Algorithme pour l'étude des jeux

Sommaire

Le but de ce chapitre et de voir une manière usuelle de trouver des stratégies gagnantes à des jeux à deux joueurs au tour par tour.

I Représentation par des graphes, vocabulaire

Définition 6.1: prédécesseurs, successeurs

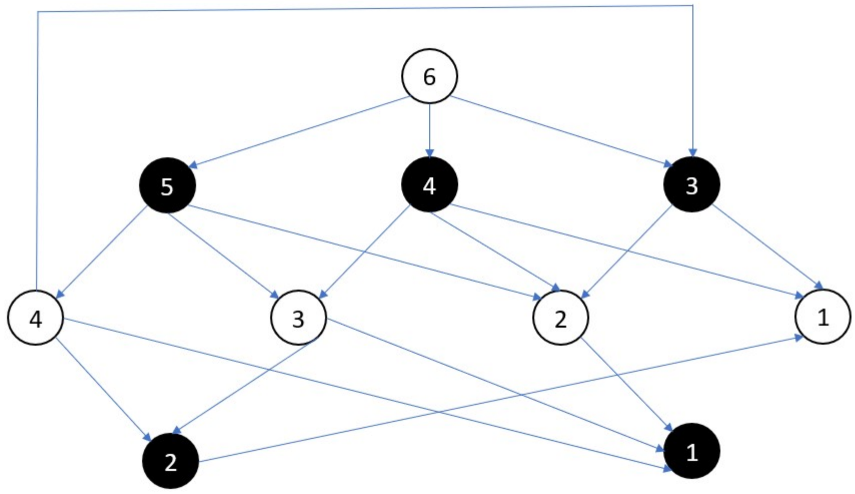
Soit $G = (S, A)$ un graphe orienté. Pour tout $s \in S$, les successeurs de s sont les extrémités des arcs dont s est l'origine. Les prédécesseurs de s sont les origines des arcs dont s est l'extrémité. Un sommet est terminal s'il n'a pas de successeur (c'est à dire si son degré sortant est nul).

Définition 6.2: graphe biparti

Soit $G = (S, A)$ un graphe (orienté ou pas, étiqueté ou pas). On dit que G est biparti s'il existe deux sous-ensembles de sommets, S_0 et S_1, formant une partition de S et tels que pour tout arc ou arête a $\in A$, l'origine et l'extrémité sont dans des S_i différents.

Exemple 6.3

Le graphe représentant une partie de Nim est un graphe biparti : si on note S_0 et S_1 les ensembles de sommets contrôlés par les joueurs J_0 et J_1 respectivement, on constate qu'il n'y a aucun arc de S_i vers lui-même.



GRAPHE DE LA PARTIE POUR UN NOMBRE DE BÂTONS DE SIX. LE NUMÉRO DU SOMMETS CORRESPOND AU NOMBRE DE BÂTONS RESTANTS. SI LE JOUEUR J_0 JOUE EN PREMIER, S_0 EST L'ENSEMBLE DES SOMMETS EN BLANC, S_1 L'ENSEMBLE DES SOMMETS EN NOIR.

Notation 6.4. *Par la suite, on notera xN le sommet de numéro x en noir, xB pour celui en blanc.*

Définition 6.5: graphe de jeu ou arène pour deux joueurs

Un graphe de jeu à deux joueurs ou arène est une structure $\mathcal{G} = (G, S_0, S_1)$ où $G = (S, A)$ est un graphe fini et biparti, S_0, S_1 sont comme dans la définition précédente. Par convention, on associe à tout graphe de jeu deux joueurs J_0 et J_1, et on dit que S_i est l'ensemble des sommets contrôlés par le joueur J_i.

