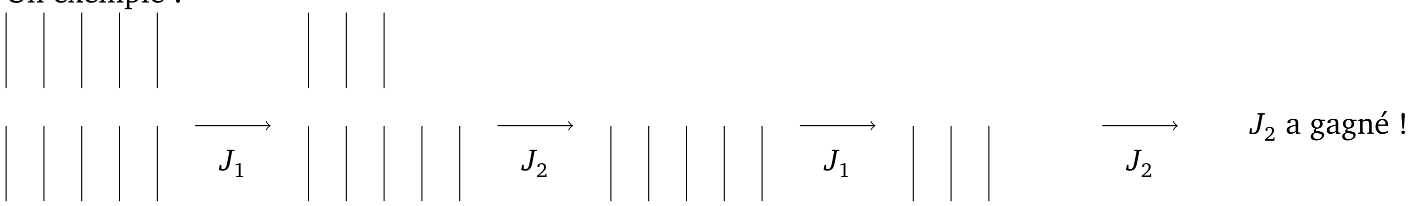
Jeux d'accessibilité à deux joueurs sur un graphe

Les progrès de l'intelligence artificielle se sont manifestés à différentes époques lorsque des machines ont battu des joueurs humains au jeu de dames (années 1950), aux échecs (Deep Blue bat le champion du monde Garry Kasparov en 1997) ou au jeu de Go (Alpha Go bat les meilleurs joueurs du monde en 2016 et 2017).

1 Jeux sur un graphe

1.1 Le jeu de Nim

Dans sa version la plus simple, le jeu de Nim oppose deux joueurs qui prennent à tour de rôle une, deux ou trois allumettes. Le joueur qui prend la dernière allumette a gagné.
Un exemple :



1.2 Cadre général

On s'intéresse aux jeux à deux joueurs qui possèdent les propriétés suivantes :

- le jeu est à information complète : les deux joueurs ont la même vue sur le jeu ;
- le jeu est sans hasard ;
- les deux joueurs jouent à tour de rôle.

Les jeux de Nim, des dames, d'échecs, de Go, morpion, puissance 4 sont des exemples de tels jeux.

Définition

Un graphe (S,A) est **biparti** lorsque l'ensemble S des sommets est la réunion disjointe de deux sous-ensembles S_1 et S_2 et que chaque arête a une extrémité dans S_1 et l'autre dans S_2 (il n'existe pas d'arête reliant deux sommets de S_1 ou deux sommets de S_2).

Ce type de jeu est associé à un graphe orienté (S,A). Chaque sommet de S représente une configuration du jeu pour le joueur J_1 ou pour le joueur J_2. On note S_1 l'ensemble des sommets contrôlés par J_1 (c'est au tour de J_1 de jouer) et S_2 l'ensemble des sommets contrôlés par J_2.
Une arête $a = (s,t) \in A$ reliant deux sommets indique la possibilité pour un joueur de passer de la position s à la position t en un coup. Comme les joueurs jouent à tour de rôle, une arrête relie soit un sommet de S_1 à un sommet de S_2, soit un sommet de S_2 à un sommet de S_1. Le graphe est donc biparti.
On précise de plus la position de départ s_0 (contrôlée par J_1) et pour chaque état final (sans arrête sortante) : les états gagnants pour J_1, les états gagnants pour J_2 et les états de match nul.

Définition

Un **jeu d'accessibilité** à deux joueurs J_1 et J_2 est défini par :

- un graphe fini orienté (S, A) biparti selon la partition $S = S_1 \cup S_2$ (états contrôlés par J_1 et J_2 respectivement) ;
- un état de départ $s_0 \in S$;
- un recouvrement disjoint de l'ensemble F des états finaux sous la forme $F = G_1 \cup G_2 \cup N$ (états finaux gagnants pour J_1, pour J_2, et nuls).

Une partie est un chemin qui part de s_0 et qui est soit infini, soit termine en un état de F (état final).

Exemple :

Représenter le graphe du jeu de Nim pour 10 allumettes.

1.3 Stratégie gagnante et position gagnante

Définition

Une **stratégie** pour J_1 sur un jeu d'accessibilité est une fonction $f : S_1 \backslash F \longrightarrow S_2$ telle que pour tout $s \in S_1 \backslash F, \quad (s, f(s)) \in A$. Ainsi une stratégie consiste à donner, pour chaque position (non finale) contrôlée par J_1, le coup à jouer.

