

Intelligence Artificielle (IA)

Les jeux, recherche avec horizon (I)

Akka Zemmari

LaBRI, Université de Bordeaux

2023 - 2024

Introduction d'*heuristiques*

Idée

On ne développe l'arbre de recherche que jusqu'à une certaine profondeur maximale p . Les feuilles de l'arbre ne sont plus forcément des positions finales du jeu.

On ne peut donc plus se contenter de l'information « gagné » ou « perdu ». Il faut **évaluer** les positions.

Introduction d'*heuristiques*

Idée

On ne développe l'arbre de recherche que jusqu'à une certaine profondeur maximale p . Les feuilles de l'arbre ne sont plus forcément des positions finales du jeu.

On ne peut donc plus se contenter de l'information « gagné » ou « perdu ». Il faut **évaluer** les positions.

Définition intuitive

Les heuristiques sont des **fonctions statiques** associant un **réel** aux **plateaux de jeu**. Dans les jeux avec adversaire, plus l'heuristique est grande et positive, plus on est proche de la victoire. Plus elle est négative et petite, plus on est proche de la défaite.

Exemple des dominos 3×3

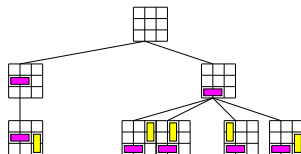
Idée

- ▶ On développe l'arbre (ou graphe) de recherche jusqu'à une profondeur donnée (déterminée suivant le temps que l'on a pour jouer et le facteur de branchement estimé du jeu)
- ▶ On évalue la position aux feuilles (dépend du plateau et du joueur)
- ▶ On fait remonter l'évaluation pour *trouver le meilleur coup garantissant au moins l'estimation de la feuille*

Exemple des dominos 3 × 3

Idée

- ▶ On développe l'arbre (ou graphe) de recherche jusqu'à une profondeur donnée (déterminée suivant le temps que l'on a pour jouer et le facteur de branchement estimé du jeu)
- ▶ On évalue la position aux feuilles (dépend du plateau et du joueur)
- ▶ On fait remonter l'évaluation pour *trouver le meilleur coup garantissant au moins l'estimation de la feuille*



Exemple d'heuristiques : les Échecs dès 1950

- (1) The relative values of queen, rook, bishop, knight and pawn are about 9, 5, 3, 3, 1, respectively. Thus other things being equal (!) if we add the numbers of pieces for the two sides with these coefficients, the side with the largest total has the better position.
- (2) Rooks should be placed on open files. This is part of a more general principle that the side with the greater mobility, other things equal, has the better game.
- (3) Backward, isolated and doubled pawns are weak.
- (4) An exposed king is a weakness (until the end game).

These and similar principles are only generalizations from empirical evidence of numerous games, and only have a kind of statistical validity. Probably any chess principle can be contradicted by particular counter examples. However, from these principles one can construct a crude evaluation function. The following is an example: -

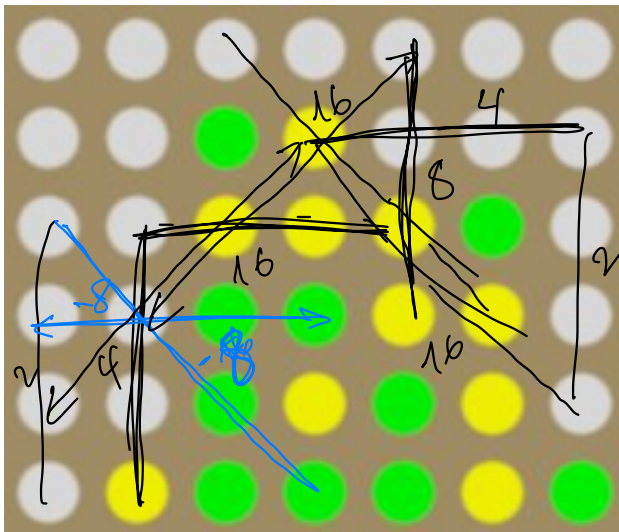
$$f(P) = 200(K-K') + 9(Q-Q') + 5(R-R') + 3(B-B'+N-N') + (P-P') - \\ 0.5(D-D'+S-S'+I-I') + \\ 0.1(M-M') + \dots$$

in which: -

- (1) K, Q, R, B, N, P are the number of White kings, queens, rooks, bishops, knights and pawns on the board.
- (2) D, S, I are doubled, backward and isolated White pawns.
- (3) M = White mobility (measured, say, as the number of legal moves available to White).