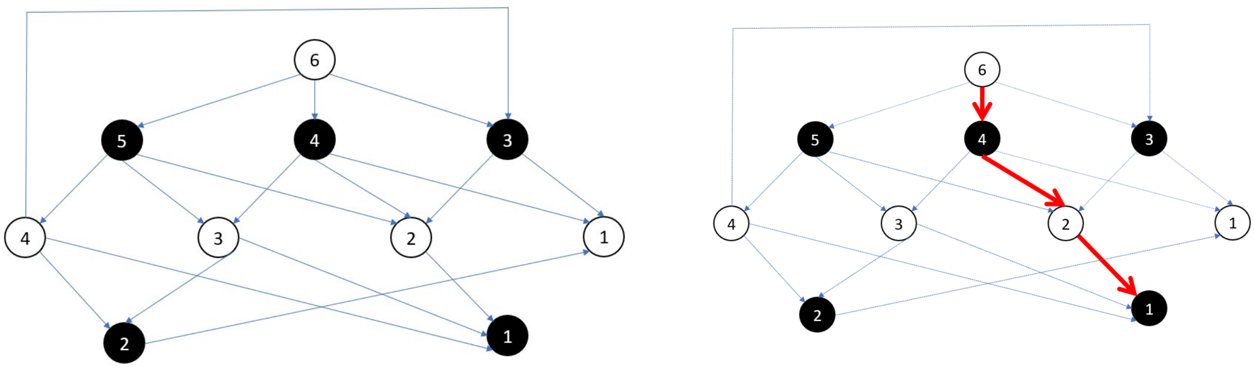
Exemple 6.6

Exemples : Revenons au jeu de Nim : le joueur J_0 joue le premier et contrôle S_0, l'ensemble des sommets blancs. J_1 contrôle les autres sommets. Pour J_i, contrôler un sommet ou un état, signifie que c'est à son tour de jouer lorsque la partie est dans cet état.

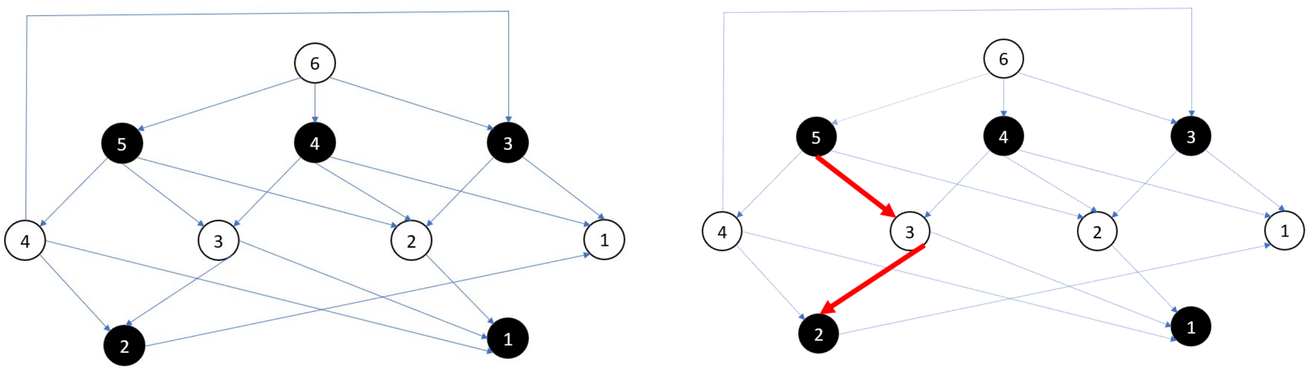
Les graphes de jeux bipartis tels que nous les avons définis permettent de représenter des jeux dans lesquels deux joueurs jouent en alternance, où chaque coup conduit à une position contrôlée par le joueur adverse. Maintenant que le déroulement du jeu est en place (l'ensemble des coups permis est égal à A), il nous reste à définir formellement les **parties**, les **conditions de victoire**. Nous pourrons alors parler de **parties gagnantes,** de **positions gagnantes** et de **stratégies**.

Définition 6.7

- Un chemin dans un graphe orienté G, est **maximal** lorsqu'il est infini ou lorsque l'extrémité de son dernier arc est terminale
- Soit $\mathcal{G} = (G, S_0, S_1)$ un graphe de jeu. On appelle "partie débutant en s_0" un chemin maximal de G et dont le premier arc a pour origine s_0.
- Une **partie partielle** débutant en s_0 est un chemin fini dont le premier arc a pour origine s_0.



EXEMPLE DE CHEMIN MAXIMAL DANS LE GRAPHE ORIENTÉ PRÉCÉDENT : (6B,4N,2B,1N) EST UN CHEMIN MAXIMAL CAR 1N EST TERMINAL. C'EST UNE PARTIE DÉBUTANT EN 6B



LE CHEMIN MIS EN ÉVIDENCE EST UNE PARTIE PARTIELLE DÉBUTANT ÉGALEMENT EN 5N.

