Théorie des jeux

I. Introduction

Dans ce TP, on étudie les jeux présentant l'ensemble des propriétés suivantes.

- Deux joueurs antagonistes jouent alternativement sans pouvoir passer leur tour : le jeu se termine soit par la victoire d'un seul des deux joueurs, soit par un match nul.
- Il y a un nombre fini de configurations du jeu.
- Le jeu est **à information complète** : chaque joueur a une vision complète de l'état du jeu à tout instant.
- Le jeu est déterministe : dans une situation donnée, une décision amène toujours à la même situation.

Exemple
Les hypothèses ci-dessus :

× permettent d'étudier le jeu du morpion, du Puissance 4, les dames, les échecs, ou encore le go.

× excluent l'étude du shifumi, du Monopoly et la plupart des jeux de cartes.

Remarque
On appellera les deux joueurs J_0 et J_1, et on supposera que le joueur J_0 est le premier à jouer.

Dans ce chapitre, nous allons étudier comment modéliser ce type de jeux par des graphes et comment déterminer une stratégie pour l'un ou l'autre des joueurs.

II. Généralités

II.1. Représentation par un graphe biparti

On rappelle qu'un graphe G est un couple (S, A) où :

× l'ensemble S est l'ensemble des sommets du graphe G,

× l'ensemble $A \subset S^2$ décrit l'ensemble des arêtes du graphe G.

Définition

Soit $G = (S, A)$ un graphe.

Le graphe G est dit **biparti** s'il existe une partition de S en deux parties S_0 et S_1 telle qu'aucune arête de S ne relie deux sommets de S_0 ou deux sommets de S_1.

Remarque
Un jeu vérifiant les hypothèses de l'énoncé peut naturellement se représenter par un graphe biparti (S, A) de la manière suivante :

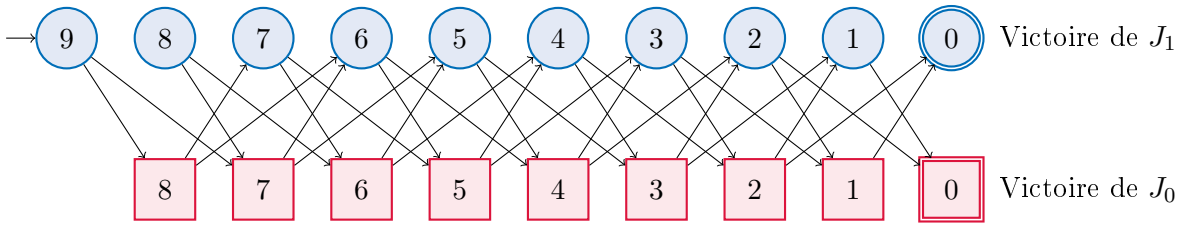
× l'ensemble S_0 des sommets représentant les états du jeu contrôlés par le joueur J_0 ;

× l'ensemble S_1 des sommets représentants les transitions possibles entre les états du jeu, *i.e.* les choix que les joueurs peuvent effectuer à chacun de leur tour.

Le sommet initial représente le début du jeu tandis que les sommets finaux (les sommets sans successeurs) représentent les différentes fins de parties.

Exemple : Le jeu des allumettes
On installe 9 allumettes sur un plateau de jeu. Les joueurs J_0 et J_1 jouent alternativement : durant son tour, chaque joueur peut retirer 1 ou 2 allumettes de la table. Le joueur qui ne peut plus jouer a perdu et l'autre joueur a gagné la partie.

► Représenter ce jeu par un graphe biparti : le numéro de chaque sommet indique le nombre d'allumettes restantes sur le plateau. On représentera les états contrôlés par J_0 par des ronds bleus et les états contrôlés par J_1 par des carrés rouges.



On remarque qu'on aurait pu supprimer le sommet 8 pour le joueur J_0 : il correspond à une situation impossible.

Remarque
De façon générale, l'ensemble des sommets finaux F du graphe biparti (S, A) se partitionne en trois parties :

× l'ensemble F_0 des sommets finaux représentant une victoire de J_0 ;
× l'ensemble F_1 des sommets finaux représentant une victoire de J_1 ;
× l'ensemble F_N des sommets finaux représentant un match nul.

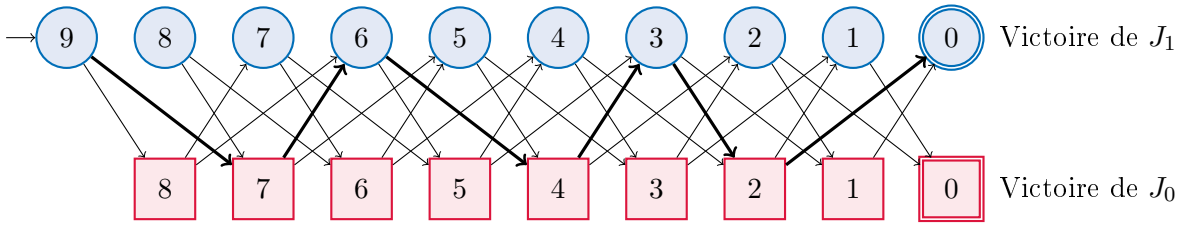
► Dans le jeu des allumettes, décrire les ensembles F_0, F_1 et F_N.

