RENDU DU TP DE FONDEMENT DE L'IA

MANOBA Ougadja Georges

M1 TNSID

TITRE DU PROJET: **OTHELLO**

Sommaire

- Introduction

I. Modélisation et Algorithmes

II. Validation des Résultats

III. Discussion des Résultats Obtenu

- Conclusion

- Annexes

- Bibliographie

INTRODUCTION

Le présent rapport décrit le développement d'une intelligence artificielle (IA) pour le jeu d'Othello, en mettant en œuvre l'algorithme Minimax avec élagage alpha-bêta. Ce jeu de plateau classique nécessite des stratégies avancées pour anticiper les mouvements de l'adversaire et maximiser ses propres gains. L'objectif principal est de développer

une IA capable de jouer contre un joueur humain ou d'autres stratégies IA, et d'évaluer ses performances.

I. Modélisation et Algorithmes

La modélisation du jeu a été réalisée en utilisant Python avec la bibliothèque Tkinter pour l'interface graphique. Le plateau est représenté par une matrice 8x8 et les pièces sont codées comme des constantes (VIDE, NOIR, BLANC). L'algorithme Minimax avec élagage alpha-bêta a été implémenté pour que l'IA prenne des décisions optimales.

L'analyse détaillée de l'algorithme et des structures de données utilisées pour représenter le plateau et les mouvements possibles est présentée en annexe.

II. Validation des Résultats

Pour valider l'efficacité de l'IA, plusieurs critères ont été pris en compte, notamment le temps de calcul, le nombre de nœuds générés et la comparaison des différentes stratégies IA entre elles, ainsi qu'une confrontation entre l'IA et un joueur humain. Des mesures numériques ont été obtenues pour évaluer les performances de l'IA dans des scénarios variés.

Ceci est l'interface graphique du jeu d'othello.

