Informatique

Théorie des jeux

PSI : Lycée Rabelais

Pré-requis

Cours d'informatique sur les dictionnaires

Cours sur les graphes (1ère année)

Objectifs

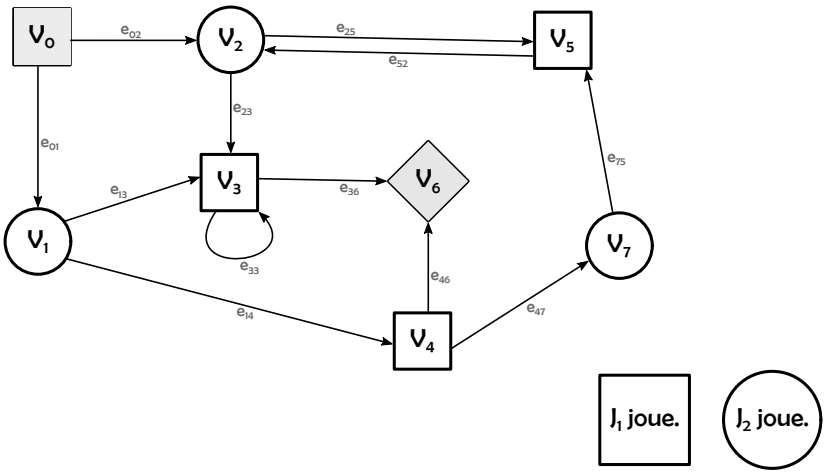
Comprendre le fonctionnement d'une arène d'un jeu et déterminer une stratégie gagnante

Être capable de créer un arbre de jeu et d'utiliser une méthode heuristique pour établir une stratégie

1 Introduction au chapitre et vocabulaire

La théorie des jeux est une branche des mathématiques relativement récente (années 1920) s'appliquant à de nombreuses situations réelles : économie, politique, stratégie militaire... Il s'agit d'une formalisation mathématique des situations "de jeux", c'est-à-dire mettant en opposition plusieurs adversaires aux objectifs antagonistes.

Dans le cadre du programme de classe préparatoire, on se limite à l'étude des jeux dits d'**accessibilité**. Il existe bien entendu, un nombre beaucoup plus important de type de jeux. Ces jeux d'accessibilité peuvent se représenter par des graphes. Pour introduire les différentes notions, nous utiliserons le jeu représenté par le graphe ci-dessous :



Règles du jeu : Il s'agit d'un jeu à deux joueurs notés J_1 et J_2. Un jeton est initialement placé sur la case (**nœud**) de départ V_0 et peut se déplacer en suivant les **arêtes** du graphe. Si le jeton est sur un nœud carré alors c'est à J_1 de jouer sinon, c'est J_2 qui doit jouer. On dira alors que J_k est l'ensemble des nœuds jouables par le joueurs k.

But du jeu : Pour gagner, J_1 doit rejoindre le nœud en forme de losange. J_2 doit empêcher J_1 de gagner.

1.1 Formalisme

Le graphe représenté ci-dessus sera appelé **arène de jeu**. Il sera noté $G(\mathscr{V}, \mathscr{E})$ où $\mathscr{V}$ est l'ensemble des sommets et $\mathscr{E}$ l'ensemble des arêtes. Pour un jeu à deux joueurs avec des cases associées aux tours de J_1 et J_2, on dira que l'arène est une arène **bipartite** car $\mathscr{V}$ peut se décomposer en deux sous-ensembles complémentaires $\mathscr{V}_1$ et $\mathscr{V}_2$ où $\mathscr{V}_k$ est l'ensemble des nœuds associés au joueur J_k.

Dans le jeu proposé, le joueur J_1 doit atteindre le nœud V_6. On définira alors l'**ensemble des conditions de gain** $\Omega = \{V_6\}$ qui est l'ensemble des cases permettant au joueur J_1 de gagner. On pourrait avoir, pour des jeux plus complexes, plusieurs cases permettant la victoire de J_1.

Un **jeu** sera donc un couple **(arène de jeu , conditions de gain)** $= (G, \Omega)$. Une **partie**, notée Λ, représente l'ensemble des actions effectuées par les joueurs (arêtes parcourues). Par exemple : $\Lambda = \{e_{01}, e_{13}, e_{34}, e_{46}\}$.