Dans le jeu des allumettes :

× l'ensemble F_0 contient un seul sommet : le sommet rouge 0 ;
× l'ensemble F_1 contient un seul sommet : le sommet bleu 0 ;
× l'ensemble F_N est vide : il n'y a pas de match nul possible.

Remarque
Chaque partie d'un jeu est naturellement représentée par un chemin $(s_0, s_1, \ldots, s_n)$ du graphe (S, A) commençant au sommet initial s_0 et se terminant par un sommet final $s_n \in F$.

► Dans le jeu des allumettes, représenter, un exemple de partie gagnée par le joueur J_1.



II.2. Stratégie et positions gagnantes

Dans cette sous-partie, on considère un jeu modélisé par un graphe biparti (S, A) avec $S = S_0 \cup S_1$. On note toujours F l'ensemble de ses sommets finaux.

Définition Soit $k \in \{0, 1\}$.

Une **stratégie** pour le joueur J_k est une application $\varphi_k : S_k \setminus F \to S_{1-k}$ telle que, pour tout $s \in S_k \setminus F$, le couple $\big(s, \varphi_k(x)\big)$ est une arête du graphe biparti.

Remarque

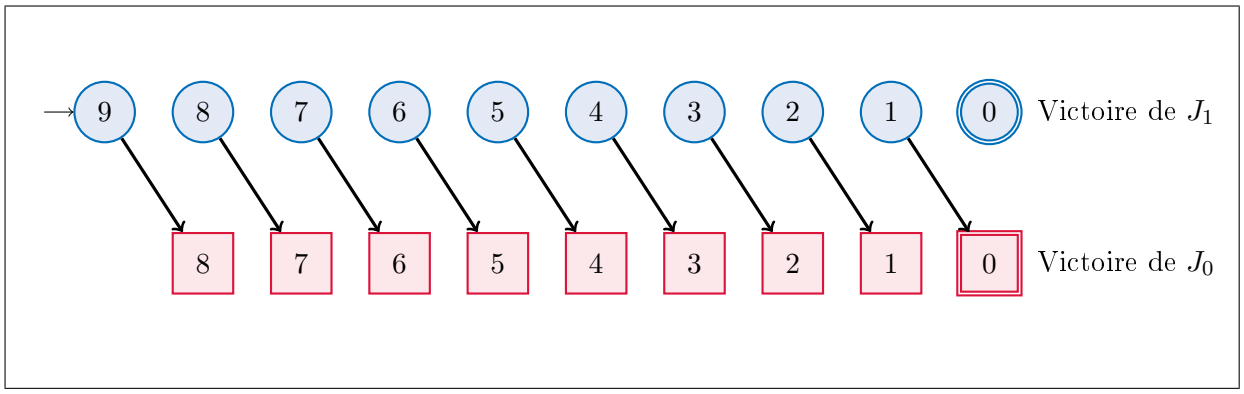
a) Une stratégie pour le joueur J_k est donc la donnée du choix effectué par J_k pour chaque état de jeu qu'il contrôle.

b) Dans le programme de CPGE scientifiques, on ne considère que des stratégies **sans mémoire** : le choix effectué par un joueur ne dépend que de l'état actuel du jeu (et pas de tout l'historique de la partie en cours).

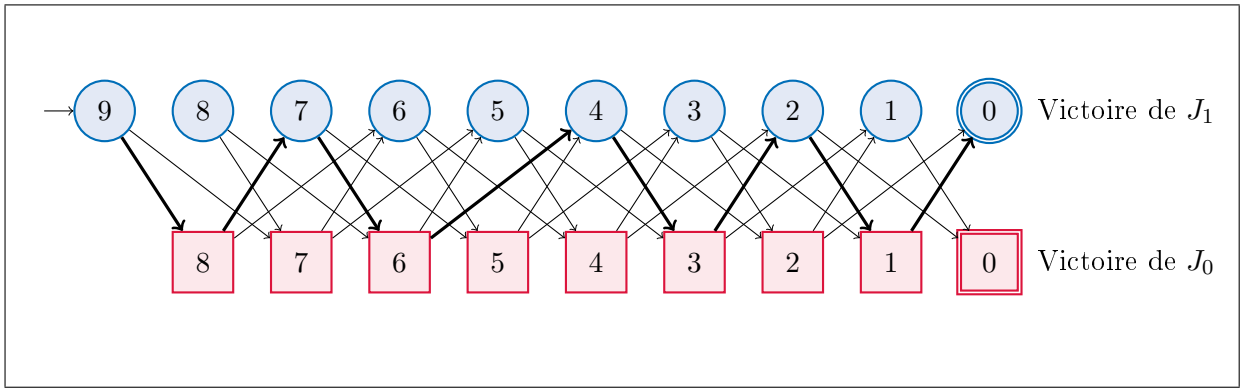
c) Une partie $(s_0, s_1, \ldots, s_n)$ du jeu est dite **jouée suivant la stratégie** φ_k si :

$$\forall i \in [\![0, n-1]\!], \quad s_i \in S_k \;\Rightarrow\; s_{i+1} = \varphi_k(s_i)$$