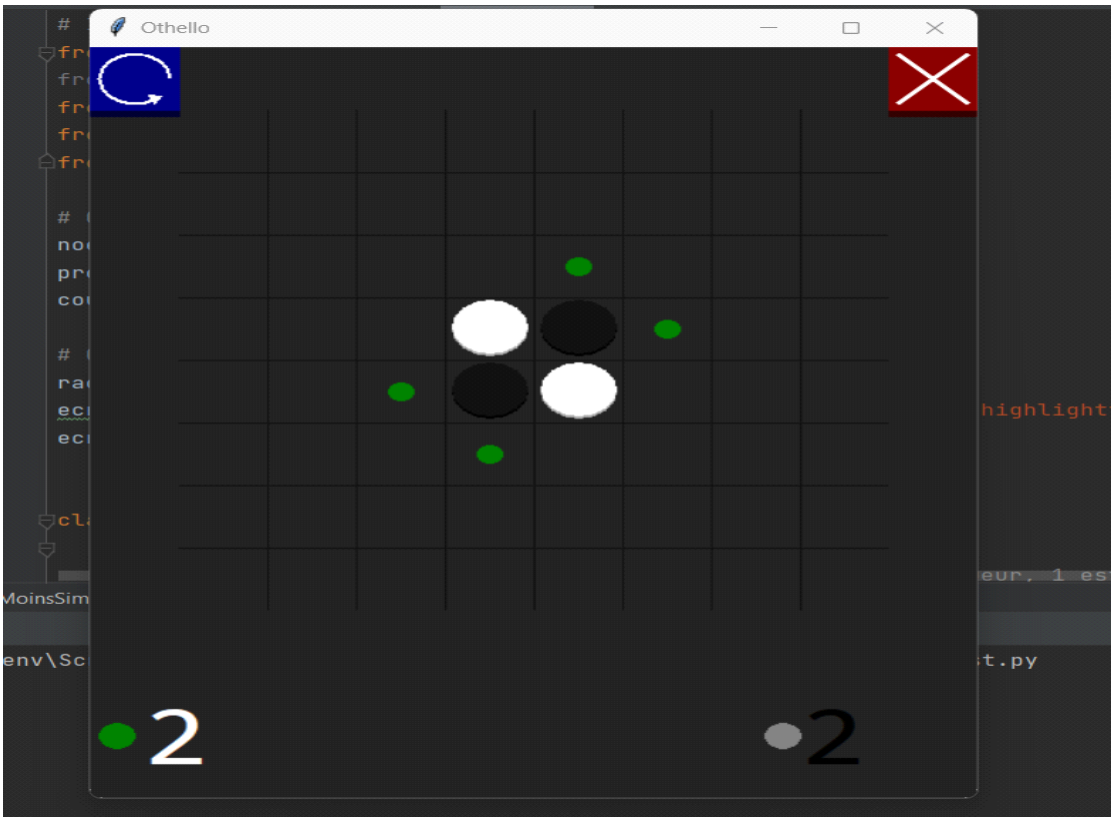


figure 1

III. Discussion des Résultats Obtenu

Cette section analyse les résultats obtenus et évalue l'efficacité des stratégies employées. Elle met en lumière les performances du joueur humain comparé aux différentes stratégies d'IA, souligne les principaux défis et limites du projet, et propose des pistes d'amélioration pour répondre à ces limites.

```
Noirs: 4, Blancs: 1
Noirs: 6, Blancs: 0
Noirs: 4, Blancs: 1
```

figure 3

5. Conclusion

Le projet Othello a abouti à une application fonctionnelle avec une IA performante grâce à Minimax et à l'élagage alpha-bêta. Malgré ses

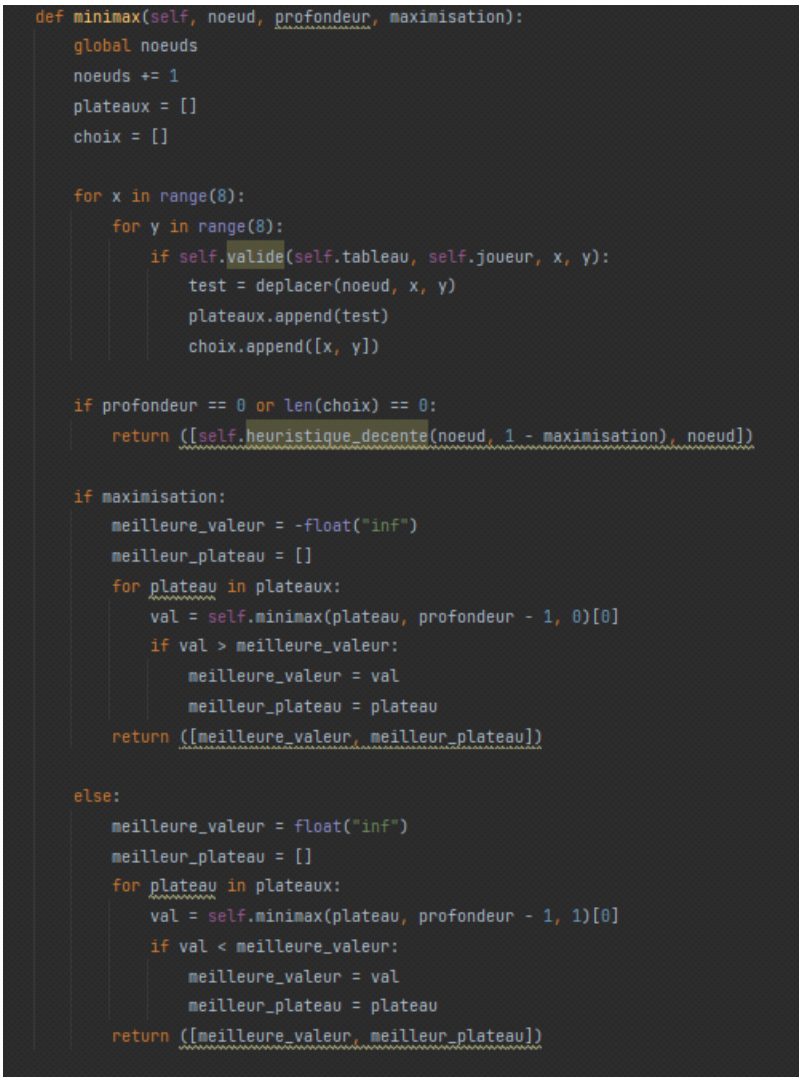
succès face à l'humain, des pistes d'amélioration subsistent, notamment pour optimiser davantage les stratégies d'IA. Ce projet représente une exploration enrichissante des défis algorithmiques des jeux de stratégie, ouvrant la voie à des améliorations futures pour renforcer l'efficacité de l'IA dans le jeu Othello.