Notation 6.8. *Nous noterons $\mathcal{P}$ l'ensemble des parties, $\mathcal{P}^*$ l'ensemble des parties partielles, $\mathcal{P}_0, \mathcal{P}_1$ les ensembles formés des parties partielles qui aboutissent à un sommet/état/position appartenant à S_0 ou S_1 respectivement. On pourra représenter une partie par la succession des sommets visités : $P = (s_0, s_1, \dots, s_{p-1})$ représente donc le chemin $(s_0, s_1), (s_1, s_1), \dots, (s_{p-2}, s_{p-1})$.*

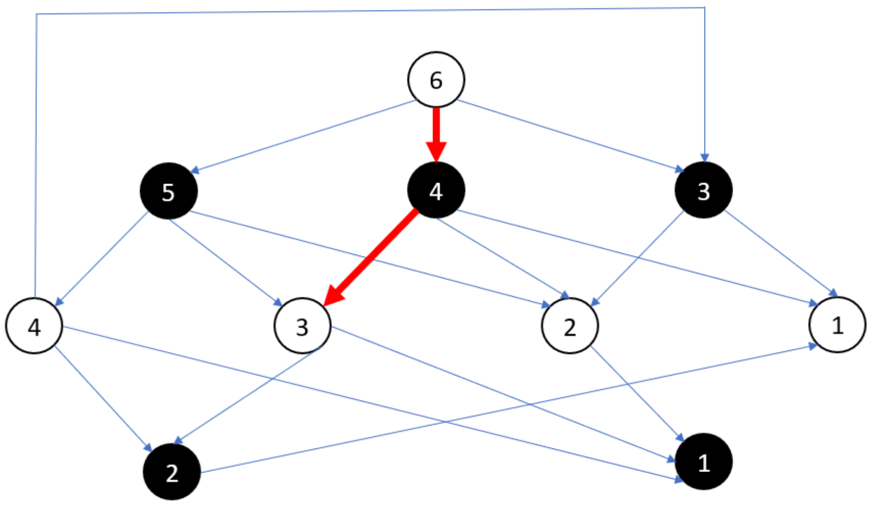
Définition 6.9: stratégies, stratégies sans mémoire

Soit $((S, A), S_0, S_1)$ un graphe de jeu.

- Une stratégie pour le joueur J_i est une application $\sigma : \mathcal{P}_i \longrightarrow S_{1-i}$ qui à toute partie partielle $(s_0, s_1, \dots, s_{p-1})$ aboutissant en $s_{p-1} \in S_i$, associe un sommet s_p tel que $(s_{p-1}, s_p) \in A$. (s_{p-1}, s_p) est donc un coup jouable pour J_i.
- Une partie $(s_0, s_1, \dots, s_t)$ est conforme à une stratégie pour le joueur i, si pour tout k tel que $0 \leq k < t$ et $s_k \in S_i$ on a $s_{k+1} = \sigma(s_0, s_1, \dots, s_k)$.
- Une stratégie sans mémoire est une stratégie $\sigma : \mathcal{P}_i \longrightarrow S_{1-i}$ telle que pour tout couple de parties partielles $(s_0, s_1, \dots, s_k), (v_0, v_1, \dots, v_\ell) \in \mathcal{P}_i^2$,
$$s_k = v_\ell \Rightarrow \sigma((s_0, s_1, \dots, s_k)) = \sigma((v_0, v_1, \dots, v_\ell))$$

Disposer d'une stratégie σ permet de savoir quel coup (unique, car σ est une application) jouer en toute circonstance du jeu. Dire qu'une stratégie est sans mémoire c'est dire que pour jouer un coup dans une partie conforme, seul l'état actuel du jeu est à prendre en considération. Il existe une règle aux échecs qui nous dit qu'une partie est nulle si une même position des pièces sur l'échiquier se répète trois fois. Si on considère que les états sont les différentes positions des pièces, il n'y a donc pas de stratégie utile sans mémoire.

Si on continue avec l'exemple du jeu de Nim, imaginons qu'on a effectuer la partie partielle (6B,4N,3B).



Une stratégie σ pour le joueur J_0, va être de choisir en connaissant la partie partielle (6B,4N,3B) le choix à effectuer pour le joueur. Ici, le joueur doit prendre deux bâtons pour gagner la partie, d'où une stratégie "intéressante" sera d'avoir $\sigma((6B, 4N, 3B)) = 1N$, ce qui est possible car $(3B, 1N) \in A$.
De manière générale, une "bonne" stratégie pour le jeu de Nim consiste à faire en sorte qu'il reste toujours un multiple de 4 + 1 bâtons sur la table, ce qui est toujours possible lorsqu'on débute (si on est le joueur J_0 ici). Cette stratégie est sans mémoire : elle n'utilise que l'état actuel du jeu (le nombre de bâtons restants sur la table).

Définition 6.10: condition de gain, position gagnante

Soit $((S, A), S_0, S_1)$ un graphe de jeu.

