TD - Introduction à l'Intelligence Artificielle n° 1
(Correction)

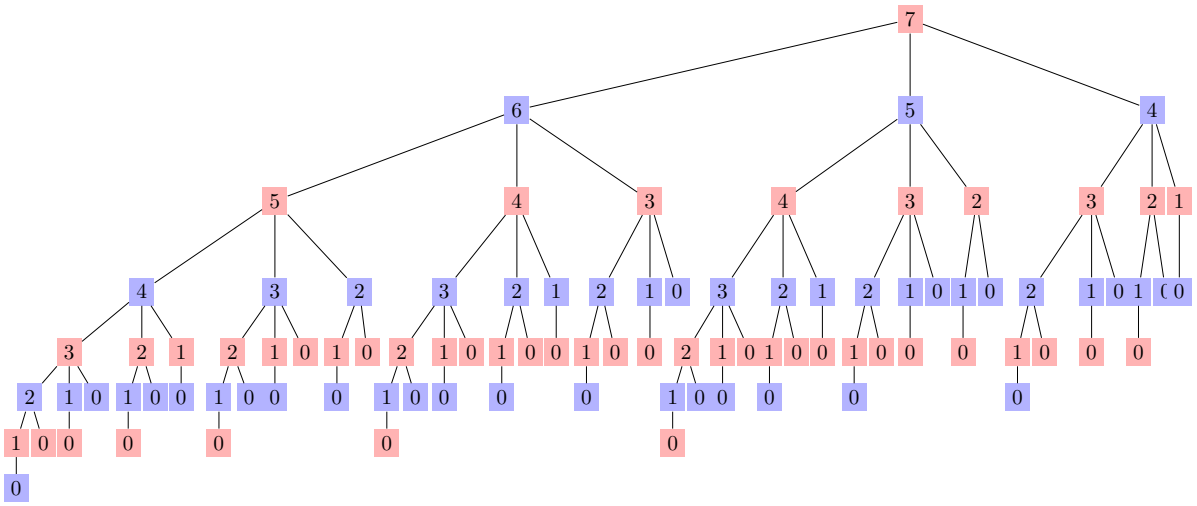
Algorithmes pour les jeux (1)

Exercice 1 Le jeu de Nim

On s'intéresse ici à la modélisation d'une des nombreuses variantes du jeu de Nim, dans laquelle on dispose au départ d'un ensemble de N allumettes et où chaque joueur peut tour à tour enlever $1, 2$ ou 3 allumettes. Le joueur perdant est celui qui ramasse la dernière allumette.

1. Développez l'arbre de recherche entier pour $N = 7$.

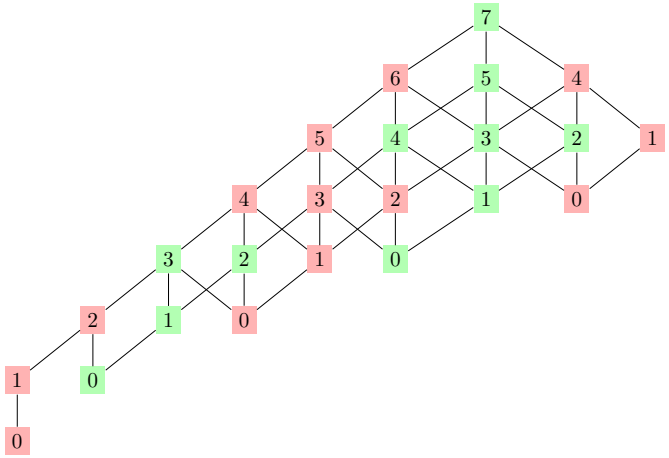
   Correction :
   On peut représenter l'arbre complet de la façon suivante :



   Les noeuds ami (i.e. max) figurent ici en rouge et les noeuds ennemi (i.e. min) en bleu.
   Note : En TD les noeuds ami sont représentés par des ronds, et les noeuds ennemi par des carrés.

