

# ÉTUDE DES JEUX

B. Landelle

## Table des matières

<b>I</b>	<b>Préambule</b>	<b>2</b>
1	Un premier exemple . . . . .	2
2	Définitions . . . . .	4
<b>II</b>	<b>Jeux d'accessibilité</b>	<b>7</b>
1	Accessibilité, stratégie . . . . .	7
2	Attracteur . . . . .	10
3	Calcul d'attracteur . . . . .	12
<b>III</b>	<b>Algorithme du minimax</b>	<b>16</b>
1	Arbre de jeu . . . . .	16
2	Minimax . . . . .	19
3	Heuristique . . . . .	22

# I Préambule

## 1 Un premier exemple

On considère le *jeu des bâtonnets* aussi appelé *jeu de Nim*, popularisé par le jeu télévisé *Fort Boyard*. On dispose d'un tas de 20 bâtonnets et deux joueurs s'affrontent à tour de rôle en retirant 1, 2 ou 3 bâtonnets à chaque coup. Le joueur qui enlève le dernier bâtonnet perd la partie.

$$20 \xrightarrow{J_0} 17 \xrightarrow{J_1} 14 \xrightarrow{J_0} 11 \xrightarrow{J_1} 9 \xrightarrow{J_0} 6 \xrightarrow{J_1} 5 \xrightarrow{J_0} 4 \xrightarrow{J_1} 1 \xrightarrow{J_0} 0$$

FIGURE 1 – Exemple de partie opposant les joueurs  $J_0$  et  $J_1$

Dans l'exemple ci-avant, le joueur  $J_0$  démarre et retire 3 bâtonnets, il en reste 17, le joueur  $J_1$  retire 3 bâtonnets, il en reste 14, etc . . . , il en reste 5, le joueur  $J_0$  retire 1 bâtonnet, il en reste 4, le joueur  $J_1$  retire 3 bâtonnets, il en reste 1 et le joueur  $J_0$  perd en retirant le dernier bâtonnet.

On peut observer qu'une fois arrivé à 5 bâtonnets, le joueur  $J_0$  peut anticiper son échec puisque, quoi qu'il fasse, le joueur  $J_1$  dispose à coup sûr d'un choix qui lui garantit la victoire : si le joueur  $J_0$  retire 1 bâtonnet, alors le joueur  $J_1$  en retire 3, si le joueur  $J_0$  retire 2 bâtonnets, alors le joueur  $J_1$  en retire 2 et si le joueur  $J_0$  en retire 3, alors le joueur  $J_1$  en retire 1 et il gagne la partie.

On peut modéliser l'ensemble des parties possibles par un graphe orienté. Pour simplifier, on considère la configuration d'un tas de 9 bâtonnets.

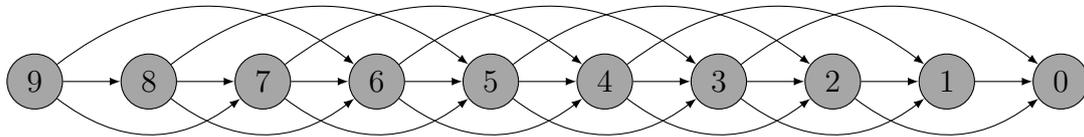


FIGURE 2 – Jeu de Nim

On peut alléger cette représentation en considérant qu'un joueur ne se « suicide » pas : il ne décide pas de perdre, une défaite est toujours subie. Par conséquent, l'état à 0 bâtonnet n'est plus pertinent dans la représentation et la partie se termine lorsqu'il ne reste plus qu'un seul bâtonnet à l'un des joueurs.

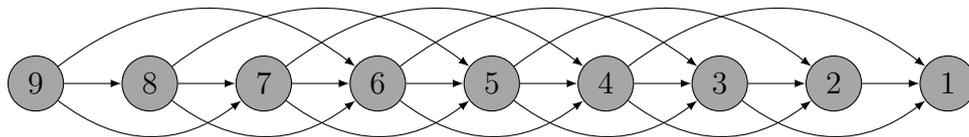


FIGURE 3 – Jeu de Nim, représentation allégée

Pour suivre le déroulé d'une partie, on peut *dédoubler* les états afin de savoir qui joue. Avec le dédoublement des états, une partie se lit complètement dans le choix d'un chemin de ce nouveau graphe orienté *biparti* : un sommet est contrôlé par un joueur et un seul et tout arc passe d'un sommet contrôlé par un joueur à un sommet contrôlé par l'autre.

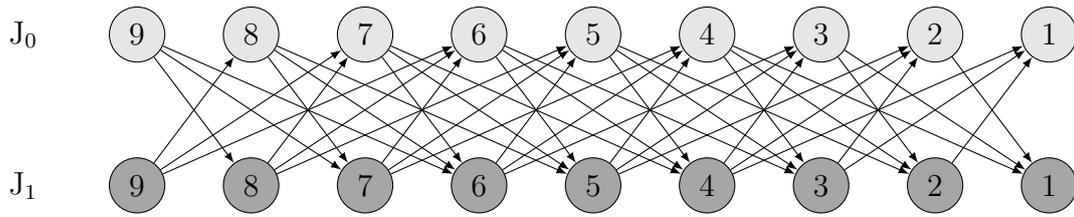


FIGURE 4 – Jeu de Nim

Dans l'exemple qui suit, on peut lire une partie démarrée et gagnée par le joueur  $J_0$ . On verra plus loin que le joueur  $J_1$  aurait pu tirer partie de cette configuration initiale particulière.

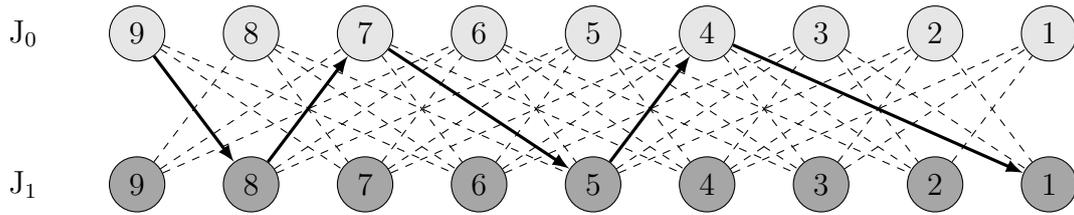


FIGURE 5 – Partie gagnée par  $J_0$

Comme on l'avait mentionné précédemment, si un joueur atteint l'état de 5 bâtonnets, alors la défaite lui est assurée : il laissera au joueur suivant 4, 3 ou 2 bâtonnets et celui-ci n'aura plus qu'à « compléter » le retrait pour ne laisser qu'un seul bâtonnet à son adversaire.

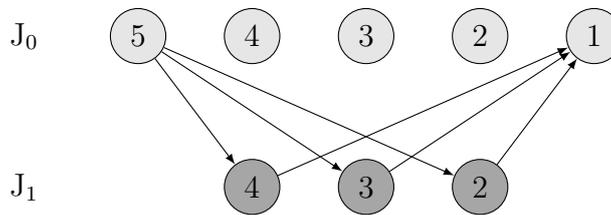


FIGURE 6 – Jeu de Nim

Il s'agit d'un fait plus général : si un joueur atteint l'état de  $4n + 1$  bâtonnets ( $n$  entier non nul), alors son adversaire est assuré de pouvoir le maintenir sur des états de  $4k + 1$  bâtonnets avec  $k \in \llbracket 0; n - 1 \rrbracket$  jusqu'à atteindre 1 bâtonnet et donc gagner. En effet, le joueur avec  $4n + 1$  bâtonnets en retire  $\ell$  avec  $\ell \in \llbracket 1; 3 \rrbracket$  et son adversaire n'a plus qu'à retirer  $4 - \ell \in \llbracket 1; 3 \rrbracket$ . Le tas passe ainsi de  $4n + 1$  à  $4n + 1 - \ell$  puis de  $4n + 1 - \ell$  à  $4(n - 1) + 1$  bâtonnets, etc

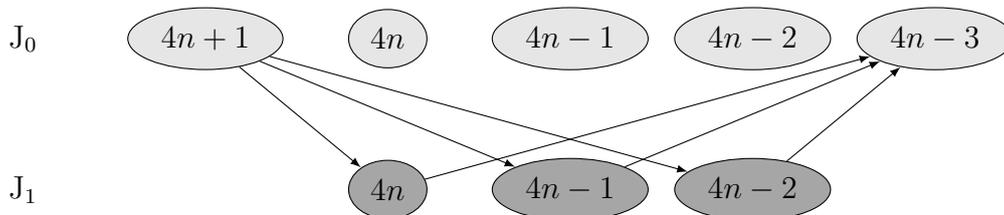


FIGURE 7 – Jeu de Nim

Une stratégie gagnante consiste donc à laisser à son adversaire un tas de  $4k + 1$  bâtonnets avec  $k$  entier. Si le tas initial ne contient pas  $4n + 1$  bâtonnets, alors le joueur qui commence et suit la règle de laisser  $4k + 1$  bâtonnets à son adversaire gagne. En revanche, si le tas initial contient

$4n + 1$  bâtonnets et que le deuxième joueur suit la règle des  $4k + 1$  bâtonnets, c'est lui qui gagne. Cette configuration est celle illustrée dans l'exemple 8.