- Une condition de gain pour le joueur J_i dans ce jeu est un ensemble de parties $\Omega_i \subset \mathcal{P}$: J_i gagne une partie π ssi $\pi \in \Omega_i$.
- Une stratégie est gagnante pour le joueur J_i si toute partie conforme à la stratégie est gagnée par J_i.
- Une stratégie est gagnante à partir d'une position d pour le joueur J_i si toute partie conforme à la stratégie et qui débute en d est gagnée par J_i (appartient à Ω_i). On dit que d est une position gagnante pour J_i.

L'ensemble des parties gagnantes pour J_0 est l'ensemble des parties dont le dernier sommet est 1N :

- (6B,5N,4B,1N)
- (6B,5N,3B,1N)
- (6B,5N,2B,1N)
- (6B,4N,2B,1N)
- (6B,4N,3B,1N)
- (6B,3N,2B,1N)
- (6B,5N,4B,3N,2B,1N)

II Calcul des attracteurs

II.1 Jeu d'accessibilité

Définition 6.11: jeu d'accessibilité

Soit $\mathcal{G} = ((S, A), S_0, S_1)$ un graphe de jeu à deux joueurs, et Ω_i une condition de gain sur $\mathcal{G}$ pour le joueur J_i. On dit que le jeu ainsi défini est un jeu d'accessibilité lorsque

- il existe un ensemble de sommets terminaux F_i (ensemble cible) tel que les parties gagnantes pour J_i sont celles qui se terminent sur un sommet de F_i, on a donc
$$\Omega_i = \{(s_0, \ldots, s_t) \in \mathcal{P}; s_t \in F_i\}$$
- il n'y a pas de match nul (c'est à dire que $\Omega_{1-i} = \mathcal{P} \backslash \Omega_i$).

C'est le cas pour le jeu de Nim, toute partie mène aux sommets 1N ou 1B car dans une partie, le numéro du sommet décroit strictement.

