Université Paris Cité

Rapport Final
Projet d'intelligence artificielle

Année : 2024-2025

Groupe N°3 :

GOURMELEN Thomas, PONNOUSSAMY Valentin

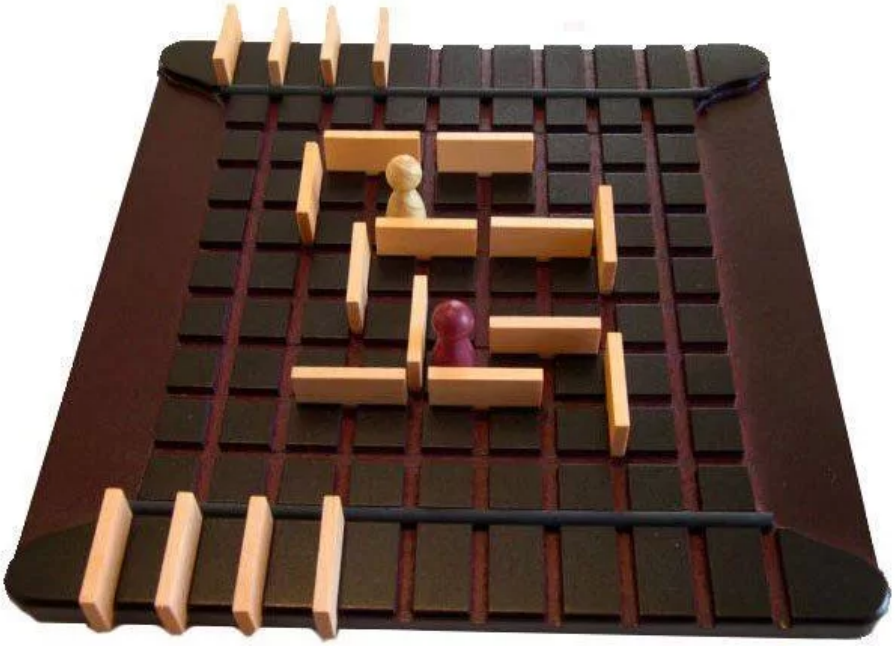
Enseignante : Elise Bonzon

1) Introduction.

Dans le cadre de notre projet en intelligence artificielle, nous avons pu développer le jeu Quoridor et des intelligence artificielles pour ce dernier. Afin d'être en accord avec les consignes du projet, notre choix c'est porté sur ce jeu à 2 joueurs et à connaissances parfaites. (Ce jeu fut également un cadeau d'anniversaire très apprécié…)

Quoridor est un jeu de stratégie combinatoire abstrait à 2 ou 4 joueurs. Dans notre projet nous nous sommes concentré sur la version à 2 joueurs.

Voici une représentation du jeux au cours d'une partie à 2 joueurs:



Sur cette image nous pouvons observer:

- Les pions des deux joueurs
- Les murs placés par les joueurs.
- Les murs en stock pour chaque joueur.

2) Description du jeu.

2.1) Règle du jeu.

2.1.1) But du jeu.

Le but du jeux est d'atteindre le premier la ligne opposée à sa ligne de départ. Le problème est que le joueur adverse peut poser des barrières pour ralentir l'avancée et inversement. Toutefois il est obligatoire de laisser un au moins un passage libre.

2.1.2) Comment jouer.

Les joueurs se partagent les barrières et chacun pose son pion au centre de sa ligne de départ. A tour de rôle, chacun déplace son pion d'une case, ou pose une barrière afin de ralentir l'adversaire. Les pions doivent contourner les barrières, qui créent un labyrinthe dont il faut sortir.

2.1.3) Règles.

Début de la partie :

Les barrières sont remisées dans leur zone de stockage (10 par joueur). Chaque joueur pose son pion au centre de sa ligne de départ (la première devant lui). Les 2 joueurs se font face. Un tirage au sort détermine qui commence.
A tour de rôle, chaque joueur choisit de déplacer son pion d'une case **ou** de poser une de ses barrières.

Déplacement des pions :

Les pions se déplacent d'une seule case, horizontalement ou verticalement, en avant ou en arrière, mais jamais en diagonale. Les barrières doivent être contournées et non sautées.

Face à face :

Quand les 2 pions se retrouvent en vis-à-vis sur 2 cases voisines non séparées par une barrière, le joueur dont c'est le tour peut sauter son adversaire. Si une barrière est située derrière le pion sauté, alors le pion sauteur devra être posé sur n'importe quelle autre case adjacente au pion sauté.

