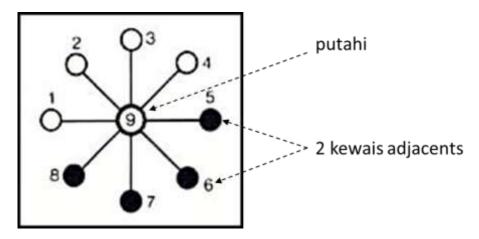
Mini-Projet d'Intelligence Artificielle :

Apprendre à jouer au Mu Torere par un apprentissage par renforcement

Le Mu Torere est un jeu dont les origines remontent aux tribus Maoris de Nouvelle-Zélande. C'est un jeu de stratégie basé sur un nombre faible de pion (4 pions par joueur) avec 9 emplacements.



- 1. La figure ci-dessus indique la position initiale des pièces (le trou central "putahi" est vide)
- 2. Les noirs commencent et on joue à tour de rôle.
- 3. On peut déplacer un pion:
 - a) d'un kewai vers le putahi, mais seulement si l'une ou les deux cases contiguës du kewai sont occupées par un ou deux pions adverses;
 - b) du putahi vers un kewai.
 - c) d'un kewai vers un kewai adjacent.
- 4. Une case ne peut accueillir qu'un seul pion à la fois.
- 5. Un pion n'a pas le droit de sauter au dessus un autre pion qu'il soit ami ou ennemi.
- 6. Pour gagner il faut bloquer tous les pions adverses en sorte qu'ils se trouvent dans l'impossibilité de se déplacer.

Exemple de partie:

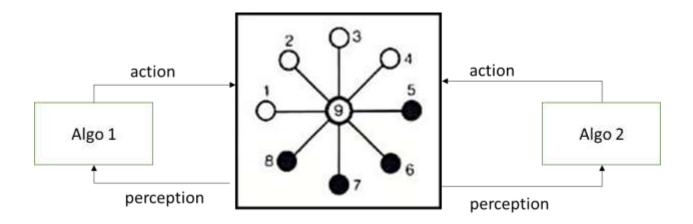
- **1.** 5-9, 4-5;
- **2.** 9-4, 3-9;
- **3.** 4-3, 9-4;
- **4.** 3-9, 2-3;
- **5.** 9-2, 1-9;
- **6.** 2-1, 3-2;

Les blancs ont gagné.

Voir une explication en vidéo ici: https://www.youtube.com/watch?v=zxlB6CqeE7E

Objectif:

L'objectif du projet est de faire un sorte qu'un programme apprenne à jouer (et à gagner) en s'entrainant contre lui-même grâce à un algorithme de renforcement. Si on se base sur le code fournit durant le TP (ce qui est une possibilité mais pas une obligation), il faut considérer que l'on a en fait 2 agents (et donc 2 algorithmes) qui font des parties (voir figure ci-dessous).



Ainsi l'état perçu par l'algo est simplement l'état du plateau de MU Torere dans que les actions sont au nombre de 9 (placer un pion à une place vide). Le renforcement est obtenu lorsque l'agent a gagné (renforcement positif) ou perdu (renforcement négatif). Attention, il y a une différence importante avec le problème du labyrinthe vu en TP: Lors que l'agent du labyrinthe fonce dans un mur, il reçoit un renforcement négatif et n'est pas déplacé. Dans un jeu à deux, ce cas pourrait correspondre à un coup que l'agent n'a pas le droit de faire (à cause des règles). Là il y a deux stratégies :

- 1) Soit on considère que l'agent connait les règles et élimine parmi les 9 actions possibles celles qui ne peuvent pas avoir lieu (à cause des règles). Ceci implique que l'apprentissage sera celui de la stratégie. Il faut néanmoins définir les actions possibles avant d'en faire sélectionner une par l'algo.
- 2) Soit on considère que l'agent ne connait pas les règles et qu'il va les apprendre (ce qui est assez intéressant aussi). Pour cela, il faut quand même veiller à respecter le fait qu'un coup interdit soit 'puni' et surtout, ne soit pas joué (donc si un agent propose un coup interdit le plateau de jeu n'évolue pas mais surtout, cet agent doit proposer à nouveau un coup et ce tant qu'il ne propose pas de coup autorisé.

Dans les deux cas, le code du SarsaSituatedAgent peut être une bonne base. Il faut créer deux agents et revoir toute la partie « perception / action » qui se trouve en fait dans la classe mère « SituatedAgent » (remplacer la variable « _maze » par un « _muTorere » et les 4 actions de déplacements par les 9 coups possibles.

Contact: deloor@enib.fr

Références:

Deux articles sur l'usage de l'apprentissage par renforcement pour les jeux

https://www.researchgate.net/publication/331938898 Reinforcement Learning for Real-Time Strategy games

http://107.167.189.191/~vishnu/gameResearch/Al/Ghory04%20RL%20in%20board%20game.pdf

Deux pages sur le Mu Torere

http://www.lecomptoirdesjeux.com/regle-mutorere-nouvelle-zelande.htm

http://jeuxstrategieter.free.fr/Mu_torere_complet.php