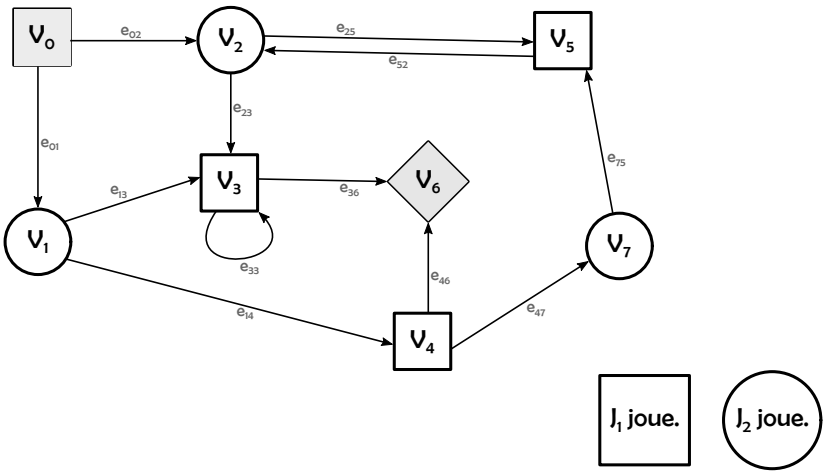
1.2 Notion de stratégies

L'intérêt de la théorie des jeux est d'élaborer des **stratégies**. Une stratégie, au sens mathématique, est une application ψ qui indique l'action à effectuer en fonction des actions précédentes. Dans le cadre du programme, on étudiera seulement les stratégies *sans mémoire*. Cela signifie que l'application ψ indique le coup à jouer e_{jk} en fonction du coup précédent e_{ij} (et uniquement de celui-ci). On peut donc écrire :

$$\psi : \begin{cases} \mathscr{E} & \to & \mathscr{E} \\ e_{ij} & \to & e_{jk} \end{cases}$$

On dira qu'**une stratégie est gagnante** pour le nœud V_j si elle permet de gagner **systématiquement** (sans compter sur une erreur de l'adversaire) à partir du nœud V_j. Dans ce cas, le nœud V_j sera lors une **position gagnante**.

On dira également qu'un jeu est **résolu** si une stratégie gagnante est trouvée pour chacun des nœuds de l'arène.



Positions gagnantes pour J_1 en bleu

Positions gagnantes pour J_2 en rouge

Pour les jeux d'accessibilité, il existe, pour chaque nœud, un stratégie gagnante pour l'un ou l'autre des joueurs.

2 Notion d'attracteur

2.1 Formalisme général

Pour identifier les positions gagnantes, la notion d'attracteur est souvent utilisée. On appellera **attracteur** (pour le joueur J_1), la suite $(A_n)_{n\in\mathbb{N}}$ telle que A_i est l'ensemble des nœuds à partir desquels J_1 gagne en moins de i coups. On dira que cette suite est :

croissante car les ensembles A_i contiennent de plus en plus d'éléments au fur et à mesure que i augmente ;
stationnaire car il existe un entier n_0 tel que, $\forall i > n_0, A_i = A_{n_0}$.

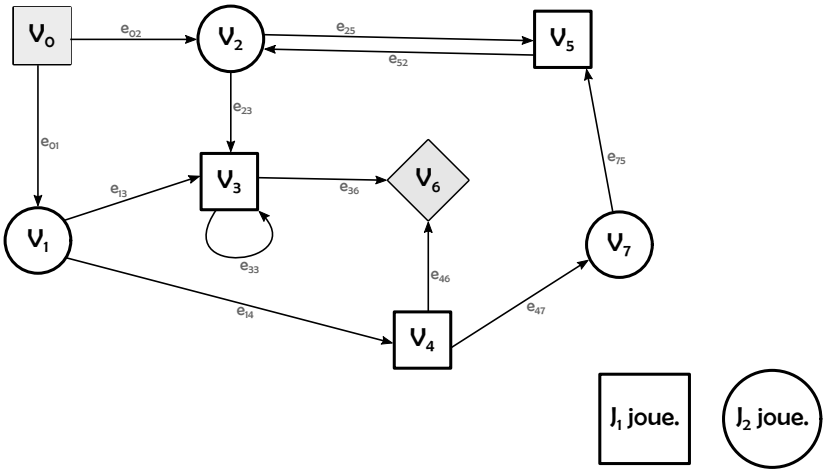
L'ensemble A_{n_0} sera l'ensemble de toutes les positions gagnantes pour J_1.

