Game Theory with Computer Science Applications Homework 1

Junhao Dai daijunhao@sjtu.edu.cn

April 9

Problem 1.

Let player 2 play his three strategies with probabilities p_L, p_M, p_R respectively. If player 1 mixes between U and M, then

$$\begin{aligned} U_1(U) &= p_L - 2p_M \\ U_1(M) &= -2p_L + p_M \\ \Rightarrow \quad U_1(U) &= U_1(M) \\ \Rightarrow \quad p_L &= p_M \end{aligned} \tag{1}$$

then $U_1(U) = U_1(M) < 0$, while $U_1(D) \geq 0$, and that's same for player 2, so player 2 can't mix L and M.

Suppose that player 2 does not play M with positive probability, then player 1 cannot play M with positive probability because $U_1(M) = -2p_L \leq 0$. But then if player 1 only mixes between U and D, then player 2 cannot play L because $U_2(L) = -2p_U < 0$. So player 2 will only play R to get a positive payoff, and player 1 cannot play U because $U_1(U) = 0$. Thus, player 1 will only play D to get a positive payoff.

Therefore, there is no mix strategy equilibrium, the game has a unique pure strategy equilibrium (D, R).

Problem 2.

By the definition of sup:

$$\sup_{x\in X}\left(\inf_{y\in Y} f(x,y)\right) - \epsilon < \inf_{y\in Y} f(x^*, y) \tag{2}$$

for some $x^* \in X$. Meanwhile we have:

$$\inf_{y\in Y} f(x^*, y) \leq \inf_{y\in Y}\left(\sup_{x\in X} f(x,y)\right) \tag{3}$$

because $f(x^*, y) \leq \sup_{x\in X} f(x,y)$ for every y. Combine the two above we have:

$$\sup_{x\in X}\left(\inf_{y\in Y} f(x,y)\right) - \epsilon < \inf_{y\in Y}\left(\sup_{x\in X} f(x,y)\right) \tag{4}$$

This holds for any $\epsilon > 0$. Therefore,

$$\sup_{x\in X}\left(\inf_{y\in Y} f(x,y)\right) \leq \inf_{y\in Y}\left(\sup_{x\in X} f(x,y)\right) \tag{5}$$

Problem 3.

By the zero-sum game, we define $A = \begin{pmatrix} 4 & 3 & 1 & 4 \\ 2 & 5 & 6 & 3 \\ 1 & 0 & 7 & 0 \end{pmatrix}$. Using the minimax throrem we can get the value of the game is :

$$\min_y \max_x x^T Ay = \max_x \min_y x^T Ay \tag{6}$$

To find the saddle points, we need to obtain the corresponding LPs :

$$\begin{aligned} \max_{x,v_1} \quad & v_1 \\ \text{s.t.} \quad & v_1 \leq \left(x^T A\right)_i \quad \forall i \\ & \sum_i x_i = 1 \\ & x \geq 0 \end{aligned} \tag{7}$$

.

and

$$\begin{aligned} \min_{y,v_2} \quad & v_2 \\ \text{s.t.} \quad & v_2 \geq (Ay)_j \quad \forall j \\ & \sum_j y_j = 1 \\ & y \geq 0 \end{aligned} \tag{8}$$

Using code , we get the saddle point is $x = \left(\frac{4}{7}, \frac{3}{7}, 0\right)^T$, $y = \left(\frac{5}{7}, 0, \frac{2}{7}, 0\right)^T$, and the value of the game is $\frac{22}{7}$.

Problem 4.

We use the same code as in the previous question, with parameter and constraint changes, and the obtained saddle point is $x = \left(\frac{52}{143}, \frac{50}{143}, \frac{41}{143}\right)^T$, $y = \left(\frac{44}{7}, \frac{42}{143}, \frac{57}{143}\right)^T$, and the value of the game is $\frac{96}{143}$.

Problem 5.

Assuming that player 1 plays strategies $a1, a2, a3, a4$ with probabilities m, n, p, q respectively, and player 2 plays strategies $b1, b2, b3, b4$ with probabilities x, y, z, t respectively.

We observe that for player 1, strategies a3 and a4 are strictly dominated by strategies a1 and a2. Therefore, player 1 will never choose strictly dominated strategies, so we can ignore a3 and a4. Similarly, for player 2, strategies b2 and b3 are strictly dominated by strategies b1 and b4. Therefore, we can ignore b2 and b3. Then we get:

$$\begin{array}{ccc} & b_1 & b_4 \\ a_1 & (-2,2) & (5,-6) \\ a_2 & (-4,0) & (4,-3) \end{array} \tag{9}$$

The updated matrix indicates that for player 2, strategy b4 is strictly dominated by b1, and for player 1, strategy a2 is strictly dominated by a1. Therefore, the final Nash Equilibrium is (-2,2).

Problem 6.

