Algorithmes pour l'étude des jeux. Exemples et applications

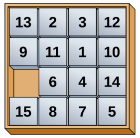
I. Jeux à un joueur [BAR 11.5.1]

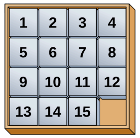
A. Formalisation d'un jeu

Formalisation 1 **Jeu** On formalise un jeu sous forme d'un graphe $G = (S, A)$, les états représentant les configurations possibles du problème, et les arêtes, les actions possibles permettant de passer d'un état à l'autre.

Formalisation 2 **Résolution** La résolution d'un tel jeu consiste alors à trouver un chemin dans ce graphe partant d'un état initial à l'un des états finaux.

Exemple 3 **Jeu de Taquin**





Un des exemples les plus simples et caractéristiques est le jeu du taquin. Il s'agit d'un jeu à un joueur en forme de damier, composé de 15 carreaux numérotés de 1 à 15 (ou représentant des morceaux d'une image), glissant dans un carré de 4×4 cases. L'objectif est de remettre les nombres dans le bon ordre.

Définition 4 **Puzzle** Un tel jeu à un joueur est appelé puzzle.

Remarque 5 **Explosion** Si il est possible d'énumérer toutes les parties possibles, cela résulte en une explosion combinatoire et n'est pas utilisé en pratique.

B. L'algorithme A* [BAR 11.5.2]

Définition 6 **L'algorithme A*** est une variation de l'algorithme de Dijkstra où la recherche est guidée par une fonction heuristique.

Exemple 7 **Cercles concentriques Dijkstra vs A***

Définition 8 On appelle **heuristique pour l'algorithme A*** toute fonction $h : S \to \mathbb{R}$ qui pour chaque sommet $s \in S$ estime la taille du meilleur chemin entre s et s_f.

Définition 9 **Une heuristique est admissible** si pour tout sommet s du graphe d'état, $h(s)$ est une borne inférieure de la plus courte distance séparant s de s_f.

Définition 10 **Une heuristique est consistante** si pour tout arc $s \to p$ du graphe d'état, $h(s) \leq h(p) + C(s \to p)$

Définition 11 **Une heuristique est monotone** si l'estimation du coût total d'un chemin ne décroît pas lors du passage d'un sommet à ses successeurs : étant donné un chemin $s_0 \to \ldots \to s_n$ on a pour tout $0 \leq i < j \leq n, f(s_j) \geq f(s_i)$.

Exemple 12 Pour le jeu du taquin, on peut par exemple penser aux heuristiques suivantes :

- nombre de chiffres mal placés,
- somme des distances L1 (ou Manhattan) des cases à leur position souhaitée.

C. Réduction à SAT

Remarque 13 Une méthode alternative pour résoudre un puzzle dans NP est de le réduire à SAT et d'utiliser un logiciel de résolution de formule SAT comme `bitwuzla` ou encore `z3`.

DEV

[TOR p1031]

Exemple 14 **Le puzzle du Sudoku**

Réduction 15 **Sudoku à SAT**

Algorithme 16 **DPLL**

II. Jeux à deux joueurs [TOR 9.7.2]

A. Définitions

1. Motivation [BAR 11.4]

Définition 17 **Un jeu à information complète** est un jeu lors duquel les deux joueurs connaissent toutes les informations du jeu.

Définition 18 **Un jeu à coups asynchrone** est un jeu ou chaque joueur joue alternativement.

Motivation 19 Derrière ces notions se cachent de nombreux jeux de plateau, comme les échecs, le go, le morpion, puissance 4,...

2. Formalisation [TOR 9.7.2]

Définition 20 **Un jeu à deux joueurs** est un graphe orienté biparti $G = (S_0 \uplus S_1, A)$. Les sommets de S_0 représentent les états contrôlés par le joueur 0, et ceux de S_1 les états contrôlés par le joueur 1. Un arc $(x \to y) \in A$ représente un coup possible. Un état, $s_{\text{init}} \in (S_0 \uplus S_1)$, est désigné comme l'état initial.

Exemple 21 **Le graphe modélisant le morpion** contient 5478 états et 16167 arcs. Il s'agit d'un graphe orienté acyclique.

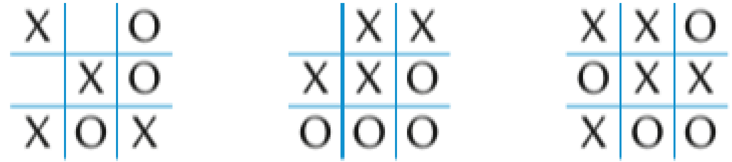
Définition 22 **Un état terminal** est un sommet sans arc sortant.

