TD no5 bonus	**L'algorithme Alpha-Bêta pour mieux jouer au Puissance 4**

Objectif de la séance :
Compléter le TP sur le Puissance 4 pour qu'une IA utilise l'algorithme Alpha-Bêta.

1 Avant propos

1.1 Constats

Comme vous avez vous pu le constater avec le précédent TP, pour qu'une IA jouant au jeu du puissance 4 avec l'algorithme Min-Max joue convenablement, il faut que la profondeur soit au moins de 6. Hors dans ce cas, le temps de réponse est assez long. Ne pourrait on pas l'améliorer ?

Étudions de nouveau l'algorithme Min-Max sur l'exemple théorique présenté par la figure 1.

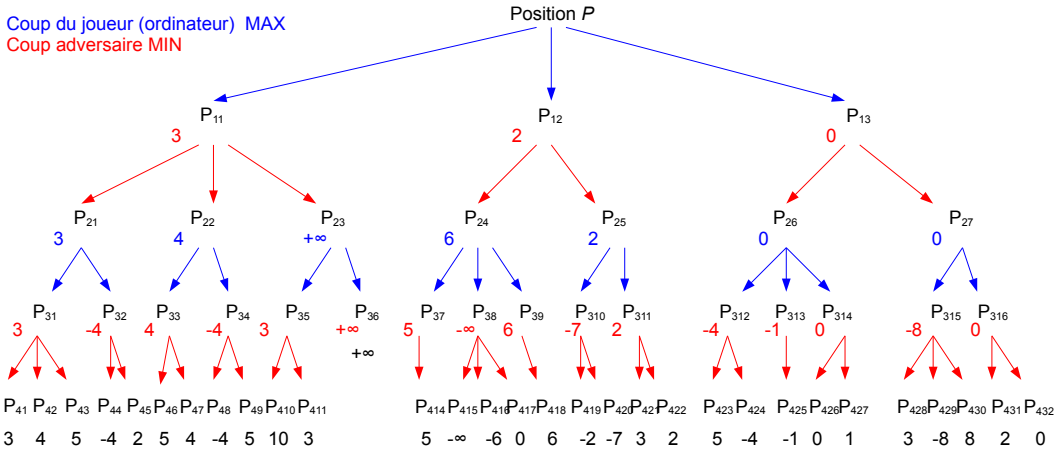

FIGURE 1 – Résultat de l'algorithme MinMax

Focalisons nous sur les positions P_{21}. Le score du premier coup, qui amène à la position P_{31}, est de 3. Le calcul du score du deuxième coup amène à la position P_{32}. Le score du premier coup de cette position P_{32} est de -4. On remarque alors qu'il n'est pas nécessaire de continuer à évaluer les coups de cette position, et que l'on peut remonter tout de suite -4. En effet comme à ce niveau (rouge) le calcul d'un score retourne le *min* des scores, la valeur retournée pour la position P_{32} ne peut donc pas être plus grande que -4. Or comme au niveau supérieur, au niveau de la position P_{21}, on retourne le *max* (bleu), le score remonté ne peut pas être plus petit que 3. Il est donc inutile de calculer le score du deuxième coup de P_{32}, celui amenant à la position P_{45}.

On peut appliquer le même raisonnement à la position P_{11}. Le score du premier coup est de 3. Le score du premier coup de la position P_{22} est de 4. Le score du coup amenant

à la position P_{22} sera donc au moins de 4. Mais comme le score du coup ayant amené la position P_{11} ne peut pas être supérieur à 3, il est inutile de calculer le score du deuxième coup de la position P_{22}.

La figure 2 présente l'ensemble des coupures que l'on peut faire dans l'arbre des appels sans changer les valeurs retournées.

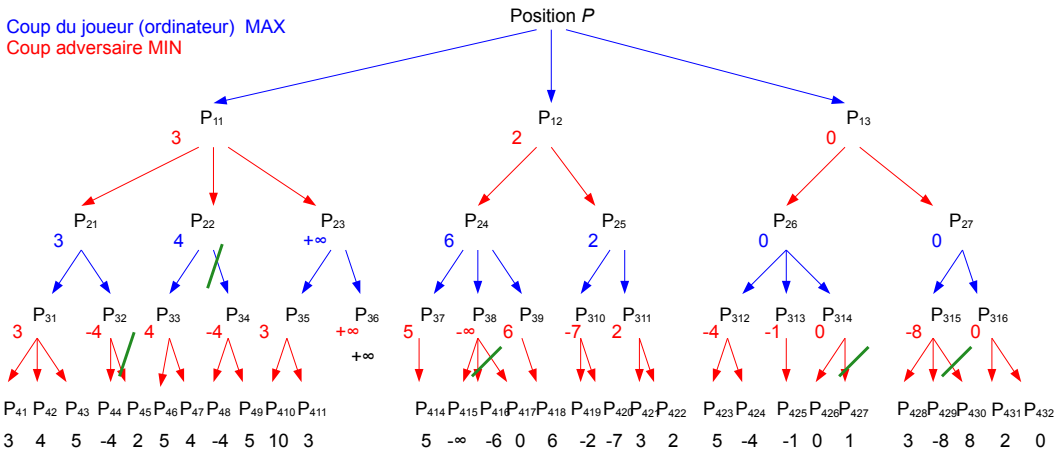

FIGURE 2 – Coupures possibles (en vert) qui ne changent pas le résultat de l'algorithme Min-Max

1.2 L'algorithme Alpha-Bêta

L'algorithme de la fonction Alpha-Bêta calcule le score d'un coup amenant à une position non terminale en réalisant ces coupures. Cette fonction reprend la signature de la fonction *min-max* en ajoutant deux paramètres formels :

alpha qui est utilisé dans une position *min* et il représente la valeur courante de la position *max* du dessus ;

bêta est utilisé dans une position *max* et il représente la valeur courante de la position *min* du dessus.

