Directions:

Write your solutions using and LaTeX. Then submit the files `hw8.tex`, `hw8.pdf`, `symmetric_games.ipynb`, and `AGGs.ipynb`.

Problem 1

Consider the following congestion game.

- Resources: $R = \{1, 2, 3\}$

- $A_i = \{(1, 2), (1, 3), (2, 3)\} \; \forall i \in P$

- Congestion Costs: $c_1(x) = x$ $c_2(x) = 2x - 2$ $c_3(x) = x^2$

(a) Fill in the symmetric data structure for this game when $|P| = 4$.

configs:

3	2	2	1	1	1	0	0	0	0
0	1	0	2	1	0	3	2	1	0
0	0	1	0	1	2	0	1	2	3

(b) Identify a pure-strategy Nash equilibrium in this game when $|P| = 4$ and when $|P| = 16$.

(a) payoffs:

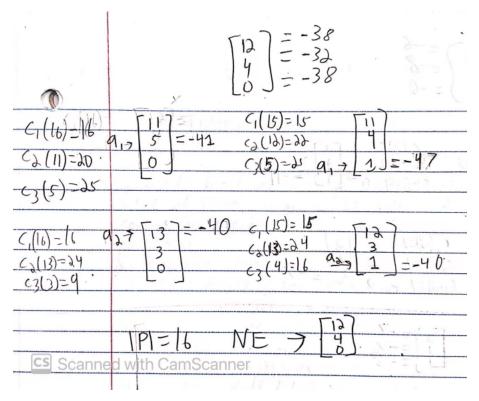| -10	-8	-9	-8	-7	-8	-4	-5	-6	-7
-5	-8	-7	-13	-12	-11	-20	-19	-18	-17
-7	-8	-10	-11	-13	-15	-16	-18	-20	-22

(b) The Nash Equilibrium when $|P| = 4$ is [2, 1, 1]. The Nash Equilibrium when $|P| = 16$ is [12, 4, 0].

Work for proving [2,1,1] is Nash of 4 player game:

$C_1(3)=3$
$C_2(3)=4$
$C_3(2)=4$

$\begin{bmatrix} ? \end{bmatrix} = \begin{matrix} -7 \\ -7 \\ -8 \end{matrix}$

$C_1(3)=3$
$C_2(2)=2$
$C_3(3)=9$

$a_1 \rightarrow \begin{bmatrix} 1 \\ 2 \\ 1 \end{bmatrix} = -12$ (x)

$C_1(2)=2$
$C_2(3)=4$
$C_3(3)=9 \rightarrow_{a_1} \begin{bmatrix} 1 \\ 1 \\ 2 \end{bmatrix} = -13$ (x)

$C_1(3)=3$
$C_2(4)=6$
$C_3(1)=1$

$a_3 \rightarrow \begin{bmatrix} 3 \\ 0 \\ 1 \end{bmatrix} = -9$ (x)

$C_1(2)=2$
$C_2(4)=6$
$C_3(2)=4 \rightarrow_{a_2} \begin{bmatrix} 2 \\ 0 \\ 2 \end{bmatrix} = -10$

$C_1(4)=4$
$C_2(3)=4$
$C_3(1)=1$

$a_3 \rightarrow \begin{bmatrix} 3 \\ 1 \\ 0 \end{bmatrix} = -8$

$C_1(4)=4$
$C_2(2)=2$
$C_3(2)=4 \rightarrow_{a_3} \begin{bmatrix} 2 \\ 2 \\ 0 \end{bmatrix} = -8$

Let a_1 or a_2 or a_3 denote the utility by playing that action in the original mixture - this is the number which we are comparing the deviations to. As can be seen by this work, there is no action change for any player that will cause a positive gain by deviating. There is one player who will have the same utility, however Nash Equilibrium is always greater-than-or-equal-to, since we expect players to sometimes be indifferent. Therefore, this is a Nash Equilibrium.

Work for proving [12,4,0] is Nash of 16 player game:

2

$$\begin{bmatrix} 12 \\ 4 \\ 0 \end{bmatrix} \begin{matrix} = -38 \\ = -32 \\ = -38 \end{matrix}$$

$c_1(16)=16$ $a_{1,2}$ $\begin{bmatrix} 11 \\ 5 \\ 0 \end{bmatrix} = -41$ $c_1(15)=15$
$c_2(11)=20$ $c_2(12)=22$
$c_3(5)=25$ $c_3(5)=25$ $a_{1,7}$ $\begin{bmatrix} 11 \\ 4 \\ 1 \end{bmatrix} = -47$

$c_1(16)=16$ $a_{2,7}$ $\begin{bmatrix} 13 \\ 3 \\ 0 \end{bmatrix} = -40$ $c_1(15)=15$
$c_2(13)=24$ $c_2(13)=24$
$c_3(3)=9$ $c_3(4)=16$ $a_{2,9}$ $\begin{bmatrix} 12 \\ 3 \\ 1 \end{bmatrix} = -40$

$|P|=16$ NE → $\begin{bmatrix} 12 \\ 4 \\ 0 \end{bmatrix}$

Let a_1 or a_2 denote the utility by playing that action in the original mixture - this is the number which we are comparing the deviations to. As can be seen by this work, there is no action change for any player that will cause a positive gain by deviating. Therefore, this is a Nash Equilibrium.

Problem 2

In the Jupyter notebook symmetric_games.ipynb, implement the functions to compute deviation payoffs and deviation gains for a symmetric mixture in a symmetric game. Then adapt your Nash local search code to find symmetric mixed-strategy Nash equilibria.

Problem 3

In the Jupyter notebook AGGs.ipynb, implement the functions to compute deviation payoffs and deviation gains for a role-symmetric mixture in a symmetric game. Then adapt your Nash local search code to find symmetric mixed-strategy Nash equilibria.

Note that the ActionGraphGame class has several updates relative to the one from class. In particular, you'll find the new attributes self.role_config_indices and self.neighbor_masks helpful for operating over an action's neighborhood. The former provides a list of which indices in the configuration belong to each role; the latter provides a mask for which entries in a profile give probabilities for actions in the neighborhood. Some examples of how to use them have been provided.

3

Also note that there are some extra functions to implement, but they should both be easy: `deviation_payoffs` and `deviation_gains`. Finally, the interface to the helper functions `config_prob` and `repetitions` has changed. I think this one will be easier to use now that you have the above-attributes for help with indexing, but if you prefer the old interface, you may revert to it.