INSA LYON

IAT Projet

Apprentissage par Renforcement

- Olga Cosculluela Palacín
- Jade Archer

---

1. Formalisation

Afin de formaliser le problème dans le but de le résoudre par apprentissage, nous devons définir un état. La possibilité de tirer avec précision dépend bien sûr de la position de l'alien cible et de l'endroit où nous nous trouvons. C'est pourquoi la position est une donnée fondamentale de l'État. Nous devons préciser que la position qui nous intéresse est celle de l'alien qui se trouve le plus près du joueur, puisque c'est celle qui nous fera perdre la partie le plus rapidement. Donc, lorsque nous nous référons à l'alien, nous nous référons à celui qui est le plus proche. Toutefois, nous n'avons pas besoin d'une très grande précision, car nous devons tenir compte de la taille de l'extraterrestre et de ses déplacements à l'écran. Pour cette raison, nous avons décidé de discrétiser l'écran, afin d'obtenir la position de l'alien par rapport au joueur de manière approximative.

La position est importante à connaître, mais pour notre programme, il est vraiment important de connaître la distance à l'extraterrestre. Ainsi, au lieu d'inclure toutes les positions dans l'État, nous avons décidé d'inclure la distance discrète à laquelle il se trouve. Dans la dernière section, nous verrons comment cette décision optimise les résultats. Pour inclure la distance dans l'état, il suffit de calculer la distance à celui-ci avec le théorème de Pythagore et de la discrétiser. Pour ce faire, nous avons calculé la distance maximale qu'il peut théoriquement atteindre, soit 1000, et l'avons divisée par la taille approximative de la diagonale de l'alien. Par conséquent, nous obtenons 15 positions possibles où l'alien peut se trouver, et c'est cette information que nous incluons dans l'état.

En plus de la distance, il est important de connaître la direction dans laquelle l'extraterrestre se déplace, car la balle doit être tirée lorsque l'extraterrestre n'est pas encore juste au-dessus d'elle. Le fait de savoir dans quelle direction il se déplace permet au programme de savoir qu'en fonction de cette direction, il est préférable de tirer ou non.

Un autre élément d'information qui aurait pu être pris en compte est l'état de la balle, cependant, nous le considérons simplement pour voir si nous pouvons tirer une autre balle ou non. Que l'on tue l'alien ou non est complètement indépendant de l'état de la balle.

En conclusion, l'état que nous avons choisi est distance discrétisée entre l'alien et le joueur et la direction dans laquelle l'alien se déplace.

2. Algorithme

Parmi les nombreux algorithmes d'apprentissage par renforcement existants, nous avons choisi le Q-Learning. Le Q-Learning est une méthode d'apprentissage par renforcement pour résoudre des problèmes de décision séquentiels dans lesquels l'utilité d'une action dépend d'une séquence de décisions et où il existe une incertitude sur la dynamique de l'environnement dans lequel se trouve l'agent.

Cet algorithme converge vers Q* avec une représentation tabulaire et diverge avec une représentation basée sur Q-Networks. Nous mettons en évidence la non-stationnarité des objectifs et la corrélation entre les échantillons dans cette méthode, ce qui en fait un algorithme très approprié pour résoudre notre problème.

3. Apprentissage / Hyperparamétrage

Les hyperparamètres sont des paramètres dont les valeurs contrôlent le processus d'apprentissage et déterminent les valeurs des paramètres du modèle qu'un algorithme d'apprentissage finit par apprendre. C'est pourquoi il est important de les choisir correctement. Les hyperparamètres sont optimisés simplement en entraînant un modèle sur une grille de valeurs d'hyperparamètres possibles et en prenant celle qui est la plus performante sur un échantillon de validation.

Les hyperparamètres les plus importants sont : alpha, gamma et epsilon.

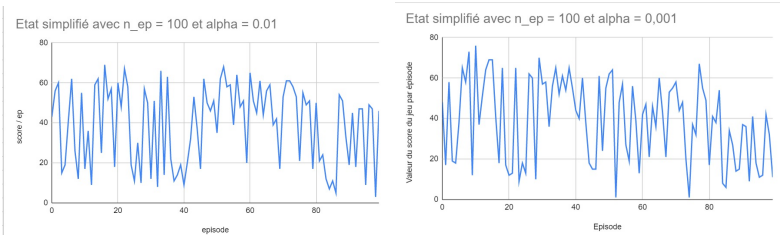
Le taux d'apprentissage (alpha) est un outil qui peut être utilisé pour déterminer dans quelle mesure nous conservons nos connaissances antérieures de notre expérience qui doit être conservée pour nos paires état-action.

