium 1: (D, d) with utilities: (1, 1)
+++ Equilibrium 2: (C, c) with utilities: (5, 5)
```

3 Exploring the Iterated Prisoner's Dilemma (I)

Different strategies exist for the Iterated Prisoner's Dilemma. Here are a few:

1. Always cooperate
2. Always defect
3. Cooperate and defect at random
    - If you wish, you may assume the random draw always causes defection for turn 1.

Building upon your `findPNE()` function to find all PNE for these strategies.

4 Exploring the Iterated Prisoner's Dilemma (II)

Now do the same for two more strategies:

1. Grim trigger
    - Start with C at t_1 then cooperate until partner defects
    - If partner defects, then always defect
2. Tit for tat
    - Start with C at t_1
    - Starting at t_2, play what partner played at t_{n-1}