Coopération de Q-Agents dans le dilemme du prisonnier

Lachance A., Toulouse F.

Résumé

Le dilemme du prisonnier est un problème intéressant dans le cas de l'apprentissage machine. C'est un problème dont la solution optimale n'est pas évidente du point de vue du joueur. Dans nos expériences, nous tentons d'apprendre à des Q-agents à coopérer dans le dilemme du prisonnier itératif. Pour cela nous nous sommes inspirés des expériences de Sandholm et Crites, qui ont entraîné des RNNs sur le même problème. Or, dans notre cas, nous utilisons des LSTMs. Les résultats avec un LSTM de base sont similaire aux résultats obtenus par Sandholm et Crites. Pour inciter la coopération, nous avons donc essayé trois méthodes de sélection des agents: La sélection "Best-Worst", qui consiste à remplacer les agents avec le moins de points par ceux ayant le plus de points, La sélection par famine, qui consiste à établir une limite sous laquelle les agents meurent et sont ramplacés par des copies des agents au-dessus de la limite et, finalement, la sélection sociale, qui consiste à établir un niveau de confiance au cours de la partie avec les autres agents et chaque partie est suivie d'un vote envers l'agent en lequel on fait le moins confiance, celui recevant le plus de votes est remplacé par l'agent en qui les autres ont le plus confiance, qui est déterminé par le même processus. En terme de résultat, la sélection "Best-Worst" et par famine éliminent les agents toujours coopératifs en premier, ce qui piège les autres agents dans le cas trahison-trahison du dilemme du prisonnier itératif. Par contre, la sélection sociale donne toujours des agents coopératifs si on utilise un taux d'apprentissage assez bas.

Introduction

Le dilemme du prisonnier itératif (DPI) est un cas intéressant dans le domaine de l'apprentissage machine par renforcement: non seulement permet-il à plusieurs agents de s'affronter entre eux, mais la solution optimale d'ensemble au problème, c'est à dire la façon de faire pour maximiser le pointage total, est difficile à atteindre du point de vue d'un joueur.

Se basant sur le DPI et utilisant des agents RNNs, Sandholm et Crites ont réalisés quelques expériences d'apprentissage. Ils ont d'abord entraîné des agents à jouer contre tit-for-tat. Pour cela, ils les ont fait jouer 100 parties de 100 000 itérations chacunes. Pour une valeur de gamma de 0.05 à 0.2, les agents ont appris à toujours trahir. Pour un gamma de 0.25 à 0.65, les agents ont appris une certaine séquence de coopération-trahison. Pour un gamma de 0.7 à 0.95, les agents ont appris à coopérer avec tit-for-tat.

Ils ont ensuite entraîné les agents à jouer les uns contre les autres. Ils testé 2 types d'agents: un utilisant un lookup table et l'autre utilisant un RNN. Sur ces 2 types d'agents, 3 tests ont été effectués. Le premier test avait pour paramètre un taux d'apprentissage de 0.2 avec un taux d'exploration de 0.999. Le deuxième test avait un taux d'apprentissage de 1 et un taux d'exploration de 0.999. et le 3e test avait un taux d'apprentissage de 0.2 et un taux d'exploration de 0.9999. En résumé, ces tests ont rapporté que les RNN tombaient plus facilement dans le piège de la trahison mutuelle, qu'un taux d'apprentissage élevé encourage la trahison mutuelle et qu'un taux d'exploration élevé encourage les autres stratégies que la trahison mutuelle.

Théorie

L'approche Q-agent est une méthode d'apprentissage machine par renforcement. L'apprentissage machine par renforcement consiste à apprendre à un agent, qui contient un réseau de neurone, à prendre des décisions dans un contexte donné. Pour s'y faire, chaque choix que l'agent prend cause un effet auquel est rattaché une récompense. La logique de prise de décision d'un agent est ce que l'on appelle "politique". Les récompenses sont distribuées de sorte que la politique soit optimisée, de sorte qu'elle donne l'effet désiré.

L'apprentissage de Q-agent est dirigée par la formule suivante:

$$Q_{i+1}(s,a) = Q_i(s,a) + \alpha\left[\underbrace{\left(r(s') + \gamma \max_{a'} Q_i(s',a')\right)}_{\text{critic}} - \underbrace{Q_i(s,a)}_{\text{actor}}\right]$$

À chaque itération, le Q-agent apprend selon les récompenses attribuées et les actions qui maximisent cette récompense, multiplié par le facteur de vitesse d'apprentissage (α).