De même, Une stratégie pour J_2 est une fonction $f : S_2 \backslash F \longrightarrow S_1$ telle que pour tout $s \in S_2 \backslash F, \quad (s, f(s)) \in A$.
Pour un joueur J_i ($i \in \{1, 2\}$), suivre la stratégie f consiste, à chaque tour en position non finale $s \in S_i$, à jouer le coup $(s, f(s))$.

Exemple :

On désigne chaque sommet du graphe pour le jeu de Nim par les couples (k, i) où k est le nombre d'allumettes restantes et i est le numéro du joueur qui contrôle la position. Une stratégie pour le joueur 1 est $f : (k, 1) \mapsto (k-1, 2)$, elle consiste à prendre systématiquement une allumette.

Définition

Une stratégie f pour J_1 sur un jeu d'accessibilité est une **stratégie gagnante** lorsque pour toute partie telle que J_1 suit la stratégie f est finie et termine en un état de G_1 (gagnant pour J_1).
De même, une stratégie g est gagnante pour J_2 lorsque pour toute partie telle que J_2 suit la stratégie g est finie et termine par un état de G_2.

Exemple :

Déterminer une stratégie gagnante pour J_1 au jeu de Nim pour 10 allumettes.

Remarque :

L'existence d'une stratégie gagnante pour un joueur dépend en général de l'état initial.

Définition

Une position $s \in S$ est une **position gagnante pour** J_1 lorsqu'il existe une stratégie gagnante pour J_1 pour une partie débutant au sommet s. Dans ce cas, on dit aussi que s est une **position perdante pour** J_2.

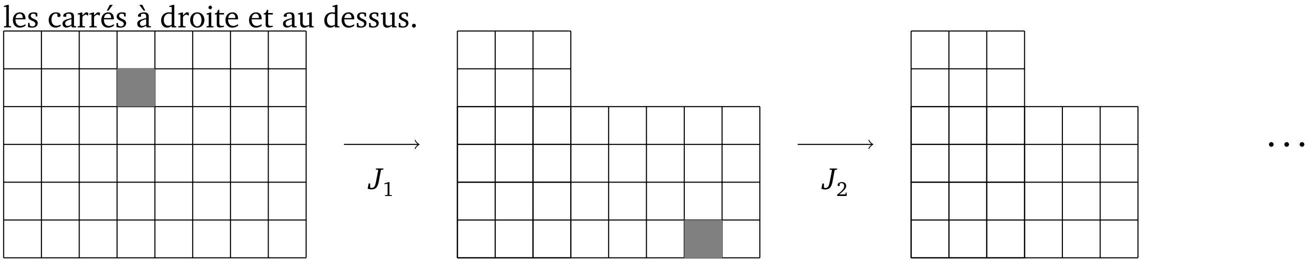
De même, une position $s \in S$ est une **position gagnante pour** J_2 lorsqu'il existe une stratégie gagnante pour J_2 pour une partie débutant au sommet s. Dans ce cas, on dit aussi que s est une **position perdante pour** J_1.

Exemple :

Pour le jeu de Nim, les positions $(10,1)$, $(9,1)$, $(7,1)$, $(6,1)$, $(5,1)$, $(3,1)$, $(2,1)$, $(1,1)$, $(8,2)$, $(4,2)$ et $(0,2)$ sont des positions gagnantes pour J_1.

1.4 Jeu du « croque-chocolat » ou jeu de « Chomp »

On se donne une tablette de chocolat rectangulaire qui contient $n \times m$ carrés de chocolat. À tour de rôle, chaque joueur prend une partie de la tablette : s'il choisit le carré (i,j) alors il mange ce carré ainsi que tous les carrés à droite et au dessus.



Le dernier carré étant empoisonné, celui qui le mange a perdu (chaque joueur doit prendre au moins un carré à chaque tour).

Au jeu du « croque-chocolat » de taille (n,m) (avec $(m,n) \neq (1,1)$), il existe une stratégie gagnante pour J_1. En effet, commençons par remarquer qu'une partie de croque-chocolat est toujours finie (le nombre de carrés décroît strictement à chaque tour) et qu'il ne peut y avoir de match nul. Cela implique que toute position est soit gagnante soit perdante pour J_1.

Si J_1 choisit le carré en haut à droite :

- soit ce coup est gagnant pour J_1 (il mène à une position perdante pour J_2) ;
- soit ce coup est perdant pour J_1 (il mène à une position gagnante pour J_2), ce qui signifie que J_2 dispose d'un coup gagnant (i_0, j_0) dans cette position. Mais dans ce cas, il suffit à J_1 de jouer directement (i_0, j_0) au premier tour.

Dans tous les cas, le premier joueur a la possibilité de jouer un coup gagnant.