Cas particuliers 23 On considérera deux types de jeux:

- dans les **jeux d'accessibilité** l'ensemble des états finaux est partitionné en W_0 l'ensemble des états faisant gagner le joueur 0, W_1 l'ensemble des états faisant gagner le joueur 1 et W_N l'ensemble des états pour lesquels il y a un match nul.
- dans les **jeux à somme nulle**, une fonction de score $w : E \to \mathbb{R}$ donne le score du joueur 0. Le score du joueur 1 est l'opposé du score du joueur 0.

Remarque 24 Un jeu d'accessibilité est un cas particulier de jeux à somme nulle, en posant $w(e) = \begin{cases} 1 \text{ si } e \in W_0 \\ 0 \text{ si } e \in W_N \\ -1 \text{ si } e \in W_1 \end{cases}$

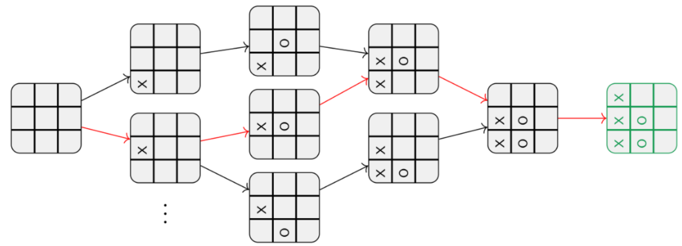
Exemple 25 **Le morpion est un jeu d'accessibilité**



Un état gagnant pour X — Un état gagnant pour O — Un match nul

Définition 26 **Une partie** est un chemin dans G, depuis s_{init} vers un état terminal.

Exemple 27 Le chemin rouge représente un exemple de partie de morpion.



B. Traitement exhaustif

1. Cas des jeux d'accessibilité

Définition 28 **Une stratégie sans mémoire** pour le joueur $X \in \{0, 1\}$ est une fonction partielle de $S_X \to A$ qui à tout $s \in S_X$ non terminal, associe un coup $f(s)$.

Remarque 29 On défini ici les stratégies sans mémoire, qui ne dépendent que de l'état courant. Dans une définition plus générale, le domaine d'une stratégie pourrait être un historique de coups.

Définition 30 On dit qu'un chemin $s_1 \to \ldots \to s_n$ de G **suit une stratégie** f pour le joueur X si pour tout $s_i \in S_X$ non-terminal, on a $f(s_i) = s_i \to s_{i+1}$.

Définition 31 Une stratégie f pour le joueur X est **gagnante** depuis un état s si tout chemin suivant f entre s et un état final abouti à un état gagnant pour X.

Définition 32 Un état est appelé une **position gagnante** pour le jouer X s'il existe une stratégie gagnante pour X depuis cet état. L'ensemble des positions gagnantes est appelé l'**atracteur** de X.

Remarque 33 Les attracteurs peuvent être caractérisés comme les limites des suites suivantes:

$$-A_0^X = W_X$$

$$-A_{i+1}^X = A_i^X \cup \{s \in S_X \mid \exists s' \in A_i^X \text{ tel que } s \to s'\} \cup \{s \in S_{1-X} \mid \forall s' \text{ tel que } s \to s' \text{ alors } s' \in A_i^X\}$$

Complexité 34 Ces points fixes peuvent être calculés en $O(|S|^2)$.

Remarque 35 [GRA] Les attracteurs peuvent être calculés avec une complexité linéaire en la taille du graphe de jeu. DEV

2. Cas des jeux à somme nulle: algorithme min-max [TOR 9.7.2.1]

Hypothèse 36 On se limitera pour la suite aux jeux dont le graphe **ne contient pas de cycles**.

Algorithme 37 **Min-max**

```
Entrée: un état s du graphe de jeu
Sortie: meilleur score que peut réaliser le joueur 0 depuis s,
si le joueur 1 joue de façon optimale

MinMax(s):
  Si s est un état terminal:
    renvoyer w(s)
  Si s est contrôlé par le joueur 0:
    renvoyer le maximum, pour s' successeur de s, de MinMax(s')
  Si s est contrôlé par le joueur 1:
    renvoyer le minimum, pour s' successeur de s, de MinMax(s')
```

Remarque 38 L'ordre dans lequel l'algorithme MinMax parcours le graphe de jeu forme un arbre. On parle d'**arbre de parcours**. Dans cet arbre, on appelle **nœud min** un nœud contrôlé par le joueur 1 (sur lequel l'algorithme calcule un minimum) et **nœud max** un nœud contrôlé par le joueur 0 (sur lequel l'algorithme calcule un maximum).

Complexité 39 Comme le calcul des attracteurs pour les jeux d'accessibilité, l'algorithme MinMax a une complexité linéaire en la taille du graphe de jeu.