Dans notre étude, les récompenses attribuées correspondent aux points donnés par le dilemme du prisonnier, additionné d'un offset (O). Le dilemme du prisonnier itératif est un jeu à 2 joueurs. À chaque itération, chaque joueur a le choix de coopérer (C) ou "defect" (D), aussi nommé "trahir". Le tableau suivant décrit comment les pointages sont distribués selon le choix de chaque joueur:

		Joueur 2	
		C	D
Joueur 1	C	R=3 + O	S=0 + O
	D	T=5 + O	P=1 + O

Tableau 1 - Pointage attribué au joueur 1 dans le dilemme du prisonnier et leurs symbole correspondant

Il est à noter que les pointages sont symétriques pour le joueur 2.

Le dilemme du prisonnier itératif est dirigé par 2 équations sur ces pointages:
1)T> R>P>S
2) 2R > T+S > 2P

Ces deux équations montrent que, pour un joueur, il est plus avantageux de trahir alors que l'action qui génère le plus de points au total est la coopération. On peut aussi le voir ainsi: si les 2 joueurs coopèrent, chacun sait qu'il aurait pu gagner plus de points en trahissant l'autre. Si l'un coopère et l'autre trahit, celui qui coopère sait qu'il aurait gagné plus de points en trahissant. Si les deux se trahissent mutuellement, chacun sait qu'il a pris la décision optimale dans la situation, les deux joueurs sont donc satisfaits de leur choix. Cette situation se nomme l'équilibre de Nash et c'est la situation, selon la théorie du jeu, vers laquelle les actions de joueurs vont tendre.

Expériences

Pour nos expériences, l'environnement est modélisé comme un tournoi du dilemme du prisonnier itératif: deux agents de l'ensemble (N) interagissent entre eux pendant (I) itérations, en recevant en entrée soit un état initial pour l'interaction et une mémoire vierge, ou l'état précédent le la mémoire sauvegardée par l'agent et son réseau. Le total de toutes les interactions avec tous les agents du groupe détermine le succès de l'agent et sa stratégie pendant le tournoi.
Un environnement secondaire modélisé correspond à un DPI contre la stratégie fixe tit-for-tat afin de tester la capacité de nos configurations d'agents à apprendre la coopération.

Un troisième environnement de validation permet de déterminer la stratégie spécifique développée par les agents. Chaque agent interagit avec deux stratégies fixes: toujours coopérer ou toujours trahir. Cette validation permet de distinguer les agents qui développent la stratégie de coopération inconditionnelle, qui coopèrent malgré la trahison, de ceux qui apprennent une coopération conditionnelle, qui cessent de coopérer après être trahis tels Tit-for-tat ou Grim.
Pour chacun de ces environnements, les agents vont jouer un certain nombre de parties de DPI (E) chacune avec un certain nombre (I) d'itérations légèrement aléatoire.
Nous avons donc modélisé des agents avec une couche LSTM pour permettre un apprentissage indépendant de la mémoire lors d'une interaction dans l'environnement d'un tournoi DPI. Nos premières expériences en tournoi ont révélés que la coopération entre individus dépends beaucoup de l'exploration initiale, et devient virtuellement impossible avec plus de deux agents, confirmant les résultats des expériences de Sandholm et Crites sur la relation entre le rythme d'apprentissage, la quantité d'exploration et la coopération multi-agents du DPI.