Le facteur de réduction (gamma) fonctionne comme suit : à mesure que vous vous rapprochez de l'échéance, votre préférence pour la récompense à court terme devrait augmenter, car vous ne serez pas là assez longtemps pour obtenir la récompense à long terme, ce qui signifie que votre gamma devrait diminuer.

Au fur et à mesure que nous développons notre stratégie, nous avons moins besoin d'exploration et plus d'exploitation pour obtenir plus d'utilité de notre politique, donc plus les essais augmentent, plus epsilon devrait diminuer. Comme nous le verrons plus loin, nous avons choisi de créer une fonction qui fait varier l'epsilon entre 0 et 1 avec un pas de 0,1.

Le nombre d'épisodes et le nombre maximal d'actions par épisode peuvent varier en fonction du temps que l'on est prêt à attendre pour qu'il apprenne. Dans chaque épisode, il y a un moment où les connaissances acquises ne s'améliorent pas sensiblement, et c'est ce que nous avons essayé de découvrir de manière expérimentale.

La stratégie que nous avons suivie pour les optimiser a été de le faire manuellement. Nous avons gardé tous les paramètres constants, à l'exception d'un seul, que nous n'avons cessé de modifier, et nous avons conservé la valeur qui donnait les meilleurs résultats.On montre ci-dessous les graphiques en comparant alpha pour voir comment on a une notable différence entre le score qu'on obtient et pour voir comment on a choisi les paramètres.



Les valeurs que nous avons retenues après en avoir testé plusieurs sont les suivantes:

- alpha = 0.01
- gamma = 1
- epsilon = EpsilonProfile(1.0, 0.1)
- nº épisodes = 100 (plus aurait été bien mieux, mais pour les raisons evoquées dans la partie suivante çommentaires"nous n'avons pas pu faire plus).

- max. steps = 10000

4. Commentaires

Lors du développement de notre programme, nous avons fait face à de nombreux problèmes qui ont probablement beaucoup impacté les performances.

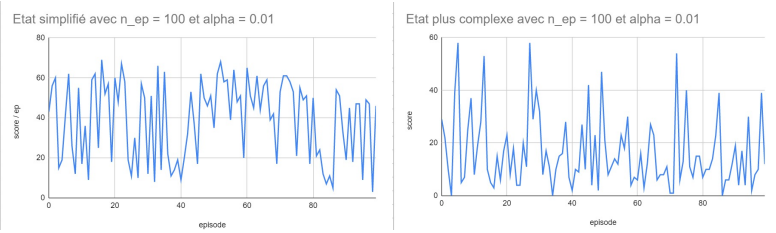
- Tout d'abord, un bug faisait qu'un alien apparaisse hors de la fenêtre (ou à la limite) ce qui faisait crasher immédiatement le programme. Nous avons pu le corriger en réduisant beaucoup la fenêtre d'apparition des aliens, mais cela n'a pas empêché qu'il continue à se produire par moment, sans faire crasher. Cette dernière version est aussi embêtante, puisque l'alien descend en ligne droite ce qui crée un game over presque instantané, ce qui risque aussi de fausser légèrement l'apprentissage de l'IA.
- La façon optimale de faire les tests aurait été de les effectuer sur un ordinateur non utilisé avec assez de mémoire et de temps pour faire au moins 1000 épisodes. Cependant, cela n'était pas possible et les tests ont été effectués sur un ordinateur qui faisait tourner d'autres choses en même temps, ce qui a pu sans doute impacter les performances. De plus, en fonction de l'état choisi chaque épisode mettait plus ou moins de temps. Lorsque nous avons essayé de faire une simulation de 1000 épisodes avec l'état simplifié (donc beaucoup plus rapide que notre état plus complexe), cela a mis entre 3 et 4h. C'est pourquoi nous n'avons pas eu le temps d'en effectuer plus et nous n'avons pas non plus pu essayer avec l'autre état qui prenait 3* plus de temps pour juste 100 épisodes.

5. Analyse des résultats

La stratégie que nous avons suivie pour optimiser les hyperparamètres a été de le faire manuellement. Nous avons gardé tous les paramètres constants, à l'exception d'un seul, que nous n'avons cessé de modifier, et nous avons conservé la valeur qui donnait les meilleurs résultats.