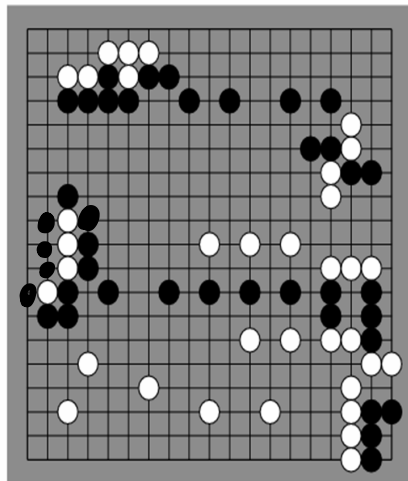
Exemple d'heuristiques : Puissance 4

$f()$



= A
B

Exemple d'heuristiques : Go



Introduction d'*heuristiques*

Nouveau but

Trouver la branche permettant de maximiser la valeur heuristique obtenue sur le plateau après les p prochains coups.

Introduction d'*heuristiques*

Nouveau but

Trouver la branche permettant de maximiser la valeur heuristique obtenue sur le plateau après les p prochains coups.

Supposition forte

On suppose que les deux joueurs jouent « bien » s'ils suivent les indications de l'heuristique.

Introduction d'*heuristiques*

Nouveau but

Trouver la branche permettant de maximiser la valeur heuristique obtenue sur le plateau après les p prochains coups.

Supposition forte

On suppose que les deux joueurs jouent « bien » s'ils suivent les indications de l'heuristique.

En pratique, il faudra prendre en compte cela pour des raisons de performances. À chaque feuille de l'arbre, on devra calculer sa valeur heuristique

Introduction d'*heuristiques*

Nouveau but

Trouver la branche permettant de maximiser la valeur heuristique obtenue sur le plateau après les p prochains coups.

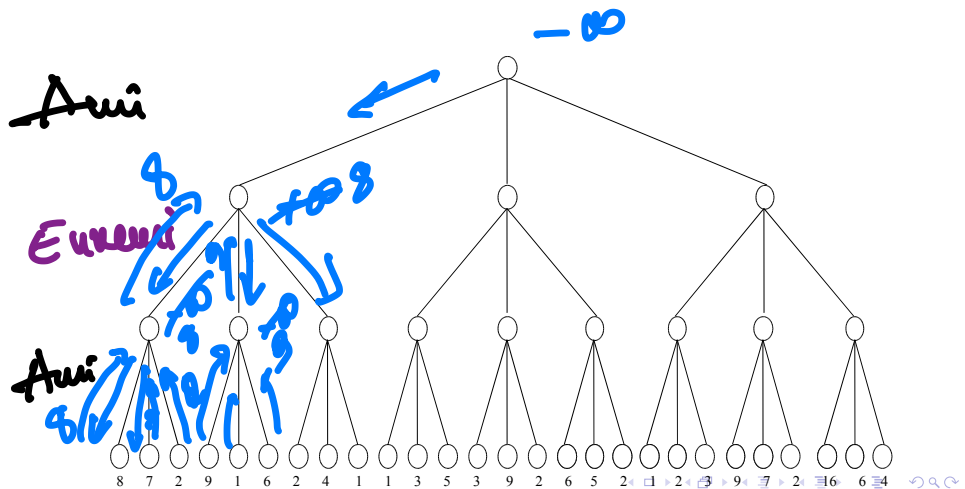
Supposition forte

On suppose que les deux joueurs jouent « bien » s'ils suivent les indications de l'heuristique.

En pratique, il faudra prendre en compte cela pour des raisons de performances. À chaque feuille de l'arbre, on devra calculer sa valeur heuristique

Problème : Comment faire remonter la meilleur feuille à la racine ?

Exemple d'arbre de jeu



Algorithme MiniMax – Fonction *MaxMin*

Idée

On appelle *MaxMin* pour évaluer la valeur MiniMax de la racine.
On doit aussi remonter le *meilleur coups à jouer* qui lui est associé.

