RAPPORT DU TP

THEORIE DES JEUX

BANDOUI NAZIM
161631055599

Dans ce rapport, il ‘s’agit de la réalisation des TP du module de Théorie des jeux avec des captures d’écrans sur la solution développée sous Python 3 uniquement.

Bibliothèques utilisées :

NumPy :

NumPy est une bibliothèque utilise pour la gestion des vecteur multidimensionnels contentant plusieurs fonction de base pour cette gestion.

Dash :

Dash est une bibliothèque développée par Plotly destinée à la création d’applications web pour la data science et le machine Learning, elle a été utilisé pour le développement front end dans ce TP et aussi dans la création et la manipulation des spreadsheets.

Afin de pouvoir faire marcher ce TP sur votre machine, veuillez ouvrir votre invite de commande (ou Terminal sous Linux), et de coller les commandes suivantes (remplacez «pip » par « pip3 » sous Linux) :

pip install numpy

pip install dash==0.21.1

pip install dash-renderer==0.13.0

pip install dash-html-components==0.11.0

pip install dash-core-components==0.23.0

pip install plotly –upgrade

pip install dash-bootstrap-components

Représentation utilisée :

En théorie des jeux, un jeu avec 2 joueurs avec N stratégies pour le joueur 1 et M stratégies pour le joueur 2 est souvent représenter par une matrice NxM ou les stratégies du joueur 1 représentent les lignes de la matrice et les stratégies du joueur 2 représentent les colonnes , voici un exemple du dilemme du prisonnier avec 2 joueurs ou chaque joueur possède 2 stratégies , et donc une matrice 2x2 :

B A	B se tait	B parle
A se tait	-1 -1	0 -3
A parle	-3 0	-2 -2

Pour 3 joueurs, nous utiliserons un tableau de 3 dimensions, cela est parfaitement réalisable mais risque de causer une confusion lors de la lecture de ce tableau par une personne.

A partir de n joueurs, les tableaux de n-dimensions ne seront plus utilisables et par conséquent, le résultat non-visible, ce qui nous a poussés à réfléchir à une nouvelle méthode pour représenter un jeu a n-joueurs.

La méthode trouvée est inspirée du code binaire, elle consiste à créer une matrice ou les lignes représentent toutes les combinaisons possibles entre les stratégies des N joueurs.

Pour exemple prenons le dilemme du prisonnier représenté par la figure ci-dessus, notre représentation donnera :

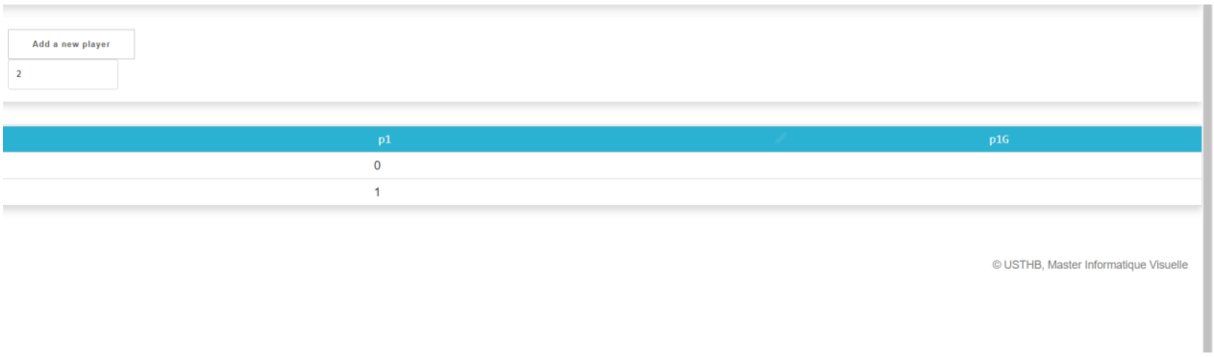
Stratégie A	Stratégie B	Gain A	Gain B
0	0	-1	-1
0	1	-3	0
1	0	0	-3
1	1	-2	-2

Ou 0 est la stratégie «Se taire » et 1 « Parler ».