Pose des barrières :

Les barrières doivent être posées exactement entre 2 blocs de 2 cases. Elles ne peuvent pas se chevaucher. Une fois posées, elles ne peuvent plus être retirées ni bougées de la partie.

La pose des barrières a pour but de se créer son propre chemin ou de ralentir l'adversaire, mais il est interdit de lui fermer totalement l'accès à sa ligne d'arrivée : il faut toujours lui laisser une solution.

Lorsqu'un joueur n'a plus de barrières, il est obligé de déplacer son pion.

Fin de la partie :

Le premier joueur qui atteint une des 9 cases de la ligne opposée à sa ligne de départ gagne la partie.

2.2) Programme.

2.2.1) Mode de jeux :

Plusieurs modes de jeux sont possibles pour jouer :

- Humain *VS* Humain.
- Humain *VS* IA.
- IA *VS* IA.

Pour un joueur IA, il y'a 3 niveaux de difficultés qui correspondent a certaines valeurs de paramétrage :

- Niveau facile :
  - Profondeur : 1
  - Epsilon : 0,4
  - Poids avancer : 1
  - Poids bloquer : 0,5
  - Poids murs : 0,2

- Niveau moyen :
  - Profondeur : 2
  - Epsilon : 0,2
  - Poids avancer : 1,2
  - Poids bloquer : 0,8
  - Poids murs : 0,3

- Niveau difficile :
  - Profondeur : 3
  - Epsilon : 0,1
  - Poids avancer : 1,5
  - Poids bloquer : 1
  - Poids murs : 0,4

Nous avons choisi ce paramétrage suite à de nombreux tests de performance de l'IA avec différentes paramètres.

2.2.2) Déroulement d'une partie :

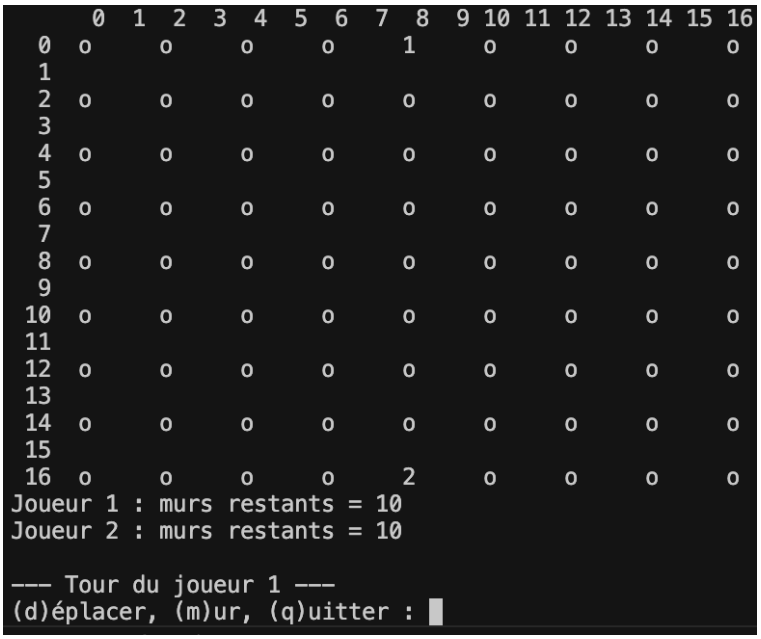
Avant le début de partie :

1. Au lancement du programme, l'utilisateur choisi son mode de jeux :

2. Si l'utilisateur choisi le mode de jeux 2 ou 3, il devra choisir le niveau de difficulté de l'IA ou des IAs :

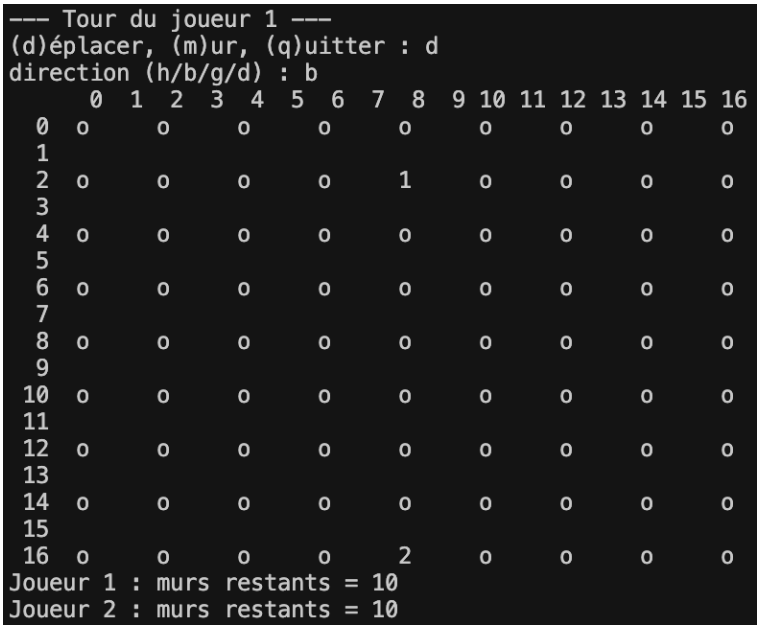
<u>Début de partie :</u> (ici nous prendrons en exemple le déroulement d'une partie Humain *VS* IA)