► En reprenant le jeu des allumettes, représenter la stratégie « stupide » du joueur J_0 consistant à ne prendre qu'une allumette à chacun de ses tours.



► Représenter maintenant une partie jouée où le joueur J_0 a joué suivant la stratégie « stupide ».

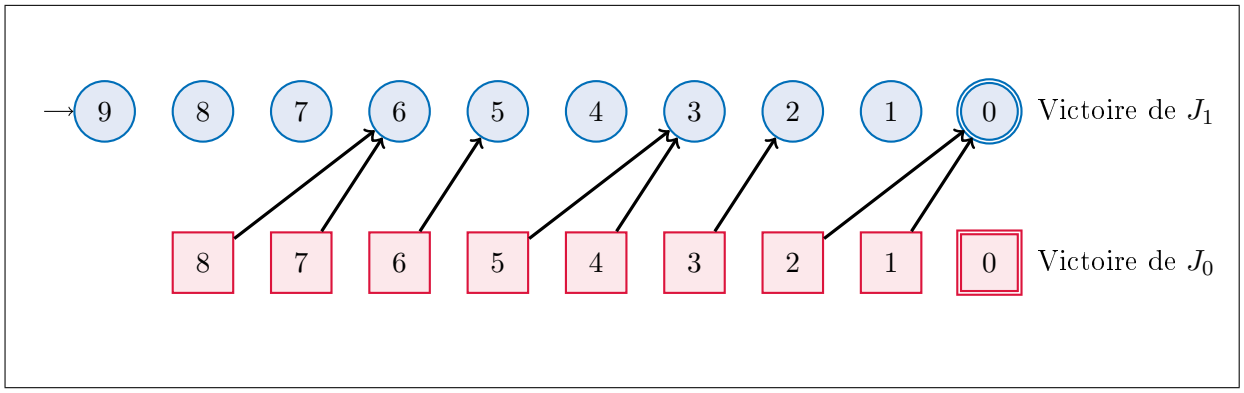


Définition

Soit $k \in \{0, 1\}$.

Une stratégie est dite **gagnante** pour J_k si toute partie jouée en suivant cette stratégie est gagnante pour J_k.

▶ Dans le jeu des allumettes, on peut vérifier que J_0 n'a pas de stratégie gagnante. Proposer une stratégie gagnante pour J_1.

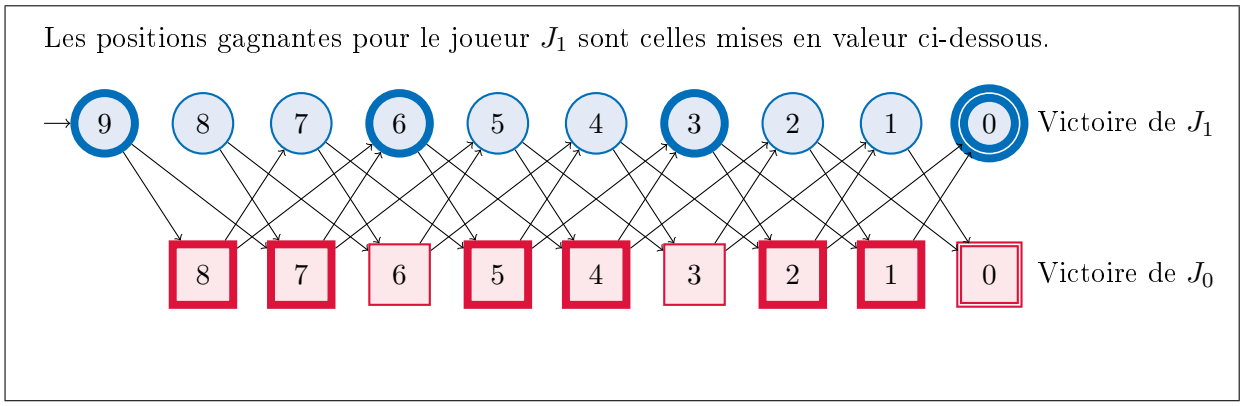


Définition

Soit $k \in \{0, 1\}$.

