



*Le 11 mars 2018*

# **Conception d'un jeu intelligent**

*TP2 conception d'un jeu intelligent IFT-2003*

**Appoline Bibie Yamatchui Kamaha (111 096 135)**

**Paul Pilote (111 104 263)**

**Éric Cogoluenhès (111 162 638)**

# **1. Introduction**

## **1.1.Motivation**

Jouer est une activité qui procure du plaisir et permet parfois de relaxer. Beaucoup de jeux sont en duel, c'est-à-dire qu'ils se jouent à deux avec un partenaire physique. Il existe des jeux dont le support est numérique et dont l'adversaire de jeu est remplacé par l'ordinateur. On utilise l'intelligence artificielle pour simuler l'adversaire dans les jeux numériques en duel.

Nous citerons en exemple les jeux comme Puissance 4, le jeu de go, les échecs etc.

Dans ce travail nous nous intéresserons à la conception et la réalisation du jeu Puissance 4 avec le langage Prolog.

## **1.2.Objectif**

Dans le cadre du cours IFT-2003 d'introduction à l'intelligence artificielle, notre travail consiste à :

- la conception d'un jeu avec une technique d'intelligence artificielle,
- la modélisation de ce jeu en espaces d'états et
- à l'implémentation de ce jeu en langage Prolog.

Dans une première partie, nous présenterons le jeu puissance 4, qui fait l'objet de notre étude. Puis, nous modéliserons le jeu Puissance 4 sous forme d'un espace d'états, en expliquant la technique de recherche et les fonctions heuristiques utilisées. Alors, il sera possible d'implanter la solution en langage Prolog. Enfin; nous testerons notre programme afin de vérifier qu'il répond à ce que nous nous étions fixé comme objectifs.

## **1.3.Ce que nous avons voulu faire**

Nous avons voulu réaliser un jeu de puissance 4 avec un joueur et une intelligence artificielle (IA). Les gains se font lorsque 4 jetons de même couleur sont alignés horizontalement, verticalement ou diagonalement.

## **1.4.Ce que nous avons vraiment réalisé**

Nous avons réalisé un jeu de puissance 4 avec un joueur et une IA. Les possibilités de gain que nous avons obtenu sont : 4 jetons de même couleur alignés horizontalement et verticalement.

## 2. Description du jeu puissance 4

### 2.1.Principe du jeu

Le jeu Puissance 4 est un jeu de stratégie qui se joue à deux. Dans notre cas, l'un des adversaires est l'ordinateur. Le but du jeu est d'aligner des jetons de même couleur. Les jetons d'un joueur sont tous de la même couleur, et les joueurs ont des jetons de couleur différentes de ceux de l'autre. L'alignement des jetons peut être faite à la verticale, à l'horizontal ou en diagonale. Un joueur gagne quand il parvient à aligner quatre jetons selon l'une des positions citées précédemment.

### 2.2.Description du jeu

Le jeu est composé d'une grille de 7 colonnes et 6 lignes. Initialement, la grille est vide.

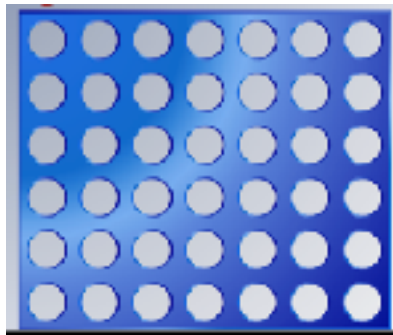


FIGURE 2.11: GRILLE INITIALE D'UN JEU PUISSANCE DE 4

À tour de rôle, chaque joueur doit mettre un jeton de sa couleur (rouge ou jaune) dans la grille. Le joueur jaune joue toujours en premier. Le joueur sera considéré comme gagnant s'il aligne quatre de ses jetons verticalement, horizontalement ou diagonalement.

---

<sup>1</sup>Image de la grille provient du site <http://www.stratozor.com/puissance-4/>



*Figure 2.2<sup>2</sup> : EXEMPLE DE GRILLE FINALE GAGNANTE POUR LE JOUEUR ROUGE*

Pour placer un pion, le joueur doit l'insérer dans la colonne choisie. Le jeton glissera alors jusqu'à la case la plus basse inoccupée. Un placement de pion est défini par sa ligne et sa colonne. Une case n'est jamais accessible directement.