2. Existe-t-il des stratégies gagnantes pour ce jeu ?

   Correction : Pour savoir ici s'il existe une stratégie gagnante, on peut faire tourner l'algorithme ET/OU vu en cours. On étiquette chaque noeud feuille gagnante (i.e. $+\infty$ pour une victoire du joueur Ami) et perdant (i.e. $-\infty$ pour une victoire du joueur Ennemi) et on fait remonter les estimations vers la racine en étiquetant chaque noeud interne Ami gagnant si l'un de ses fils est lui même gagnant et en perdant si tous ses fils sont perdants. On procède inversement pour les noeud Ennemi.
   Dans l'arbre ci-dessous, les noeuds gagnants sont étiquetés en vert ($+\infty$) et les perdants en rouge ($-\infty$).
   NB : Pour alléger la représentation, l'arbre de jeu est ici représenté sous la forme d'un graphe.



3. En prenant comme heuristique $h(n) = N - n$ pour $n > 1$ (n étant le nombre d'allumettes restantes), $h(1) = -\infty$, et $h(0) = +\infty$ pour l'évaluation d'une configuration du jeu où c'est à AMI de jouer, et $h(1) = +\infty$, et $h(0) = -\infty$ pour l'évaluation d'une configuration du jeu où c'est à ENNEMI de jouer, simulez une partie entière pour $N = 10$ entre deux I.A., l'une voyant à 2 demi-coups, et l'autre à 4 demi-coups.

Correction :

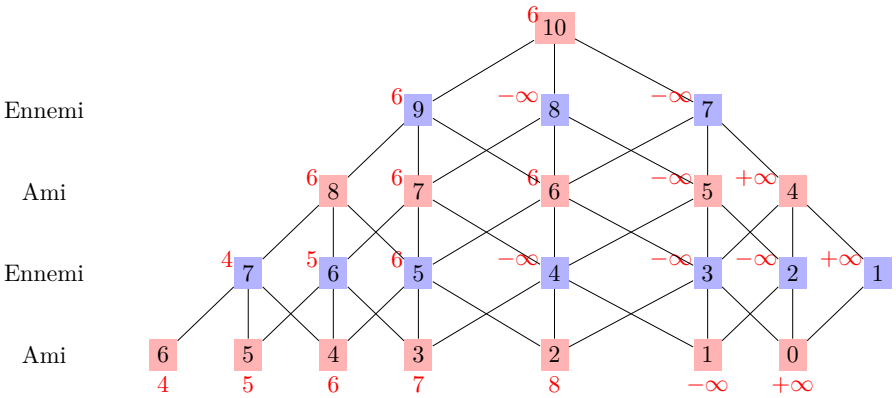
Faire faire les deux développements possibles aux étudiants, en alternant dans les rangs :

- L'un fait un développement en prenant pour premier joueur l'IA à 4 demi coups (nommée "Alice" ci-dessous) et pour le second joueur, l'IA à 2 demi coups (nommée "Bob" ci-dessous).
- Le voisin fait l'inverse : premier joueur à 2 demi coups et second joueur à 4 demi coups.

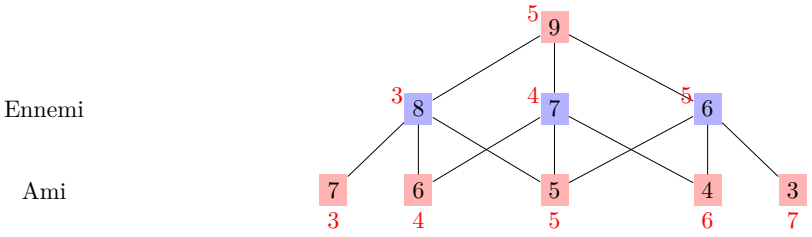
Insister sur le fait qu'après chaque coup joué par un joueur, il faut recommencer un nouvel arbre dont la racine correspond à l'état résultant du coup précédent.