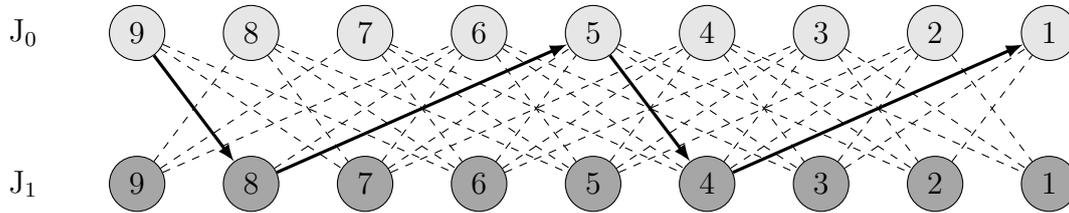


FIGURE 8 – Partie gagnée par  $J_1$

Il existe de nombreuses variantes au jeu des bâtonnets exposé ci-avant :

- le même format de jeu mais celui qui enlève le dernier bâtonnet gagne (la version où celui qui enlève le dernier bâtonnet perd est appelée version *misère* ou *qui perd gagne* au sens où le dernier qui joue perd en réalité) ;
- le jeu de Marienbad qui est un jeu de Nim à plusieurs tas, jeu résolu par le mathématicien Charles Bouton<sup>1</sup> en 1901 ;
- jeu de Grundy<sup>2</sup> où, partant d'un tas initial, chaque joueur sépare un tas en deux tas de tailles distinctes, jusqu'à ce qu'il ne reste que des tas à un ou deux éléments ;
- jeu de Wythoff (voir [2]).

## 2 Définitions

Avant de fournir une définition formelle de la notion de jeu, on recense différentes propriétés sur les jeux à deux joueurs.

**Définition 1.** *Un jeu à deux joueurs est dit :*

- séquentiel ou asynchrone ou encore alternatif si les adversaires jouent à tour de rôle (sinon, le jeu est dit simultané ou synchrone) ;
- à somme nulle si la somme des gains et des pertes des joueurs est nulle (donc un gain pour l'un constitue une perte pour l'autre) ;
- impartial ou non partisan si le jeu est séquentiel et si les coups autorisés et les gains obtenus dépendent uniquement de la position et non du joueur ;
- déterministe si le hasard n'intervient pas.

### Dilemme du prisonnier

Pour illustrer certaines notions, on citera le *dilemme du prisonnier* : deux individus A et B sont arrêtés, suspectés d'un délit. Afin de les inculper, la police a besoin d'aveux. Le marché proposé aux suspects est le suivant :

- si l'un des suspects dénonce l'autre, il est libéré et son complice est condamnée à 6 ans de prison ;
- si les deux suspects se dénoncent mutuellement, ils écopent chacun d'une peine de 3 ans de prison ;
- si les deux se taisent, ils écopent chacun d'une peine de 1 an de prison, faute de preuve.

1. Charles Leonard Bouton, 1869-1922, mathématicien américain.

2. Patrick Michael Grundy, 1917-1959, mathématicien britannique, co-découvreur du théorème de Sprague-Grundy qui démontre que tout jeu impartial fini sans partie nulle est équivalent à un jeu de Nim à un tas.

La représentation de ce « jeu » sous forme dite *normale*, c'est-à-dire sous forme de matrice, est décrite par :

A \ B	se taire	dénoncer
se taire	(1,1)	(6,0)
dénoncer	(0,6)	(3,3)

Ce « jeu » est un dilemme car chaque suspect peut être tenté de dénoncer l'autre pour se voir libéré sauf que si les deux optent pour cette stratégie, ils écopent d'une peine plus lourde que s'ils avaient chacun gardé le silence (le silence est d'or ...).

**Exemples :** 1. Les échec, le go sont des jeux séquentiels. Le jeu pierre-feuille-ciseaux, le dilemme du prisonnier sont des jeux simultanés.

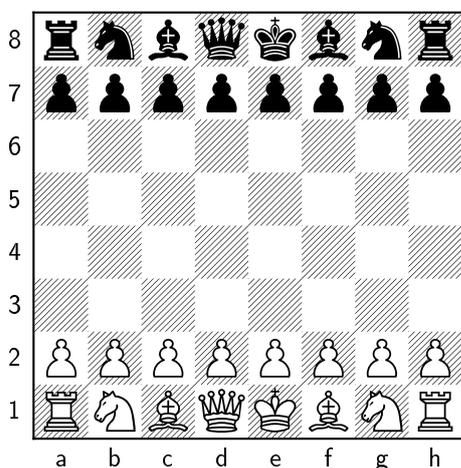


FIGURE 9 – Jeu d'échecs

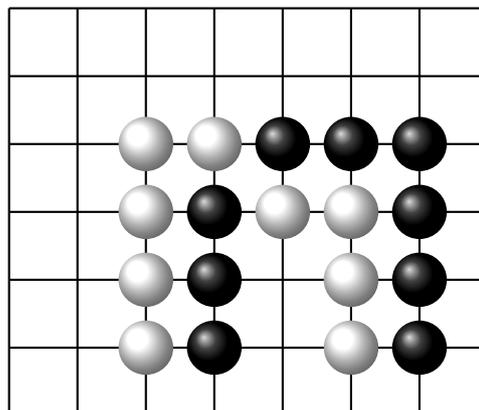


FIGURE 10 – Jeu de go

2. Les échecs sont un jeu à somme nulle si on considère pour un joueur un gain égal à 1 s'il est victorieux, égal à 0 en cas de partie nulle et à  $-1$  s'il perd. Au dilemme du prisonnier, si on considère que le gain d'un suspect est égal à  $3 - d$  avec  $d$  la durée de la peine, alors le jeu est à somme non nulle et la situation où chacun se tait est plus profitable aux deux individus avec la somme de leurs gains est égale à 4. Beaucoup de situations en économie, en politique, ... sont des jeux à somme non nulle.

3. Le jeu de Nim, le jeu de Grundy, le jeu de Chomp sont des jeux impartiaux. Les échecs, le go, le puissance 4, le morpion sont des jeux partisans : depuis une position donnée, les coups autorisés et les gains obtenus diffèrent selon le joueur considéré.

**Définition 2.** *Un jeu à deux joueurs est dit :*

- à information complète si chaque joueur connaît parfaitement la structure du jeu : les règles, les gains résultants de chaque action, les motivations de l'autre joueur (les jeux à information incomplète sont appelés jeux bayésiens) ;
- à information parfaite si le jeu est séquentiel et chaque joueur a une connaissance parfaite de la structure du jeu et des actions passées de l'autre joueur (pas d'action cachée) ;

**Remarque :** Un jeu à information parfaite est en particulier à information complète. Un jeu à information imparfaite peut être à information complète ou incomplète.

**Exemples :** 1. Aux échecs, le jeu est à information complète et parfaite.

2. Le dilemme du prisonnier est un jeu à information complète mais imparfaite du fait de la simultanéité des choix des joueurs.

3. La bataille navale est un jeu à information complète mais imparfaite puisque chaque joueur ignore la disposition initiale des navires de son adversaire.

**⚠** Dans tout ce qui suit, les jeux considérés sont des jeux à deux joueurs séquentiels, à somme nulle, déterministe et à information parfaite.

On va modéliser de tels jeux au moyen de graphes *bipartis*.

**Définition 3.** Un graphe  $G = (S, A)$  orienté ou non est dit biparti s'il existe une partition  $S = S_0 \sqcup S_1$  telle que toute arête (ou arc) du graphe a une extrémité dans  $S_0$  et l'autre dans  $S_1$ .