Si la grille est remplie complètement, de telle sorte qu'aucun autre jeton ne peut être inséré avant qu'un joueur n'aie rencontré une condition de victoire, alors la partie sera considérée comme nulle.

---

<sup>2</sup> Image de la grille provient du site <http://www.stratozor.com/puissance-4/>

### 3. Modélisation

L'espace d'état de ce problème est décrit par l'ensemble des sept listes  $\{L1, L2, L3, L4, L5, L6, L7\}$ , tel que chacune d'elle ne peut contenir au maximum que 6 éléments. Chaque élément de chaque liste ne pourra avoir que trois valeurs : vide, R pour un jeton rouge et J pour un jeton jaune.

L'état initial est  $\{[ ], [ ], [ ], [ ], [ ], [ ], [ ]\}$ . La grille est vide au commencement du jeu.

Les états finaux sont tous les états avec quatre jetons de même couleur alignés que ce soit verticalement, horizontalement ou diagonalement ou un état où la grille est pleine sans vainqueur. Il en existe trois par colonne ( $3 * 7 = 21$ ), quatre par ligne ( $4 * 6 = 24$ ) et douze en diagonale. Il existe donc 57 cas possibles de victoire par joueur et beaucoup plus d'états permettant de les modéliser. De même, il existe beaucoup trop d'états différents représentant une grille pleine pour tous les modéliser dans ce travail.

Les opérateurs à utiliser pour résoudre le problème sont les suivants :

#	Opération	Nom opérateur	Conditions
1	$L_n \rightarrow [R L_n]$	Mettre un jeton R (rouge) dans la colonne n	la colonne n ( $L_n$ ) n'est pas pleine (ne contient pas 6 éléments avant de jouer)
2	$L_n \rightarrow [J L_n]$	Mettre un jeton J (jaune) dans la colonne n	la colonne n ( $L_n$ ) n'est pas pleine (ne contient pas 6 éléments avant de jouer)

La fonction heuristique utilisée permet de trouver la valeur heuristique des états, ce qui permet à l'ordinateur de prendre des décisions sur les coups qu'il jouera. Plus la valeur est basse, plus les chances de gagner sont grandes. Si l'état est gagnant, on retourne 0.

La technique de recherche heuristique que nous allons utiliser est le Minimax, dans laquelle Max tente de gagner sauf s'il ne peut gagner au coup suivant et que l'adversaire est sur le point de gagner au coup suivant. Dans ce cas précis, il empêchera l'adversaire de gagner.