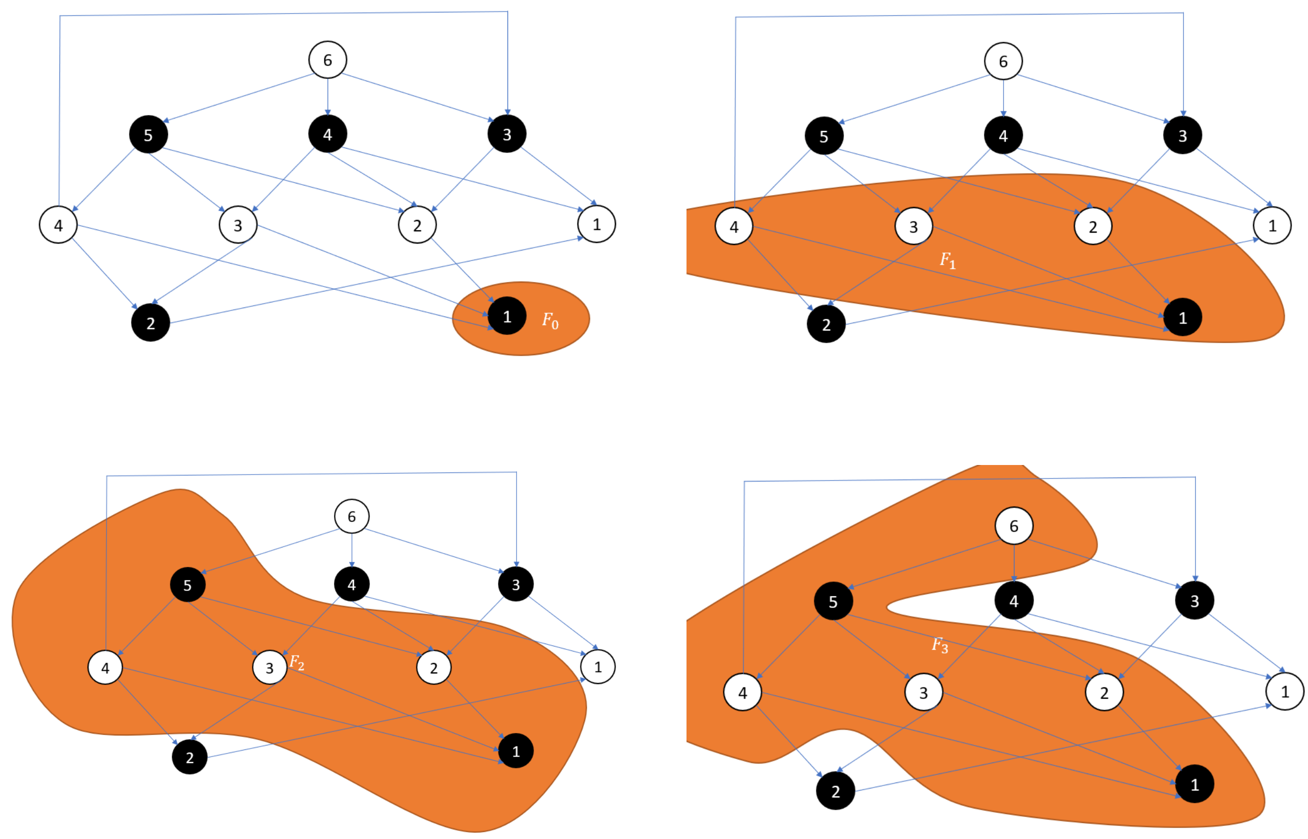
II.2 Attracteurs et pièges

Considérons un jeu d'accessibilité dans lequel la condition de victoire pour le joueur J_0 est que la partie aboutisse à F_0. Pour une position donnée, J_0 est à un coup de gagner avec certitude, soit si c'est à son tour de jouer et s'il peut choisir un coup qui le conduit à F_0, soit si c'est à son adversaire de jouer et que tous les coups jouables mènent à F_0.

Cela nous incite à construire l'ensemble des positions gagnantes dans un tel jeu de la façon suivante :

1. On définit deux applications de des parties de S dans lui-même, en posant pour tout ensemble de sommets, $X \subset S$:
$$\mathrm{Pr}(X) = \left\{s \in S_0, \exists \left(s, s'\right) \in A, s' \in X\right\} \cup \left\{s \in S_1, \forall \left(s, s'\right) \in A, s' \in X\right\}$$
et $\mathcal{F}(X) = X \cup \mathrm{Pr}(X)$
$\mathrm{Pr}(X)$ est l'ensemble des positions qui permettent au joueur J_0 ou qui imposent au joueur J_1 d'amener le jeu en X.
2. On définit alors par récurrence une suite d'ensembles en posant :
   (a) $A_0 = F_0$;
   (b) $A_{n+1} = A_n \cup \mathrm{Pr}\left(A_n\right) = \mathcal{F}\left(A_n\right)$.

Observons que, lorsque $s \in A_i$, le joueur J_0 est à i coups au plus de la victoire. La suite $(A_i)_i$ est croissante pour l'inclusion et comme S est fini, elle est stationnaire : il existe un rang n_0 à partir duquel $A_{n_0} = A_{n_0+p}$ pour tout $p \in \mathbb{N}$. On note $\mathrm{Attr}\left(F_0, J_0\right) = A_{n_0}$.



CALCUL DE L'ATTRACTEUR DU COUPLE (F_0, J_0) QUI EST F_3

Définition 6.12: attracteur et rang

- L'ensemble $\text{Attr}\,(F_0, J_0)$ est le bassin d'attraction (ou attracteur) de F_0 pour le joueur J_0.
- Pour tout $s \in \text{Attr}\,(F_0, J_0)$, on définit le rang de s : $rg(s) = \min\,\{i \in \mathbb{N}, s \in A_i\}$. Lorsque $s \notin \text{Attr}\,(F_0, J_0)$, on pose : $rg(s) = \infty$.

THÉORÈME 6.13: BASSIN ATTRACTION = POSITION GAGNANTE

T

Soit $\mathcal{G} = (S, A, S_0, S_1)$ un graphe de jeu d'accessibilité à deux joueurs pour lequel F_0 est la condition de gain pour le joueur J_0. Alors,

- $W_0 = \text{Attr}\,(F_0, J_0)$ est l'ensemble des positions gagnantes pour le joueur J_0.
- $W_1 = S \backslash \text{Attr}\,(F_0, J_0)$ est l'ensemble des positions gagnantes pour le joueur J_1.

Remarque 6.14. *Ce théorème nous dit que le bassin d'attraction* $\text{Attr}\,(F_0, J_0)$ *est l'ensemble des positions gagnantes pour* J_0*. Un algorithme, avec une complexité linéaire en* $|S| + |A|$*, permet de le calculer.*

III Algorithme du minimax, heuristiques

Nous envisageons maintenant l'étude des jeux en déployant l'ensemble des parties possibles comme un arbre. Un nœud représentant un état, les branchements, les différents coups permis au joueur qui contrôle cet état.