1. Affichage du plateau de jeux :



Quand la partie commence, les joueurs peuvent visualiser le plateaux de jeux, la position initial de leur pion ainsi que le nombre de murs en leur possession.

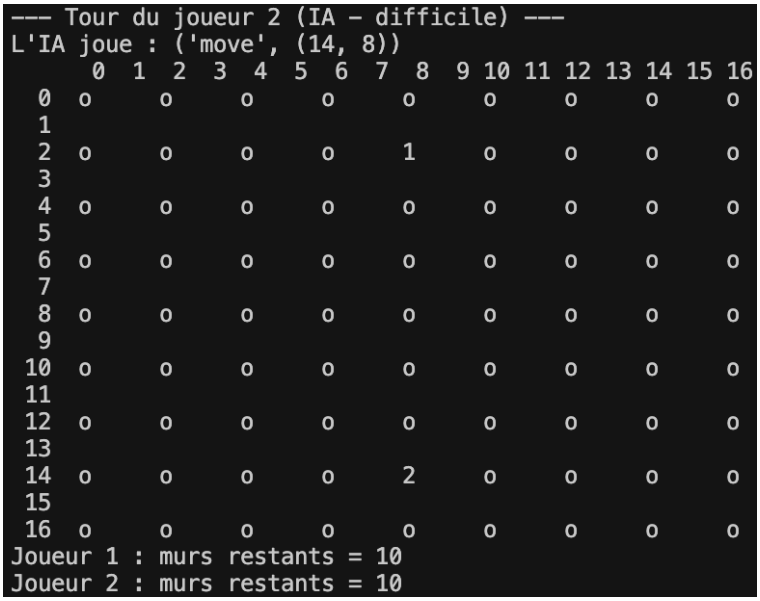
2. Action (déplacement) :



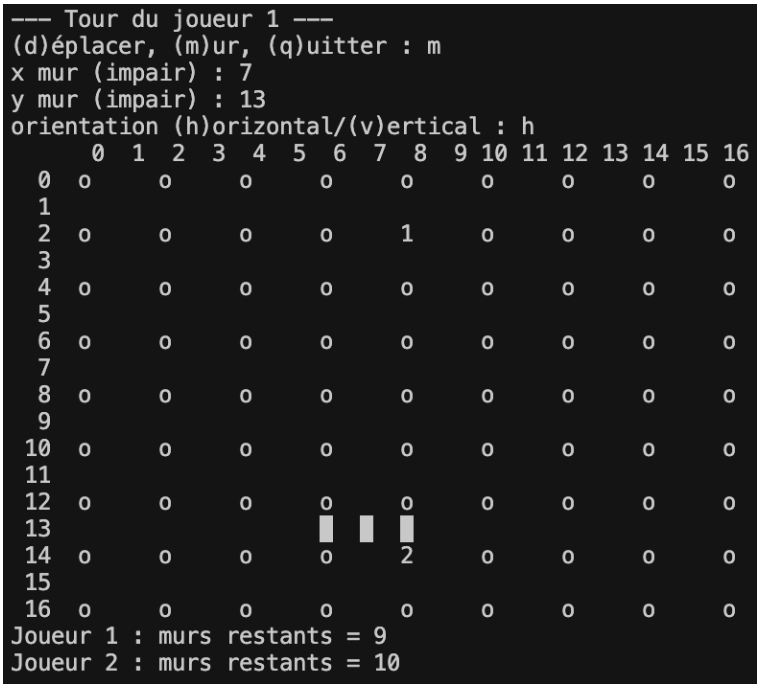
En choisissant l'action « se déplacer », l'utilisateur doit choisir la direction de son déplacement (haut, bas, gauche, droit).

(Dans notre exemple le joueur à choisi de faire un déplacement vers le bas.)