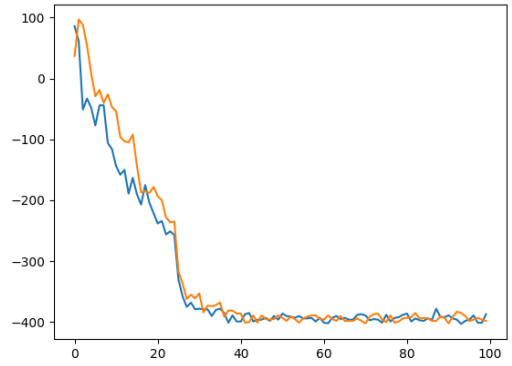


Fig. 1 - Agents LSTM traîtres

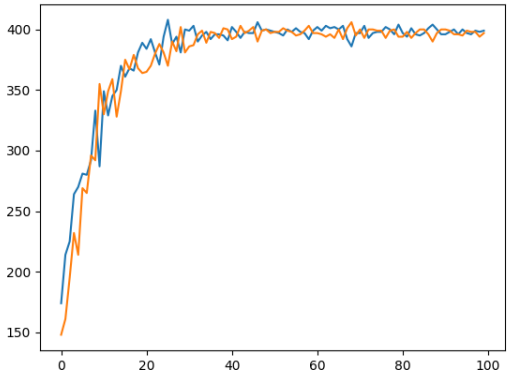


Fig. 2 - Agents LSTM coopérateurs

Pour valider le modèle du LSTM, nous avons fait jouer un échantillon d'agents dans l'environnement secondaire, contre la stratégie tit-for-tat. Ces expériences sont réalisées avec des échantillons de 10 agents à configurations variable afin de déterminer quelle impact le nombre de couches après le LSTM (L) et la taille de la couche cachée (H) ont sur la capacité d'apprendre la coopération. Les résultats suivants montre les différentes combinaisons avec H=16, 32 ou 64 et L=1 ou 2.

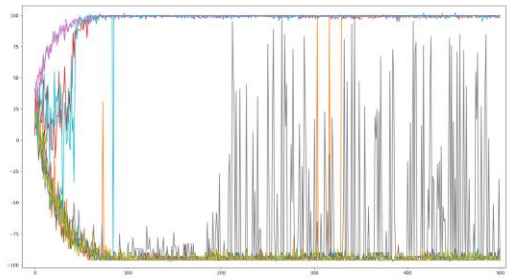


Fig. 3 - Pointage des agents contre TFT, H=16, L=1, 5 Coopérateurs

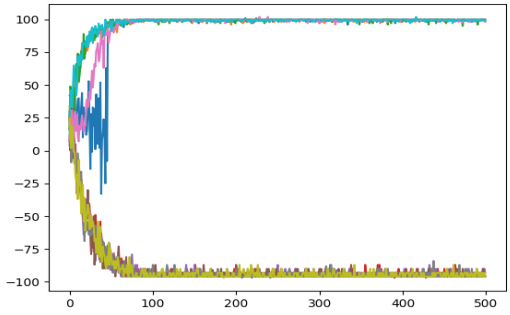


Fig. 4 - Pointage des agents contre TFT, H=16, L=2, 5 Coopérateurs

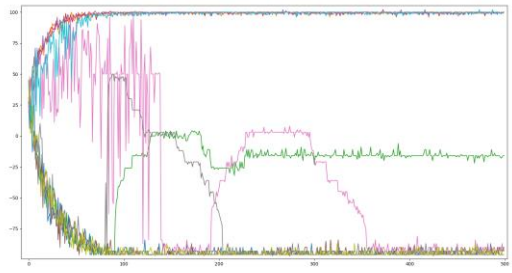


Fig. 5 - Pointage des agents contre TFT, H=32, L=1, 4 Coopérateurs

Fig. 6 - Pointage des agents contre TFT, H=32, L=2, 4 Coopérateurs

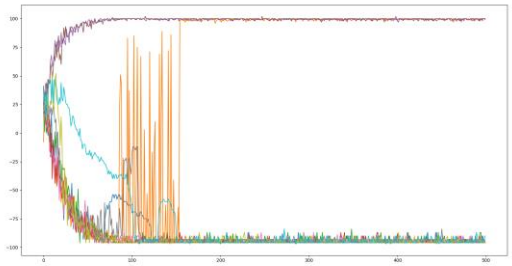


Fig. 7 - Pointage des agents contre TFT, H=64, L=1, 3 Coopérateurs

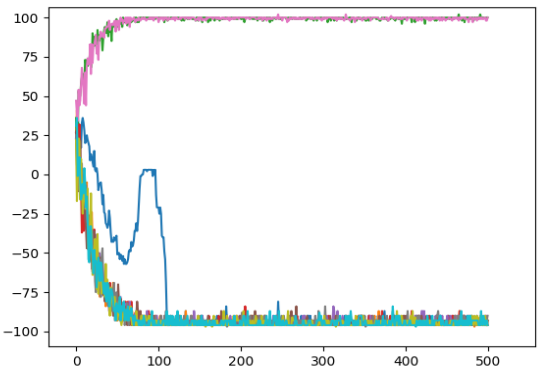


Fig. 8 - Pointage des agents contre TFT, H=64, L=2, 2 Coopérateurs

Selon ces expériences, nous avons pu déterminer qu'une plus petite couche cachée améliore la capacité des agents de coopérer. Les paramètres pour les réseaux LSTM lors des prochaines expériences resteront fixés à H=16 et L=1 pour faciliter et accélérer l'apprentissage.