Pour déterminer les attracteurs A_n, une méthode par récurrence est utilisée. On écrira alors :

Initialisation : $A_0 = \Omega$ (conditions de gain) ;
Récurrence : $A_{n+1} = A_n \cup A_{J_1 \to A_n} \cup A_{J_2 \Rightarrow A_n}$ avec :

- A_n et A_{n+1} les attracteurs aux rangs n et $n+1$;
- $A_{J_1 \to A_n}$ les nœuds de J_1 qui permettent d'accéder à un nœud de A_n ;
- $A_{J_2 \Rightarrow A_n}$ les nœuds de J_2 qui mènent **obligatoirement** à un nœud de A_n.

2.2 Mise en œuvre



3 Jeux plus complexes

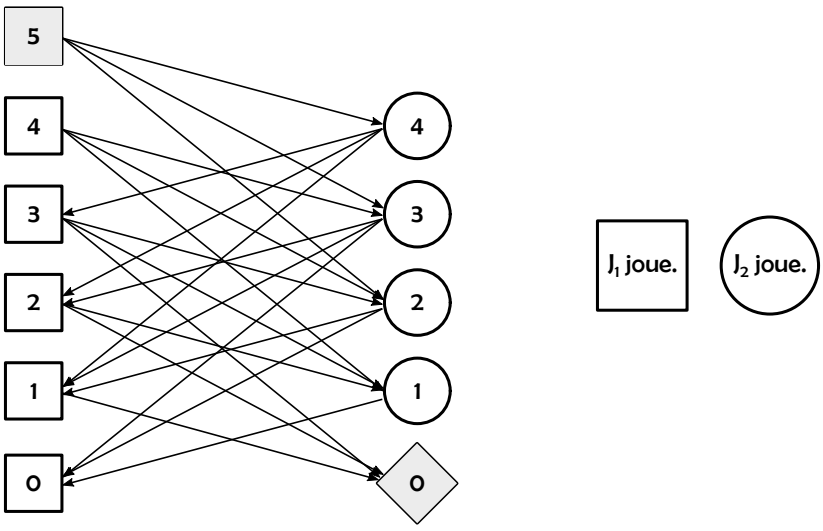
Pour certains jeux, il n'est pas possible de définir de positions gagnantes. Il est cependant possible de déterminer quel est le meilleur coup à jouer. On parle dans ce cas de méthode **heuristique**. L'algorithme **min-max** permet de gérer ce type de problème.

3.1 Exemple du jeu de Nim à un tas

On considère un tas de 5 allumettes. Chaque joueur, à tour de rôle, choisit d'enlever 1, 2 ou 3 allumettes. Le dernier joueur à retirer une ou des allumettes a gagné.

3.1.1 Représentation avec un graphe

Une première méthode serait d'utiliser un graphe comme présenté précédemment. Le numéro de la case correspond au nombre d'allumettes présentes dans le tas.



On peut alors déterminer les positions gagnantes pour le joueur J_1 (coloriées en bleu sur le graphe ci-dessus).