Le joueur IA a joué !!! Il faut absolument bloquer sa progression pour éviter qu'il ne gagne.



3. Action (placement d'un mur) :



En choisissant l'action « placement d'un mur », le jour doit renseigné la position x et y à quelle il souhait placer son mur. Il doit également préciser l'orientation du mur (horizontal, vertical).

(Ici le joueur a décider de placer un mur à la position x=7 et y=13. Nous pouvons remarqué que son nombre de murs restants a diminué.)

4. Action (déplacement au dessus d’un joueur) :

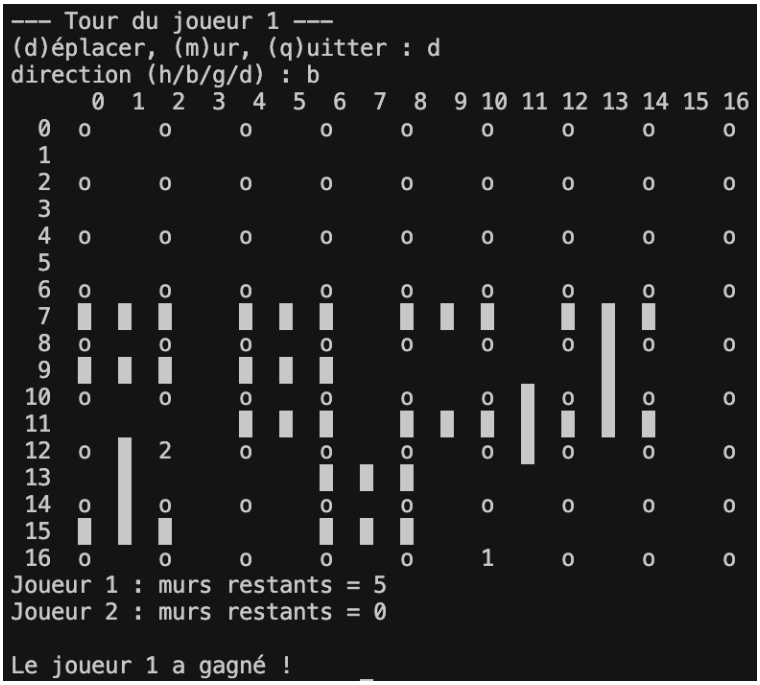
```
--- Tour du joueur 2 (IA - difficile) ---
L'IA joue : ('move', (8, 10))
      0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
  0  o     o     o     o     o     o     o     o     o
  1
  2  o     o     o     o     o     o     o     o     o
  3
  4  o     o     o     o     o     o     o     o     o
  5
  6  o     o     o     o     o     1     o     o     o
  7
  8  o     o     o     o     o     2     o     o     o
  9
 10  o     o     o     o     o     o     o     o     o
 11
 12  o     o     o     o     o     o     o     o     o
 13                  ▌  ▌  ▌
 14  o     o     o     o     o     o     o     o     o
 15
 16  o     o     o     o     o     o     o     o     o
Joueur 1 : murs restants = 9
Joueur 2 : murs restants = 10

--- Tour du joueur 1 ---
(d)éplacer, (m)ur, (q)uitter : d
direction (h/b/g/d) : b
      0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16
  0  o     o     o     o     o     o     o     o     o
  1
  2  o     o     o     o     o     o     o     o     o
  3
  4  o     o     o     o     o     o     o     o     o
  5
  6  o     o     o     o     o     o     o     o     o
  7
  8  o     o     o     o     o     2     o     o     o
  9
 10  o     o     o     o     o     1     o     o     o
 11
 12  o     o     o     o     o     o     o     o     o
 13                  ▌  ▌  ▌
 14  o     o     o     o     o     o     o     o     o
 15
 16  o     o     o     o     o     o     o     o     o
Joueur 1 : murs restants = 9
Joueur 2 : murs restants = 10
```

Dans cette situation, le joueur 1 peut décider, en effectuant un déplacement vers le bas, de sauté au-dessus du joueur adverse.

Le joueur 1 (humain) a sauté au dessus du joueur adverse lui faisant gagné 2 places.

Fin de partie :



Victoire le joueur 1 a gagné !!!

3) Description des IAs implémentées.

3.1) Architecture Générale

Notre implémentation du jeu Quoridor intègre plusieurs niveaux d'intelligence artificielle basés sur l'algorithme Minimax avec élagage Alpha-Beta. Chaque IA utilise une fonction d'évaluation paramétrable et une table de transposition pour optimiser les performances.