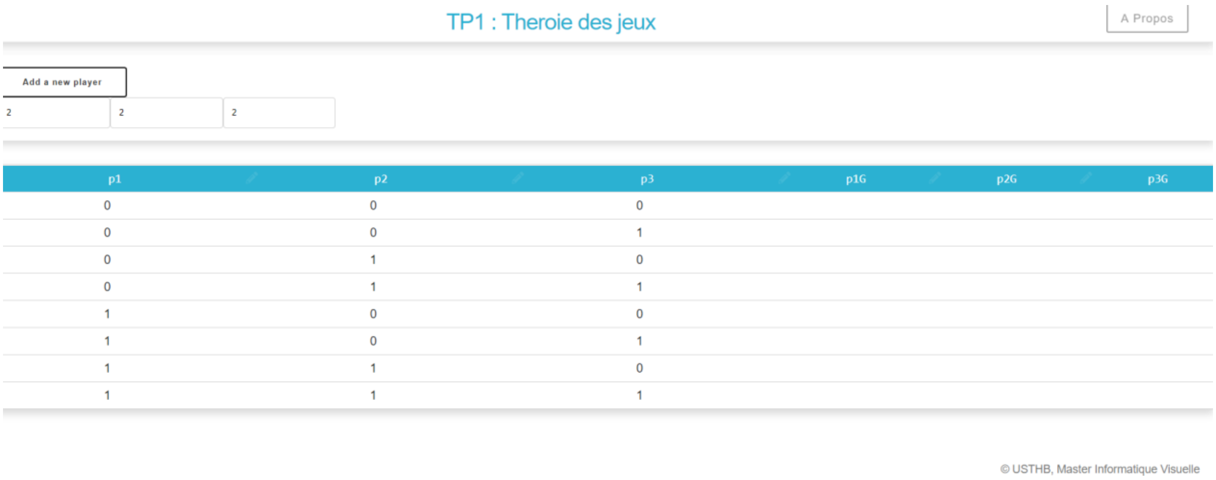
Présentation de l’interface :

L’interface développée dispose d’un spreadsheet qui sers d’input avec tous les combinaisons possibles prédéfinie et l’utilisateur n’as qua remplir les gains là où il faut.

L’interface aussi dispose d’un bouton « Ajouter un nouveau joueur » qui permet comme son nom l’indique d’ajouter un nouveau joueur, et de modifier le nombre de stratégies, le tout sera mis à jour automatiquement dans le spreadsheet comme le montre les captures ci –dessous :



p1	p1G
0	
1	



p1	p2	p3	p1G	p2G	p3G
0	0	0			
0	0	1			
0	1	0			
0	1	1			
1	0	0			
1	0	1			
1	1	0			
1	1	1			

p1	p2	p3	p1G	p2G	p3G
0	0	0			
0	0	1			
0	0	2			
0	1	0			
0	1	1			
0	1	2			
0	2	0			
0	2	1			
0	2	2			
0	3	0			
0	3	1			
0	3	2			
1	0	0			

TP 1

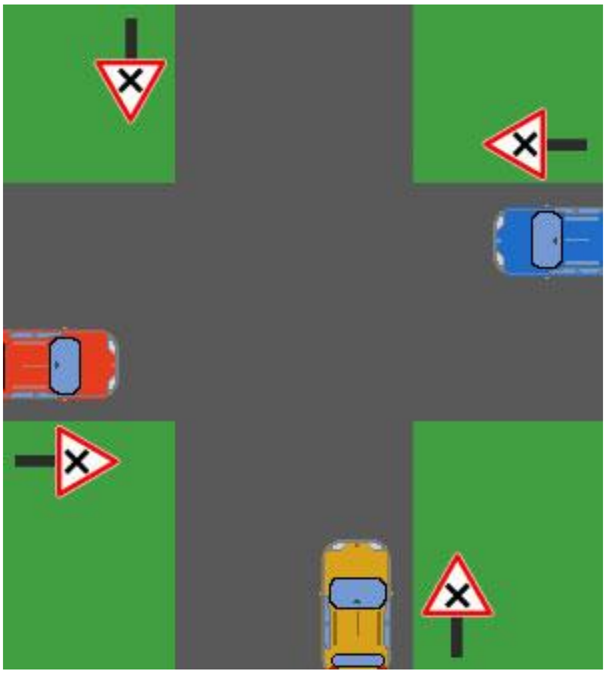
Dans ce TP nous allons commencer par développer toutes les notions vues en cours en stratégies pures dans un jeu à 2 joueurs et plus, à savoir la détermination d'une stratégie strictement dominante, stratégie faiblement dominante, équilibres par élimination successives des stratégies fortement dominantes et faiblement dominantes, équilibres de Nash et profils Pareto dominants et Niveau de sécurité d'une stratégie ainsi que d'un joueur.