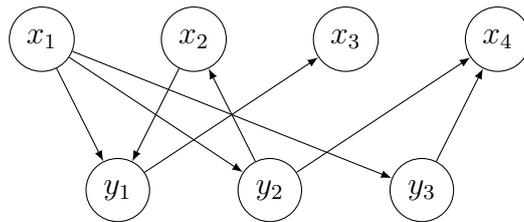


FIGURE 11 – Un graphe biparti

**Proposition 1.** Soit  $G = (S, A)$  un graphe orienté. Un graphe est biparti si et seulement si on peut colorier ses sommets avec seulement deux couleurs sans que deux sommets adjacents aient la même couleur.

**Définition 4.** Soit  $G = (S, A)$  un graphe orienté biparti avec  $(S_0, S_1)$  une partition de  $S$ . Le triplet  $(G, S_0, S_1)$  est appelée arène ou graphe de jeu. Un état final du graphe  $G$  est un sommet de degré sortant nul. On note  $F$  l'ensemble des états finals de  $G$ .

**Exemple :** Pour le jeu de Nim avec un tas initial de 9 bâtonnets, l'arène est

$$S_0 = \{(1)_0, (2)_0, \dots, (9)_0\} \quad S_1 = \{(1)_1, (2)_1, \dots, (9)_1\}$$

et les états finals sont  $\{(1)_0\}$  et  $\{(1)_1\}$ .

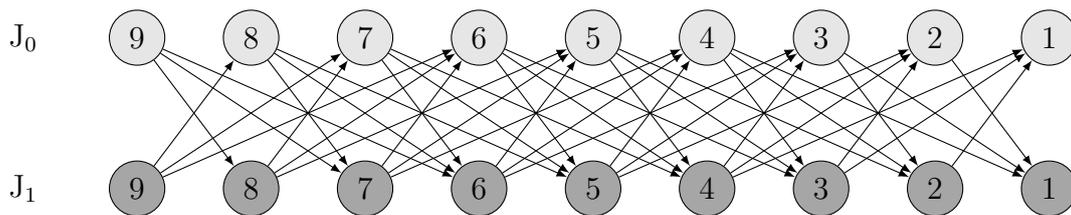


FIGURE 12 – Jeu de Nim : arène, états finals

**Définition 5.** Soit  $(G, S_0, S_1)$  une arène. Une partie est un chemin  $(s_0, s_1, s_2, \dots)$  fini ou infini, maximal dans le graphe  $G$ , avec  $s_0 \in S_0, s_1 \in S_1, \dots$

**Remarques :** 1. Le chemin est dit *maximal* au sens où on ne peut pas le compléter : soit parce qu'il est infini, soit parce qu'il aboutit à un état final.

2. Par convention, le joueur  $J_0$  commence d'où  $s_0 \in S_0$  et comme le graphe est biparti, on a nécessairement  $s_1 \in S_1, s_2 \in S_0, \dots$

**Exemple :** Le joueur  $J_0$  retire 1 bâtonnet puis le joueur  $J_1$  retire 1 bâtonnet puis le joueur  $J_0$  retire 2 bâtonnets puis le joueur  $J_1$  retire 1 bâtonnet puis le joueur  $J_0$  retire 3 bâtonnets et gagne.

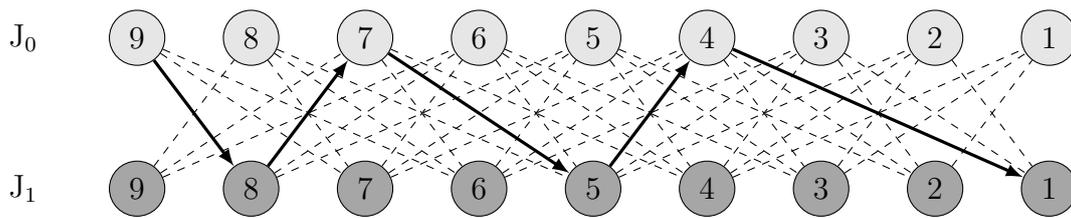


FIGURE 13 – Partie  $\{(9)_0, (8)_1, (7)_0, (5)_1, (4)_0, (1)_1\}$

**Proposition 2.** Soit  $(G, S_0, S_1)$  une arène. Si le graphe  $G$  est acyclique, toute partie est finie.

**Définition 6.** Soit  $(G, S_0, S_1)$  une arène. Une condition de gain ou condition de victoire est un sous-ensemble de l'ensemble des parties. Un jeu à deux joueurs est la donnée d'une arène et d'une condition de victoire pour chaque joueur.

Par convention (version normale, au contraire de la version *misère*), pour une partie finie qui n'est pas nulle, le joueur qui contrôle l'état final et ne peut plus jouer a perdu.

**Remarque :** Pour le jeu de Nim, le fait de ne pas représenter l'état à 0 bâtonnet permet d'avoir une représentation en version normale où le joueur qui ne peut plus jouer a perdu.

## II Jeux d'accessibilité

### 1 Accessibilité, stratégie

**Définition 7.** Soit  $(G, S_0, S_1)$  une arène. Une condition d'accessibilité pour le joueur  $J_i$  avec  $i \in \{0, 1\}$  est déterminée par  $V_i$  un sous-ensemble de  $F$  et représente l'ensemble des parties finies se terminant en  $V_i$ . On appelle  $V_i$  l'ensemble des états finals gagnants de  $J_i$ .

**Remarque :** D'après la convention mentionnée précédemment (pour une partie non nulle, le joueur qui ne peut plus jouer a perdu), étant donnée  $V_i$  une condition d'accessibilité pour le joueur  $J_i$  avec  $i \in \{0, 1\}$ , on a  $V_i \subset F \cap S_{1-i}$ .

**Exemple :** Pour le jeu de Nim avec un tas initial de 9 bâtonnets, on a

$$V_0 = \{(1)_1\} \quad \text{et} \quad V_1 = \{(1)_0\}$$

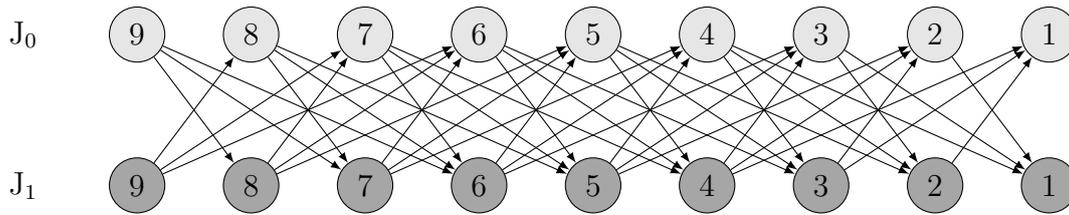


FIGURE 14 – Jeu de Nim, conditions d’accessibilité

**Définition 8.** Un jeu d’accessibilité à deux joueurs  $J_0$  et  $J_1$  est la donnée d’une arène et de conditions d’accessibilités déterminées par  $V_0, V_1$  respectivement pour les joueurs  $J_0$  et  $J_1$ .

**Exemples :** Les échecs, le go, le Hex, le puissance 4, le morpion sont des jeux d’accessibilité.

**Définition 9.** Dans un jeu d’accessibilité à deux joueurs, une partie infinie ou une partie finie dont l’état final est un cul-de-sac, c’est-à-dire un sommet dans  $F \setminus (V_0 \sqcup V_1)$  est une partie nulle.

**Remarque :** Dans un jeu d’accessibilité dont le graphe est acyclique et sans cul-de-sac, toute partie est finie avec un état final dans  $F = V_0 \sqcup V_1$ . Par conséquent, il y a un gagnant et un perdant.

**Exemples :** 1. Le jeu de Nim est un jeu d’accessibilité dont le graphe est acyclique et sans cul-de-sac.

2. On considère le jeu d’accessibilité à deux joueurs dont le graphe de jeu est le suivant :

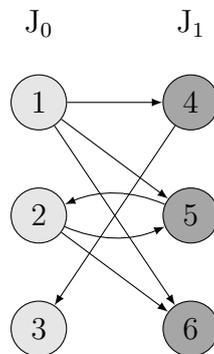


FIGURE 15 – Jeu d’accessibilité avec cycle

avec  $J_0$  qui démarre à 1 ou  $J_1$  qui démarre à 4 et les conditions d’accessibilité  $V_0 = \{6\}$  et  $V_1 = \{3\}$ . Une partie peut être infinie (et donc nulle) du fait de l’existence du cycle  $(2, 5)$ .