- 1: **Fonction** *MaxMin*(*etat*) ▷ Évaluation niveau AMI
- 2: *etat* : Plateau de jeu courant
- 3: **Si** *EstFeuille*(*etat*) **Alors**
- 4: **Retourner** *evalue*(*AMI*, *etat*) ▷ Évaluation heuristique
- 5: **Fin Si**
- 6: *Meilleur* $\leftarrow -\infty$
- 7: **Pour Tout** successeur *s* de *etat* **Faire**
- 8: *Meilleur* $\leftarrow \max(\text{Meilleur}, \text{MinMax}(s))$
- 9: **Fin Pour**
- 10: **Retourner** *Meilleur*
- 11: **Fin Fonction**

Algorithme MiniMax – Fonction *MinMax*

Attention

Si l'évaluation de la position a lieu sur un niveau impair (c'est à *ENNEMI* de jouer), la fonction heuristique doit en tenir compte !

- 1: **Fonction** *MinMax*(*etat*) ▷ Évaluation niveau ENNEMI
- 2: *etat* : Plateau de jeu courant
- 3: **Si** *EstFeuille*(*etat*) **Alors**
- 4: **Retourner** *evalue*(*ENNEMI*, *etat*) ▷ Évaluation
 heuristique
- 5: **Fin Si**
- 6: *Pire* ← $+\infty$
- 7: **Pour Tout** successeur *s* de *etat* **Faire**
- 8: *Pire* ← *min*(*Pire*, *MaxMin*(*s*))
- 9: **Fin Pour**
- 10: **Retourner** *Pire*
- 11: **Fin Fonction**

Les deux fonctions MiniMax

Fonction *MaxMin*(*etat*)

etat : Plateau de jeu courant

Si *EstFeuille*(*etat*) **Alors**

Retourner *evalue*(*AMI*, *etat*)

Fin Si

Meilleur $\leftarrow -\infty$

Pour Tout successeur *s* de *etat* **Faire**

Meilleur $\leftarrow \max(\text{Meilleur}, \text{MinMax}(s))$

Fin Pour

Retourner *Meilleur*

Fin Fonction

Fonction *MinMax*(*etat*)

etat : Plateau de jeu courant

Si *EstFeuille*(*etat*) **Alors**

Retourner *evalue*(*ENNEMI*, *etat*)

Fin Si

Pire $\leftarrow +\infty$

Pour Tout successeur *s* de *etat* **Faire**

Pire $\leftarrow \min(\text{Pire}, \text{MaxMin}(s))$

Fin Pour

Retourner *Pire*

Fin Fonction

▷ Évaluation niveau AMI

▷ Évaluation heuristique

▷ Évaluation niveau ENNEMI

▷ Évaluation heuristique

Limitations des principes MiniMax

- ▶ Que se passe-t-il si un joueur
 - ▶ Ne joue pas « bien » ?
 - ▶ Ne joue pas selon l'estimation heuristique ?
- ▶ Problèmes de collaboration :

- ▶ Dilemme du prisonnier

	Silence	Accusation
Silence	0 0	10 1
Accusation	1 10	5 5

- ▶ Amplification de valeurs heuristiques
 - ▶ Une valeur faible mais isolée dans une zone dangereuse sera préférée à une valeur presque identique dans une zone plus « amie »
 - ▶ En pratique cet effet est contre-balancé par les valeurs heuristiques qui sont théoriquement éloignées aux feuilles.
- ▶ Le résultat est le même, après n'importe quelle application d'une fonction monotone sur les valeurs heuristiques.

Recherche dans les arbres de jeux

L'espace des états atteignables peut être gigantesque. Il dépend :

- ▶ Facteur de branchement b du jeu
- ▶ Nombre de coups n d'une partie

Idée des tailles de certains jeux

- ▶ Échecs : $b \sim 35$ et $n \sim 30$ coups au minimum
|graphe d'états| = 35^{30}
Soit 2099139642966190174953146230280399322509765625
- ▶ Othello : $5 \leq b \leq 15$
- ▶ Go : $b \sim 360$

Il faut retrouver l'idée d'élagage de l'arbre de recherche.