L'objectif est de développer des algorithmes efficaces avec une complexité algorithmique moindre, avec une interface simple et interactive pour l'utilisateur où il y'aura un guide expliquant les procédures.

Il sera aussi demander de rajouter ***des exemples célèbres de jeu avec des interfaces explicites ou il sera question de tester les notions en supra***.

Dans la suite de ce TP, nous allons suivre l'exemple suivant :

3 voitures sont arrivées en même temps dans une intersection T, chacune d'une route différente comme le montre la figure ci-dessous (La priorité à droite n'est pas prise en compte) :



- Si une voiture passe, son gain est de 0
- Si une voiture s’arrête, son gain est de -1
- Si des voitures se percutent, leur gain est de -5
- Les voitures bleu et rouge peuvent passer en même temps sans se percuter
- Si une voiture passe avec la voiture jaune, elle se percute à elle

Voici la matrice qui représente le jeu :

p1	p2	p3	p1G	p2G	p3G
0	0	0	-1	-1	-1
0	0	1	-1	-1	0
0	1	0	-1	0	-1
0	1	1	-1	-5	-5
1	0	0	0	-1	-1
1	0	1	0	-1	0
1	1	0	-5	-5	-1
1	1	1	-5	-5	-5

Ainsi que les calculs demandés :

Results :

[0 1 0] [1 0 1]
Nash Equilibrium :

[0 1 0] [1 0 1]
Pareto's Optimum :

-1 -1 -1
Security level:

Les équilibres de Nash sont les issues :

- 1 0 1
- 0 1 0

Les optimums de Pareto sont les issues :

- 1 0 1
- 0 1 0

Le niveau de sécurité est :

- Joueur 1 : -1
- Joueur 2 : -1
- Joueur 3 : -1

Dans cet exemple, il n’y a aucune stratégie fortement ou faiblement dominante.

Voici un exemple trivial ou il existe des stratégies dominantes :

p1	p2	p3	p1G	p2G	p3G
0	0	0	1	2	1
0	0	1	2	3	-1
0	1	0	3	4	2
0	1	1	4	6	-2
1	0	0	5	7	3
1	0	1	6	5	4
1	1	0	7	4	-6
1	1	1	8	87	7

Results :

[1 1 1]
Nash Equilibrium :

[1 1 1]
Pareto's Optimum :

5 4 -2
Security level:

Player 0 Strategy 1 strictly dominates Player 0 Strategy 0

Player 2 Strategy 1 strictly dominates Player 2 Strategy 0

p1	p2	p1G	p2G
0	0	1	2
0	1	2	3
0	2	0	3
1	0	2	2
1	1	2	1
1	2	3	2
2	0	2	1
2	1	0	0
2	2	1	0

Player 0 Strategy 1 weakly dominates Player 0 Strategy 2

Player 1 Strategy 2 weakly dominates Player 1 Strategy 1

TP 2

TP2 : calcul de l'équilibre de Nash en mixtes + la valeur dans un jeu à somme nulle

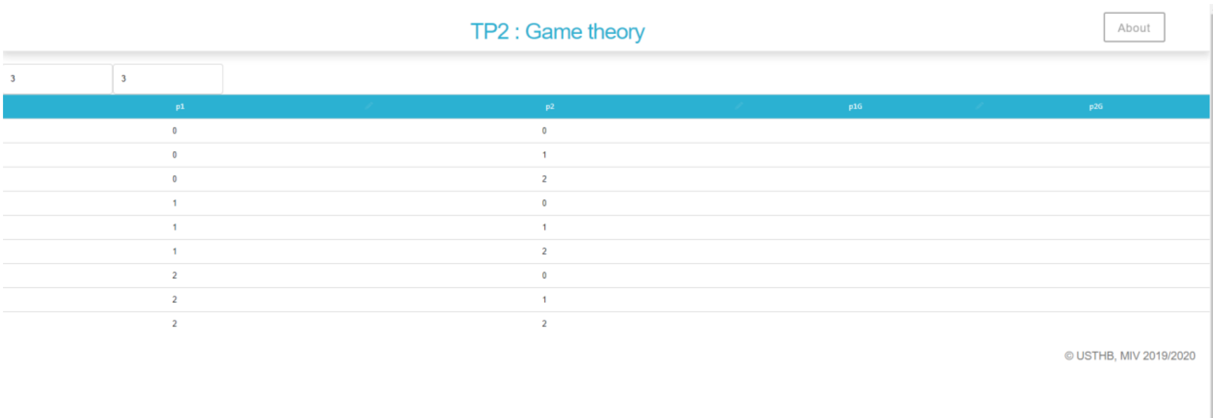
Résumé

Dans ce TP nous allons développer une petite application qui permet de calculer l'équilibre de Nash mixte dans un jeu à deux joueurs avec au maximum 3 stratégies chacun, cela permettra à l'ensemble des étudiants de comprendre le principe de l'algorithme de calcul.