3.1.2 Représentation avec un arbre de décision (arbre de jeu)

Présentation générale

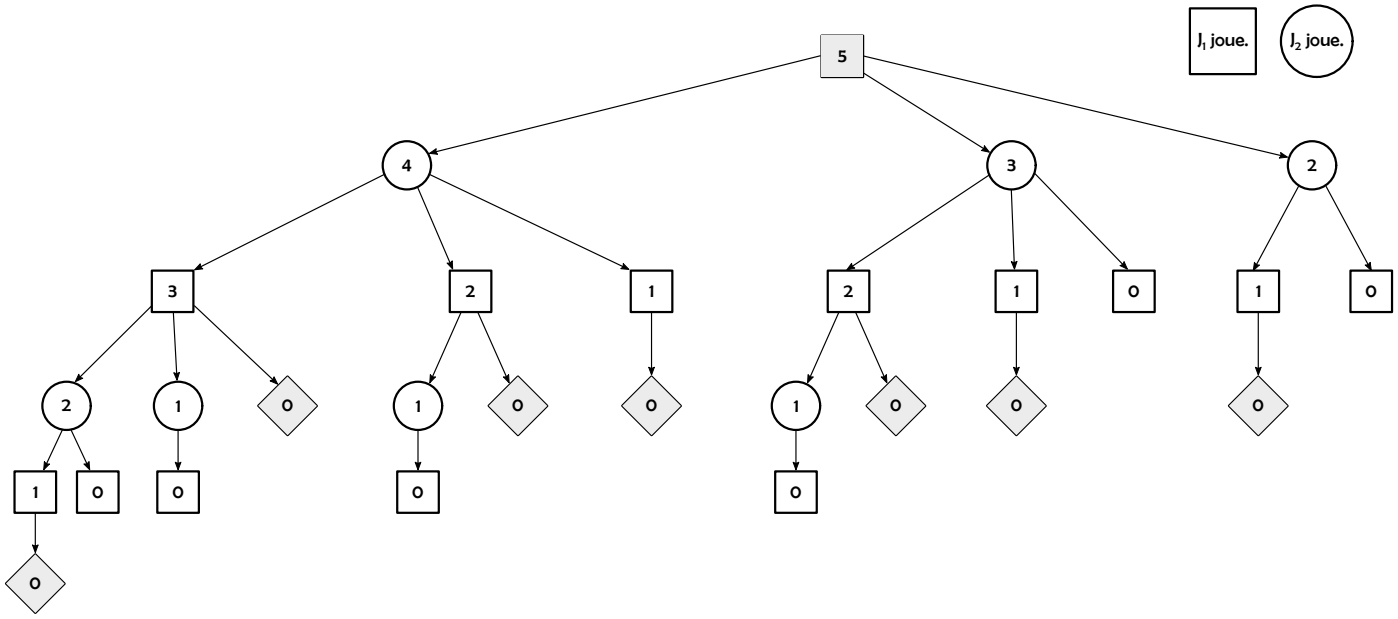
Une autre manière de représenter le jeu est d'utiliser un arbre de décision (ou arbre de jeu). Les sommets de l'arbre sont les situations du jeu et les arêtes représentent les actions effectuées.

La *racine* est la position initiale. Les nœuds "extrêmes" sont les *feuilles*.

On peut alors parcourir l'arbre comme on parcourt un graphe pour chercher à partir de n'importe quelle position le meilleur coup à jouer. Pour trouver le meilleur coup, il sera nécessaire d'énumérer tous les coups possibles et de leur associer un coût. Ce coût représente le rapport bénéfice/perte liée à l'action.

Dans l'algorithme **min-max** ou **minimax**, pour le joueur J_1, l'algorithme visera à emprunter le chemin qui maximise les gains de ce joueur J_1 et partant du principe que J_2 essayera aussi de minimiser ses pertes.