3. On considère le jeu d’accessibilité à deux joueurs dont le graphe de jeu est le suivant :

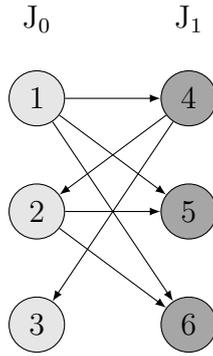


FIGURE 16 – Jeu d’accessibilité avec cul-de-sac

avec  $J_0$  qui démarre à 1 ou  $J_1$  qui démarre à 4 et les conditions d’accessibilité  $V_0 = \{6\}$  et  $V_1 = \{3\}$ . Une partie qui se termine en 5 est nulle puisqu’il s’agit d’un cul-de-sac.

**Notation :** Soit  $(G, S_0, S_1)$  une arène. Pour  $i \in \{0, 1\}$ , on note  $S_i^{>0}$  l’ensemble des sommets de  $S_i$  de degrés sortants strictement positifs.

**Définition 10.** Dans un jeu d’accessibilité à deux joueurs, une stratégie pour  $J_i$  avec  $i \in \{0, 1\}$  est une application  $\varphi : S_i^{>0} \mapsto S_{1-i}$  telle que pour tout sommet  $s \in S_i^{>0}$ , on a  $(s, \varphi(s)) \in A$  (ensemble des arcs du graphe).

**Remarque :** Il s’agit d’une stratégie *sans mémoire* (les seules au programme) : le prochain sommet joué ne dépend que du sommet courant et pas des précédents.

**Vocabulaire :** Lors d’une partie  $\pi = (s_0, s_1, \dots)$ , on dit que le joueur  $J_i$  avec  $i \in \{0, 1\}$  respecte la stratégie  $\varphi$  si

$$\forall j \in \llbracket 0; n \rrbracket \quad s_j \in S_i^{>0} \implies s_{j+1} = \varphi(s_j)$$

avec  $|\pi| = n + 1$ ,  $n \in \mathbb{N} \cup \{\infty\}$  où  $|\pi|$  désigne la longueur de  $\pi$ .

**Définition 11.** Dans un jeu d’accessibilité à deux joueurs, une stratégie  $\varphi$  est gagnante pour  $J_i$  avec  $i \in \{0, 1\}$  depuis le sommet  $s_0$  si toute partie jouée depuis le sommet  $s_0$  où le joueur  $J_i$  respecte la stratégie  $\varphi$  est gagnée par  $J_i$ .

**Commentaire :** Si un joueur dispose d’une stratégie gagnante depuis un sommet et qu’il la respecte, quels que soit les coups joués par l’adversaire, sa victoire est assurée.

**Exemple :** Pour le jeu de Nim avec un tas initial de 9 bâtonnets, le joueur  $J_0$  dispose d’une stratégie gagnante depuis les sommets 8, 7, 6, 4, 3, 2.

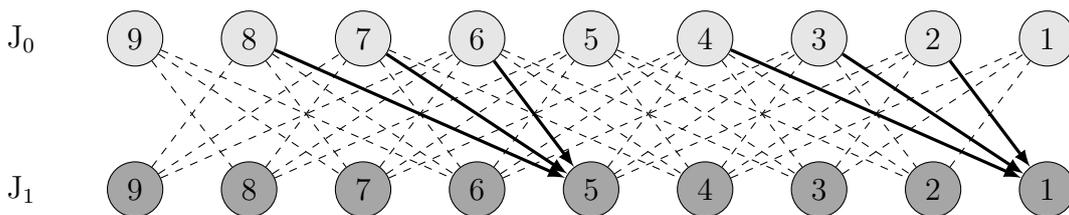


FIGURE 17 – Stratégie gagnante pour le joueur  $J_0$

En revanche, le joueur  $J_0$  ne possède pas de stratégie gagnante depuis les sommets 9 et 5 puisque, par symétrie, le joueur  $J_1$  dispose de stratégie gagnantes depuis les sommets successeurs des sommets 9 et 5.

**Définition 12.** Dans un jeu d'accessibilité à deux joueurs, un sommet  $s \in S_i$  avec  $i \in \{0, 1\}$  est une position gagnante pour le joueur  $J_i$  si celui-ci possède une stratégie gagnante depuis le sommet  $s$ .

**Exemple :** Pour le jeu de Nim avec un tas initial de 9 bâtonnets, les sommets 8, 7, 6, 4, 3, 2 sont des positions gagnantes pour les joueurs  $J_0$  et  $J_1$ .

**Définition 13.** Un jeu à deux joueurs est dit déterminé si l'un des joueurs possède une stratégie gagnante.

## 2 Attracteur

Pour  $i \in \{0, 1\}$ ,  $k$  entier et un objectif  $U \subset S$ , on va définir la notion d'attracteur  $\text{Attr}_k^i(U)$  comme l'ensemble des états du jeu à partir desquels, le joueur  $J_i$  possède une stratégie qui le fait atteindre l'ensemble  $U$  en  $k$  étapes au plus. Ainsi

- au rang  $k = 0$ , l'attracteur est exactement l'objectif;
- au rang  $k + 1$ , on ajoute les états contrôlés par le joueur  $J_i$  où au moins un successeur atteint  $U$  en  $k$  étapes ainsi que les états contrôlés par l'adversaire où tous les successeurs atteignent  $U$  en  $k$  étapes.

Formellement, on a la définition :

**Définition 14.** Dans un jeu d'accessibilité à deux joueurs, on définit l'attracteur du joueur  $J_i$  avec  $i \in \{0, 1\}$  d'ordre  $k$  entier d'une partie  $U \subset S$  noté  $\text{Attr}_k^i(U)$  par récurrence avec  $\text{Attr}_0^i(U) = U$  et

$$\forall k \in \mathbb{N} \quad \text{Attr}_{k+1}^i(U) = \text{Attr}_k^i(U) \cup \left\{ s \in S_i : \exists t \in \text{Attr}_k^i(U) \text{ et } (s, t) \in A \right\} \\ \cup \left\{ s \in S_{1-i} : \forall t \in S_i \quad (s, t) \in A \implies t \in \text{Attr}_k^i(U) \right\}$$

et on définit l'attracteur du joueur  $J_i$  d'une partie  $U$  par

$$\text{Attr}^i(U) = \bigcup_{k \in \mathbb{N}} \text{Attr}_k^i(U)$$

**Remarque :** La suite  $(\text{Attr}_k^i(U))_k$  est une suite croissante pour l'inclusion.

**Théorème 1.** Soit un jeu d'accessibilité à deux joueurs  $J_0$  et  $J_1$  d'arène  $(G, S_0, S_1)$  et conditions d'accessibilité déterminées par  $V_0, V_1$  respectivement pour les joueurs  $J_0$  et  $J_1$ . On suppose le graphe  $G$  acyclique et on note  $N = F \setminus (V_0 \sqcup V_1)$  l'ensemble des culs-de-sac (associés aux parties nulles). On a

$$\text{Attr}^0(V_0) \sqcup \text{Attr}^1(V_1 \sqcup N) = \text{Attr}^0(V_0 \sqcup N) \sqcup \text{Attr}^1(V_1) = S$$

En particulier, si  $N = \emptyset$ , on a

$$\text{Attr}^0(V_0) \sqcup \text{Attr}^1(V_1) = S$$

**Corollaire 1.** Les jeux d'accessibilité à deux joueurs sans match nul, c'est-à-dire avec un graphe acyclique et sans cul-de-sac, sont déterminés.

**Exemples :** 1. Au jeu de Nim, un des joueurs a une stratégie gagnante puisque c'est un jeu d'accessibilité avec un graphe acyclique et sans cul-de-sac. Par exemple, avec tas de 9 bâtonnets, le deuxième joueur  $J_1$  possède une stratégie gagnante.