Exemple : Alice est le premier joueur

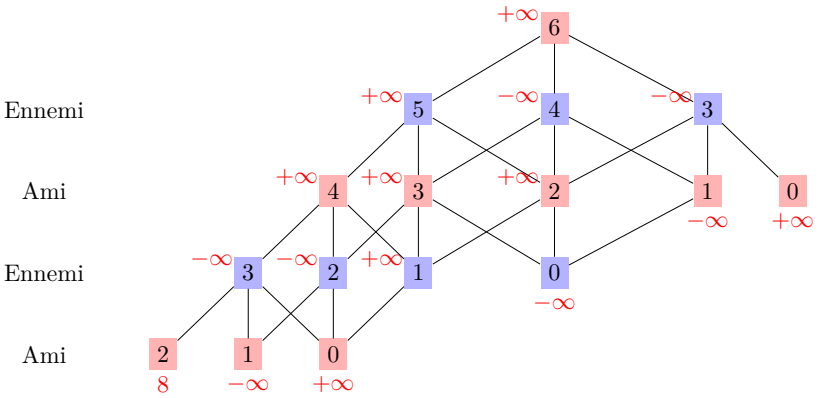
Avec Alice, le premier coup de jeu donne lieu à l'arbre suivant :



La valeur remontée à la racine est 6 et correspond au coup qui consiste a prendre une allumette. Alice joue donc ce coup et Bob lance son IA (avec seulement 2 demi-coups) à partir de l'état 9. Notons que pour Bob les rôles sont inversés. Le joueur Ami (du point de vue d'Alice) devient le joueur ennemi, et lui devient le joueur Ami.



Bob en déduit donc le meilleur coup à jouer pour lui des de prendre 3 allumettes.
Alice repart alors de cet état :



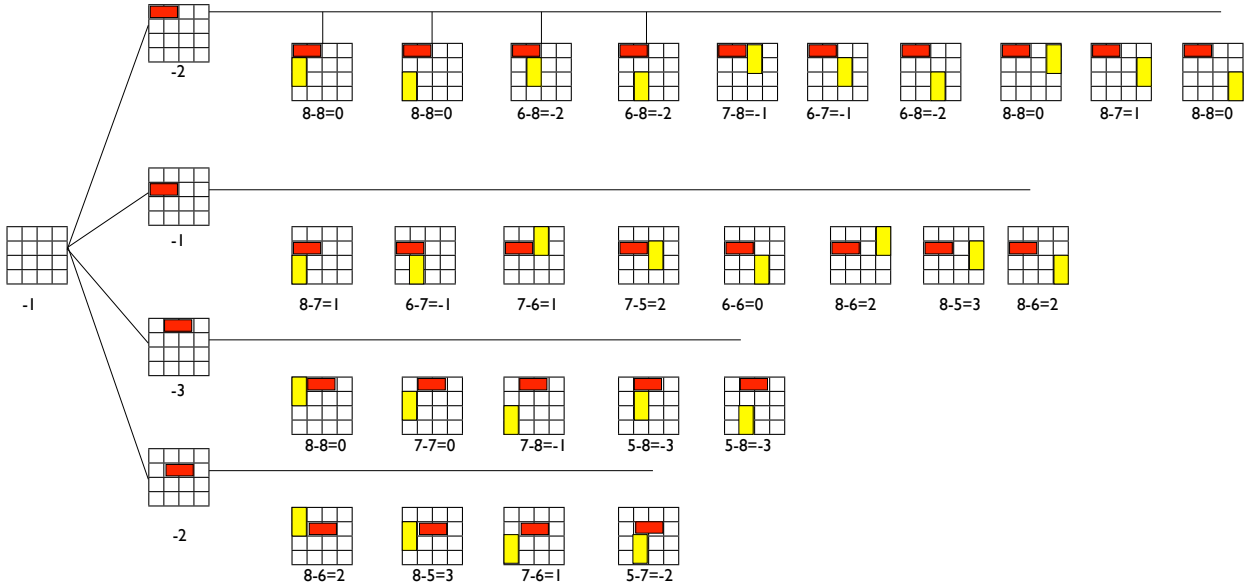
Comme la valeur $+\infty$ est remontée à la racine elle sait à ce stade qu'elle a une stratégie gagnante et qu'elle doit enlever une allumette pour jouer en 5.
Bob pourra alors jouer n'importe quoi, la réponse sera de jouer le complémentaire à 4 pour ramener l'adversaire dans l'état où il ne reste plus qu'une allumette.

