Vincent Beaudoin (111 103 778)

Alexandre Picard-Lemieux (111 103 625)

Gabriel Legault ()

Clément Spies ()

Intelligence artificielle I

IFT-2003

TP #2

Concevoir un jeu en utilisant l'IA

Travail présenté à

Laurence Capus

Département d’informatique et de génie logiciel

Université Laval

Hiver 2016

# Introduction

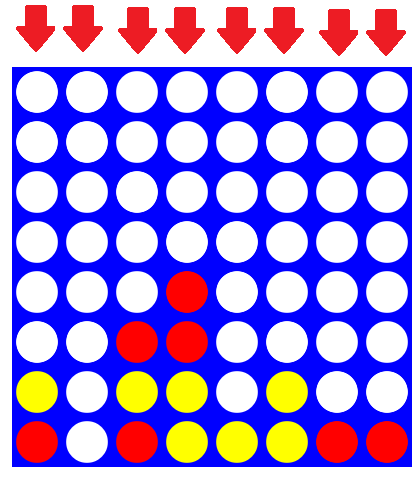
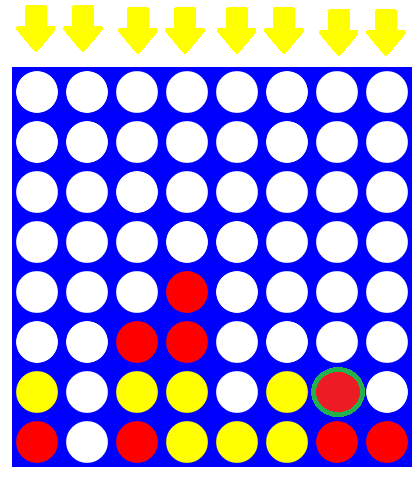
L’intelligence artificielle est utilisée dans de nombreuses applications. Dans ce travail, elle sera utilisée pour créer un adversaire dans le jeu de Puissance 4. Nous voulons donc analyser ce jeu et créer un joueur intelligent à l’aide du langage Prolog. Pour cela, il sera utilisé la démarche de résolution de recherche intelligente par espace d’états.

# Description du jeu choisi (10%)[[1]](#footnote-1)[[2]](#footnote-2)

## Présentation du jeu

Puissance 4 est un jeu de stratégie commercialisé en 1974. Le but du jeu est de faire des connexions de quatre jetons dans une grille de sept colonnes et de 6 rangées. Dans cet exemple, le jeu possède 8 colonnes et 8 rangées.

Voici quelques exemples du jeu:

Grille où c’est au tour du joueur rouge Grille où c'est au tour du joueur jaune

## Description des règles

Chaque joueur possède vingt-et-un jetons d’une couleur différente. Ces couleurs sont en général jaune et rouge. Les joueurs placent un jeton dans une colonne à chaque tour. La pièce va se placer à la rangée la plus basse qu’elle peut aller. Un joueur ne peut pas placer de jeton dans une colonne pleine. Pour gagner, il faut faire un alignement horizontal, vertical ou diagonal de quatre jetons de sa couleur. La partie est déclarée nulle s’il est impossible de mettre d’autres jetons et qu’il n’y a aucun gagnant.

Voici quelques illustrations :



Grille avec les positions gagnantes Grille avec les positions gagnantes

# Modélisation du problème et de la solution (30%)

## Description du problème

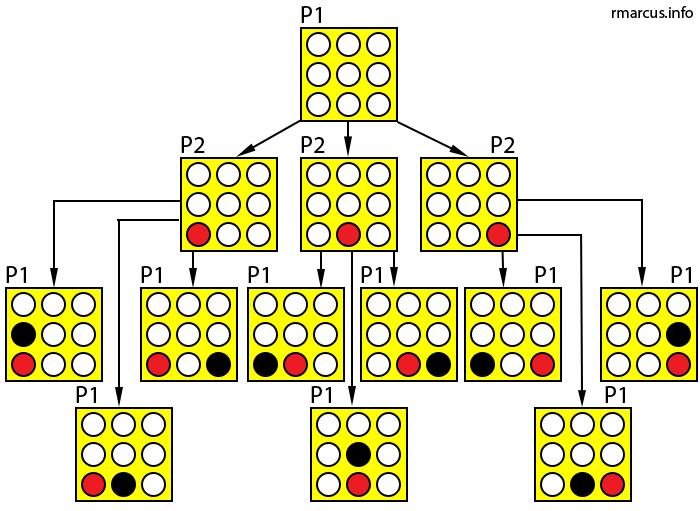
L’espace d’états de ce problème est décrit par l’ensemble des grilles m x n, possédant des cases vides, des cases avec des jetons rouges et des cases avec des jetons jaunes. Dans le jeu original, m = 6 et n = 7. Cela signifie donc que chaque état est l’ajout d’un jeton potentiel et chaque nœud des profondeurs est un tour.