Une position de jeu $s \in S$ est dite **gagnante** pour J_k lorsqu'il existe une stratégie gagnante pour J_k pour une partie débutant au sommet s.

▶ Dans le jeu des allumettes, quelles sont les positions gagnantes pour le joueur J_1 ?



Remarque

L'ensemble des sommets du graphe biparti (S, A) se partitionne en 3 parties :

× l'ensemble des sommets représentant une position gagnante pour J_0 ;

× l'ensemble des sommets représentant une position gagnante pour J_1 ;

× l'ensemble des sommets pour lesquels aucun joueur n'a de stratégie gagnate : en partant d'un de ces sommets, si les deux joueurs choisissent une stratégie optimale, la partie se finira par un match nul.

II.3. Détermination des positions gagnantes et d'une stratégie gagnante

On considère maintenant un jeu modélisé par un graphe biparti (S, A) où $S = S_0 \cap S_1$.
Pour tout $k \in [0, 1]$, on note F_k l'ensemble de ses sommets finaux représentant une victoire de J_k.
Objectif : Proposer un algorithme donnant l'ensemble des positions gagnantes d'un joueur donné.

Algorithme

Pour déterminer les positions gagnantes de J_0, l'idée est la suivante : on part des sommets de F_0 (on rappelle qu'ils sont tous gagnants pour J_0), puis on « remonte » dans le graphe biparti (S, A) en suivant les règles suivantes :

× un sommet de S_0, menant vers un sommet gagnant pour J_0, est lui-même gagnant pour J_0. En effet, il suffit au joueur de faire ce choix ;

× un sommet de S_1, dont toutes les arêtes en partant mènent vers un sommet gagnant pour J_0, est lui-même gagnant. En effet, le joueur J_1 ne peut alors que faire un choix gagnant pour J_0.

Définition

Soit $k \in \{0, 1\}$.

On définit une suite d'ensemble $\left(\mathscr{A}_i^{(k)}\right)_{i\in\mathbb{N}}$ par :

$$\begin{cases} \mathscr{A}_0^{(k)} = F_k \\ \forall i \in \mathbb{N},\ \mathscr{A}_{i+1}^{(k)} \ = \ \mathscr{A}_i^{(k)} \cup \left\{ s \in S_k \mid \exists s' \in \mathscr{A}_i^{(k)},\ (s, s') \in A \right\} \cup \left\{ s \in S_{1-k} \mid \forall s' \in S_k,\ \big((s, s') \in A\big) \ \Rightarrow \ s' \in \mathscr{A}_i^{(k)} \right\} \end{cases}$$

On appelle **attracteur** pour le joueur J_k l'ensemble $\mathscr{A}^{(k)} = \bigcup\limits_{i\in\mathbb{N}} \mathscr{A}_i^{(k)}$

Remarque

- L'ensemble $\mathscr{A}_{i+1}^{(k)}$ est construit à partir des sommets de $\mathscr{A}_i^{(k)}$ en ajoutant :
  - × les sommets contrôlés par J_k dont au moins un successeur est dans $\mathscr{A}_i^{(k)}$,
  - × les sommets contrôlés par J_{1-k} dont tous les successeurs sont dans $\mathscr{A}_i^{(k)}$.
- Par construction, les sommets de $\mathscr{A}_i^{(k)}$ sont les positions gagnantes pour le joueur J_k lui permettant de gagner en au plus i coups.
- L'attracteur $\mathscr{A}^{(k)}$ est l'ensemble des positions gagnantes pour le joueur J_k.
- Il existe une stratégie gagnante pour le joueur J_k si et seulement si le sommet initial appartient à l'attracteur $\mathscr{A}^{(k)}$. Dans ce cas, en notant F l'ensemble des sommets finaux du graphe biparti (S, A), on peut définir une stratégie gagnante $\varphi : S_k \setminus F \to S_{1-k}$ en définissant, pour tout $s \in S_k \setminus F$, $\varphi(s)$ de la façon suivante :
  - × si $s \in \mathscr{A}^{(k)}$, alors on considère un indice $i \in \mathbb{N}^*$ minimal tel que : $s \in \mathscr{A}_i^{(k)}$ et on définit $\varphi(s)$ comme un successeur de s dans $\mathscr{A}_{i-1}^{(k)}$;
  - × si $s \notin \mathscr{A}^{(k)}$, on définit $\varphi(s)$ comme un successeur quelconque de s (le choix n'a aucune incidence, car cette portion de la stratégie ne sera pas utilisée au cours de la partie).
- Comme la suite $\left(\mathscr{A}_i^{(k)}\right)_{i\in\mathbb{N}}$ est croissante pour l'inclusion et que tous ses éléments sont des parties de l'ensemble fini des sommets S, on en déduit qu'elle est stationnaire :

