Rapport IA du jeu Oshi-Zumo

Introduction

Le jeu Oshi-Zumo, également connu sous le nom de Sumo Push, est un jeu stratégique traditionnel japonais. Il oppose deux joueurs qui doivent utiliser leurs pièces (ou "monnaies") pour pousser un lutteur vers le côté adverse du plateau. Chaque joueur commence avec un nombre identique de pièces, et à chaque tour, ils misent secrètement un certain nombre de pièces. Le joueur qui mise le plus pousse le lutteur d'une case vers le côté adverse. Le jeu se termine lorsque le lutteur est poussé hors du plateau ou lorsque les deux joueurs n'ont plus de pièces à miser. Le but est de pousser le lutteur hors du plateau de l'adversaire ou de faire en sorte que l'adversaire n'ait plus de pièces pour continuer à jouer.

Description des Composants de l'IA Oshi-Zumo

Fonction `solve_game`

La fonction **`solve_game`** prend en entrée un indicateur de joueur et utilise la programmation linéaire pour calculer la stratégie optimale dans ce jeu à somme nulle. Elle renvoie un vecteur de probabilité et la valeur attendue de la stratégie mixte.

RewardMethodEnum (Classe d'Énumération)

Cette classe définit différentes méthodes de récompense pour le jeu :

- **`stepReward`** : Attribue une récompense à l'un des joueurs à chaque tour.
- **`endReward`** : Attribue la récompense à la fin de la partie.
- **`winTieLoseReward`** : Prend en compte la position du lutteur et le nombre de pièces restantes pour évaluer la récompense.

OpponentPolicyEnum (Classe d'Énumération)

Cette classe définit la politique de l'adversaire dans le jeu :

- Affronter une autre IA.
- Affronter un joueur aléatoire (random) avec des ajustements pour rendre le jeu plus intéressant.

Classe `OshiZumoState`

Cette classe représente l'état du jeu avec les attributs suivants :

- Position du lutteur.
- Nombre de pièces de chaque joueur.
- Taille du plateau.
- Indication si les deux joueurs ont misé 0, ce qui signale la fin du jeu.

Fonctions clés :

- **`is_wrestler_off_board`** : Vérifie si le lutteur est hors du plateau.
- **`no_coins`** : Vérifie s'il reste des pièces sur le plateau.
- **`print_state`** : Affiche l'état actuel du jeu.

Classe `OshiZumoGame`

Cette classe gère le jeu en prenant les paramètres suivants :

- Nombre de pièces initiales pour chaque joueur (**`self.N`**).
- Mise minimale (**`self.M`**).
- Taille du plateau (**`self.K`**) et calcul de la longueur de la piste (**`self.TrackLengthK`**).
- Méthode de récompense à utiliser (**`self.rewardMethod`**).

Fonctions importantes :

- **`start_state`** : Initialise le jeu.
- **`is_game_over`** : Détermine si la partie est terminée.
- **`max_action`** et **`min_action`** : Définissent les actions possibles pour les deux joueurs.
- **`successors`** : Génère les nouveaux états du jeu après chaque tour.

Fonction `evaluation_reward`

Cette fonction évalue la récompense en utilisant la méthode **`winTieLoseReward`**, en prenant en compte le nombre de pièces et la position du lutteur, puis en appliquant une formule utilisant la tangente hyperbolique.

Fonction `minmax_value`

Implémente l’algorithme Minimax avec élagage alpha-beta pour calculer la valeur d’un état du jeu. Utilise la récursivité pour explorer les possibilités de jeu et choisir la stratégie optimale.

Fonction `play_game`

Lance une simulation de partie :

- Initialise le jeu avec **`start_state`**.
- Boucle jusqu'à la fin de la partie.
- Calcule la solution optimale pour chaque joueur à chaque tour.
- Affiche le vainqueur et la stratégie adoptée.

Conclusion

Le projet de création d'une IA pour le jeu Oshi-Zumo utilise des concepts avancés de programmation linéaire et de théorie des jeux pour simuler et optimiser les stratégies. Les différentes classes et fonctions sont conçues pour modéliser fidèlement les règles et dynamiques du jeu, permettant de créer une IA capable de jouer de manière optimale ou de simuler des adversaires variés. Ce projet met en lumière l'intersection entre la théorie des jeux, la programmation et l'algorithmique, offrant une plateforme intéressante pour étudier et améliorer les stratégies dans les jeux compétitifs.