Nous avons par la suite tenté de comparer avec un modèle RNN plus classique, qui conserve en mémoire les deux dernières entrées pour concaténer avec l'état présent. Ce modèle est significativement plus lent à entraîner, demandant un taux d'apprentissage réduit et un temps d'exploration augmenté (taux de diminution a = 0.999) afin de donner un apprentissage de coopération significatif.

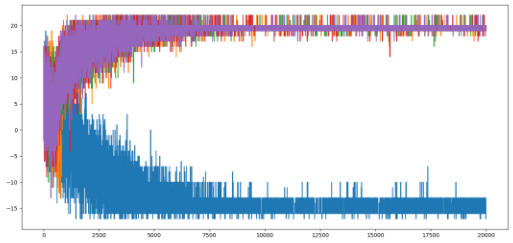


Fig. 9 - Pointage des agents RNNs contre TFT, 20 000 parties, exploration augmentée

Pour pouvoir entraîner plusieurs agents en même temps, nous avons créé un système de compétition où chaque agent confronte chacun des autres agents présents. Cela été fait de façon similaire dans les expériences de Sandholm et Crites, mais donnait des résultats aléatoires.
Pour tenter d'atteindre de façon constante le pointage total maximum du DPI, nous avons pensé à 3 expériences de sélection d'agent. C'est à dire qu'à chaque époque d'apprentissage, un certain nombre d'agent sont tués selon un certain critère de sélection et remplacé par le "meilleur" agent, aussi choisi selon un certain critère. Le premier critère est inspiré de la théorie de l'évolution: Une certains nombre D (dead) d'agents, ceux qui ont le moins de points, sont remplacés par des copies du meilleur agent, celui qui a eu le plus de points. On a nommé ce type de sélection "Best-Worst".
Nous avons donc entraîné une dizaine d'agents avec sélection Best-Worst avec N= 10, E = 200 et I = 100. On remarque que le pointage diminue presque immédiatement après la première partie du DPI d'entraînement (voir fig. 10).

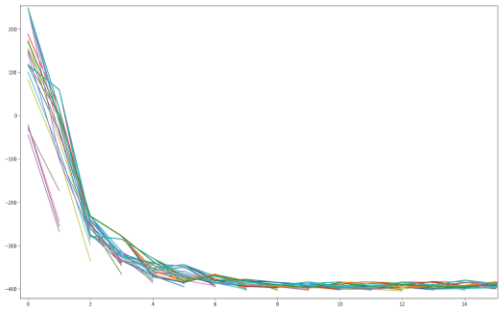


Fig. 10 - Pointage des agents à chaque partie avec sélection Best-Worst

La sélection a donc été configurée pour qu'elle ne commence qu'après un certain nombre de parties (δ) de DPI, ici établi à 25, afin de permettre un certain degré d'exploration initiale. Ainsi on pourrait mieux voir la cause de la baisse immédiate de points. Vous pouvez voir les résultats de cette expérience à la figure 11.

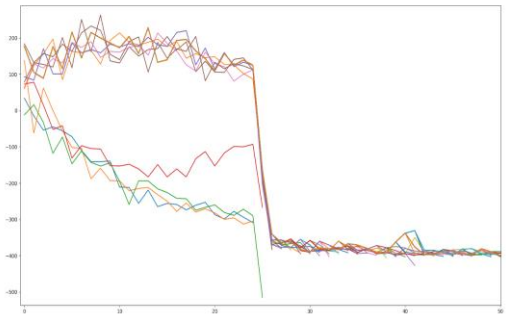


Fig. 11 - Pointage des agents à chaque parties avec sélection Best-Worst avec délai d=25

Comme on peut voir, certains agents baissent normalement en points alors que d'autres grimpent. C'est ce qu'on observait dans nos expériences sans sélection. Alors, quand la sélection arrive, les agents qui ont un pointage bas meurent et sont remplacés par ceux ayant un pointage élevé. Malheureusement, les agents ayant un pointage élevé n'ont ce pointage que parce qu'ils profitaient des agents qui n'avaient pas de points, ceux qui coopèrent toujours (cas DC). En retirant les agents coopératifs, qui avaient le moins de points, il ne reste que les agents traîtres, qui ne peuvent plus gagner autant de points puisque personne ne coopère. Ce type de sélection ne nous donne donc pas l'effet voulu; une stratégie de coopération conditionnelle n'a pas les bonnes conditions pour émerger.