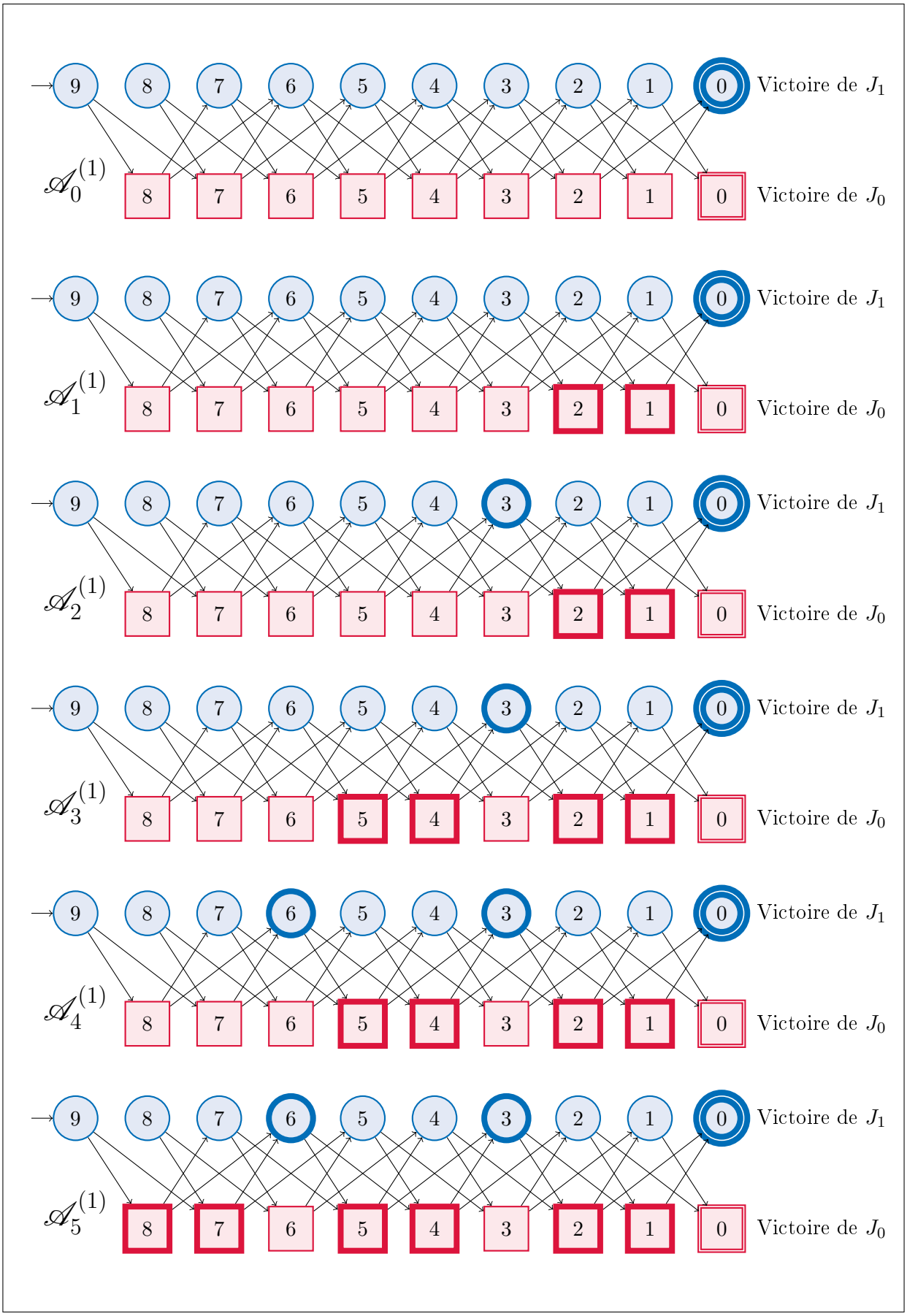
  $$\exists i_0 \in \mathbb{N}, \qquad \mathscr{A}^{(k)} = \mathscr{A}_{i_0}^{(k)}$$

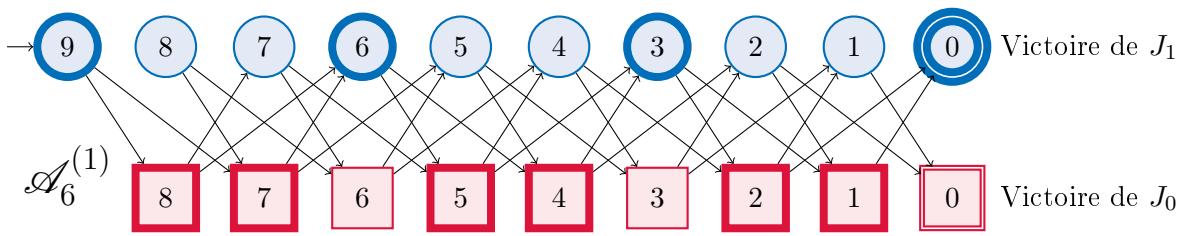
  En pratique, comme la suite $\left(\mathscr{A}_i^{(k)}\right)_{i\in\mathbb{N}}$ est définie par récurrence, on peut considérer :

  $$i_0 \ = \ \min\left(\left\{i \in \mathbb{N} \mid \mathscr{A}_i^{(k)} = \mathscr{A}_{i+1}^{(k)}\right\}\right)$$