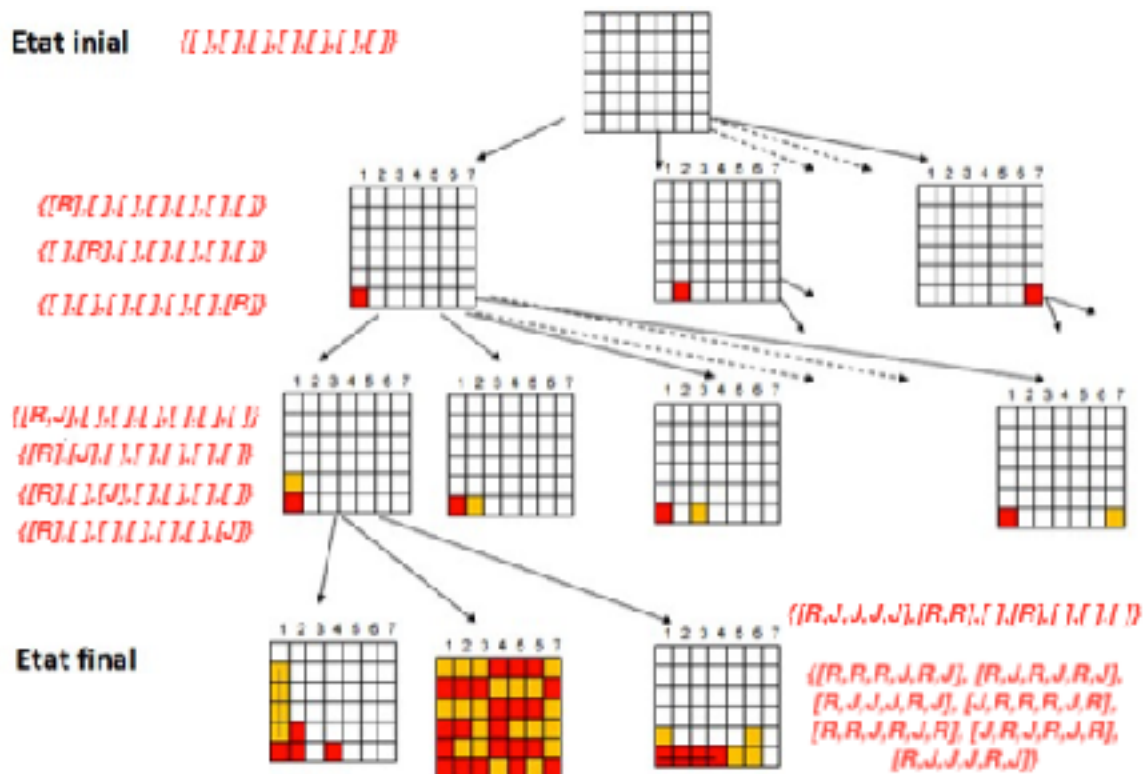


FIGURE 3 : MODÉLISATION DES ÉTATS

### 3.1 Guide d'utilisation du programme Prolog

?- jouer.

```

X X X X X X X X
X X X X X X X X
X X X X X X X X
X X X X X X X X
X X X X X X X X
X X X X X X X X

```

- La question à poser est : « ?- jouer. ».
- Le joueur a les jetons jaunes et l'IA a les jetons rouges.

|: 6.

```

X X X X X X X X
X X X X X X X X
X X X X X X X X
X X X X X X X X
X X X X X X X X
X X X X X X X X

```

- Ensuite il faut choisir la colonne où mettre le jeton en entrant un entier compris entre 1 et 7, suivi de « . ». Par exemple : « |: 6. ».

```

X X X X X X X X
X X X X X X X X
X X X X X X X X
X X X X X X X X
X X X X X X X X
X X X X X X X X

```

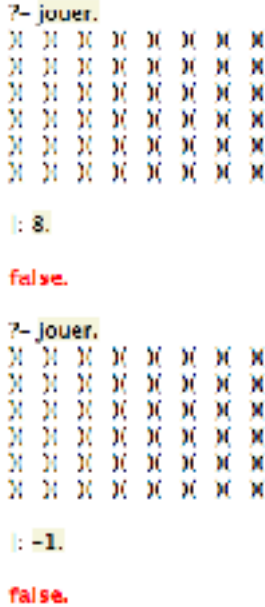
- L'intelligence artificielle joue ensuite son coup.
- Puis, c'est à nouveau au joueur de jouer en saisissant un entier compris entre 1 et 7, suivi de « . ».

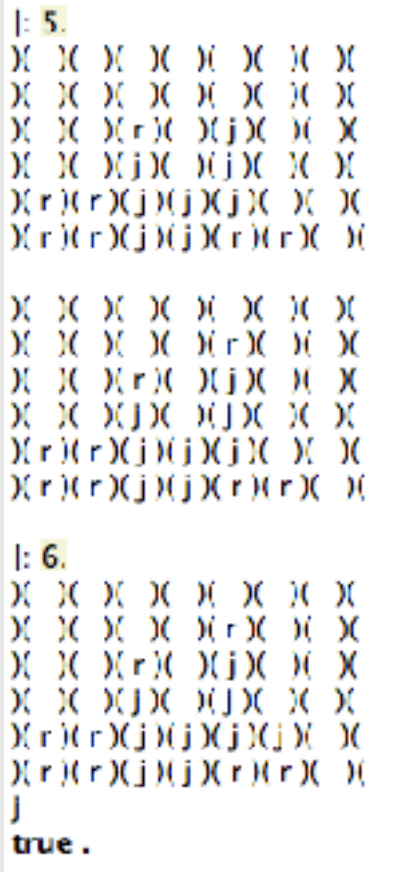
FIGURE 3.1 : EXEMPLE DE DÉBUT DE PARTIE

|:

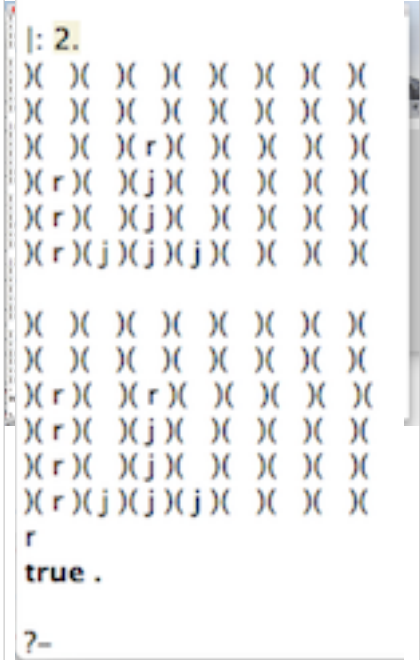
## 4. Discussion sur les résultats obtenus