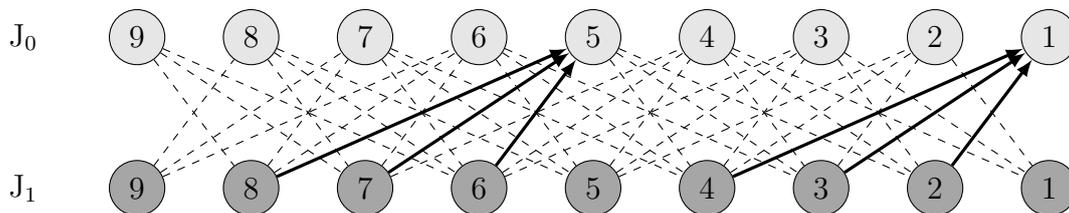


FIGURE 18 – Victoire du joueur  $J_1$  qui respecte la stratégie

2. Le jeu de Hex est un jeu d'accessibilité avec un graphe acyclique (c'est un jeu de remplissage) et sans cul-de-sac. En effet, on note  $R$  l'ensemble des pions rouges reliés au bord rouge du bas par un chemin de pions rouges (composante connexe). Si cet ensemble  $R$  est relié au bord rouge du haut, alors les rouges gagnent. Sinon, cet ensemble  $R$  est bordé par un chemin de pions bleus reliant les deux bords bleus et les bleus gagnent. Plus précisément, on peut démontrer, par une méthode dite de *vol de stratégie* que le premier joueur a une stratégie gagnante. En revanche, pour de grands plateaux, on ne sait pas déterminer une telle stratégie.

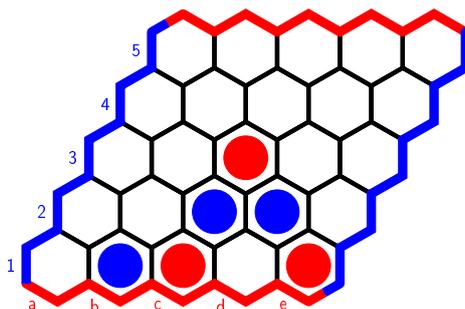


FIGURE 19 – Jeu de Hex

**Corollaire 2.** Dans un jeu d'accessibilité fini à deux joueurs avec match nul, c'est-à-dire avec un graphe acyclique ayant éventuellement des culs-de-sac, l'un des joueurs a une stratégie non perdante.

**Exemple :** Aux échecs, soit les blancs ont une stratégie gagnante, soit les noirs ont une stratégie gagnante, soit les deux peuvent forcer une partie nulle. En revanche, la complexité du jeu est telle qu'on ne sait pas déterminer dans quelle configuration on se trouve.

En présence de cycles et donc de parties infinies considérées nulles (autres que des parties se finissant dans un cul-de-sac), la situation est plus délicate. Les unions d'attracteurs peuvent être égales à  $S$  comme elles peuvent ne pas l'être. On considère les jeux d'accessibilité à deux joueurs dont les graphes de jeu sont les suivants :

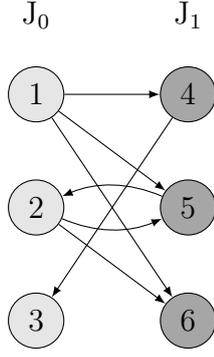


FIGURE 20 – Jeu avec cycle

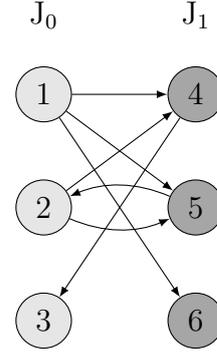


FIGURE 21 – Jeu avec cycle

avec  $J_0$  qui démarre à 1 ou  $J_1$  qui démarre à 4 et les conditions d'accessibilité  $V_0 = \{6\}$  et  $V_1 = \{3\}$ . Pour le jeu décrit par le graphe de la figure 20, on a

$$\text{Attr}^0(V_0) = \{1, 2, 5, 6\} \quad \text{Attr}^1(V_1) = \{3, 4\}$$

d'où  $\text{Attr}^0(V_0) \sqcup \text{Attr}^1(V_1) = S$

Depuis le sommet 2 contrôlé par  $J_0$ , ce dernier a une stratégie gagnante triviale (aller en 6) tandis que depuis le sommet 5 contrôlé par  $J_1$ , ce dernier n'a pas d'autre choix qu'aller en 2. Ainsi, même si des parties infinies sont possibles, il y a depuis les sommets 2 et 5 une stratégie gagnante pour  $J_0$ .

En revanche, pour le jeu décrit par le graphe de la figure 21, on a

$$\text{Attr}^0(V_0) = \{1, 6\} \quad \text{Attr}^1(V_1) = \{3, 4\}$$

d'où  $\text{Attr}^0(V_0) \sqcup \text{Attr}^1(V_1) \subsetneq S$

Depuis le sommet 5 contrôlé par  $J_1$ , ce dernier n'a pas d'autre choix qu'aller en 2. Depuis le sommet 2 contrôlé par  $J_0$ , celui-ci peut aller en 5 ou en 4. Mais depuis le sommet 4, le joueur  $J_1$  dispose d'une stratégie gagnante triviale (aller en 3). Ainsi, pour éviter la défaite, la stratégie du joueur  $J_0$  depuis le sommet 2 consiste à aller en 5. Cette stratégie non perdante pour  $J_0$  consiste donc à réaliser une partie infinie avec le cycle (2, 5).

Plus généralement, pour un jeu d'accessibilité à deux joueurs avec match nul, on pose

$$I = S \setminus (\text{Attr}^0(V_0 \sqcup N) \cup \text{Attr}^1(V_1 \sqcup N))$$

L'union  $\text{Attr}^0(V_0 \sqcup N) \cup \text{Attr}^1(V_1 \sqcup N)$  est l'ensemble des sommets depuis lesquels un des joueurs dispose d'une stratégie non perdante finie. Le complémentaire est donc l'ensemble des sommets depuis lesquels aucun joueur ne dispose de stratégie non perdante finie, autrement dit chaque joueur dispose d'une stratégie non perdante infinie (partie nulle). On peut alors établir

$$\text{Attr}^0(V_0 \sqcup N \sqcup I) \sqcup \text{Attr}^1(V_1) = \text{Attr}^0(V_0) \sqcup \text{Attr}^1(V_1 \sqcup N \sqcup I) = S$$

**Théorème 2.** Dans un jeu d'accessibilité à deux joueurs avec match nul, l'un des joueurs a une stratégie non perdante.

### 3 Calcul d'attracteur

**Définition 15.** Soit  $G = (S, A)$  un graphe orienté. On appelle graphe transposé de  $G$  le graphe noté  $G^\top$  défini par  $G^\top = (S, A^\top)$  avec

$$A^\top = \{(y, x) \mid (x, y) \in A\}$$

**Interprétation :** Dans le graphe transposé, les rôles entre successeurs et prédécesseurs sont échangés.

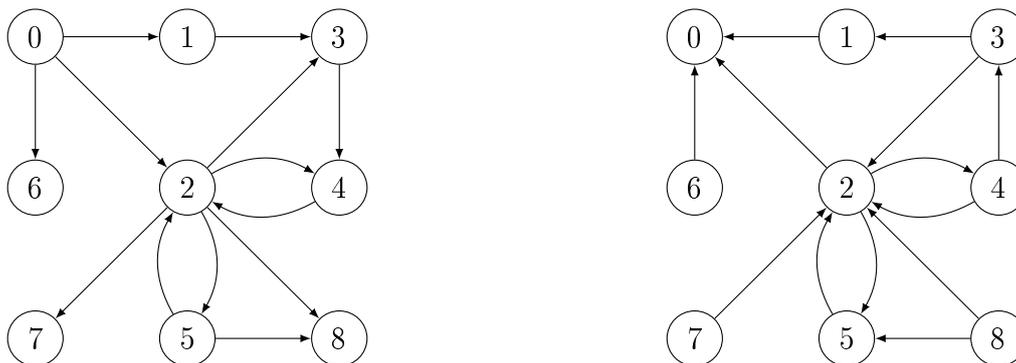


FIGURE 22 – Graphe et son transposé

Dans ce qui suit, on déroge à l'usage habituel. On considère qu'un graphe  $G$  est décrit directement par sa liste d'adjacence et on continue de noter  $S$  l'ensemble de ses sommets. Ceci permet de « libérer » la lettre  $A$  pour l'utiliser comme nom de variable pour le calcul d'attracteur. On utilisera un dictionnaire pour l'implémentation.