- Le calcul de l'attracteur $\mathscr{A}^{(k)}$ peut se faire en utilisant un parcours en largeur du graphe transposé de (S, A). Il peut donc s'effectuer avec une complexité en $\underset{\rightarrow}{O}\big(\operatorname{Card}(S) + \operatorname{Card}(A)\big)$.

▶ Dans le jeu des allumettes, déterminer l'attracteur du joueur J_1.





De plus : $\mathscr{A}_7^{(1)} = \mathscr{A}_6^{(1)}$. Ainsi, la suite $\left(\mathscr{A}_i^{(1)}\right)_{i\in\mathbb{N}}$ est stationnaire à partir du rang 6. On en déduit : $\mathscr{A}^{(1)} = \mathscr{A}_6^{(1)}$.
Comme le sommet initial appartient à l'attracteur $\mathscr{A}^{(1)}$, il existe une stratégie gagnante pour le joueur J_1.

III. Algorithme min-max

Dans cette partie, on considère un jeu modélisé par un graphe biparti (S, A) où : $S = S_0 \cup S_1$.

En théorie, le calcul des attracteurs présenté dans la partie précédente permet de déterminer les stratégies gagnantes et de jouer de manière parfaite. En pratique, l'algorithme permettant de les déterminer n'est pas utilisable pour des jeux complexes, c'est-à-dire lorsque le graphe associé (S, A) est trop gros. Par exemple, on estime que le nombre d'états du jeu est de l'ordre de 10^{32} pour les dames, d'au moins 10^{46} pour les échecs et de 10^{100} pour le go.

Pour contourner cette difficulté, nous allons présenter l'algorithme min-max qui ne nécessite pas une exploration complète du graphe du jeu. En contre-partie, la stratégie obtenue via cette approche ne sera plus parfaite en générale.

III.1. Heuristique

L'algorithme min-max repose sur une fonction permettant d'évaluer la qualité de chaque position du jeu.

Définition
Une **heuristique** pour le jeu étudié est une fonction $h : S \to \mathbb{R} \cup \{-\infty, +\infty\}$ de sorte que pour toute position $p \in S$ du jeu :

- plus $h(p)$ est grand, meilleure est la position pour J_0 ;
- plus $h(p)$ est petit, meilleure est la position pour J_1.

Remarque

- Le mot « heuristique » signifie « qui sert à la découverte » ou « qui est propre à guider une recherche ».
- En pratique :
  - si $s \in S$ est un sommet final représentant une victoire de J_0, alors on pose : $h(s) = +\infty$;
  - si $s \in S$ est un sommet final représentant une victoire de J_1, alors on pose : $h(s) = -\infty$.
- Toujours en pratique, une heuristique est souvent construite de manière expérimentale.
- Plus l'heuristique utilisée sera pertinente, meilleure sera la qualité de la stratégie obtenue.

Pour illustrer cette notion, on s'intéresse au jeu Puissance 4. Le but du jeu est d'aligner une suite de 4 pions de même couleur dans une grille rectangulaire composée de 6 lignes et 7 colonnes. Chaque joueur dispose de 12 pions d'une même couleur. Les deux joueurs placent tour à tour un pion dans la colonne de leur choix. Le pion coulisse alors jusqu'à la position la plus basse possible dans ladite colonne, à la suite de quoi, c'est à l'adversaire de jouer. Le vainqueur est le joueur qui réalise en premier un alignement (horizontal, vertical ou diagonal) consécutif d'au moins 4 pions de sa couleur. Si toutes les cases de la grille de jeu sont remplies et qu'aucun des deux joueurs n'a réalisé un tel alignement, la partie est déclarée nulle.

Dans la suite, on suppose que J_0 joue avec des pions jaunes et que J_1 joue avec des pions rouges.

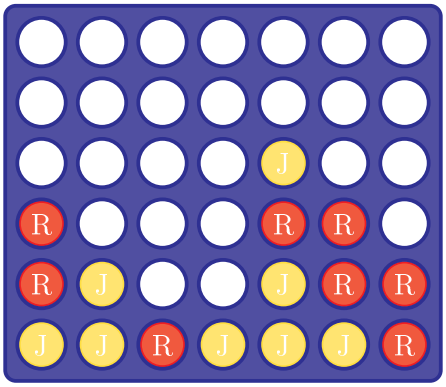


Fig. 1 Une position du jeu Puissance 4

On commence par construire une heuristique pour ce jeu. Pour ce faire, on attribue à chaque case du jeu le nombre d'alignements possibles de 4 jetons contenant cette case. Le valeurs attribuées sont reportées dans le tableau ci-dessous.