Lors du premier appel à la fonction *alpha-beta*, le paramètre effectif pour *alpha* vaut $-\infty$ et il vaut $+\infty$ pour *bêta*.

Dans une position *min*, lorsque le calcul de score d'un coup s est plus petit que *alpha*, alors il n'est pas nécessaire de calculer les scores des autres coups, il suffit de retourner s. Si ce n'est pas le cas, *bêta* est mis à jour pour le coup suivant, *bêta* vaut alors le `min` entre *bêta* et s.

Dans une position *max*, lorsque le calcul de score d'un coup s est plus grand que *bêta*, alors il n'est pas nécessaire de calculer les scores des autres coups, il suffit de retourner s. Si ce n'est pas le cas, *alpha* est mis à jour pour le coup suivant, *alpha* vaut alors le `max` entre alpha et s.

La figure 3 présente les valeurs des paramètres effectifs des paramètres *alpha* et *bêta* dans les calculs des scores depuis la position P_{11} de la figure 2 :

— Lors du calcul du score du premier coup depuis la position P_{11} *alpha* vaut $-\infty$ et *bêta* $+\infty$. Il en est de même pour tous les premiers de coups depuis les positions P_{21}, P_{31} et P_{41} ;

— Lors du calcul du deuxième coup de P_{31}, *bêta* vaut 3 car P_{31} est un *min* et que 3 n'est pas plus petit que *alpha*. Au calcul du score du troisième coup de P_{31}, *bêta* vaut toujours 3 car le score du deuxième coup, qui vaut 4, n'est pas plus petit que 3 ;
— Lors du calcul du deuxième coup de P_{21}, *alpha* vaut 3 car P_{21} est un *max* et que 3 n'est pas plus grand que *bêta*. C'est aussi le cas lors du calcul du score du premier coup de P_{31}. En revanche comme le score de ce premier coup est de -4, qui est plus petit que *alpha*, l'algorithme ne calcule pas le score du deuxième coup, il y a une coupure, il retourne directement cette valeur -4 ;
— Ainsi le score du premier coup de P_{11} vaut 3, ce qui modifie la valeur de *bêta*. Cette valeur de *bêta* est utilisée pour le calcul du deuxième coup de P_{11}. Mais comme le score du premier coup de P_{22} vaut 4, qui est plus grand que *bêta*, alors l'algorithme ne calcule pas le score du deuxième coup de P_{22} et il retourne directement la valeur 4 comme score pour le deuxième coup de P_{11}.

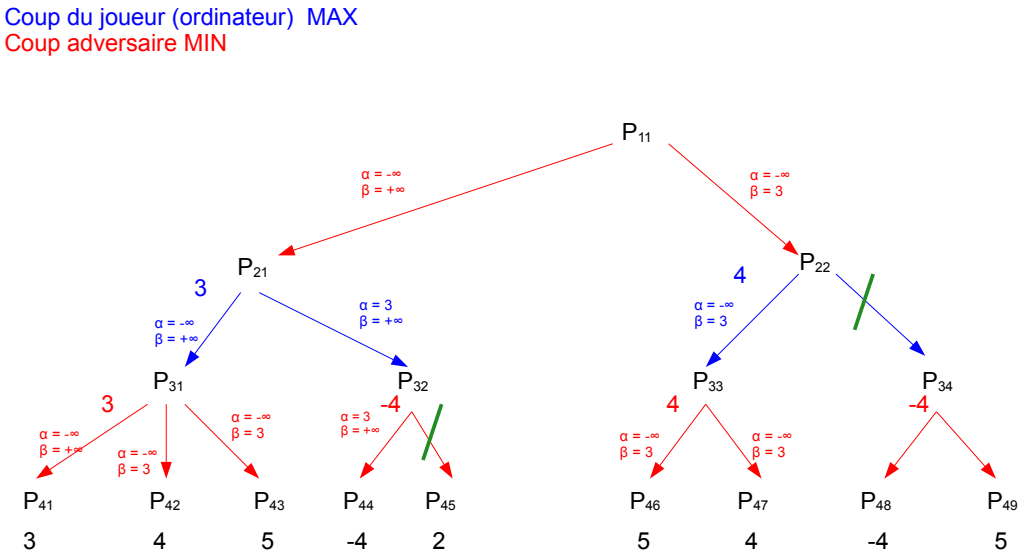

FIGURE 3 – Valeur des paramètres *alpha* et *bêta* dans les calculs des scores depuis la position P_{11} de la figure 2

2 Implémentation pour le puissance 4

Téléchargez l'archive sur Moodle. Elle reprend l'architecture du précédent TP avec toutefois les modifications suivantes :
— au lancement du script `puissance4_ihm_txt`, le programme demande maintenant quelle type d'algorithme on veut utiliser pour l'IA ;
— dans le module `ia_puissance` ;
  — les fonctions précédemment utilisées pour l'algorithme Min-Max (`score_d_un_coup` , `obtenir_meilleur_coup`) on été renommées (ajout du suffixe `_min_max`)
  — trois fonctions ont été ajoutées pour l'IA Alpha-Bêta :
    1. `score_d_un_coup_alpha_beta` ;
    2. `obtenir_meilleur_coup_alpha_beta` ;

3. `alpha_beta`).

1. Complétez le corps de ces trois fonctions. À l'image du code utilisé pour l'algorithme Min-Max, si l'IA joue le premier coup de la partie, celui ci est aléatoire pour éviter d'avoir des parties déterministes
2. Comparez les algorithmes Min-Max et Alpha-Bêta en faisant jouer ces deux IA l'une contre l'autre avec un temps de « réflexion » équivalent.