Since the definition of NE, we have

$$\begin{aligned} U_1\left(x^*, y^*\right) \geq U_1\left(x_i, y^*\right) \quad \forall x_i \in X \\ U_2\left(x^*, y^*\right) \geq U_2\left(x_i, y^*\right) \quad \forall x_i \in X \end{aligned} \tag{10}$$

and we know that $\hat{x} \in X$. We notice that regardless of the chosen strategies, we always have $U_1 + U_2 = 2$. Therefore, we can conclude:

$$\begin{aligned} 2-U_2\left(x^*, y^*\right) \geq 2-U_2\left(\hat{x}, y^*\right) \Rightarrow U_2\left(x^*, y^*\right) \leq U_2\left(\hat{x}, y^*\right) \\ 2-U_1\left(x^*, y^*\right) \geq 2-U_1\left(\hat{x}, y^*\right) \Rightarrow U_1\left(x^*, y^*\right) \leq U_1\left(\hat{x}, y^*\right) \end{aligned} \tag{11}$$

From the above equation, we can derive:

$$\begin{aligned} U_1\left(x^*, y^*\right) = U_1\left(\hat{x}, y^*\right) \\ U_2\left(x^*, y^*\right) = U_2\left(\hat{x}, y^*\right) \end{aligned} \tag{12}$$

Thus

$$\begin{aligned} U_1\left(\hat{x}, y^*\right) \geq U_1\left(x_i, y^*\right) \quad \forall x_i \in X \\ U_2\left(\hat{x}, y^*\right) \geq U_2\left(x_i, y^*\right) \quad \forall x_i \in X \end{aligned} \tag{13}$$

Therefore, $(\hat{x}, y^*)$ satisfy the definition of Nash equilibrium, thus proving that $(\hat{x}, y^*)$ are Nash equilibria. Similarly, the same can be proved for $(x^*, \hat{y})$.

Problem 7.

First we show that we can’t have both (1) and (2). Note that $y^T A x = y^T (Ax) = y^T b < 0$ since by (1), $Ax = b$ and by (2) $y^T b < 0$. But also $y^T A x = \left(y^T A\right) x = \left(A^T y\right)^T x \geq 0$ since by (2) $A^T y \geq 0$ and by (1) $x \geq 0$.

Now we must show that if (1) doesn’t hold, then (2) does. To do this, let $v_1, v_2, \ldots, v_n$ be the columns of A. Define

$$Q = cone\left(v_1, \ldots, v_n\right) \equiv \left\{ s \in \mathbb{R}^m : s = \sum_{i=1}^{n} \lambda_i v_i, \lambda_i \geq 0, \forall i \right\} \tag{14}$$

This is a conic combination of the columns of A, which differs from a convex combination since we don' t require that $\sum i = 1^n \lambda_i = 1$. Then $Ax = \sum_{i=1}^{n} x_i v_i$, there exists an x such that $Ax = b$ and $x \geq 0$ if and only if $b \in Q$.

So if (1) does not hold then $b \notin Q$. We show that condition (2) must hold. We know that Q is nonempty (since $0 \in Q$), closed, and convex, so we can apply the separating hyperplane theorem. The theorem implies that there exists $\alpha \in \mathbb{R}^m, \alpha \neq 0$, and β such that $\alpha^T b > \beta$ and $\alpha^T s < \beta$ for all $s \in Q$. Since $0 \in Q$, we know that $\beta > 0$. Note also that $\lambda v_i \in Q$ for all $\lambda > 0$. Then since $\alpha^T s < \beta$ for all $s \in Q$, we have $\alpha^T \left(\lambda v_i\right) \in Q$ for all $\lambda > 0$, so that $\alpha^T v_i < \beta / \lambda$ for all $\lambda > 0$. Since $\beta > 0$, as $\lambda \to \infty$, we have that $\alpha^T v_i \leq 0$. Thus by setting $y = -\alpha$, we obtain $y^T b < 0$ and $y^T v_i < 0$ for all i. Since the v_i are the columns of A, we get that $A^T y \geq 0$. Thus condition (2) holds.

Problem 8.

Proof: Since $C \in \mathbb{R}$ is a convex, closed and bounded set, $f : C \to C$, we assume

$$f : [a, b] \to [a, b] \tag{15}$$

Define $g\left(x\right) = x - f\left(x\right)$, then $g\left(a\right) = a - f\left(a\right) \leq 0$, and $g\left(b\right) = b - f\left(b\right) \geq 0$. So by the Intermediate Value Theorem there must be a point x_0 so that $g\left(x_0\right) = 0$. We then have $f\left(x_0\right) = x_0$.

Appendix

```python
from pulp import LpMaximize, LpMinimize, LpProblem, LpVariable, value
from fractions import Fraction

model = LpProblem(name="right", sense=LpMaximize)

variables = []
coefficients = [[4, 2, 1], [3, 5, 0], [1, 6, 7], [4, 3, 0]]
for i in range(3):
    variables.append(LpVariable(name=f"x{i+1}", lowBound=0))

v = LpVariable(name="v")

model += v

for i in range(4):
    model += (v <= sum(coefficients[i][j] * variables[j] for j in range
        (3)), f"constraint{i+1}")

model += sum(variables) == 1

```

```python
model.solve()

v_value = Fraction(value(v)).limit_denominator()
x_values = [Fraction(value(variables[i])).limit_denominator() for i in
    range(3)]

print(f"v={v_value}, x={x_values}")

model = LpProblem(name="right", sense=LpMinimize)

variables = []
coefficients = [[4, 3, 1, 4], [2, 5, 6, 3], [1, 0, 7, 0]]
for i in range(4):
    variables.append(LpVariable(name=f"y{i+1}", lowBound=0))

v = LpVariable(name="v")

model += v

for i in range(3):
    model += (v >= sum(coefficients[i][j] * variables[j] for j in range
        (4)), f"constraint{i+1}")

model += sum(variables) == 1

model.solve()

v_value = Fraction(value(v)).limit_denominator()
y_values = [Fraction(value(variables[i])).limit_denominator() for i in
    range(4)]

print(f"v={v_value}, y={y_values}")
```