3	4	5	7	5	4	3
4	6	8	10	8	6	4
5	8	11	13	11	8	5
5	8	11	13	11	8	5
4	6	8	10	8	6	4
3	4	5	7	5	4	3

Fig. 2 Tableau des valeurs attribuées à chaque case

Ensuite, pour calculer l'heuristique d'une position (non finale) du jeu, on calcule la somme des valeurs des cases contrôlées par J_0 à laquelle on retranche la somme des valeurs des cases contrôlées par J_1.

► Quelle est l'heuristique de la position présentée en Figure 1 ?

$$h(s) = (3+4+7+5+4+6+8+11) - (5+3+4+6+4+5+11+8) = 2$$

III.2. Principe de l'algorithme

Dans cette sous-partie, on suppose que l'on dispose d'une heuristique $h : S \to \mathbb{R} \cup \{-\infty, +\infty\}$.

Pour illustrer le principe de l'algorithme min-max, on reprend l'exemple du jeu Puissance 4. On représente les états contrôlés par J_0 par des ronds bleus et les états contrôlés par J_1 par des carrés rouges. Afin d'avoir des arbres de taille raisonnable, on limite les coups possibles des deux joueurs à placer un pion dans une des trois colonnes centrales.

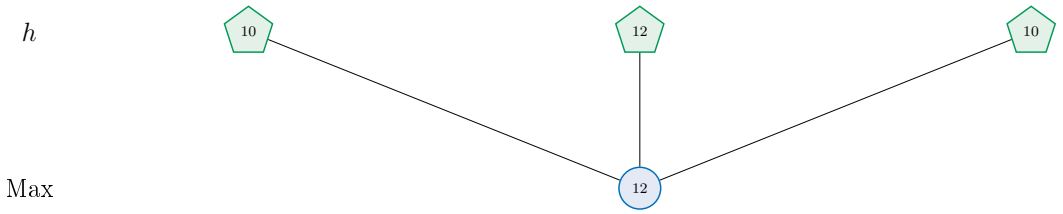
On part de la position du jeu Puissance 4 représentée en Figure 1 : c'est au joueur J_0 de jouer.
Une première possibilité pour effectuer le choix du coup à jouer est de calculer l'heuristique de chacune des positions atteignables, puis de rejoindre celle dont l'heuristique est maximale. En effet, le joueur J_0 cherche à se trouver dans une position avec une forte heuristique.

► Quel coup parmi les 3 possibles (on rappelle qu'on considère seulement des coups dans l'une des 3 colonnes centrales) doit jouer le joueur J_0 avec cette heuristique ?

On ajoute à l'heuristique de la position précédente (2) l'heuristique de la case où l'on va jouer.

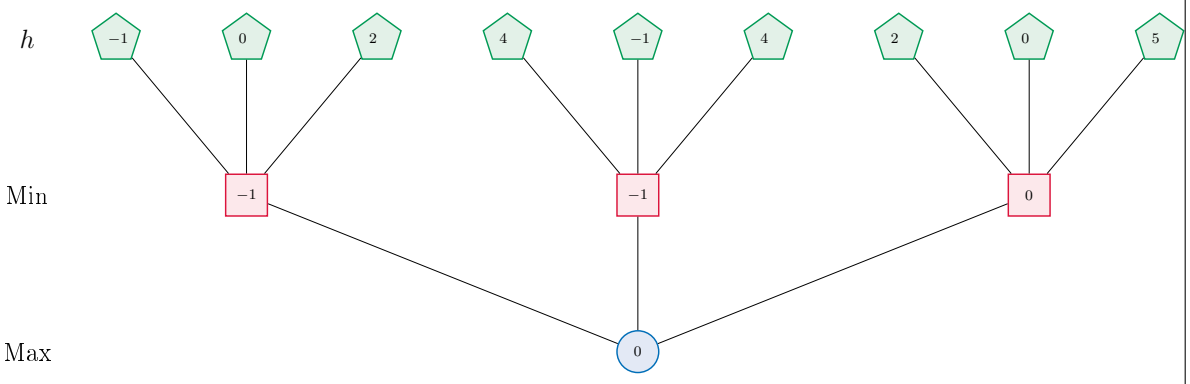
- L'heuristique du coup joué en première colonne est alors : $2 + 8 = 10$.
- L'heuristique du coup joué en deuxièee colonne est alors : $2 + 10 = 12$.
- L'heuristique du coup joué en troisième colonne est alors : $2 + 8 = 10$.

Le joueur J_0 va donc choisir de jouer en colonne centrale. On pouvait représenter la situation à l'aide d'un arbre.