6. Annexes

Dans cette section se trouvent le code source de l'algorithme Minimax, ainsi que des détails techniques sur la modélisation du jeu.

l'algorithme minimax explore les mouvements possibles à chaque niveau de l'arbre, en alternant entre la maximisation et la minimisation. La fonction renvoie la meilleure valeur évaluée et le meilleur coup à jouer à partir de la position actuelle du tableau.

Voici un illustration du code de cet algorithme



```
def minimax(self, noeud, profondeur, maximisation):
    global noeuds
    noeuds += 1
    plateaux = []
    choix = []

    for x in range(8):
        for y in range(8):
            if self.valide(self.tableau, self.joueur, x, y):
                test = deplacer(noeud, x, y)
                plateaux.append(test)
                choix.append([x, y])

    if profondeur == 0 or len(choix) == 0:
        return ([self.heuristique_decente(noeud, 1 - maximisation), noeud])

    if maximisation:
        meilleure_valeur = -float("inf")
        meilleur_plateau = []
        for plateau in plateaux:
            val = self.minimax(plateau, profondeur - 1, 0)[0]
            if val > meilleure_valeur:
                meilleure_valeur = val
                meilleur_plateau = plateau
        return ([meilleure_valeur, meilleur_plateau])

    else:
        meilleure_valeur = float("inf")
        meilleur_plateau = []
        for plateau in plateaux:
            val = self.minimax(plateau, profondeur - 1, 1)[0]
            if val < meilleure_valeur:
                meilleure_valeur = val
                meilleur_plateau = plateau
        return ([meilleure_valeur, meilleur_plateau])
```

figure 4

L'algorithme Alpha-Beta est une amélioration de l'algorithme Minimax, visant à réduire le nombre de nœuds évalués dans l'arbre de recherche. Cela permet d'accélérer la recherche tout en garantissant les mêmes résultats qu'un Minimax complet.

L'idée principale de l'algorithme Alpha-Beta est d'éliminer les branches de l'arbre qui ne peuvent pas influencer la décision finale. Pour ce faire, il utilise deux valeurs, alpha et bêta, qui représentent les limites inférieure et supérieure de la valeur d'un nœud.

Voici un illustration du code de cet algorithme

```python
def minimax(self, noeud, profondeur, maximisation):
    global noeuds
    noeuds += 1
    plateaux = []
    choix = []

    for x in range(8):
        for y in range(8):
            if self.valide(self.tableau, self.joueur, x, y):
                test = deplacer(noeud, x, y)
                plateaux.append(test)
                choix.append([x, y])

    if profondeur == 0 or len(choix) == 0:
        return ([self.heuristique_decente(noeud, 1 - maximisation), noeud])

    if maximisation:
        meilleure_valeur = -float("inf")
        meilleur_plateau = []
        for plateau in plateaux:
            val = self.minimax(plateau, profondeur - 1, 0)[0]
            if val > meilleure_valeur:
                meilleure_valeur = val
                meilleur_plateau = plateau
        return ([meilleure_valeur, meilleur_plateau])

    else:
        meilleure_valeur = float("inf")
        meilleur_plateau = []
        for plateau in plateaux:
            val = self.minimax(plateau, profondeur - 1, 1)[0]
            if val < meilleure_valeur:
                meilleure_valeur = val
                meilleur_plateau = plateau
        return ([meilleure_valeur, meilleur_plateau])
```

7. Bibliographie

Une liste des références bibliographiques utilisées pour l'élaboration et l'analyse du rapport est présentée dans cette section.

https://kevinragonneau.fr/blog/jeux/othello

https://www.ffothello.org/othello/regles-du-jeu/