Limite 40 La taille des graphes modélisant la plupart des jeux à deux joueurs sont suffisamment grandes pour qu'une complexité linéaire en la taille du graphe se révèle irréalisable en pratique.

C. Traitement non-exhaustif

1. Élagage alpha-beta

Remarque 41 Dans l'algorithme MinMax, on peut s'épargner des calculs en faisant les deux constats suivants:

- Pour un nœud min s, si f un des fils de s renvoie un résultat plus faible que celui déjà renvoyé par g l'un des grand-frères de s, alors il n'y a pas besoin d'explorer les autres fils de s. En effet, on aura $\text{MinMax}(s) \leq \text{MinMax}(f) \leq \text{MinMax}(\mathrm{g})$ et $\max(\text{MinMax}(s), \text{MinMax}(g)) = \text{MinMax}(g)$.
- Pour un nœud max s, si f un des fils de s renvoie un résultat plus grand que celui déjà renvoyé par g l'un des grand-frères de s, alors il n'y a pas besoin d'explorer les autres fils de s. En effet, on aura $\text{MinMax}(s) \geq \text{MinMax}(f) \geq \text{MinMax}(\mathrm{g})$ et $\min(\text{MinMax}(s), \text{MinMax}(g)) = \text{MinMax}(g)$.

Algorithme 42 **L'élagage alpha-beta** est une variante de l'algorithme MinMax qui exploite ces deux constats.

2. Estimation

Définition 43 **Une heuristique pour un jeu à somme nulle** est une fonction $h : S \to \mathbb{R}$ qui *estime* le score que le joueur 1 peut espérer depuis chaque état.

Variante 44 Il existe une **variante de l'algorithme MinMax** dans laquelle, au delà d'un certain nombre d'appels récursifs, on remplace les appels récursif par l'appel à une heuristique.

Remarque 45 **Cette variante de l'algorithme MinMax renvoie une estimation et non pas un résultat exact.** On remarquera le contraste avec l'algorithme A*, qui bien qu'utilisant une heuristique renvoie un résultat exact, lorsque l'heuristique est admissible.

Algorithmes pour l'étude des jeux. Exemples et applications

I. Jeux à un joueur [BAR 11.5.1]

A. Formalisation d'un jeu

1 Formalisation Jeu
2 Formalisation Résolution
3 Ex Jeu de Taquin
4 Def [NAN] Puzzle
5 Rem Explosion

B. L'algorithme A* [BAR 11.5.2]

6 Def L'algorithme A*
7 Ex Cercles concentriques Dijkstra vs A*
8 Def Heuristique pour A*
9 Def Une heuristique est admissible
10 Def Une heuristique est consistante
11 Def Une heuristique est monotone
12 Ex Heuristique pour le Taquin

C. Réduction à SAT

13 Rem [NAN]
14 Ex Le puzzle du Sudoku
15 Réduction Sudoku à SAT
16 Algo DPLL

II. Jeux à deux joueurs [TOR 9.7.2]

A. Définitions

17 Def Un jeu à information complète
18 Def Un jeu à coups asynchrone
19 Motiv
20 Def Un jeu à deux joueurs
21 Ex Le graphe modélisant le morpion
22 Def Un état terminal
23 Cas particuliers Jeux d'accessibilité et jeux à somme nulle
24 Rem Accessibilité est somme nulle
25 Ex Le morpion est un jeu d'accessibilité
26 Def Une partie
27 Ex Exemple de partie

B. Traitement exhaustif

28 Def Une stratégie sans mémoire
29 Rem Stratégie sans mémoire
30 Def Chemin suivant une stratégie
31 Def Stratégie gagnante
32 Def Position gagnante & Attracteur
33 Rem Attracteur par point fixe
34 Complex
35 Rem Complexité du calcul de l'attracteur
36 Hypothèse Graphe sans cycle
37 Algo Min-max
38 Rem Arbre de parcours, nœuds min et max
39 Complex Min-Max linéaire
40 Limite Traitement exhaustif impraticable

C. Traitement non-exhaustif

41 Rem alpha-beta améliore min-max
42 Algo L'élagage alpha-beta
43 Def Une heuristique pour un jeu à somme nulle
44 Variante Min-max à profondeur bornée
45 Rem Cette variante de l'algorithme MinMax renvoie une estimation et non pas un résultat exact.

Bibliographie

[BAR] V. Barra, *Informatique MPI/MP2I.*
[TOR] T. Balabonski & S. Conchon & J. Filliâtre & K. Nguyen & L. Sartre, *MP2I MPI, Informatique Cours et exercices corrigés.*
[GRA] E. Grädel, *Finite Model Theory and Its Applications.*

Alexis Hamon & Santiago Sara Bautista