III.1 Arbres

Nous donnons deux définitions équivalentes de la structure d'arbre. La première les présente comme des graphes particuliers, la seconde liée à la théorie des ensembles.

Définition 6.15

Un arbre est un graphe orienté dans lequel

- il existe un sommet et un seul, appelé racine, n'ayant pas d'antécédent ;
- tous les autres sommets admettent un prédécesseur et un seul.

Définition 6.16

On appelle arbre un ensemble $\mathcal{A}$ non vide, dont les éléments seront appelés nœuds, sur lequel est définie une relation $\mathcal{P}$ ($x\mathcal{P}y$ se dit "x est parent de y") telle que :

- il existe un nœud, r, et un seul, appelé racine, n'ayant pas de parent (c'est-à-dire, tel que $\forall x \in \mathcal{A}, \text{non}(x\mathcal{P}r)$) ;
- tous les autres nœuds admettent un(e) parent et un (e) seul(e) $(\forall y \in \mathcal{A}, \exists ! x \in \mathcal{A}, (x\mathcal{P}y))$

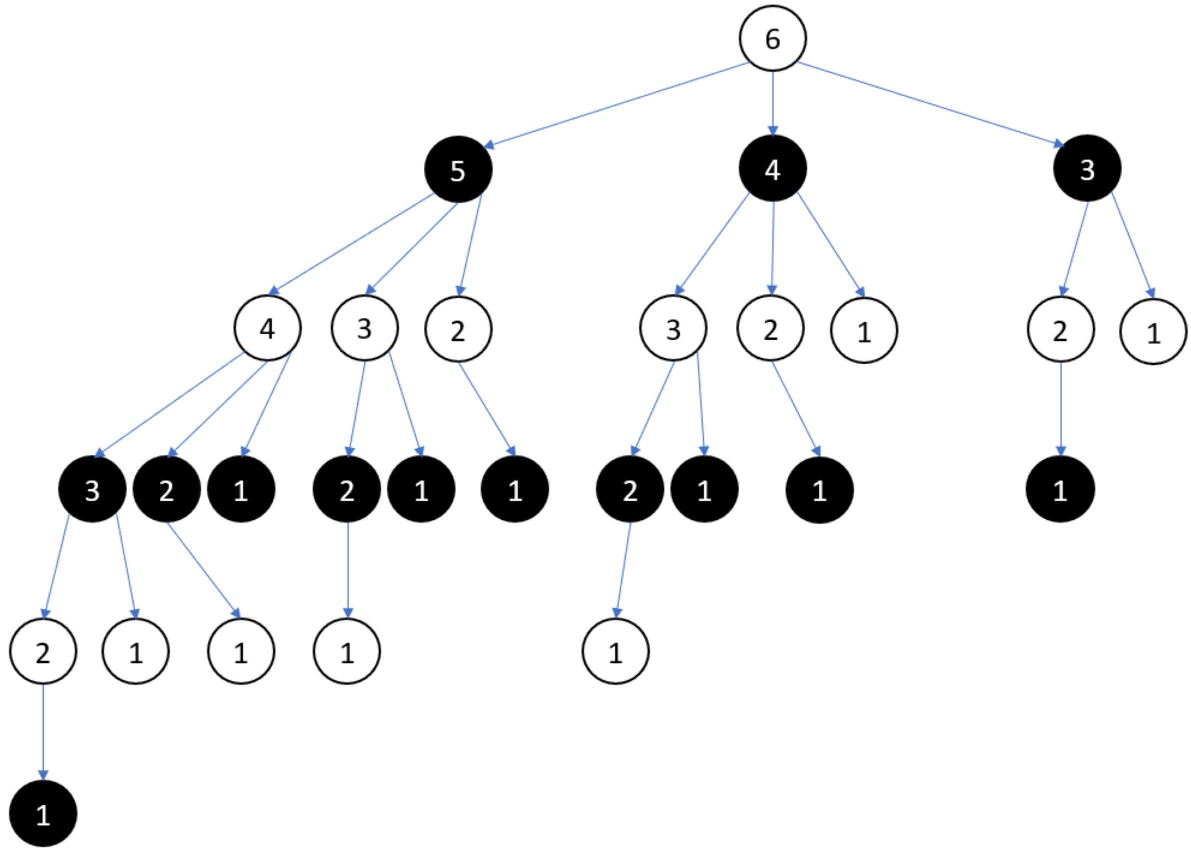
Théorème 6.17

Il existe un unique chemin qui nous permet de remonter d'un nœud quelconque jusqu'à la racine. Formellement : pour tout noeud $x \in \mathcal{A}\backslash\{r\}$, il existe une suite finie $x_0 = r, \ldots, x_p = x$ telle que pour tout i tel que $0 \leq i \leq p-1$, $(x_{i-1}\mathcal{P}x_i)$ (on dit que les x_i sont les ascendants de x).