Le deuxième type de sélection est la sélection par famine (starvation). Pour cette sélection, on ne regarde pas comment les agents se comparent les uns les autres. Plutôt, on les compare à une limite fixe de points t (threshold). Si le pointage de l'agent est en dessous de t, il meurt et est remplacé par un agent qui a un pointage plus élevé que t.
Pour nos expériences, nous avons choisi comme paramètres N = 10, E = 200, I = 100, O = -2 et t = 0. Ici, O est utilisé pour facilement déterminer le seuil selon lequel les agents survivent à la sélection: avec O = -2, les récompenses P et S sont négatives alors que R et T sont positives. Avec un seuil de zéro, l'environnement va être hostile aux coopérateurs inconditionnels et à la trahison mutuelle, ce qui en théorie rends les stratégies exploitant la récompense T inviables.

		Joueur 2	
		C	D
Joueur 1	C	R=1	S=-2
	D	T=3	P=-1

Tableau 2 - Pointage attribué au joueur 1 dans le dilemme du prisonnier, avec O = -2

Malheureusement, les agents étaient tous morts dès la deuxième ou troisième partie (voir fig. 12).

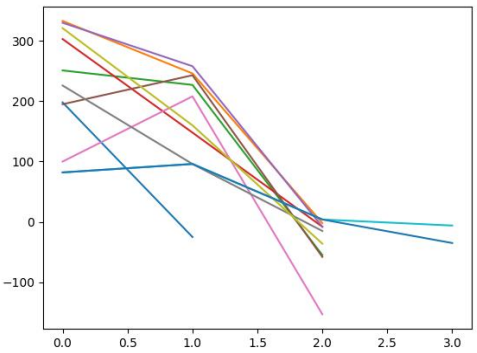


Fig. 12 - Pointage des agents à chaque partie avec sélection par famine

Comme avec l'expérience Best-Worst, nous avons introduit un délai avant que la famine s'applique. Le graphe résultant se trouve à la figure 13.

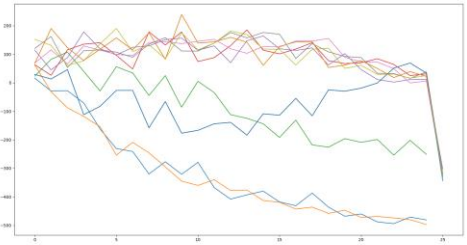


Fig. 13 - Pointage des agents à chaque partie avec sélection par famine avec délai δ=25

Comme on peut voir, dès que la famine s'installe, les agents avec le moins de points, encore ceux qui coopèrent toujours, meurent. Cela entraîne le groupe au complet à se trahir mutuellement dès la prochaine partie. Ils ont alors un pointage sous t et meurent tous. Ce type de sélection ne nous donne donc pas l'effet voulu la plupart du temps. Il y a eu quelques cas où cette sélection est arrivée à une coopération mutuelle des agents, mais cela est assez rare (voir fig. 14).

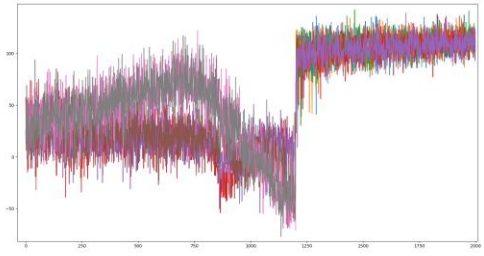


Fig. 14 - Les agents survivants ont appris à coopérer avec la sélection par famine

En examinant les agents survivants, quand il y a lieu, à l'aide de l'environnement de validation, nous avons pu découvrir un certain équilibre entre les stratégies de coopération inconditionnelle, coopération conditionnelle et trahison constante s'établit et survit à la sélection. Dans certains cas observés, cet équilibre semble s'être rompu et a entraîné le groupe vers la famine (fig. 15).

