IA41 – Teeko

Rapport de projet

Barbara SCHIAVI
Paul-Émile BRETEGNIER

Printemps 2015

Introduction

Dans l'UV IA41 nous sommes ammenés a réalisé un projet de programmation en Prolog afin d'utiliser les connaissances acquises dans l'unité de valeur ce semestre. Nous avons choisi le jeu de plateau Teeko dont nous détaillerons les règles par la suite.

Dans ce rapport, nous regrouperons notre analyse du problème, la solution que nous y apportons, ainsi que l'ensemble des difficultés rencontrées, sans oublier les parties clefs du code.

Le jeu Teeko est un jeu pour deux joueurs, sur un plateau carré de 25 cases. Le jeu se déroule en une à deux phases successives. L'objectif de chaque joueur est d'aligner successivement ses 4 pions sur une ligne, une colonne ou une diagonale ou de faire un carré (2*2). Lors de la première phase, chaque joueur dépose chacun son tour un de ces pions. Si aucun vainqueur n'émerge après que chacun d'entre eux ait posé ses 4 pions la seconde phase commence. À ce moment là, toujours à tour de rôle, les joueurs doivent déplacer leurs pions afin de réaliser une des conditions de victoire. Un pion peut uniquement être déplacé sur une case vide adjacente à sa position de départ au début du tour.

Sommaire

Introduction....2
1 Formalisation du problème....4
1.1 Utilisation de l'algorithme min/max....4
1.2 Utilisation d'alpha/bêta pour élaguer....4
2 Représentation des données....5
2.1 Plateau de jeu....5
2.2 Joueurs et pions....5
2.3 Conditions de victoire....5
3 Placement des pions....7
3.1 Lors de la phase initiale....7
3.2 Déplacement dans la seconde phase....7
4 Problèmes rencontrés....7
4.1 Définition du plateau....7
4.2 Déplacement dans la seconde phase....7

1 Formalisation du problème

1.1 Utilisation de l'algorithme min/max

Pour programmer l'Intelligence Artificielle, nous avons dû chercher l'algorithme le plus adapté pour trouver les meilleurs coups. L'algorithme min/max, présenté en cours, analyse l'ensemble des coups possibles, et détermine le meilleur coup. De plus teeko, répond aux critères pour l'utilisation de l'algorithme. C'est un jeu asynchrone à deux joueurs, à somme nulle et dont l'information est complète et parfaite.

L'algorithme min/max génère l'arbre de jeu à partir de la situation actuelle et crée une branche fille, pour chaque coup possible. Chaque nœud fils génère une nouvelle série de nœud fille pour chaque nouveau coup possible, et ainsi de suite. Jusqu'à la victoire de chaque branche. L'arbre est donc caractérisé par sa profondeur **n** qui représente le nombre de coups faits. Le meilleur chemin est celui où le joueur minimise ses pertes sachant que l'adversaire tente de les maximiser (principe du jeu à somme nulle).

Pour évaluer un coup, il nous faut une fonction d'évaluation permettant de savoir si c'est un bon coup ou non et dans quelle mesure.

Dans l'exemple si dessous, le nœud racine est un nœud max. Ses fils sont des nœuds min, leurs fils, des nœuds max et ainsi de suite jusqu'au dernier nœud. Une fois ces nœuds finaux évalués, l'information remonte dans les nœuds parents. Le meilleur nœud enfant d'un nœud max est le *plus grand* de ses enfants nœuds min. à l'inverse, le meilleur enfant d'un nœud min est *le plus petit* de ses enfants nœuds max.