Remarque :

Cette démonstration utilise le principe de *vol de stratégie* car le deuxième joueur se fait voler toute stratégie potentielle par le premier joueur. Par contre, elle est non constructive : on a montré qu'il existe une stratégie gagnante, mais on ne sait pas comment gagner...

1.5 Attracteurs et calcul des positions gagnantes

Pour calculer une stratégie gagnante, on définit la notion d'attracteur par récurrence : $\mathscr{A}_1^n$ est l'ensemble des positions (de $S = S_1 \cup S_2$) à partir desquelles il existe une stratégie gagnante pour le joueur J_1 en au plus n coups.

Définition

Pour chaque joueur J_i (avec $i \in \{1;2\}$), on définit l'attracteur d'ordre $n \in \mathbb{N}$ par récurrence : $\mathscr{A}_i^0 = G_i$ et pour tout $n \geq 1$:

$$\begin{aligned}\mathscr{A}_1^{n+1} = & \mathscr{A}_1^n \\ & \cup \{s \in S_1 \mid \exists t \in \mathscr{A}_1^n, (s,t) \in A\} \\ & \cup \{s \in S_2 \mid \forall t \in S_1, (s,t) \in A \Rightarrow t \in \mathscr{A}_1^n\}\end{aligned}$$

et

$$\begin{aligned}\mathscr{A}_2^{n+1} = & \mathscr{A}_2^n \\ & \cup \{s \in S_2 \mid \exists t \in \mathscr{A}_2^n, (s,t) \in A\} \\ & \cup \{s \in S_2 \mid \forall t \in S_2, (s,t) \in A \Rightarrow t \in \mathscr{A}_2^n\}\end{aligned}$$

Remarque :

Les positions qui permettent de gagner au joueur 1 en au plus $n+1$ coups sont celles :

- qui permettent de gagner en au plus n coups ;
- les positions contrôlées par le joueur 1 (c'est à J_1 de jouer) à partir desquelles J_1 peut atteindre une position gagnante en au plus n coups (il existe un coup qui permet d'atteindre une position gagnante en au plus n coups) ;
- les positions contrôlées par le joueur 2 (c'est à J_2 de jouer) à partir desquelles J_2 ne peut atteindre que des positions gagnantes pour J_1 en au plus n coups (tous les coups possibles mènent à une position gagnante pour J_1 en au plus n coups).

Par définition, la suite des attracteurs pour le joueur i : $(\mathscr{A}_i^n)_{n\in\mathbb{N}}$ est croissante pour l'inclusion. L'ensemble des parties de S étant fini (si le jeu n'autorise pas de boucler sur le graphe), cette suite est stationnaire. On note n_i le premier rang à partir duquel la suite est constante et on appelle **attracteur pour le joueur** J_i l'ensemble $\mathscr{A}_i^{n_i}$ noté $\mathscr{A}_i$.

Théorème

Pour $i \in \{1,2\}$, les positions gagnantes pour J_i sont les éléments de l'attracteur $\mathscr{A}_i$ pour le joueur J_i.

Si s_0 est une position gagnante pour J_1, on peut déterminer une stratégie gagnante pour J_1 à partir de la suite des attracteurs de J_1. Pour toute position $x \in \mathscr{A}_1$ contrôlée par J_1 :

- si $x \in G_1$ la partie est finie ;
- sinon, en posant n minimal tel que $x \in \mathscr{A}_i^n$, alors il existe $y \in \mathscr{A}_1^{n-1}$ tel que $(x,y) \in A$. Le joueur J_1 peut jouer le coup (x,y) qui mène vers une position gagnante y (plus proche de la victoire).

Exemple :

Pour le jeu de Nim avec 10 allumettes, déterminer l'attracteur pour le joueur J_1.

2 Algorithme min-max

2.1 Heuristique

Pour des jeux d'accessibilité sur de petits graphes, il est possible de déterminer les positions gagnantes et des stratégies gagnantes. Mais dès qu'un jeu est plus compliqué la taille du graphe grandit très rapidement. Pour un jeu de morpion classique 3×3, il y a $3^9 = 19\,683$ positions possibles et $9! = 362\,880$ parties différentes. Pour les échecs on aurait de l'ordre de 30^{100} parties possibles… On ne construira pas de tels graphes.

Définition

Soit (S,A) le graphe orienté modélisant le jeu. Une **heuristique** sur le jeu est une fonction

$$h : S \mapsto \mathbb{R} \cup \{-\infty, +\infty\}.$$

L'heuristique mesure si une position est avantageuse pour un joueur :

- plus $h(p)$ est grand, meilleure est la position pour J_1 ;
- plus $h(p)$ est petit, meilleure est la position pour J_2.

