1. **MÉTHODOLOGIE**
2. **Explication du fonctionnement de l'agent :**

L'agent implémenté dans le script est conçu pour jouer au jeu d'Abalone, un jeu de société complexe qui demande stratégie et anticipation. L'intelligence de cet agent est articulée autour de l'algorithme Minimax avec Alpha-Beta Pruning, une technique standard en intelligence artificielle pour la prise de décision optimale dans les jeux à deux joueurs. Voici une briève explication de l’algorithme Minimax avec Alpha-Beta Pruning:

* Algorithme Minimax:Cet algorithme joue un rôle central dans la stratégie de l'agent. Il simule toutes les possibilités de mouvements jusqu'à une certaine profondeur définie, en alternant entre les perspectives du joueur et de l'adversaire. Le but est d'identifier le meilleur coup possible en évaluant les états futurs du jeu.
* Alpha-Beta Pruning: Cette technique améliore l'efficacité de Minimax en éliminant les branches de l'arbre de décision qui n'ont pas besoin d'être explorées, réduisant ainsi le temps de calcul nécessaire.

Cela dit, les fonctions clés du script sont les suivantes :

* *\_\_init\_\_ :* Initialise l'agent avec un type de pièce, un nom, et une limite de temps pour l'ensemble de la partie. Il définit également une limite de profondeur pour l'algorithme Minimax, ce qui est crucial pour gérer la complexité computationnelle.
* *compute\_action :* Cette fonction est appelée à chaque tour de l'agent pour déterminer son mouvement. Elle parcourt toutes les actions possibles à partir de l'état actuel du jeu et utilise Minimax pour évaluer et choisir la meilleure action. La fonction cherche à maximiser le score du joueur, en tenant compte de la limite de profondeur.
* *minimax :* Implémente l'algorithme Minimax avec Alpha-Beta Pruning. Cette fonction est récursive, explorant l'arbre de jeu jusqu'à la limite de profondeur ou jusqu'à ce qu'un état de jeu final soit atteint. Elle alterne entre maximiser le score du joueur (is\_maximizing\_player = True) et minimiser celui de l'adversaire (is\_maximizing\_player = False), en ajustant les valeurs alpha et beta pour le pruning.
* *evaluate :* Fournit une évaluation heuristique des états de jeu lorsque la limite de profondeur est atteinte ou que le jeu est terminé. Elle compte le nombre de billes pour chaque joueur et renvoie la différence, un score positif indiquant un avantage pour le joueur.

De plus, il est important de noter que la limite de profondeur définie dans l'initialisation de l'agent influence directement la profondeur de l'analyse du jeu et le temps de calcul. Une profondeur limitée à un niveau relativement superficiel (dans ce cas, 3) permet de garder les calculs rapides et gérables, mais cela limite la capacité de l'agent à anticiper des coups adverses plus éloignés et à planifier des stratégies à long terme.

1. **Principes de l’heuristique utilisée et la stratégie choisie :**

L'heuristique choisie pour l'agent dans le jeu d'Abalone est basée sur une évaluation simple mais stratégique : elle calcule la différence entre le nombre de billes restantes pour le joueur et pour son adversaire. Cette approche se concentre sur un aspect crucial d'Abalone, où le nombre de billes sur le plateau est directement lié à la capacité du joueur à manœuvrer et à exercer un contrôle stratégique sur le jeu.

La fonction *evaluate*, qui implémente cette heuristique, compte les billes de chaque joueur et retourne la différence. Un score positif signifie un avantage numérique pour le joueur, tandis qu'un score négatif indique un avantage pour l'adversaire. Cette méthode d'évaluation présente plusieurs avantages :

* Simplicité et rapidité : La fonction est simple à comprendre et rapide à calculer, ce qui est essentiel pour un algorithme qui doit évaluer de nombreux états de jeu en peu de temps.
* Alignement avec l'objectif du jeu : Dans Abalone, pousser les billes de l'adversaire hors du plateau est l'objectif principal. Ainsi, avoir un plus grand nombre de billes est intrinsèquement avantageux.

Cependant, l'heuristique présente également des faiblesses :

* Manque de nuance stratégique : Elle ne prend pas en compte la position des billes sur le plateau. Dans Abalone, certaines configurations de billes sont plus avantageuses que d'autres, indépendamment de leur nombre.
* Pas de prise en compte des menaces imminentes : L'heuristique ne détecte pas les situations où un grand nombre de billes du joueur pourrait être vulnérable à être poussé hors du plateau dans les prochains tours.

Pour rendre l'heuristique plus efficace, plusieurs améliorations pourraient être envisagées :

* Évaluation de la position : Inclure une évaluation de la position des billes sur le plateau, favorisant des configurations stratégiquement fortes, comme le contrôle du centre ou la formation de structures qui protègent les billes contre les poussées adverses.
* Prise en compte des menaces et opportunités : Évaluer la vulnérabilité des billes du joueur et celles de l'adversaire à être poussées hors du plateau, ainsi que les opportunités de pousser les billes adverses dans les tours suivants.
* Pondération des billes en fonction de leur position : Donner plus de poids aux billes situées dans des positions stratégiquement importantes ou à celles qui sont en position de pousser des billes adverses hors du plateau.