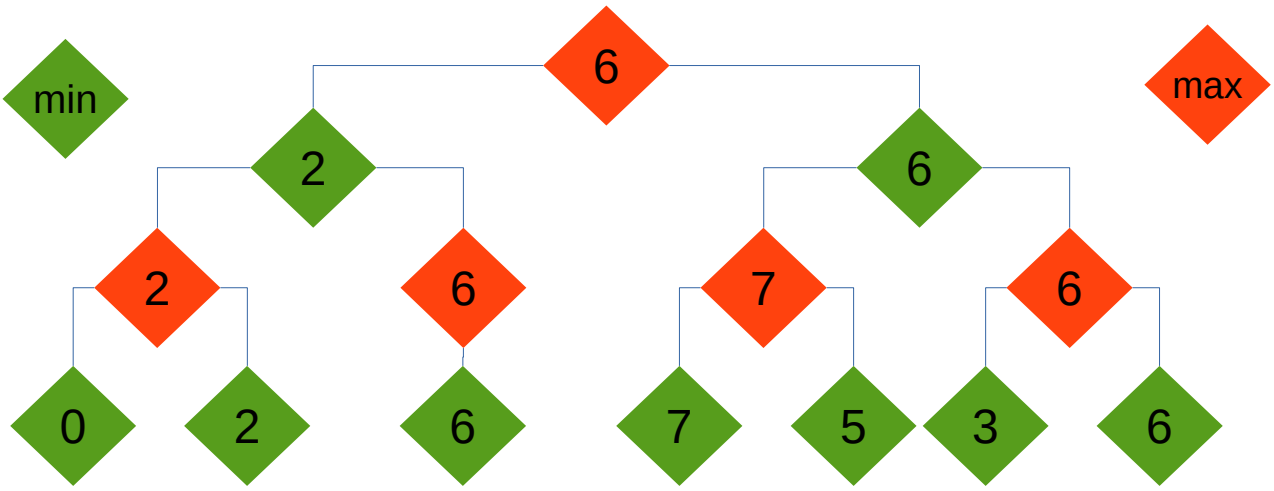

Illustration 1: Exemple de fonction de l'algorithme Min/Max

1.2 Utilisation d'alpha/bêta pour élaguer

En complément de l'algorithme min/max, afin de diminuer au maximum le nombre de coups explorés, nous avons utilisé l'algorithme Alpha-Bêta. En effet, le min/max, génère un nombre impressionnant de branches (une pour chaque coup possible). Prenons l'exemple du début de partie : le premier joueur à 24 possibilités, puis, après qu'il aie posé sont pion, son adversaire aura 23 possibilités pour poser le sien. (24 * 23 = 552) et ainsi de suite. Lorsque le second joueur aura joué 2 fois, on arrive déjà à 255024 dispositions de pion possibles sur le plateau (552 * 22 * 21 = 255024). Cela génère un nombre important de calcul, dont la grande majorité ne sont pas nécessaire !

L'élagage permet d'éviter les branches qui ne sont pas intéressantes pour le joueur. Ainsi, on ne calcule pas tout une partie de l'arbre, car chacune de ces branche est stratégiquement non viable. Comment déterminer quelle sont non viables ? Tout simplement parce que ces coups ne tendent pas à former une solution gagnante (ligne, carré…) On affecte une valeur à chaque coup afin de le valoriser (nombre positif) ou le dévaloriser(nombre négatif).

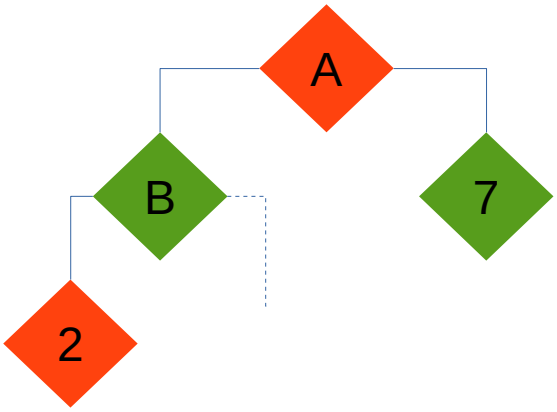

Illustration 3: Coupure alpha

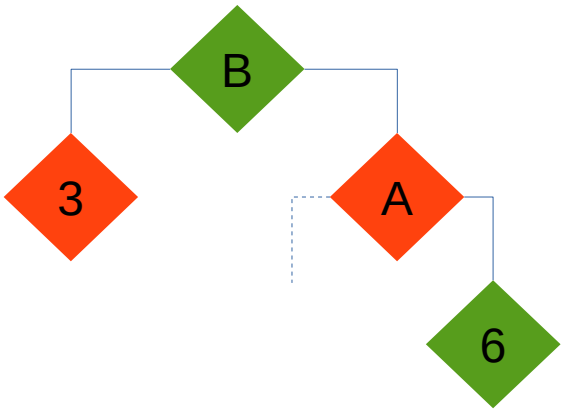

Illustration 2: Coupure bêta

Dans l'exemple ci-dessus, à gauche le noeud supérieur est un noeud max. L'un de ses fils a pour valeur 7 ; c'est à dire que la valeur de A est au **minimum** 7. Du coté de l'autre noeud fils de A, B est aussi un noeud min, donc ces noeuds enfants sont des noeud max. L'un de ceux-ci a comme valeur 2 ; cela signifique que B vaut au **maximum** 2. Donc B vaudra au maximum 2. 7 étant plus grand, A vaudra 7. Ce qui veut donc dire que la branche de B n'a plus besoin d'être explorée, elle est inutile. C'est une coupure