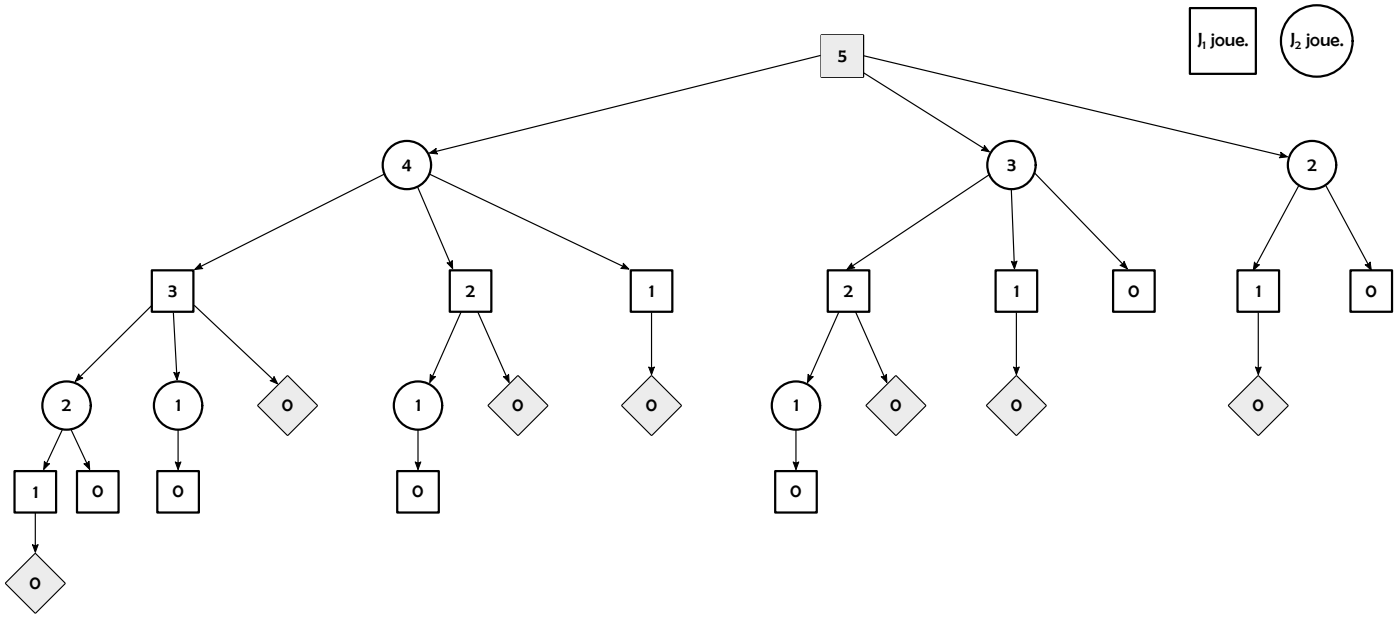
Pour le jeu de Nim, on obtient alors la représentation suivante :



Mise en œuvre

On cherche la meilleure stratégie pour J_1 !

1. Chaque feuille indique si J_1 est gagnant ou perdant. On indique +1 s'il est gagnant et -1 s'il est perdant. On pourrait indiquer 0 pour des parties nulles.
2. On remonte l'arbre de manière récursive :
    - Si le nœud est contrôlé par J_1, sa valeur est le maximum des valeurs de ses sous-arbres ;
    - Si le nœud est contrôlé par J_2, sa valeur est le minimum des valeurs de ses sous-arbres ;



Une fois les coûts indiqués, il est aisé de déterminer la stratégie gagnante pour J_1. Il suffit de suivre un chemin qui maximise toujours les gains du joueur J_1.

3.2 Notion de profondeur de limite

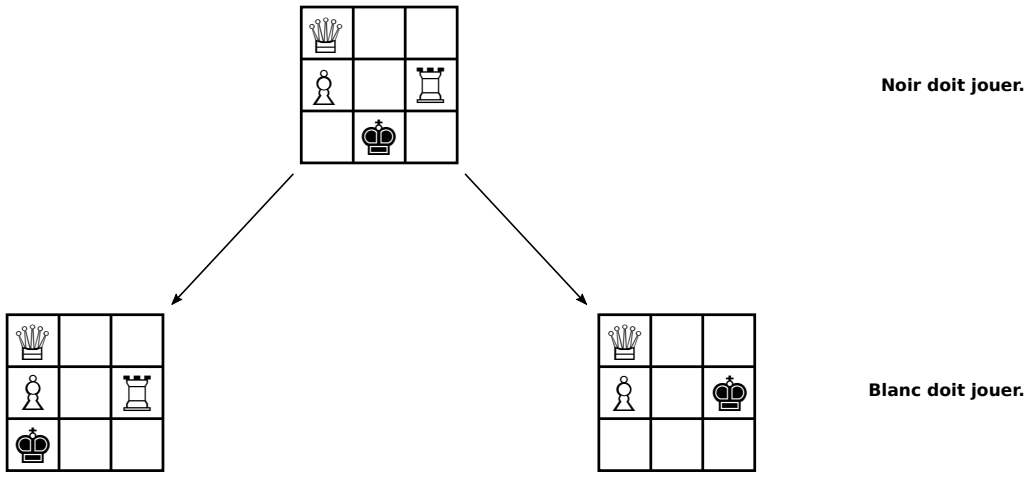
Pour certains jeux, comme les échecs ou le jeu de go, le calcul de tout l'arbre de jeu est trop gourmand en ressources informatiques. Dans ce cas, la décision est prise en fonction de la valeur de l'heuristique à une profondeur limite où

la profondeur limite est le nombre de coups suivants qui seront évalués. Il faut donc que cette heuristique puisse être déterminée sans remonter aux feuilles de l'arbre de jeu. L'heuristique est souvent déterminée de manière empirique.