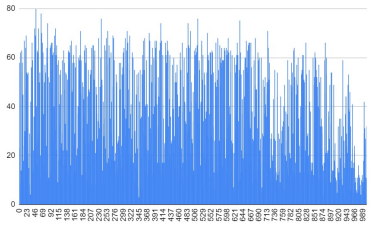
Afin de choisir un état pertinent, nous avons suivi la même stratégie, en avons testé plusieurs et avons comparé les résultats de chacun d'entre eux. Dans cette partie, nous montrerons les données obtenues à partir des différents états testés, cependant, les hyperparamètres ont été testés en parallèle.

Comme nous l'avons vu dans la première partie, l'autre état que nous avons testé était celui qui incluait la position de l'étranger. Autrement dit, au lieu de discrétiser la distance, nous discrétisons la position verticale de l'extraterrestre et les positions horizontales du joueur et de l'extraterrestre. Cependant, les résultats obtenus étaient moins bons. En effet, dans l'apprentissage par renforcement, il est important de trouver un état cible qui comporte le moins de variables possible. En obtenant un état plus simple, mais avec un degré d'information similaire, nous obtenons une meilleure et beaucoup plus rapide performance. Nous pouvons voir la différence de performance dans les graphiques ci-dessous.



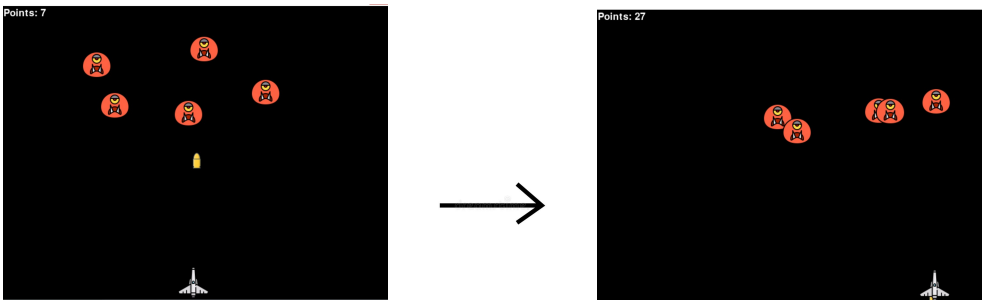
Nous pouvons voir qu'avec l'état où nous discrétisons les positions séparément (Etat plus complexe), nous obtenons des résultats plus aléatoires et plus extrêmes. En effet, pendant cette période, il n'apprend pas, mais joue au hasard. Nous obtiendrions probablement des résultats similaires à la fin, mais nous passerions beaucoup plus de temps pour les obtenir.

Avec l'état simplifié, nous voyons comment à la fin du graphique nous obtenons plus de valeurs constantes, comme résultat du processus d'apprentissage. Pour vraiment constater une amélioration du score obtenu, 100 épisodes ne suffisent pas, mais l'ordinateur utilisé n'a pas les capacités de pouvoir faire plus. Nous avons cependant réussi à exécuter une fois 1000 épisodes (ce qui à pris environ 4h), et nous avons obtenu les résultats ci-dessous (alpha = 0.01):



On remarque que le score a une tendance décroissante. En observant son comportement sur le jeu directement cependant, on observe que l'IA a bien appris des choses vue les déplacements qu'elle faisait.

En effet, près avoir exécuté le programme plusieurs fois, le comportement après l'apprentissage est tout à fait évident. Le joueur se tient très souvent à une extrémité de l'écran et tire continuellement de là, améliorant ainsi son score. Nous pouvons voir ce comportement dans l'image ci-dessous.



On remarque aussi que le joueur a tendance à beaucoup plus faire des déplacements quand les aliens se retrouvent proches de lui; alors qu'il avait tendance à rester statique lors des premiers episodes.

Nous pouvons conclure en disant que le programme suit un processus d'apprentissage, car même les graphiques ne montrent montrent des résultats pouvant sembler aléatoires et même décroissants, nous pouvons voir sur la fenêtre du jeu que la façon de jouer de l'IA n'a rien a voir entre le premier et dernier épisode. De plus, si l'on regarde la lecture du programme, on peut voir comment, au fur et à mesure des épisodes, le joueur suit une stratégie très claire : se positionner d'un côté de l'écran et bouger plus lorsque les aliens se rapprochent. Pour autant, nous pouvons voir comment le programme apprend, et comment les paramètres choisis sont effectivement les plus efficaces pour l'algorithme que nous avons développé.