L’état initial est la grille vide. Il n’y a pas de jetons d’ajouter encore, car le jeu n’est pas encore commencé. L’état final est la grille possédant une connexion de quatre jetons ou une grille pleine de jetons. La partie est donc terminée puisqu’il y a un gagnant ou la partie est nulle.

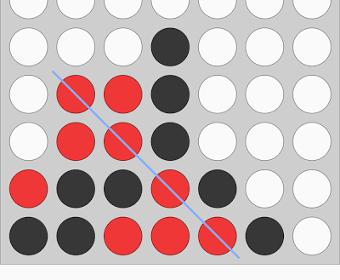
L’état initial peut aussi être rempli avec des jetons si on refait le calcul en plein milieu de la partie. L’état final peut aussi être un jeu non terminé si on limite la profondeur de l’espace d’état.

Les mouvements autorisés sont Ci qui signifie qu’on ajoute un jeton dans la colonne i. Il y a donc n mouvements autorisés par état. Il est possible d’en avoir moins s’il y a des colonnes pleines.

Par exemple, voici l’espace d’état pour une profondeur de 3 avec une grille 3x3[[3]](#footnote-3) :



L’état initial est la racine de cet arbre. Voici un état final potentiel :



## Description de la solution[[4]](#footnote-4)

La procédure minimax sera utilisée pour créer l’adversaire. On nomme les deux joueurs MIN et MAX. MAX représente le joueur qui veut gagner et MIN représente le joueur qui veut empêcher MAX de gagner. Les deux joueurs possèdent les mêmes connaissances concernant l’espace d’état.

Cette procédure consiste à descendre jusqu’à une profondeur donnée et calculer la valeur heuristique de chacun de ces nœuds. Le parent prend la valeur MAX ou MIN de ses enfants tout dépendamment du parent.

Voici un exemple d’un arbre qui utilise cet algorithme. Dans cet exemple, MAX choisit donc son deuxième enfant après le calcul des valeurs.



La fonction heuristique est donc :

01 **function** minimax(node, depth, maximizingPlayer)

02 **if** depth = 0 **or** node is a terminal node

03 **return** the heuristic value of node

04 **if** maximizingPlayer

05 bestValue := −∞

06 **for each** child of node

07 v := minimax(child, depth − 1, FALSE)

08 bestValue := max(bestValue, v)

09 **return** bestValue

10 **else** *(\* minimizing player \*)*

11 bestValue := +∞

12 **for each** child of node

13 v := minimax(child, depth − 1, TRUE)

14 bestValue := min(bestValue, v)

15 **return** bestValue

*(\* Initial call for maximizing player \*)*

minimax(origin, depth, TRUE)

La valeur heuristique est calculée de manière très simple :

* Si c’est une victoire (4 jetons connectés de MAX), on retourne 10000;
* Si c’est une défaite (4 jetons connectés de MIN), on retourne -10000;
* Si c’est nul (la grille est pleine), on retourne 0;
* Si la partie est encore en cours (grille non pleine sans victoire), on utilise 10 \* (nombre de positions qui mènent à une victoire de MAX - nombre de positions qui mènent à une victoire de MIN) + nombre de positions qui mènent à une connexion de 3 jetons de MAX – nombre de positions qui mènent à une connexion de 3 jetons de MIN.

Il faut aussi considérer que « XOX » et « XXOX » sont aussi des positions possibles pour O. Par exemple, voici les valeurs heuristiques pour ces deux grilles si MAX est jaune. Dans ces exemples, les étoiles représentent les positions pour des connexions possibles de 4 jetons.



Valeur heuristique : -23 Valeur heuristique : 29

La deuxième grille serait donc plus préférable si le jaune qui a mis son dernier jeton et si les deux grilles étaient à la même profondeur.

# Implantation (30%)

# Résultats et discussion (20%)

# Conclusion

# Bibliographie

Wikipédia. [En ligne]. <https://fr.wikipedia.org/wiki/Puissance_4> (Page consultée le 27 février 2016)

Roadtolarissa. [En ligne]. <http://roadtolarissa.com/connect-4-ai-how-it-works/> (Page consultée le 27 février 2016)

RMarcus. [En ligne]. <https://rmarcus.info/blog/2014/12/23/connect4.html> (Page consultée le 27 février 2016)

Wikipedia. [En ligne]. <https://en.wikipedia.org/wiki/Minimax> (Page consultée le 27 février 2016)

1. Wikipédia. [En ligne]. <https://fr.wikipedia.org/wiki/Puissance_4> (Page consultée le 27 février 2016) [↑](#footnote-ref-1)
2. Roadtolarissa. [En ligne]. <http://roadtolarissa.com/connect-4-ai-how-it-works/> (Page consultée le 27 février 2016) [↑](#footnote-ref-2)
3. RMarcus. [En ligne]. <https://rmarcus.info/blog/2014/12/23/connect4.html> (Page consultée le 27 février 2016) [↑](#footnote-ref-3)
4. Wikipedia. [En ligne]. <https://en.wikipedia.org/wiki/Minimax> (Page consultée le 27 février 2016) [↑](#footnote-ref-4)