UC6 2025-2026

TP : Apprentissage par Renforcement - Q-Learning (GridWorld)

Objectifs

- Comprendre l'apprentissage par renforcement en expérimentant
- Implémenter le Q-learning sur une tâche simple
- Visualiser l'apprentissage
- Analyser l'influence des hyperparamètres en lien avec l'exploration et les récompenses

Contexte

Pour ce TP, nous allons considérer un agent qui évolue dans une grille 5x5 et doit atteindre un objectif en évitant des pièges.

L'objectif est matérialisé par une case spécifique.
De même, les pièges sont des cases particulières dans son environnement c'est à dire la grille 5x5.

Dans ce contexte, l'état perçu par l'agent est sa position dans la grille.
Il possède 4 actions : aller vers le haut, vers le bas, vers la gauche ou vers la droite.

Un squelette de programme vous est fourni. Dans celui-ci un objet particulier a été défini pour représenter l'environnement (ici la grille 5x5).
Les éléments à compléter sont matérialisés par des commentaires commençant par TODO.

Travail demandé

Pour chaque point demandé vous veillerez à implémenter les éléments pas à pas en réalisant des tests au fur et à mesure.

1. Compléter l'environnement

   Vous devez ici compléter la méthode **`step`** qui doit permettre de déterminer étant donné une action le nouvel état de l'agent, sa récompense et s'il a atteint l'objectif.

2. Implémenter **`ε-greedy`**

Vous devez ici compléter le code de la fonction **ε-greedy** qui étant donné

- **Q** qui représente l'ensemble des q_valeurs pour chaque état possible (ici une grille à 3 dimensions : position dans la grille et action choisie)
- **epsilon** qui détermine la probabilité de choisir l'action au hasard
- Et **state** l'état dans lequel se trouve l'agent

retourne l'action à réaliser.

3. Implémenter le Q-learning

   Vous devez ici compléter le code de fonction **train** suivant l'algorithme du Q-learning.

4. Ajouter la visualisation des récompenses totales par épisodes lors de l'apprentissage.

   Vous compléterez ici vous programme pour que au fur à mesure des épisodes l'on puisse visualiser sur une courbe les récompenses cumulées par l'agent.

   Pour cela vous utiliserez la bibliothèque matplotlib et notamment les fonctions :

   - **ion** et **ioff** pour actionner ou terminer le mode interactif
   - **plot** pour le tracé de la courbe point à point
   - **pause** pour permettre à l'épisode suivant d'être calculé
   - **show** pour l'affichage du graphique

   Vous pourrez également pour cela vous appuyer sur le script fournit en exemple et nommé **courbe_interactive.py**

5. Ajouter la visualisation après apprentissage du parcours réalisé par l'agent suivant sa politique.

   Vous complèterez pour cela la méthode **update** de **GridGUI** pour qu'étant donné l'état courant de l'agent représenté par **env.state** une action soit choisie suivant la politique de l'agent et que son état soit mis à jour. Vous utiliserez bien entendu pour cela la méthode **step** qui vous avez complétez à la question 1.

6. Analyser les résultats et l'influence des hyperparamètres sur la vitesse de convergence et la politique optimale :
   a. Epsilon
   b. Récompenses
   c. Nombre de pièges
   d. Taille de la grille