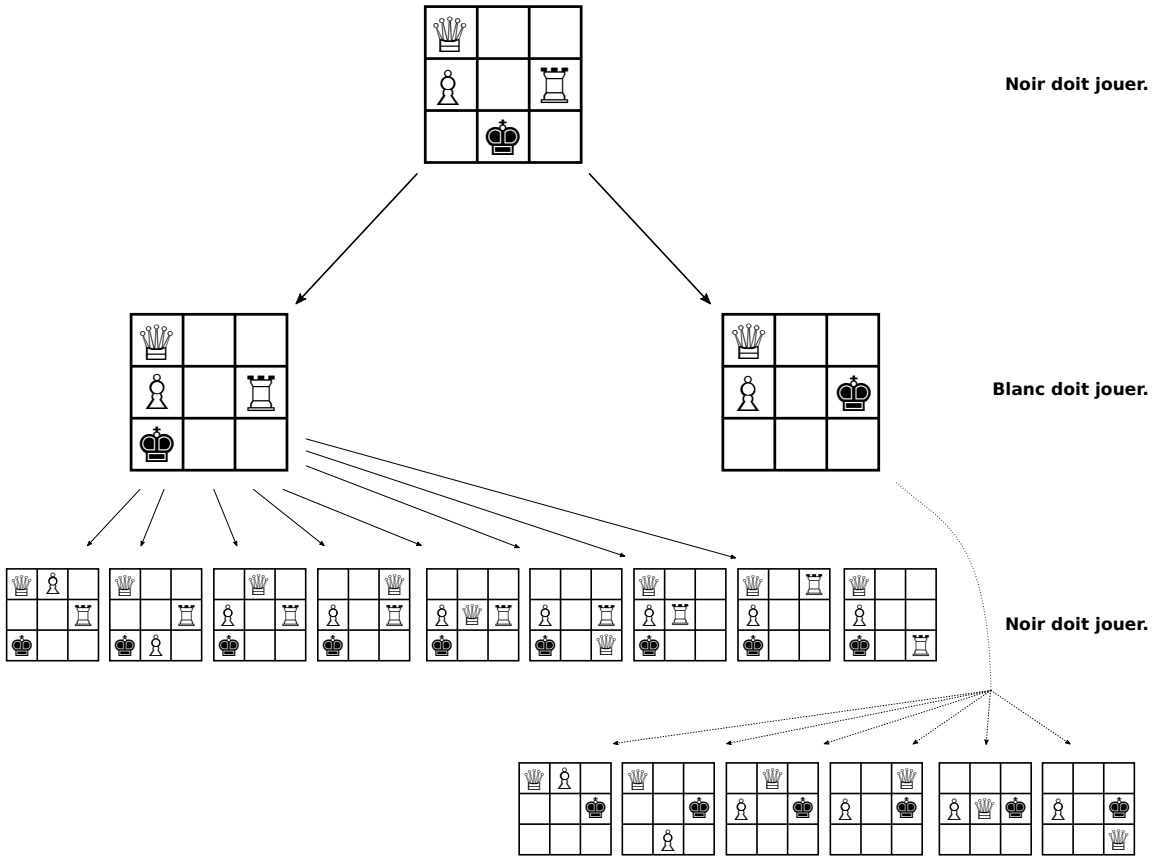
Prenons l'exemple d'une partie d'échecs. On considère que l'heuristique (très simple !) notée h est définie de la manière suivante :

- pour une configuration de fin de partie (feuilles), on imposera $h = +1000$ si le joueur gagne et $h = -1000$ s'il perd ;
- pour une autre configuration, h sera la différence entre le nombre de pièces noires et le nombre de pièces blanches.

Partons de la situation ci-dessous et déterminons, avec une profondeur limite d'**1 coup**, ce que doit jouer noir en respectant l'heuristique proposée précédemment :



Partons de la même situation et déterminons, cette fois-ci avec une profondeur limite de **2 coups**, ce que doit jouer noir en respectant l'heuristique proposée précédemment :



Toujours en partant de la même situation, déterminons, **sans profondeur limite**, ce que doit jouer noir en respectant l'heuristique proposée précédemment :