3.2) Algorithme Minimax avec élagage Alpha-Beta

L'algorithme Minimax est l'algorithme que nous avons utilisé pour l'IA. Il permet d'explorer l'arbre des possibilités de jeu jusqu'à une certaine profondeur et d'évaluer les positions de ces possibilités de jeu.
Nous avons optimisé Minimax grâce à l'élagage Alpha-Beta, ce dernier permet de réduire le nombre de nœuds explorés en évitant l'évaluation des branches qui ne peuvent pas influencer le résultat finale.

Notre implémentation présente plusieurs caractéristiques notables :

- Profondeur de recherche variable selon le niveau de difficulté
- Élagage Alpha-Beta pour optimiser les performances
- Table de transposition pour mémoriser les états déjà évalués et éviter les calculs redondants
- Paramètres ajustables pour personnaliser le comportement de l'IA

3.3) Fonction d’évaluation

La fonction d'évaluation permet de déterminer la qualité ou non d'une position. Notre heuristique prend en compte plusieurs facteurs :

1. **Distance à l'objectif** (poids_avancer) : Calcule la distance entre le joueur et la ligne d'arrivée. Plus la distance est faible, meilleur est le score.

2. **Blocage de l'adversaire** (poids_bloquer) : Valorise les positions où l'adversaire est loin de son objectif.

3. **Gestion des murs** (poids_murs) : Considère l'avantage relatif en termes de murs restants.

4. **Avance relative** (poids_avance) : Bonus accordé lorsque le joueur est plus proche de son objectif que l'adversaire de son propre objectif.

3.4) Niveaux de Difficulté

Il nous a été demandé dans le cadre de ce projet de créer 3 niveaux de difficulté que voici :

Niveau Facile :
- Profondeur de recherche limitée à 1.
- Epsilon élevé (0.4) favorisant l'exploration aléatoire.
- Poids modérés mettant en valeur le déplacement direct vers l’objectif.
- Faible considération pour le blocage de l’adversaire.

Niveau Moyen :
- Profondeur de recherche de 2.
- Epsilon modéré (0.2) permettant de trouvé un équilibre entre l’aléatoire et la recherche.
- Poids équilibrés entre progression et blocage.
- Meilleure utilisation de la stratégique des murs.

Niveau Difficile :
- Profondeur de recherche de 3.
- Faible epsilon (0.1) mettant en avant les coups optimaux.
- Valeur forte pour bloquer l'adversaire et progresser.
- Utilisation très stratégique des murs.

3.4) Optimisations Techniques

Plusieurs optimisations ont été implémentées pour améliorer les performances :

1. **Table de transposition** : Stocke les états déjà évalués pour éviter les calculs redondants
2. **Génération stratégique des coups** : Priorise les positions de murs les plus prometteuses
3. **Détection précoce des victoires** : Arrête la recherche si une victoire immédiate est possible
4. **Algorithme BFS optimisé** : Calcule efficacement les plus courts chemins vers l’objectif

3.4) Comportement Adaptatif

L'IA intègre un paramètre epsilon qui introduit une part d'aléatoire dans ses décisions, permettant d'explorer des stratégies différentes et d'éviter un comportement trop prévisible. Ce paramètre diminue avec le niveau de difficulté, rendant l'IA plus déterministe à mesure que la difficulté augmente.

4) Résultat des tournois.

5) Bilan du projet.

6) Références.

Rapport :

- ***ChatGPT 4o*** : Pour la rédaction de ce rapport, nous avons utilisez ChatGPT afin de nous aider pour la rédaction, la reformulation et la correction des fautes d'orthographes.
- **Lien**: Nous nous sommes également appuyer sur les règles du jeux Quoridor https://jeux-casse-tete.com/blog/regles-de-jeux/regle-du-jeu-quoridor

Développement :

- ***ChatGPT 4o mini-hight & Claude 3.7 Sonnet*** : Pour le développement de notre projet nous avons utilisé à certains moment des LLM afin de nous aidez dans le développement :
  - Optimisation : *Claude 3.7 Sonnet* nous a aider a ajouté une « Table de transposition » nous permettant de garder en mémoire les états déjà évalué pour éviter de recalculer plusieurs fois les mêmes positions. Mais également la mise en cache du hash d'état, pour éviter de recalculer le hash d'un état à chaque appel
  - Correction de bugs : ChatGPT 4.o mini-hight nous a beaucoup aidé pour la compréhension de nombreux bugs que nous avons eu.
  - Compréhension : Les deux LLMs nous on également aidés à comprendre certains comportement des Ins que nous avons développé.