Séance 4

1) Dernières améliorations et tests avant la compétition

Si vous avez le temps, vous pouvez :

- Chercher à optimiser vos fonctions d’évaluations existantes (cf section suivante)
- Explorer les pistes d’ameioration de minimax mentionnées sur le site du projet
- Impélementer des fonctions d’évaluations pour les échecs et abalone. Les deux jeux obligatoires sont le puissance4 et les dames (dans le cas des dames, votre IA doit fonctionner pour n’importe quelle configuration et règles, cf le wiki de aiarena)
- Si ce n’est pas déjà fait, implémenter des tests unitaires pour vos fonctions d’évaluation en les éxecutant sur des états de jeux pour lesquels vous aurez au préalable calculé le score attendu « à la main ».

Le plus important étant d’avoir bien testé vos IA pour dames et puissance4 en les faisant s’affronter sur differents jeux avec differents temps impartis, pour s’assurer qu’elles ne crashent pas et ne dépassent jamais leur temps de jeu.

2) Profilage et optimisation

Les deux point-clés de la performance de minimax sont la rapidité d'exploration et la qualité de l'estimation renvoyée par la fonction d'évaluation. Les fonctions d’exploration de l’arbre sont déjà fournies par le package aiarena donc vous n’avez pas trop la possibilité de les ameliorer. Vous pouvez cependant vous interesser à la rapidité de vos minimax et fonctions d’évaluation.

Pour déterminer les parties de votre code dans lesquelles votre IA passe le plus de temps d’exécution, vous pouvez d'utiliser un profiler. Un profiler est un outil qui permet -notamment- de mesurer le temps passé dans chaque fonction d'un programme. On peut ainsi déterminer quelles sont les lignes consommant le plus de temps de calcul, et donc évaluer les parties de code où il est possible de gagner un peu de temps.

Python dispose d’un profiler nommé cProfile. Il s’utilise comme suit (remplacer les <>) :

```
> python3 -m cProfile -o logs.cprof <chemin/vers/mon/fichier.py>
```

ou bien dans le cas d’un script définie dans un package :

```
python3 -m cProfile -o logs.cprof -m <nom.de.mon.module>
par exemple
python3 -m cProfile -o logs.cprof -m <IN104_PROJECT_X_Y.scripts.ai_vs_ai>
```

cProfile produit un fichier (nommé logs.cprof dans l’exemple précedent) listant la durée passée dans chaque morceau de code. Des outils permettent ensuite d’afficher ces résultats sous forme de graphiques. Je vous suggère Snakeviz qui est simple d'utilisation.

Installer Snakeviz:

```
> pip install snakeviz
```

Pour visualiser avec Snakeviz le rapport de profilage généré par cProfile, éxécutez :

```
> snakeviz   logs.cprof
```

Si le navigateur ne s'ouvre pas automatiquement, copier le lien snakeviz dans le terminal et copier le dans votre navigateur internet

5) Profilez vos IA (en profilant un match entre un random et votre IA par exemple) pour les differents jeux pour lesquels vous avez implémenté une fonction d'évaluation.

6) Grâce au profiler, déterminez quelles sont les fonctions les plus consommatrices en temps dans votre minimax. Des techniques

- accélérer les fonctions coûteuses en temps de calcul

- réduire le nombre de fois qu'est appelée une fonction coûteuse en l'exécutant une fois et en réutilisant le résultat (plutôt que de la ré appeler à chaque fois).

3) Rendus

a) Compétition

La compétition de fin de projet fera s'affronter les IA des differents groupes sur les jeux de dames et puissance4 uniquement (le travail effectué sur les autres jeux n'est pas inutile et peut être présenté dans le rapport et la présentation orale). Pour les jeux de dames, differentes régles et tailles de plateau seront utilisées. Pour les 2 jeux, des parties avec idfferents temps limites seront effectuées.Lors d'une partie, **une IA qui ne répondra pas dans le temps imparti sera considérée comme perdante**.

Je récupererai la version de code présente sur la branche master du répo Github de chaque groupe le vendredi 21 mai à 23h. Pour indiquer la classe que je devrais utiliser pour représenter votre groupe, vous devrez l'importer sous le nom de « default_AI » dans le fichier __init__.py situé dans le dossier contenant les sous-dossier minimax, scripts, evaluation_functions …

Par exemple, si vous vouliez utiliser votre randomBrain pour la competition, vous écririez dans votre __init__.py:

```
from .randomBrain import RandomBrain as default_AI
```

b) Rapport et présentation

Vous devrez avoir poussé sur la branche master de votre répo Github, avant samedi 22 mai à 23h, un rapport de 8 pages **maximum.**

La soutenance aura lieu le 25 mai. Votre présentation orale devra durer 10 minutes seulement, appuyée par un support visuel (ppt, pdf, …). Attention, 10 minutes, c’est très court ! Les groupes n’assistent pas à la présentation des autres groupes. L’ordre de passage n’est pas encore determiné et vous sera communiqué ulterieurement.

Concernant le contenu du rapport et de la présentation orale :

Le but n’est pas que vous réexpliquiez le fonctionnement de minimax, d’alpha-beta ou l’objectif du projet (ces choses sont communes à tous les groupes et je les connais déjà). Le plus interessant est que vous présentiez par exemple de :

- vos observations : comparaison de vos IA : leur temps de reflexion, leur « forces », ...
- vos reflexions et astuces pour concevoir le minimax le plus efficace possible, la fonction d’évaluation la plus précise possible ….
- le cas échéant : les points qui vous ont posé les plus gros problèmes et les solutions que vous avez trouvé
- toute autre initiative personnelle

bref, les choses qui n’étaient pas très détaillées dans les fiches séances et qui peuvent vous differencier des autres.