Définition 6.18

Nous serons amenés à utiliser le vocabulaire suivant :

- les nœuds sans descendant sont les feuilles de l'arbre ou nœuds terminaux ;
- le degré d'un nœud est le nombre de ses descendants ;
- la profondeur d'un nœud est le nombre de ses ascendants stricts ;
- la hauteur d'un arbre est la profondeur maximale de ses nœuds ;
- un arbre est étiqueté lorsqu'à chaque nœud on associe une information, appelée étiquette.



Arbre de l'ensemble des parties possibles pour le jeu de Nim débutant à 6 bâtons, les sommets sont étiquetés selon le joueur qui contrôle (couleur) et le nombre de bâtons restants.

III.2 L'algorithme du minimax

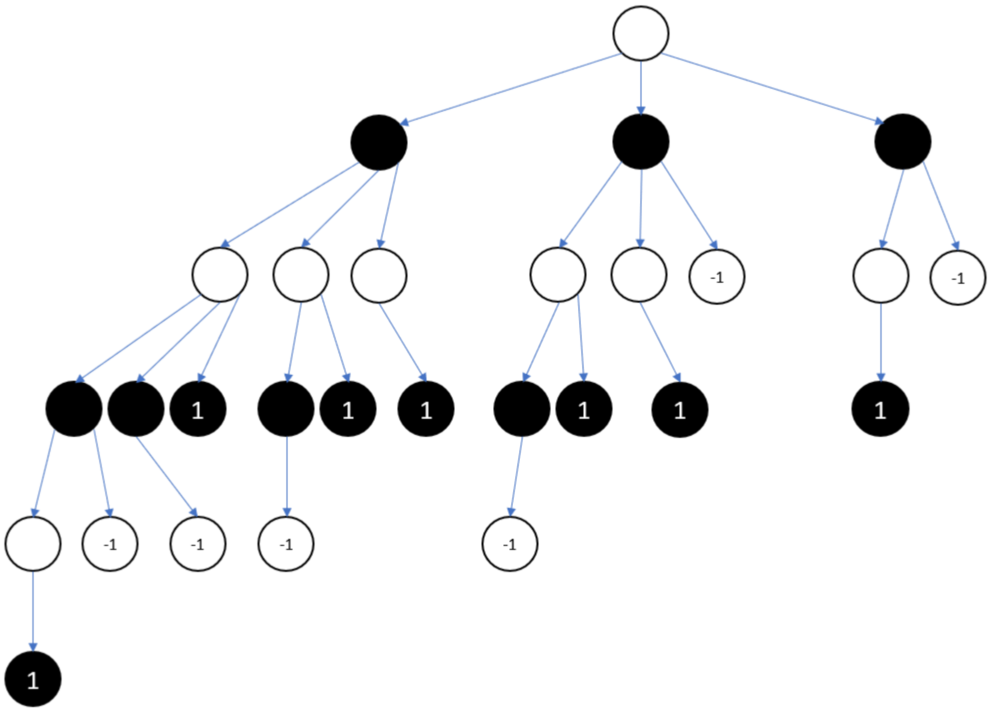
Théorème 6.19: L'algorithme du minimax

On considère, lorsque la complexité du jeu le permet, l'ensemble des parties possibles. Les feuilles d'un arbre de jeu représentent le résultat d'une partie (match gagné par J_0, perdu par J_0 ou nul), on leur attribue donc un score ou une valeur (respectivement $+1$, -1 ou 0 dans les cas où l'on ne considère que l'issue, $+$g, $-$g, 0 si les gains sont variables).