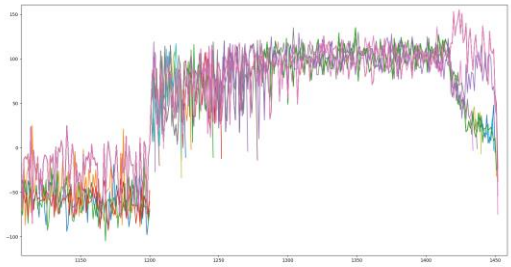


Fig. 15 - Un groupe d'agents coopératifs s'effondre avec la sélection par famine

Finalement, Notre dernière expérience avec sélection se fait avec une sélection "sociale". Ce type de sélection se fait en 2 étapes. Premièrement, lorsque les agents jouent les uns contre les autres, ils affectent la variable de confiance de chacun. À la fin d'une partie, les agents vont voter pour tuer l'agent en lequel ils font le moins confiance, celui-ci est remplacé par une copie de l'agent en lequel les autres ont le plus confiance. Cette idée fut inspirée par des jeux de rôles tels que Loup Garous de Thiercelieux et Town of Salem.

Avec ce type de sélection, un learning rate de 0.2, rate d'exploration de 0.99 et les paramètres N = 10, E = 200, I = 100, δ = 50, les agents apprennent constamment à se trahir mutuellement (voir fig. 16).

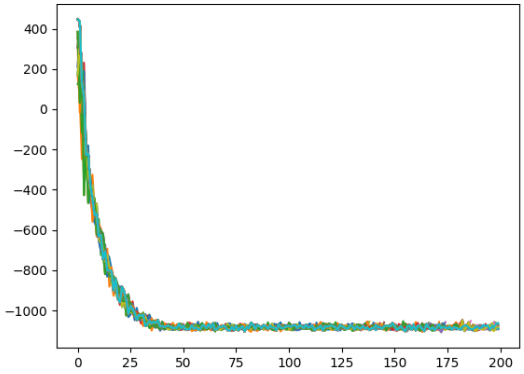


Fig. 16 - Pointage des agents avec sélection sociale et learning rate de 0.2

Nous avons donc choisi de diminuer le learning rate à 0.002 et les agents semblent alors apprendre à coopérer (voir fig.17).

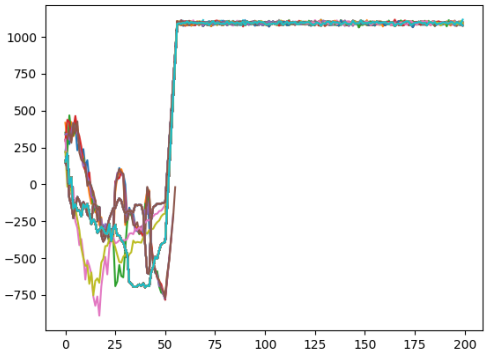


Fig. 17- Pointage des agents avec sélection sociale et learning rate de 0.002

Pour vérifier cela, nous avons fait 5 tests supplémentaires avec N = 10, E = 100, I = 20 et δ = 25 pour vérifier si le résultat obtenu est constant. Dans ces tests, les agents sont toujours arrivés à un état relativement stable (pour une situation d'agents contre agent) de coopération mutuelle (voir les graphes en annexe).

Conclusion

En conclusion, les méthodes de sélections Best-Worst et par famine ne réussissent pas à induire la coopération de façon constante: Ils tuent les agents qui coopèrent constamment en premier, ce qui ne laissent que les agents qui trahissent pour la suite. Cela rend encore plus difficile l'apprentissage de la coopération pour les agents. La sélection par famine est tout de même supérieure à la sélection «Best-Worst» dans la mesure où la survie, bien qu'improbable, mène forcément à une coopération des agents.

Par contre, la méthode de sélection sociale donne constamment des agents qui coopèrent entre eux étant donné un bon learning rate. C'est une méthode qui, selon nos expériences, est sûr d'inciter la coopération chez les agents. Toutefois, les agents résultant de la sélection sociale apprennent une stratégie de coopération inconditionnelle, qui dans un réel environnement de compétition ne serait pas une stratégie optimale.