Exercice 2 Effet horizon sur les dominos 4×4

On considère le jeux des dominos 4×4 (le graphe de tous les coups possibles fait 2120 noeuds, ce qui est bien trop pour tenir sur une feuille de TD). Donnez le coup joué par une recherche Minimax avec un horizon à 2 demi-coups et l'heuristique suivante :

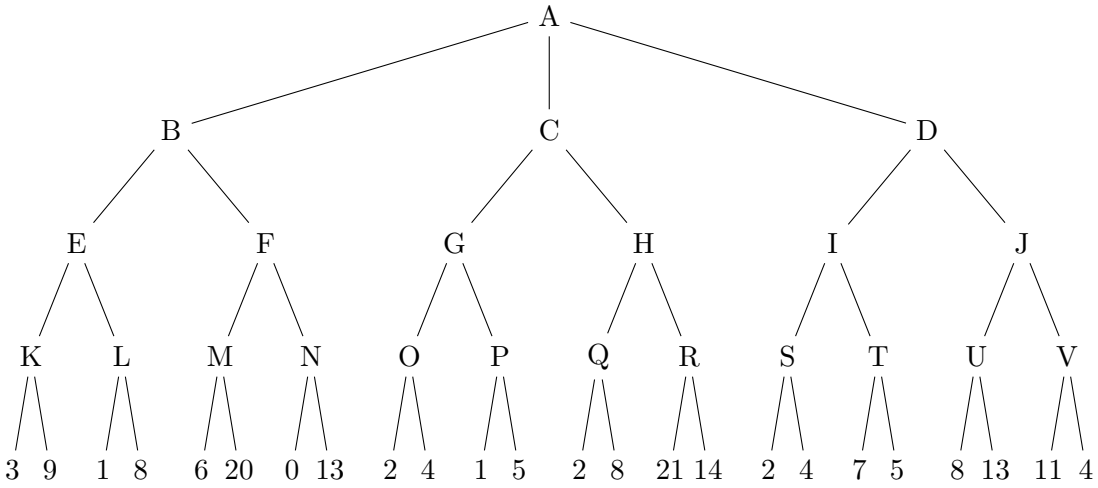
$$h(n) = nbplacements(AMI, Horizontaux) - nbplacements(ENNEMI, Verticaux)$$

Correction : Le graphe de jeu développé à profondeur 2 est le suivant :

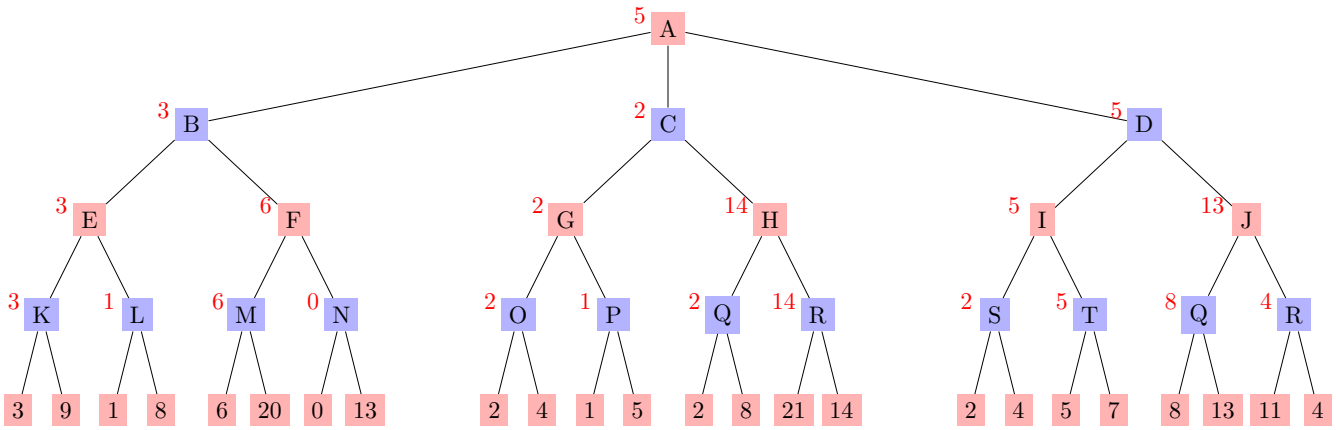


Exercice 3 Algorithmes Minimax et $\alpha\beta$

1. Appliquez rapidement Minimax sur l'arbre de jeu ci-dessous pour déterminer le coup qui semble le plus prometteur pour le premier joueur.