### 4.1 Tests réalisés

Définition du test réalisé	Copie écran	Commentaires
Ce test consiste à valider que le joueur ne peut jouer que dans les colonnes comprises entre 1 et 7 inclus.		Nous voyons bien sur l'image ci-contre que si le joueur essaie de jouer en « 8 » ou en « -1 », cela ne fonctionne pas et arrête le jeu : « false ».

Définition du test réalisé	Copie écran	Commentaires
Ce test consiste à valider que le joueur peut gagner. On vérifie par la même occasion que la partie peut être gagnée avec 4 jetons alignés horizontalement.	 <p>I: 5.</p> <p>true .</p>	Nous pouvons constater sur l'image tirée de Prolog ci-contre, qu'en jouant son jeton jaune en colonne 6, le joueur a réussi à aligner 4 jetons horizontalement et a ainsi gagné la partie.



Définition du test réalisé	Copie écran	Commentaires
<p>Ce test consiste à valider que l'intelligence artificielle (IA) peut gagner. On vérifie par la même occasion que la partie peut être gagnée avec 4 jetons alignés verticalement.</p>	 <pre> : 2. r true . ?- </pre>	<p>Nous voyons sur l'image ci-contre provenant de Prolog que l'IA, en jouant son jeton rouge à la colonne 1, a réussi à aligner 4 jetons verticalement et à gagner la partie.</p>

Définition du test réalisé	Copie écran	Commentaires
Ce test consiste à valider que la partie est nulle lorsque la grille est pleine et qu'il n'y a pas de gagnant.		<p>Nous pouvons constater que lorsque l'ensemble de la grille est remplie, la partie s'arrête sans vainqueur (« _81053 » remplace « r » ou « j » selon qui aurait pu gagner et nous indique ainsi une partie nulle).</p> <p>Ce test a aussi permis de vérifier que le joueur IA ne jouait que dans les colonnes où il y était autorisé, c'est à dire les colonnes non pleines.</p>
Ce test consiste a vérifier que la partie peut être gagnée avec 4 jetons alignés diagonalement.		<p>Nous pouvons constater sur l'image ci-contre, directement tirée de Prolog, que le joueur jaune avait déjà aligné 4 jetons jaunes diagonalement entre les colonnes 1 et 4 et les lignes 4 à 1 (de haut vers le bas). Mais notre programme n'a pas considéré cela comme un coup gagnant. En effet, nous n'avons pas eu le temps de développer la fonction qui valide le coup gagnant avec 4 jetons alignés diagonalement.</p>

## **4.2. Les buts fixés sont-ils atteints ?**

Nous n'avons pas réussi à tester la fonction heuristique que nous voulions mettre en place. La fonction est écrite dans le programme, mais n'est pas utilisée par celui-ci. Le programme utilise une technique aléatoire, excepté si le joueur est sur le point de gagner, dans ce cas, l'IA bloquera le joueur.

Cela est essentiellement dû au fait que nous nous étions chacun assigné des bouts de code à développer et que lorsque l'on a voulu les mettre ensemble, il était trop tard, nous n'avions plus le temps, pour réussir à bien connecter la fonction heuristique que nous voulions faire et le reste du code.

De plus, le manque de temps nous a aussi empêché de finir de développer la fonction pour valider un état gagnant avec 4 jetons alignés diagonalement.

## 5. Conclusion

Dans ce travail, nous voulions réaliser un jeu de Puissance 4 en Prolog qui puisse être joué par un joueur contre une intelligence artificielle (IA). Pour cela, nous souhaitons utiliser la technique de recherche heuristique Minimax.

Après avoir modélisé ce que nous voulions faire, nous avons essayé de le coder en Prolog. Pour cela, nous nous sommes réparti les bouts de code à réaliser. Par manque de temps, nous n'avons pas pu connecter la fonction heuristique que nous voulions mettre en place afin de pouvoir tester si elle était bien codée, ni finir de coder la fonction d'état gagnant diagonal. Nous aurions voulu finir proprement de développer et implémenter toutes les parties que nous avions prévu.