Références

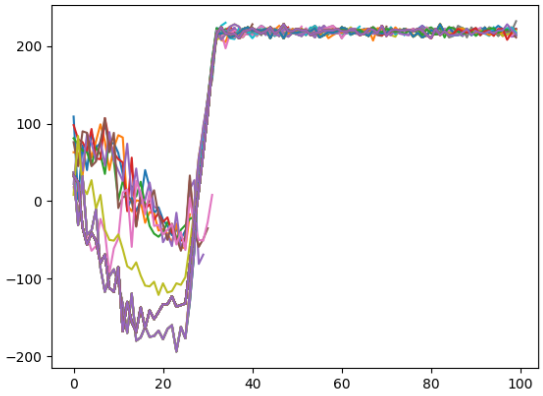
Sandholm & Crites (1996) ,Multiagent Reinforcement Learning in the Iterated Prisoner's Dilemma, http://opim.wharton.upenn.edu/~sok/papers/s/sandholm-biosystems95.pdf

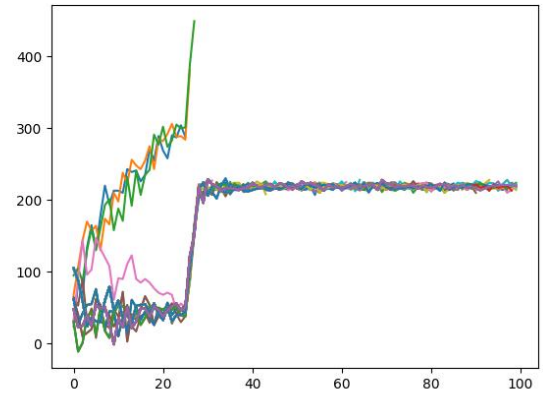
A. Agrawal & D. Jaiswal (2012),When Machine Learning Meets AI and Game Theory, http://cs229.stanford.edu/proj2012/AgrawalJaiswal-WhenMachineLearningMeetsAIandGameTheory.pdf

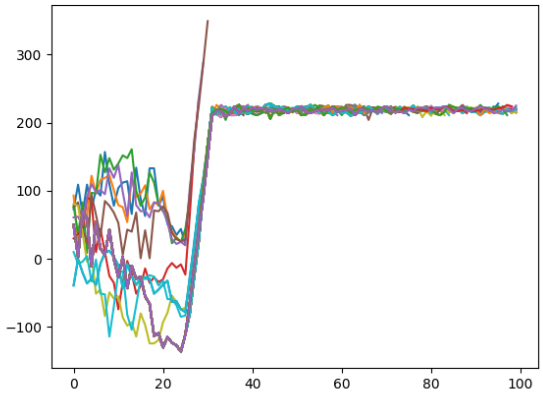
Annexe

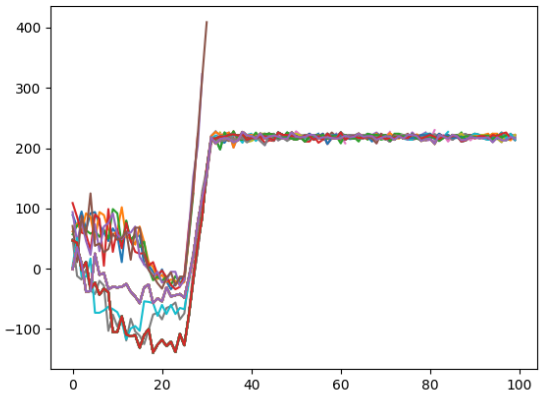
Lien vers le répertoire BitBucket: https://github.com/frtoud/multiagentIPD

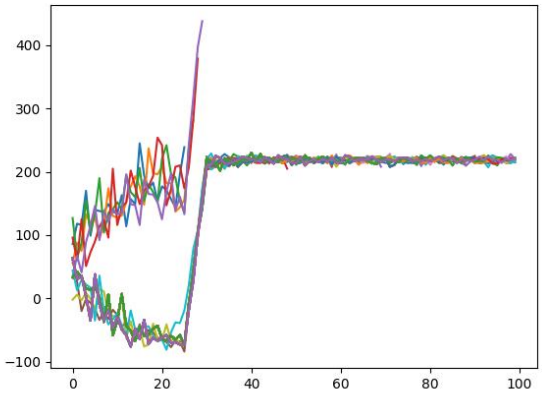
Tests supplémentaires de sélection sociale:








400
300
200
100
0
−100
0
20
40
60
80
100


400
300
200
100
0
−100
0
20
40
60
80
100