Projet Q-learning

Labyrinthe

Le but du projet est de programmer un algorithme de Q-learning pour faire sortir un robot d'un labyrinthe. Un labyrinthe sera décrit par une matrice. Il existe plusieurs types de case (toutes de même taille) :

- Une entrée.
- Des cases libres.
- Des murs (impossible de se déplacer dessus).
- Une sortie.
- Des pièges (qui pénalisent le robot).

Faire une classe labyrinthe qui :

- Peut se charger à partir d'un fichier texte.
- Renvoie la liste des déplacements possibles à partir d'une case.
- Renvoie la case d'arrivée et le renforcement à partir d'un déplacement et d'une case.

Le robot ne peut pas se déplacer en diagonale. On commencera avec un labyrinthe de taille raisonnable (par exemple 10 x 10), créé manuellement.

Q-learning

Ecrire l'algorithme de Q-learning. L'algorithme aura les paramètres suivants:

- Le coût des différentes action (se déplacer, se déplacer sur un piège, se déplacer sur une sortie, rentrer dans un mur).
- La pénalité sur le long terme (γ).
- Le nombre de déplacements total autorisé (de l'ordre de 5000 au début, à ajuster).

On veut pouvoir :

- Choisir une stratégie (en plus de l'exploration pure et de l'exploitation pure) et l'évaluer (moyenne des renforcement à la fin de l'apprentissage).
- Comparer deux stratégies.
- Choisir de faire ou non du backtracking.

Il est conseillé de commencer par une version simple, sans backtracking et avec exploitation pure, et d'étoffer le code petit à petit.

Interface Graphique

Après avoir obtenu une version basique qui fonctionne en console, faire une interface graphique afin de représenter chaque étape de l'algorithme. On devra pouvoir visualiser:

- Le labyrinthe.
- La position du robot.
- La valeur courante de max(Q) sur chaque case (par exemple avec un code couleur).

En option, pour vous aider à débugger, vous pouvez implémenter un mode où vous pouvez diriger le robot au clavier et observer les effets directement (ce mode optionnel ne sera pas noté).

Exemple : https://youtu.be/-JXxYZ5HB8U

Bonus

- Ajouter des cases portals: passer sur un portal amène aléatoirement à un autre portal.
- Faire en sorte que le robot apprenne à éviter les murs : foncer dans un mur n'est plus interdit mais le robot ne change pas de case et subit un malus.
- Générer aléatoirement le labyrinthe.
- Bonus du bonus : ajouter des cases bonus.

Modalités

Le projet se fait en **binôme**. Il peut être codé en Python ou en Java. Pour Python, veuillez utiliser Python 2.7 ou 3.5 avec le package TkInter pour l'interface graphique. Aucun package tiers n'est autorisé.
Vous devez rendre par mail à votre enseignant de TP (avec comme objet PROJET_QLEARNING_NOM1_NOM2) une archive zip contenant :

1. Les sources du projet.
2. L'exécutable. Un **menu** devra être proposé pour pouvoir tester les différentes parties du projet. Ce menu devra être lancé par défaut et ne devra pas nécessiter de modifier le code (attention aux chemins absolus vers des fichiers notamment). Le menu peut être fait en console.
3. Un rapport (.pdf) d'une page **maximum** résumant votre travail, les points importants, les difficultés rencontrées, etc.