De manière analogue, à droite ci-dessus, B est noeud min. Ses noeuds fils sont des noeuds max. L'un d'eux vaut 3, ce qui signifique que B vaudra au **maximum** 3. l'un des fils de A, un noeud min, vaut 6, donc A vaudra au **minimum** 6. 3 étant plus petit, B vaudra 3. la branche de A est donc inutile. C'est une coupure bêta.

2 Représentation des données

2.1 Plateau de jeu

Avant de commencer à réfléchir aux problèmes d'IA, nous avons chercher une solution afin de représenter le plateau. Petit à petit deux solutions ont immergés : la première, l'utilisation de deux coordonnées, une verticale et l'autre horizontale. La seconde, une liste successive de l'ensemble des positions, dans le cas présent, de 1 à 25. Chacune de ces solutions présentaient des avantages et des inconveignants

La mise en place d'un plateau sur deux axes ressemble à un jeu de plateau tel que les échecs, et semble une approche plus intuitive, en revanche sa mise en place est relativement lourde comparée à une succession de 25 positions dans une liste puisqu'il faut recueillir chacune des coordonnées x et y à chaque traitement.

À l'inverse, en utilisant une liste de positions, le recueil des coordonnées est plus léger : on diminue ainsi le nombre de paramêtre par fonction et de fonction par la même occasion. On augmente ainsi la lisibilité du code. Certains passage sont par contre un peu plus fastidieux, puisque beaucoup de chose sont répétées.

Nous avons donc choisi de représenter le plateau comme une liste de 25 cases.

2.2 Joueurs et pions

Chaque joueur possède une liste contenant la position des 4 pions. Les pions sont une liste et possède dès le départ un nom (B1, B2, etc.).

2.3 Conditions de victoire

Afin de savoir si un joueur a gagné, nous devions trouver une façon de représenter l'ensemble des solutions gagnantes, ou bien vérifier si les 4 pions du joueur sont alignés ou en carré.

Le plus simple à nos yeux, était de renseigner l'ensemble des combinaisons gagnantes : lignes, colonnes, diagonales et carré.

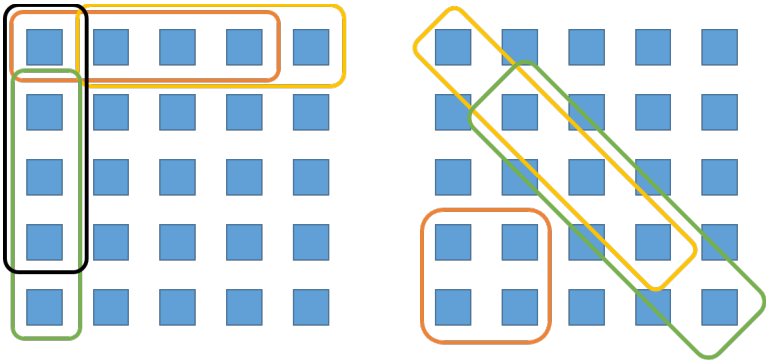

Illustration 4: Exemple de solutions gagnantes

3 Placement des pions

3.1 Initialisation

Comme expliqué dans l'introduction, la première phase du jeu consiste à placer l'ensemble de ces 4 pions. Pour cela

3.2 Déplacement dans la seconde phase

4 Problèmes rencontrés

4.1 Définition du plateau

Nous avons dû repenser notre modélisation des cases du plateau. En effet, nous avions commencés par faire un plateau sous forme de matrice (soit une liste de liste). Cependant, nous avons eu quelques problèmes dans la résolution IA, par la suite.

4.2 Déplacement dans la seconde phase

un rappel de l'énoncé du problème,

votre spécification (formalisation) du problème,

l'analyse du problème,

la méthode proposée, avec en annexe le listing du programme commenté

la description détaillée d'une ou de plusieurs situations traitées par votre programme,

les résultats obtenus par votre programme sur ces situations,

les difficultés éventuellement rencontrées,

les améliorations possibles (méthodes de résolution) et les perspectives d'ouverture possibles du sujet traité