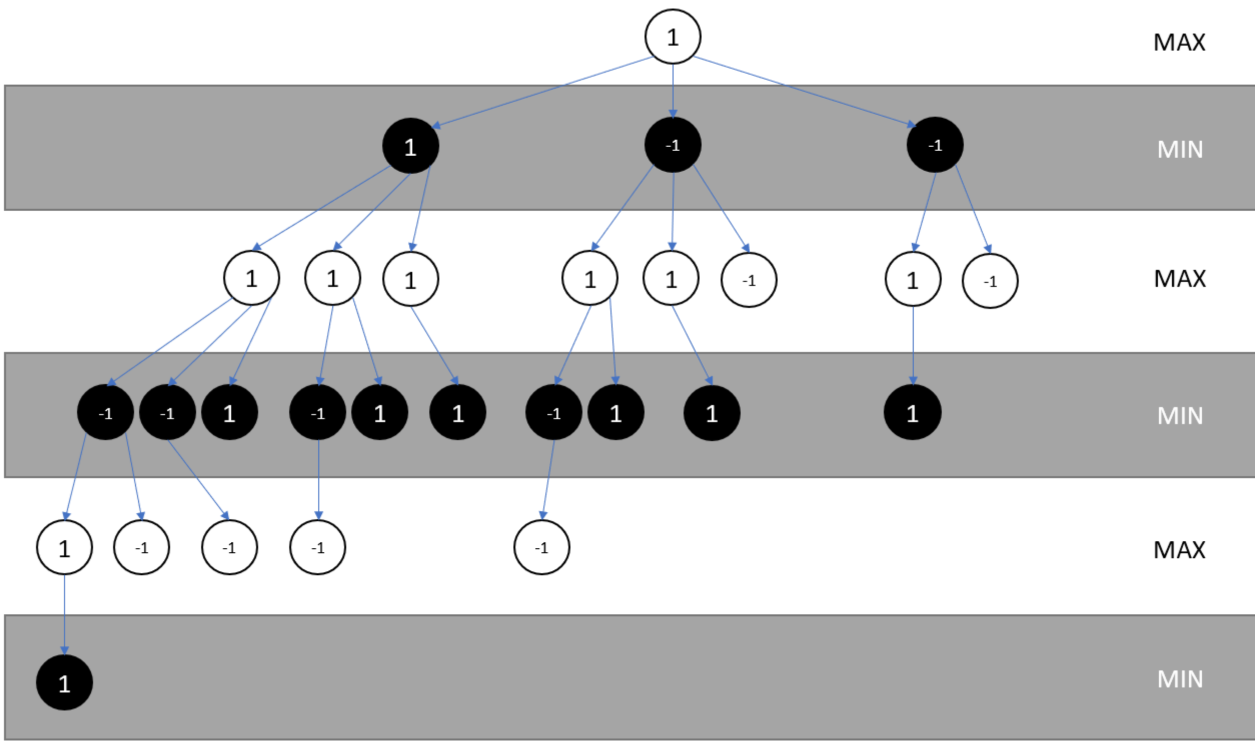
A partir de là, on définit récursivement une valeur pour chaque nœud de l'arbre :

- S'il est contrôlé par J_0, ce sera le maximum des valeurs des sous-arbres ;
- S'il est contrôlé par J_1, ce sera le minimum des valeurs des sous-arbres ;

Alors, lorsqu'un nœud a une valeur positive pour le joueur J_0, celui-ci peut imposer une partie conduisant à la victoire avec ce gain. Le score est calculé en supposant que l'adversaire joue tous ses coups de façon à minimiser ses pertes.



Score initiale dans le cadre du jeu de Nim à 6 bâtons, comme il n'y a pas de gain, les sommets terminals sont initialisés à ± 1



Application de l'algorithme du minmax au jeu de Nim à 6 bâtons, les valeurs dans les sommets sont les scores enregistrés par l'algorithme.

THÉORÈME 6.20: L'ALGORITHME DU MINIMAX AVEC FONCTION D'ÉVALUATION

T

On construit ou explore l'arbre du jeu à partir d'une position donnée en limitant la profondeur des sous-arbres à explorer (3 coups, 5 coups d'avance...). L'algorithme explore donc un arbre de jeu partiel dont les feuilles ne sont plus nécessairement des fins de parties.

On se donne une fonction d'évaluation qui est susceptible d'attribuer une valeur à tous les états du jeu. Aux échecs, elle dépendra de la nature des différentes pièces, de leurs positions (l'occupation du centre de l'échiquier ayant une meilleure valeur stratégique), etc. L'algorithme n'est pas modifié dans sa conception : il opère sur un arbre de jeu partiel et sa fonction d'évaluation est heuristique.

Ce dernier algorithme est utile lorsque l'arbre est beaucoup trop grand pour être parcouru totalement, c'est le cas, par exemple, lors d'une partie d'échec.

Définition 6.21: Heuristique

En informatique, l'heuristique est une technique conçu pour résoudre un problème plus rapidement lorsque les méthodes classiques sont trop lentes, ou pour trouver une solution approximative lorsque les méthodes classiques ne trouvent de solution exacte. Elle permet de trouver une solution dans un délai raisonnable, même si celle-ci n'est pas toujours optimale.