Par exemple, notant  $G$  le graphe de gauche de la figure 22, on choisit comme implémentation :

```
G={0: [1, 2, 6], 1: [3], 2: [3, 4, 5, 7, 8], 3: [4],
  4: [2], 5: [2, 8], 6: [], 7: [], 8: []}
```

Pour  $s \in S$ , la variable  $G[s]$  est la liste des successeurs dans  $G$  de  $s$ .

```
>>> G[2]
[3, 4, 5, 7, 8]
```

L'implémentation du graphe transposé  $G^T$  est :

```
{0: [], 1: [0], 2: [0, 4, 5], 3: [1, 2], 4: [2, 3],
  5: [2], 6: [0], 7: [2], 8: [2, 5]}
```

Pour construire le graphe transposé du graphe  $G$ , il suffit donc, lors du parcours des sommets de  $G$ , de parcourir tous les successeurs d'un sommet  $v$  et pour chacun de définir  $v$  comme successeur.

```
def transpose(G):
    """transpose(G:dict)->dict
    Calcule le graphe transposé de G décrit par liste d'adjacence"""
    tG={s:[] for s in G}
    for v in G:
        for s in G[v]:
            tG[s].append(v)
    return tG
```

Pour calculer un attracteur, on va implémenter l'algorithme qui suit :

---

**Algorithme 1** : Calcul d'attracteur

---

**Entrées** :  $G$  graphe décrit par liste d'adjacence,  $S_i$  liste de sommets contrôlés par le joueur  $J_i$ ,  $U$  objectif

**Résultat** :  $\text{Attr}^i(U)$

$\mathcal{A} \leftarrow \emptyset$ ;

**pour**  $s \in G$  **faire**

$n_s \leftarrow d^+(s)$ ;

Calculer le graphe transposé  $G^\top$ ;

**fonction** `parcours`( $s$ ) :

**si**  $s \notin \mathcal{A}$  **alors**

$\mathcal{A} \leftarrow \mathcal{A} \cup \{s\}$ ;

**pour**  $u \in G^\top[s]$  **faire**

$n_u \leftarrow n_u - 1$ ;

**si**  $u \in S_i$  **ou**  $n_u = 0$  **alors**

`parcours`( $u$ )

**pour**  $s \in U$  **faire**

`parcours`( $s$ );

**retourner**  $\mathcal{A}$

---

### Principe de l'algorithme

Au cours de la construction de l'attracteur, la variable  $n_s$  contient le nombre de successeurs de  $s$  hors de l'attracteur rencontrés lors du parcours de  $U$ . Initialement, un sommet n'a aucun successeur dans l'attracteur d'où l'initialisation de  $n_s$  à  $d^+(s)$ , le nombre de successeurs de  $s$ . Puis, à chaque ajout d'un sommet  $s$  dans l'attracteur, si  $s$  est successeur de  $u$ , alors  $n_u$  est décrémenté de 1.

C'est la fonction récursive `parcours` qui fait l'affectation des sommets concernés dans l'attracteur. L'appel `parcours(s)` place le sommet  $s$  dans l'attracteur puis place, récursivement, les prédécesseurs de  $s$  qui doivent figurer dans l'attracteur.

Le premier appel de `parcours(s)` avec  $s \in U$  place ce sommet dans l'attracteur. Ensuite, pour chaque prédécesseur  $u$  de  $s$ , on décrémente  $n_u$  de 1 et on regarde si le sommet  $u$  est contrôlé par le joueur  $J_i$  ou s'il n'a aucun successeur hors de l'attracteur ce qui revient à dire que tous ses successeurs sont dans l'attracteur. Si cette condition est vérifiée, le sommet  $u$  doit figurer dans l'attracteur et on appelle récursivement `parcours(u)`.

```

def attracteur(G,S_i,U):
    """attracteur(G:dict,S_i:list,U:list)->list
    Calcul de l'attracteur Attri(U)
    G : graphe décrit par liste d'adjacence
    S_i : sommets contrôlé le joueur J_i
    U : objectif"""
    A=[]
    nG={}
    for v in G:
        nG[v]=len(G[v])
    tG=transpose(G)
    def parcours(u):
        if u not in A:
            A.append(u)
            for v in tG[u]:
                nG[v]=nG[v]-1
                if v in S_i or nG[v]==0:
                    parcours(v)
    for u in U:
        parcours(u)
    return A

```

Pour le jeu de Nim avec un tas de 9 bâtonnets, on obtient pour le joueur  $J_0$  les résultats suivants :

```

*****
Jeu de Nim à 9 bâtonnets

Arene=
{(1, 0): [], (1, 1): [],
 (2, 0): [(1, 1)], (2, 1): [(1, 0)],
 (3, 0): [(2, 1), (1, 1)], (3, 1): [(2, 0), (1, 0)],
 (4, 0): [(1, 1), (2, 1), (3, 1)], (4, 1): [(1, 0), (2, 0), (3, 0)],
 (5, 0): [(2, 1), (3, 1), (4, 1)], (5, 1): [(2, 0), (3, 0), (4, 0)],
 (6, 0): [(3, 1), (4, 1), (5, 1)], (6, 1): [(3, 0), (4, 0), (5, 0)],
 (7, 0): [(4, 1), (5, 1), (6, 1)], (7, 1): [(4, 0), (5, 0), (6, 0)],
 (8, 0): [(5, 1), (6, 1), (7, 1)], (8, 1): [(5, 0), (6, 0), (7, 0)],
 (9, 0): [(6, 1), (7, 1), (8, 1)], (9, 1): [(6, 0), (7, 0), (8, 0)]}

S0= [(1, 0), (2, 0), (3, 0), (4, 0), (5, 0), (6, 0), (7, 0), (8, 0), (9, 0)]

V0= [(1, 1)]

Attracteur=
[(1, 1), (2, 0), (3, 0), (4, 0), (5, 1), (6, 0), (7, 0), (8, 0), (9, 1)]

```

On obtient un résultat conforme à l'analyse faite antérieurement sur ce jeu. Le joueur  $J_0$  possède une stratégie gagnante depuis les sommets 8, 7, 6, 4, 3, 2. Les sommets 9 et 5 contrôlés par

$J_1$  sont également dans l'attracteur  $\text{Attr}^0(V_0)$ . Si on considère que le joueur  $J_0$  commence en retirant 1 ou 2 bâtonnets, il dispose d'une stratégie gagnante à condition que le joueur  $J_1$  joue mal son premier coup (le second coup de la partie) en le plaçant sur un des sommets 7 ou 6.

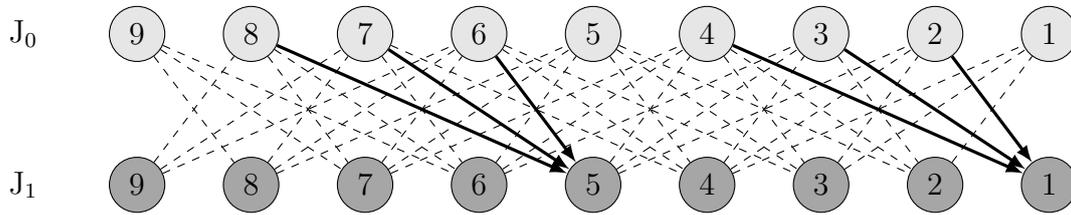


FIGURE 23 – Stratégie gagnante pour le joueur  $J_0$

**Remarque :** Concernant le choix d'implémentation, on pourrait critiquer le fait d'utiliser une liste plutôt qu'un dictionnaire pour la variable  $A$  qui reçoit l'attracteur. En effet, la version

```
def attracteur(G,S_i,U):
    A=[]
    [...]
    def parcours(u):
        if u not in A:
            A[u]=True
            [...]
    [...]
```

est théoriquement meilleure puisque le test d'appartenance `u not in A` avec  $A$  un dictionnaire est en complexité  $O(1)$  contre une complexité en  $O(\text{len}(A))$  si  $A$  est une liste. Dans les faits, le calcul d'attracteur n'est effectué que sur des « petits » graphes de jeu qu'on peut construire intégralement et l'utilisation d'une liste qui permet une meilleure lisibilité du résultat n'est donc pas réellement pénalisante.

### III Algorithme du minimax

Dans toute cette partie, on considère un jeu d'accessibilité fini à deux joueurs  $J_0$  et  $J_1$ .