Remarque :

On attribue souvent la valeur $+\infty$ aux positions de victoires pour le joueur J_1 et $-\infty$ à celles de victoire pour J_2.

Exemple :

Une heuristique naïve pour le jeu de dames : à une position donnée on attribue le nombre de pions en jeu pour le joueur J_1 moins le nombre de pions pour le joueur J_2.

2.2 min-max

Pour choisir le coup à jouer, J_1 peut évaluer l'heuristique sur chaque position atteignable à partir de la position présente et choisir le coup qui lui permet d'atteindre le meilleure score.

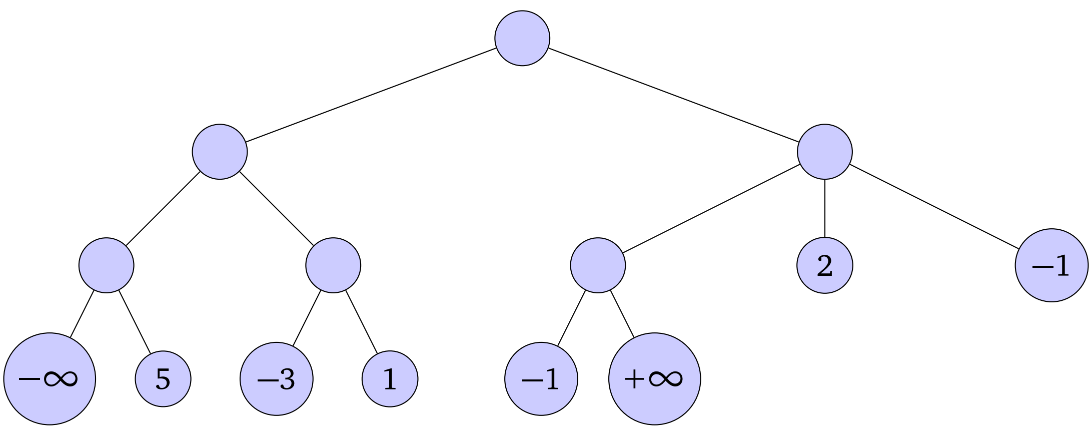
Pour voir plus loin, il peut chercher à optimiser le score après deux coups. Sachant que J_2 va chercher à minimiser l'heuristique.

Le principe de l'algorithme min-max de profondeur n est d'explorer, à partir d'une position donnée, les positions atteignables en n coups sachant qu'à chaque étape :

- J_1 cherche à maximiser l'heuristique (qui mesure ses chances de gagner) ;
- J_2 cherche à minimiser l'heuristique.

Exemple :

Compléter les scores pour déterminer le coup à jouer par J_1 en appliquant l'algorithme de min-max avec une profondeur 3



Plus précisément, en utilisant des fonctions :

- `coups_possibles(position)` de paramètre `position` qui renvoie la liste des coups possibles ;
- `jouer_coup(position, coup)` qui modifie par effet de bord le paramètre position en jouant `coup` ;
- `annuler_coup(position, coup)` qui annule le coup joué `coup` ;
- `choix_maxi(scores_coups)` et `choix_mini(scores_coups)` de paramètre une liste de couples et qui renvoie un couple qui maximise (resp. minimise) le premier élément du couple.
- `heuristique(position)` qui calcule l'heuristique.

```python
def minmax(position, joueur, p):
    """
    renvoie le score pour une profondeur p dans la position donnée ainsi que
    le coup à jouer avec position qui décrit l'état du jeu, joueur le
    numéro du joueur (1 ou 2)
    """
    L = coups_possibles(position)
    if p == 0 or len(L) == 0:
        return heuristique(position), None
    scores_coups = []  # liste des couples (score, coup) des successeurs
    for coup in L:
        jouer_coup(position, coup)
        scores_coups.append((minmax(position, p - 1)[0], coup))
        annuler_coup(position, coup)
    if joueur == 1:
        return choix_maxi(scores_coups)
    else:
        return choix_mini(scores_coups)
```

2.3 Cas des jeux de petite taille

Pour des jeux de petite taille, lorsqu'il est possible d'explorer tous les états possibles, l'algorithme min-max permet de retrouver les stratégies gagnantes. On utilise alors l'heuristique triviale :

$$h : s \mapsto \begin{cases} +\infty & \text{si } s \in G_1; \\ -\infty & \text{si } s \in G_2; \\ 0 & \text{sinon} \end{cases}$$

et on ne tient pas compte de la profondeur (c'est-à-dire qu'on enlève le paramètre `p` de la fonction `minmax`).