Correction :



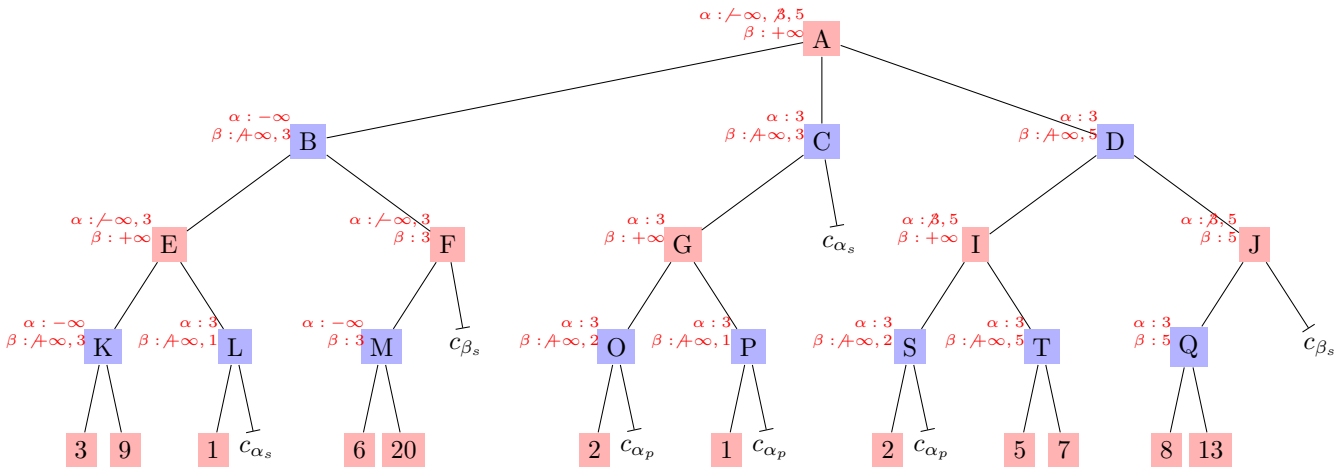
2. Appliquez l'algorithme $\alpha\beta$ sur ce même arbre.
   Vous redévelopperez complètement sur une feuille séparée l'arbre exploré. Indiquez pour chaque coupe effectuée, s'il s'agit d'une coupe α ou β et s'il s'agit d'une coupe de surface ou d'une coupe profonde.

   Correction : Expliquer que c'est bien un parcours en profondeur de gauche à droite que l'on fait et qu'il est préférable de refaire le graphe au fur et à mesure du parcours pour bien comprendre comment cela marche.

   Il est important de leur faire rédiger la solution de façon à ce que l'on puisse comprendre à postériori leur raisonnement.

   Le mieux est de leur faire marquer systématiquement au niveau de chaque noeud les valeurs des paramètres α et β. Lors de la mise à jour d'une valeur, faire en sorte de rayer la valeur mise à jour de façon à ce qu'on l'on puisse malgré tout comprendre quelles ont été les valeurs successives

   Exemple :
   $\alpha = -\not\infty, \not 2, \not 7, 15$
   $\beta = 21$



nature des coupes :
c_{α_s} : coupe-α de surface.
c_{β_s} : coupe-β de surface.
c_{α_p} : coupe-α profonde.

Coupe-α : la cause de la coupe est la valeur de α qui est supérieure à la valeur remontée pour β.
Coupe-β : la cause de la coupe est la valeur de β qui est inférieure à la valeur remontée pour α.
Coupe de surface : la valeur à l'origine de la coupe vient du parent direct
Coupe de profonde : la valeur à l'origine de la coupe vient d'un ancêtre du parent direct

3. Appliquez ensuite Nega$\alpha\beta$ sur ce même arbre.

   Correction : Même graphe que précédemment en inversant les rôles et signes de α et β sur les noeuds min.

4. Proposez un ordre de développement des fils de chaque noeud permettant de minimiser le nombre de noeuds développés par $\alpha\beta$.

   Correction : Ordonner les fils min d'un noeud max par valeurs décroissante, et les fils max d'un noeud min par valeurs croissantes.