Il sera aussi demandé de déterminer la valeur dans un jeu à somme nulle si elle existe en stratégies pures et aussi en mixtes dans le cas mixtes : 3 stratégies maximum pour chacun des joueurs.

Dans ce TP2, nous allons utiliser la même interface graphique que le TP1 qui consiste à définir le nombre de stratégies pour les deux joueurs et le remplissage automatique du spreadsheet fournie.

Vous devez remplir les gains de chaque joueur dans chaque stratégie et les résultats s'afficheront automatiquement :



p1	p2	p1G	p2G
0	0		
0	1		
0	2		
1	0		
1	1		
1	2		
2	0		
2	1		
2	2		

© USTHB, MIV 2019/2020

Pour le calcul de l'équilibre de Nash en stratégie mixte, l'algorithme d'énumération de supports a été utilisé, traitent les cas de matrices de gain de 2x2 ou de 3x3.

L'exemple ci-dessous est un Pierre-Feuille-Ciseaux ou :

- La stratégie 0 : le joueur joue Pierre
- La stratégie 1 : le joueur joue Feuille
- La stratégie 2 : le joueur joue Ciseaux

La matrice des gains est représentée ci-dessous :

3 3

p1	p2	p1G	p2G
0	0	0	0
0	1	-1	1
0	2	1	-1
1	0	1	-1
1	1	0	0
1	2	-1	1
2	0	-1	1
2	1	1	-1
2	2	0	0

Results :

L'équilibre de Nash selon l'algorithme d'énumération de supports est :

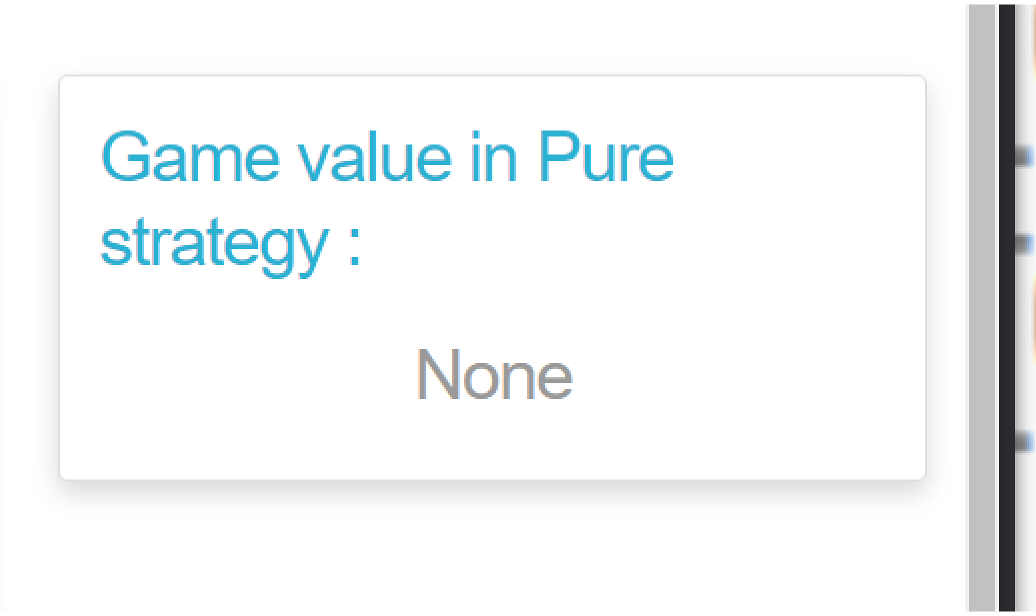
Results :

Nash equilibrium in mixed strategy (Support Enumeration) :

[0.3330.3330.333]
[0.3330.3330.333]

Le Pierre-Feuille-Ciseaux est un jeu à somme nulle, nous allons nous intéresser aux valeurs de ce jeu désormais

Pour le même exemple en stratégie pure :



Dans le cas du Pierre-Papier-Ciseaux, le j’eu n’admet pas de valeurs en stratégies pure.