#### 1 Arbre de jeu

On peut représenter un jeu d'accessibilité fini à deux joueurs par un *arbre de jeu*. On parle alors de représentation sous *forme extensive*. Par exemple, pour le jeu de Nim avec un tas initial de 6 bâtonnets et le joueur  $J_0$  qui commence, le graphe biparti est décrit par

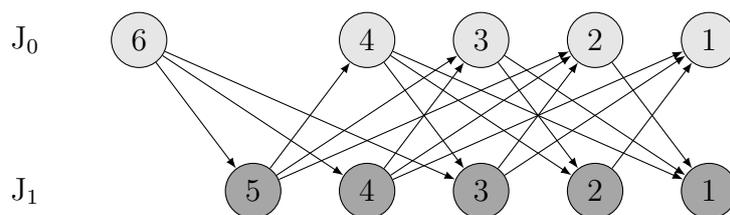


FIGURE 24 – Graphe biparti du jeu de Nim



**Définition 18.** Une arborescence aussi appelée arbre enraciné est un graphe orienté vérifiant :

- le graphe sous-jacent (celui obtenu en désorientant les arcs) est un arbre ;
- il existe un sommet de degré entrant nul et tout autre sommet a un degré entrant égal à 1.

Le sommet de degré entrant nul est appelé racine de l'arbre enraciné.

**Définition 19.** Soit  $(S, A)$  un arbre enraciné et  $(x, y) \in A$ . Le sommet  $x$  est dit père de  $y$  et le sommet  $y$  est dit fils de  $x$ . Deux sommets ayant même père sont dits frères. Un sommet ayant au moins un fils est appelé nœud interne.

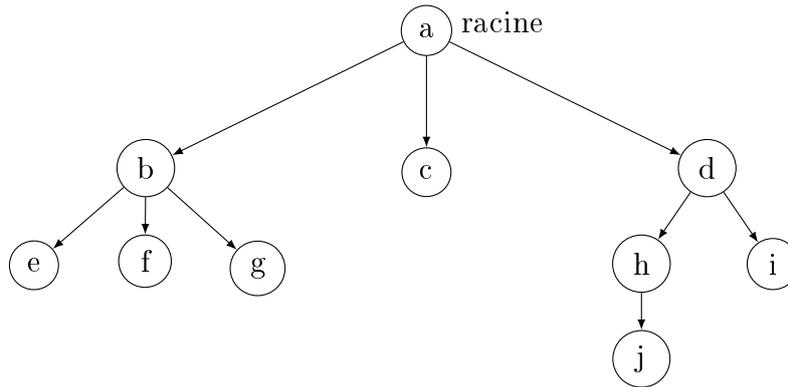


FIGURE 27 – Arbre enraciné

Le sommet  $a$  est la racine de cet arbre enraciné, les sommets  $b$ ,  $c$  et  $d$  sont ses fils et sont donc frères. Les sommets  $e$ ,  $f$ ,  $g$ ,  $j$  et  $i$  sont les feuilles de l'arbre.

**Définition 20.** Soit  $(S, A)$  un arbre enraciné et  $x \in S$ . La profondeur du sommet  $x$  est la longueur de l'unique chemin de la racine à  $x$ . Un niveau dans un graphe orienté est l'ensemble des sommets ayant une profondeur donnée.

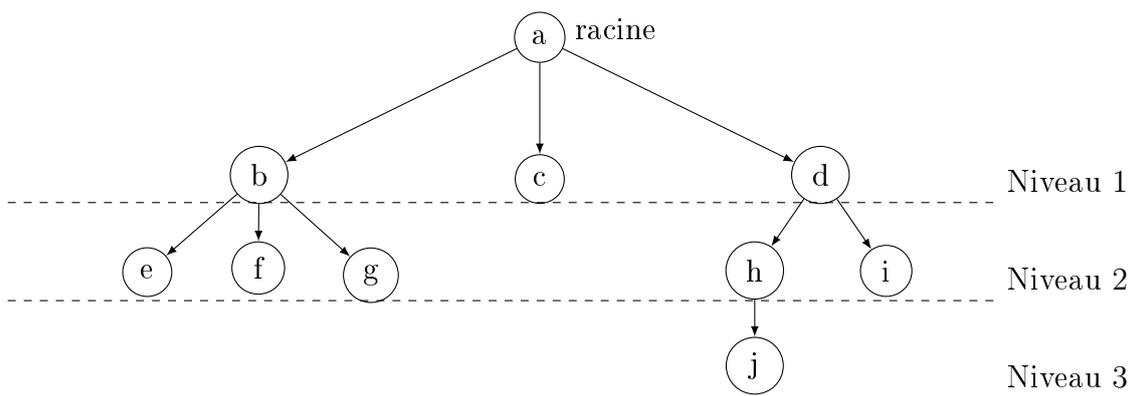


FIGURE 28 – Niveaux de profondeur

Sur l'arbre enraciné de la figure 28, le sommet  $j$  est à une profondeur égal à 3. Les sommets  $e$ ,  $f$ ,  $g$ ,  $h$  et  $i$  sont sur un même niveau de profondeur égal à 2.

**Définition 21.** Soit  $(S, A)$  un arbre enraciné et  $x \in S$ . Une branche de  $x$  est un chemin partant de  $x$  et allant jusqu'à une feuille.

Pour la plupart des jeux intéressants, le graphe de jeu est tellement énorme qu'il est inenvisageable de le construire. Considérer le point de vue d'une arborescence permet, sans chercher à la construire intégralement, d'envisager une méthode pour faire jouer la machine contre un autre joueur (ou contre elle-même si on souhaite se divertir).

## 2 Minimax

Lors d'une partie où chaque joueur joue de manière optimale, le joueur dont c'est le tour cherche la meilleure configuration sachant que son adversaire fera de même quand ce sera son tour.

En considérant l'arbre de jeu enraciné, on définit une fonction score  $\mathcal{S}$  sur l'ensemble de ses feuilles par

$$\mathcal{S}(f) = \begin{cases} +\infty & \text{si } f \text{ feuille gagnante pour } J_0 \\ -\infty & \text{si } f \text{ feuille gagnante pour } J_1 \\ 0 & \text{si } f \text{ feuille de match nul} \end{cases}$$

Dans le cas du jeu de Nim avec un tas initial de 5 bâtonnets et le joueur  $J_0$  qui commence, on obtient l'arbre de jeu :

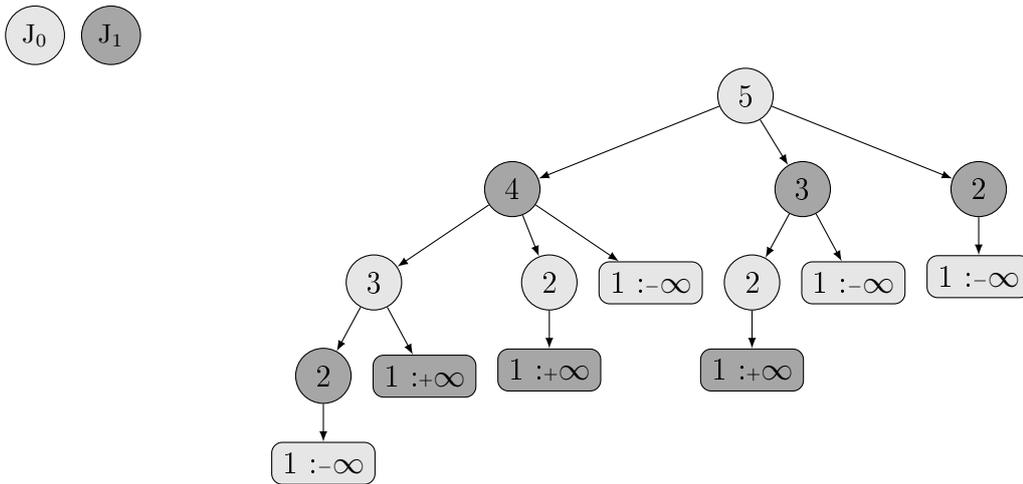


FIGURE 29 – Score des feuilles

Ensuite, on généralise cette fonction score aux autres nœuds en une fonction dite *fonction d'évaluation* notée  $\mathcal{E}$  selon le principe énoncé plus haut qui amène naturellement à une définition récursive :

$$\mathcal{E}(s) = \begin{cases} \mathcal{S}(s) & \text{si } s \text{ est une feuille} \\ \max \{ \mathcal{E}(u), u \text{ fils de } s \} & \text{si } s \text{ nœud interne contrôlé par } J_0 \\ \min \{ \mathcal{E}(u), u \text{ fils de } s \} & \text{si } s \text{ nœud interne contrôlé par } J_1 \end{cases}$$