En intégrant ces aspects, l'heuristique pourrait offrir une évaluation plus complète et stratégiquement informée de chaque état de jeu, conduisant à des décisions plus nuancées et potentiellement plus efficaces. Cela permettrait à l'agent d'anticiper et de répondre de manière plus sophistiquée aux développements du jeu, renforçant ainsi sa performance globale dans le jeu d'Abalone.

1. **Impact de la profondeur sur la performance et gestion du temps :**

L'agent, en se concentrant exclusivement sur une limite de profondeur fixe de 3 pour gérer le temps, adopte une approche simplifiée mais efficace. Cette stratégie, combinée à l’Alpha-Beta Pruning dans l'algorithme Minimax, optimise la gestion du temps en réduisant la charge computationnelle.

La limite de profondeur influence directement le temps de calcul. En se limitant à une profondeur de 3, l'agent trouve un équilibre entre profondeur de recherche et temps de réaction. Une profondeur plus élevée offrirait une meilleure anticipation des mouvements futurs mais au prix d'un temps de calcul beaucoup plus long. Inversement, une profondeur plus faible réduirait le temps de réponse mais au détriment de la qualité de la prise de décision. Une profondeur de 3 est donc un compromis qui permet d'éviter des calculs trop longs tout en offrant une anticipation raisonnable des coups à venir.

L’Alpha-Beta Pruning joue un rôle crucial dans la gestion du temps. Ce processus réduit efficacement l'espace de recherche en éliminant les branches de l'arbre de jeu qui ne nécessitent pas d'évaluation, car elles ne peuvent pas influencer le résultat final retenu par l'algorithme Minimax. Cette réduction de l'espace de recherche permet à l'agent de réaliser les calculs nécessaires plus rapidement, rendant l'utilisation d'une limite de profondeur fixe plus viable. Sans Alpha-Beta Pruning, même une profondeur de recherche modeste comme 3 pourrait conduire à des temps de calculs prohibitifs, car le nombre de scénarios à évaluer croît exponentiellement avec chaque niveau supplémentaire dans l'arbre de jeu.

Cependant, cette approche a ses limites. En se reposant uniquement sur une profondeur fixe et l’Alpha-Beta Pruning, l'agent ne dispose pas de la flexibilité nécessaire pour s'adapter à la complexité variable des différentes phases du jeu. Dans des situations de jeu moins complexes, où des mouvements évidents sont disponibles, une exploration profonde peut ne pas être nécessaire, alors que dans des scénarios plus complexes, une exploration plus approfondie pourrait être bénéfique. En ne tenant pas compte de ces variations, l'agent peut manquer des opportunités stratégiques dans des situations complexes ou perdre du temps dans des scénarios plus simples.

En résumé, la stratégie de gestion du temps de cet agent repose sur un équilibre entre une limite de profondeur fixe et l'efficacité de l’Alpha-Beta Pruning. Cette approche offre une méthode de prise de décision rapide et raisonnablement approfondie, adaptée à des parties où la rapidité est privilégiée. Toutefois, elle manque de la souplesse nécessaire pour s'adapter parfaitement à toutes les situations de jeu, ce qui pourrait être amélioré avec une gestion du temps plus dynamique.

1. **RÉSULTATS ET ÉVOLUTION DE L’AGENT :**

Lorsque l'on a commencé à développer le script pour le jeu d'Abalone, on a intégré plusieurs stratégies sophistiquées pour améliorer la gestion du temps et la performance globale de l'agent. Parmi ces stratégies, il y avait l'allocation d'un temps moyen par tour, basée sur le temps total de jeu divisé par un nombre estimé de tours. On a également envisagé d'augmenter progressivement la profondeur de recherche de l'algorithme Minimax tous les 10 tours, dans l'idée que les phases ultérieures du jeu nécessiteraient une analyse plus détaillée en raison de leur complexité croissante. En outre, on a utilisé une heuristique basée sur la distance des billes par rapport au centre du plateau, pensant que cela pourrait offrir un avantage stratégique significatif.

Cependant, au fil des tests et observations, il est devenu évident que ces méthodes initiales n'étaient pas aussi efficaces que prévu. On a constaté que la rigidité de la gestion du temps et l'augmentation programmée de la profondeur de recherche ne correspondaient pas toujours aux exigences spécifiques de chaque partie. En pratique, ces stratégies se sont avérées trop complexes et pas assez flexibles pour s'adapter aux dynamiques changeantes d'Abalone, un jeu connu pour sa profondeur stratégique et ses nombreuses possibilités.

Face à ces défis, on a opté pour une approche plus simplifiée. On a réalisé qu'une gestion du temps plus souple et une heuristique moins complexe pouvaient être plus efficaces. Cette stratégie révisée a pris en compte la nécessité d'adaptabilité et la capacité à répondre rapidement et intuitivement aux conditions de jeu changeantes, plutôt que de suivre un plan rigide et prédéterminé.

En fin de compte, la stratégie finale que l'on a adoptée a surpassé les méthodes initiales. Elle a non seulement simplifié la prise de décision, mais a également amélioré la réactivité de l'agent face à une variété de stratégies adverses. Cette expérience a souligné l'importance de l'adaptabilité en IA, en particulier dans un jeu aussi complexe et imprévisible qu'Abalone. Elle a également été un rappel précieux que, parfois, des solutions plus simples peuvent être plus efficaces, surtout lorsqu'elles permettent une plus grande flexibilité et une meilleure réactivité aux défis uniques de chaque partie.

1. **DISCUSSION**

Enlever des éléments de la méthodologie et les mettre ici.