Une autre possibilité pour effectuer le choix du coup à jouer pour J_0 est de tenir compte, en plus, du coup suivant qui sera joué par le joueur J_1 : le joueur J_0 peut calculer l'heuristique des positions atteignables en deux coups, puis choisir le coup lui permettant d'obtenir une heuristique maximale (en supposant que le joueur J_1 va jouer de sorte à minimiser l'heuristique).

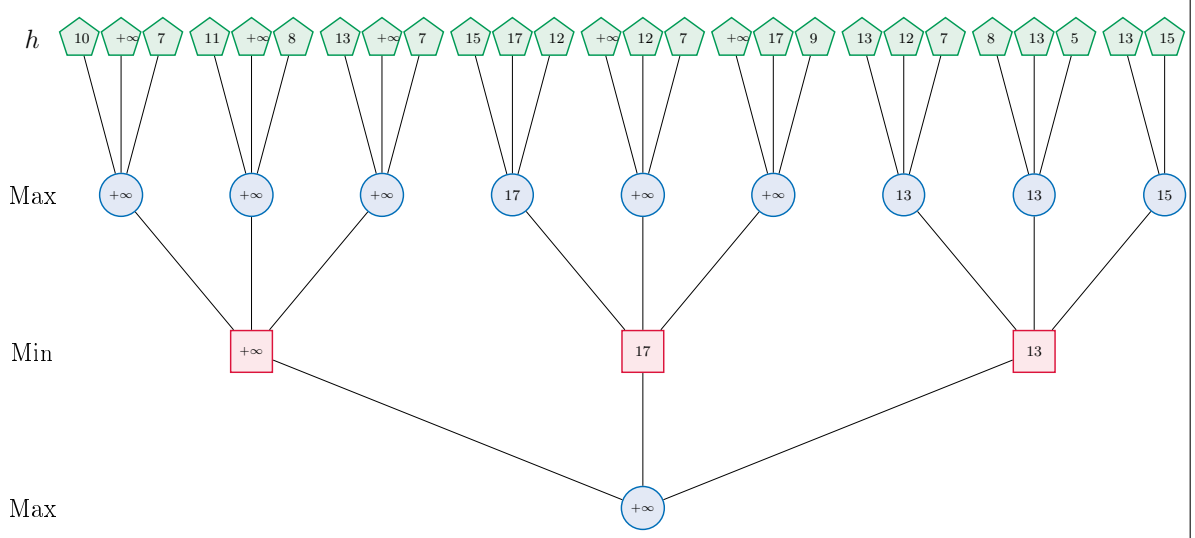
► Représenter cette stratégie à l'aide d'un arbre. Quel coup le joueur J_0 va-t-il choisir ?



En adoptant cette stratégie, le joueur J_0 choisit de jouer dans la colonne de droite.

Une troisième possibilité, pour effectuer le coup du choix à effectuer pour J_0, est d'explorer toutes les possibilités pour les trois coups suivants.

▶ Représenter cette stratégie à l'aide d'un arbre. Quel coup le joueur J_0 va-t-il choisir ?



En adoptant cette stratégie, le joueur J_0 choisit de jouer dans la colonne de gauche (et sera vainqueur en 3 coups).

Remarque

- La démarche précédente se généralise : l'algorithme min-max consiste en partant d'une position donnée du jeu à explorer toutes les possibilités après $p \in \mathbb{N}^*$ coups. L'objectif de J_0 est de choisir les positions maximisant l'heuristique, tandis que l'objectif de J_1 est de choisir les positions minimisant l'heuristique.
- Le nombre de positions à examiner a tendance à croître exponentiellement par rapport à la profondeur d'exploration de l'arbre. Il est donc nécssaire de limiter la taille de l'entier $p \in \mathbb{N}^*$.

L'algorithme ci-dessous permet d'implémenter la méthode décrite ci-dessus à l'aide d'une fonction récursive. On suppose que l'on dispose d'une fonction h permettant de calculer l'heuristique de toute position $s \in S$ du jeu.

```
1  Fonction minMax(s, p) :
2      ''Entrée : Une position s ∈ S, un profondeur p ∈ ℕ''
3      ''Sortie : La meilleure valeur possible après p coups, le coup à jouer pour l'atteindre''
4      Calculer la liste lstSucc des successeurs de s
5      si p == 0 ou lstSucc est vide :
6          retourner (h(p), -1)
7      sinon :
8          Calculer la liste lstValSucc des éléments minMax(f, p-1) où f parcourt lstSucc
9      si la position s est contrôlée par le joueur J_0 :
10         Déterminer la valeur maximale vMax et sa position indMax dans la liste lstValSucc
11         retourner (vMax, indMax)
12     sinon :
13         Déterminer la valeur minimale vMin et sa position indMin dans la liste lstValSucc
14         retourner (vMin, indMin)
```