En suivant ce schéma récursif, on détermine, en allant de gauche à droite dans l'arborescence du jeu de Nim, l'ensemble des valeurs de la fonction d'évaluation. Après avoir complètement parcouru les branches issues du nœud 4 à gauche, on obtient :

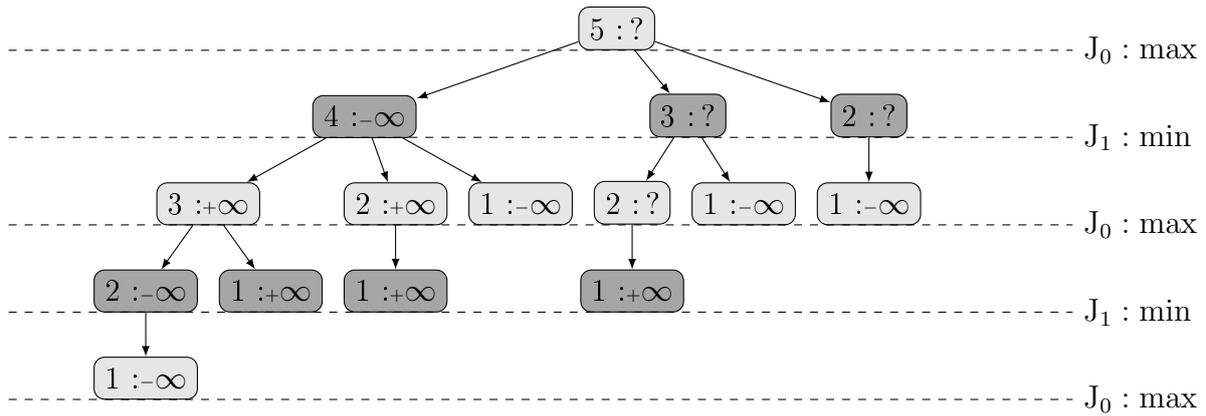


FIGURE 30 – Construction de la fonction d'évaluation

Après détermination complète de la fonction d'évaluation, on a :

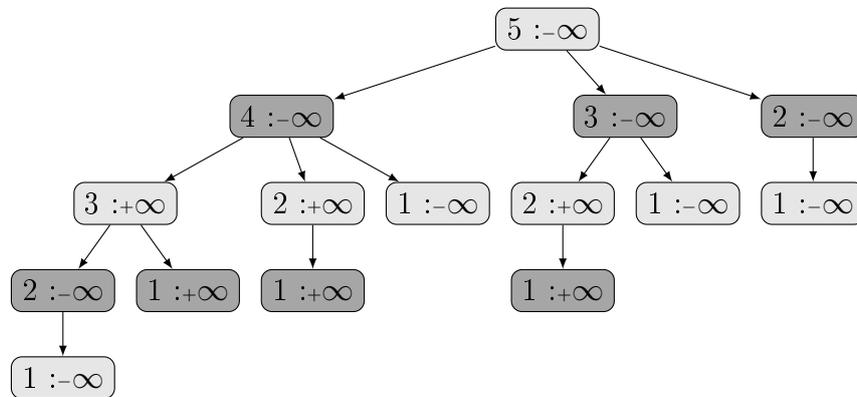


FIGURE 31 – Fonction d'évaluation

On retrouve le résultat déjà connu à savoir que pour un tas de  $5 = 4 + 1$  bâtonnets, le joueur qui ne commence pas a une stratégie gagnante. Mais la différence réside dans le fait qu'on n'a pas instruit la machine d'une telle stratégie : le meilleur coup à jouer est déterminé par la valeur de la fonction d'évaluation, sans expertise *a priori* sur le jeu.

**Définition 22.** *L'algorithme minimax est l'algorithme de calcul récursif de la fonction d'évaluation d'un arbre de jeu.*

---

**Algorithme 2** : Minimax

---

**Entrées** :  $s$  un nœud de l'arbre,  $J_i$  le joueur qui contrôle le nœud  $s$

**Résultat** :  $\mathcal{E}(s)$

**fonction** Minimax( $s, J_i$ ) :

```
    si  $s$  est une feuille alors
    | retourner  $\mathcal{S}(s)$ 
    sinon
    |  $aux \leftarrow []$ ;
    | pour  $u$  fils de  $s$  faire
    | |  $aux \leftarrow aux + [ \text{Minimax}(u, \text{adversaire}(J_i)) ]$ ;
    | si  $J_i = 0$  alors
    | | retourner  $\max(aux)$ 
    | sinon
    | | retourner  $\min(aux)$ 
```

---

On peut critiquer la construction de la liste  $aux$  puisqu'à terme, il s'agit uniquement de déterminer son maximum ou son minimum. On pourrait, selon le joueur, actualiser la valeur du maximum ou minimum au cours de la boucle sans stocker la totalité des résultats pour l'ensemble des fils de  $s$ . Dans les situations pratiques qu'on va rencontrer, cette construction de  $aux$  ne sera pas spécialement pénalisante.

Pour le jeu de Nim, on propose l'implémentation suivante :

```
def score(x):
    """score(x:int)->int
    Si J_0 a un seul batonnet, il perd -> score=-infini
    Si J_1 a un seul batonnet, J_0 gagne -> score =+infini"""
    return (2*x-1)*float('inf')
```

```
def adversaire(x):
    return 1-x
```

```
def minimax(tas, J_i):
    """minimax(tas:int, J_i:int)->float
    Renvoie la fonction d'évaluation du jeu pour une configuration donnée.
    tas : nombre de batonnets, J_i : joueur dont c'est le tour"""
    if tas==1:
        return score(J_i)
    else:
        aux=[]
        for k in range(1,4):
            reste=tas-k
            if reste>0:
                aux.append(minimax(reste, adversaire(J_i)))
        if J_i==0:
            return max(aux)
        else:
            return min(aux)
```

Dans le cas d'une partie opposant un joueur  $J_0$  contre la machine  $J_1$ , on programme la machine pour que celle-ci calcule la fonction d'évaluation sur les nœuds « jouables » qui correspondent à l'ensemble des coups que la machine peut jouer. Il lui suffit ensuite de choisir un nœud pour lequel le minimum de la fonction d'évaluation sur l'ensemble des nœuds jouables est atteint.

### 3 Heuristique

L'algorithme du minimax réalise un parcours en profondeur de l'arbre de jeu. Si celui-ci est très grand, le temps de parcours devient rédhibitoire. Une stratégie simple consiste alors à limiter la profondeur des récursions. Mais dans ce cas, quand on atteint la profondeur limite, il faut pouvoir proposer une alternative au calcul exact de la fonction d'évaluation. On utilise pour cela une *heuristique* qu'on note  $\mathcal{H}$  qui pour, un nœud  $s$  donné, renvoie une valeur  $\mathcal{H}(s)$  qui correspond à une estimation de  $\mathcal{E}(s)$  basée sur une certaine expertise du jeu.

---

**Algorithme 3** : Minimax\_depth

---

**Entrées** :  $s$  un nœud de l'arbre,  $J_i$  le joueur qui contrôle le nœud  $s$

**Résultat** : Estimation de  $\mathcal{E}(s)$

**fonction** Minimax\_depth( $s, J_i, p$ ) :

```

| si  $s$  est une feuille alors
|   retourner  $\mathcal{S}(s)$ 
| sinon si  $p = 0$  alors
|   retourner  $\mathcal{H}(s)$ 
| sinon
|    $aux \leftarrow []$ ;
|   pour  $u$  fils de  $s$  faire
|      $aux \leftarrow aux + [ \text{Minimax\_depth}(u, \text{adversaire}(J_i), p - 1) ]$ ;
|   si  $J_i = 0$  alors
|     retourner  $\max(aux)$ 
|   sinon
|     retourner  $\min(aux)$ 

```

---

## Références

- [1] <https://gfredericks.com/blog/767>
- [2] [https://fr.wikipedia.org/wiki/Jeu\\_de\\_Wythoff](https://fr.wikipedia.org/wiki/Jeu_de_Wythoff)
- [3] Laraki R., Renault J. et T. Tomala, *Théorie des Jeux*, X-UPS 2006, Éditions de l'École Polytechnique
- [4] Hans Peters, *Game Theory, a Multi-Levelled Approach*, Springer, 2015
- [5] Giacomo Bonamo, *Game Theory*, CreateSpace, 2018
- [6] A. Artasanchez, P. Joshi, *Artificial Intelligence with Python*, Packt, 2020