Les jeux

Historique

- 1770: The Turk
- 1830: The analytical engine (Babbage)
- 1910: El ajedrecisto (Leonardo Torres)
- 1950: La machine de Shannon
- 1952: Turochamp (Turing)

Donald Michie

- « La recherche sur le jeu d'échecs est le champ le plus important de la recherche cognitive. Les échecs seront pour nous ce que la drosophile a été pour les généticiens : un moyen simple et pratique de développer de nouvelles techniques. »

Points forts

- Domaine complètement spécifiable et à information totale
- Résultats facilement lisibles: victoire ou défaite
- Domaine « représentatif » de l'intelligence humaine
- Domaine « populaire » qui attire l'attention...et les financements
- Domaine riche et complexe

Principe minimax

- On génère l'ensemble des positions que l'ordinateur peut atteindre en jouant un coup (niveau 1)
- À partir de chacune des positions de niveau 1, on génère les positions que l'adversaire peut atteindre (niveau 2).
- On peut alors recommencer l'opération aussi longtemps que le permet la puissance de calcul de l'ordinateur.

Principe minimax

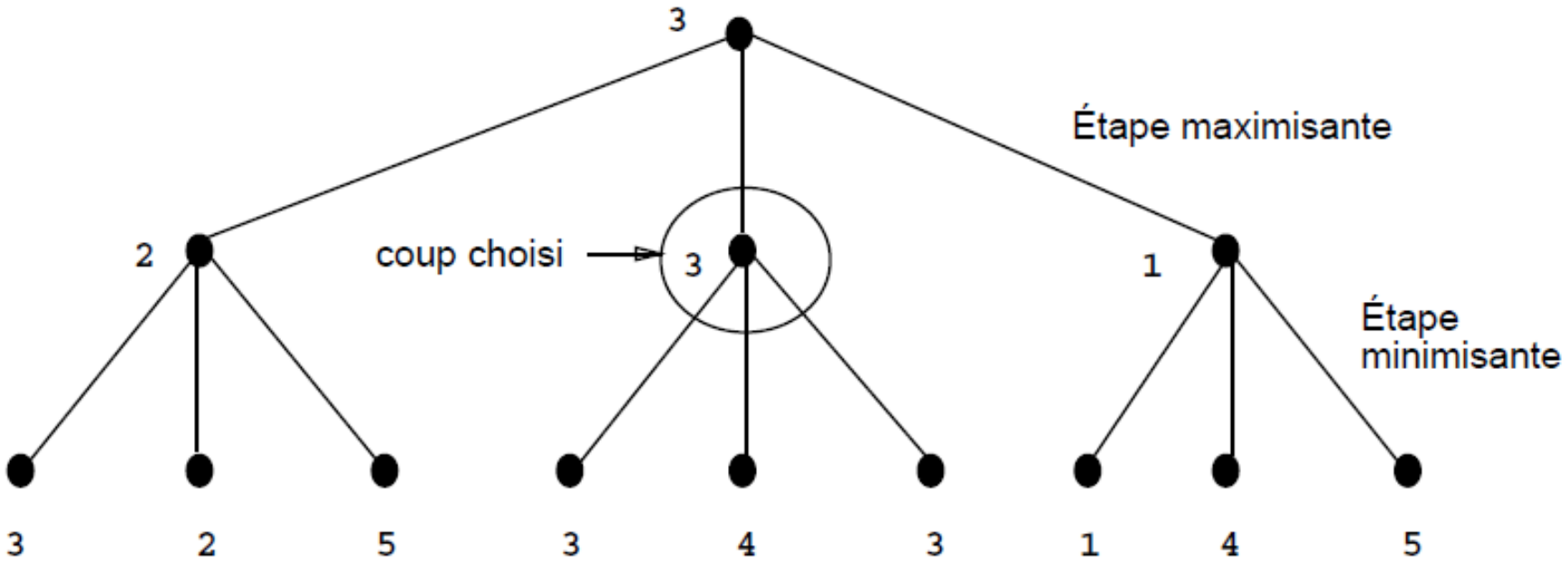
- Impossible de générer l'ensemble de l'espace d'états du problème.
- Aux échecs, le facteur de branchement est environ de 35.
- Une partie d'échecs : trente demi-coups.
- 35^30=20.991.396.429.661.901.749.543.146.230.280.399.322.509.765.625
- Temps de résolution>âge de l'univers

Fonction d'évaluation

- Limiter l'exploration à une profondeur maximale de résolution.
- Lorsqu'on a atteint cette profondeur, attribuer à chaque feuille une valeur avec une *fonction d'évaluation*.
- Cette valeur correspond à l'estimation de la position.
- Equilibre entre vitesse/qualité de la fonction

Algorithme minimax

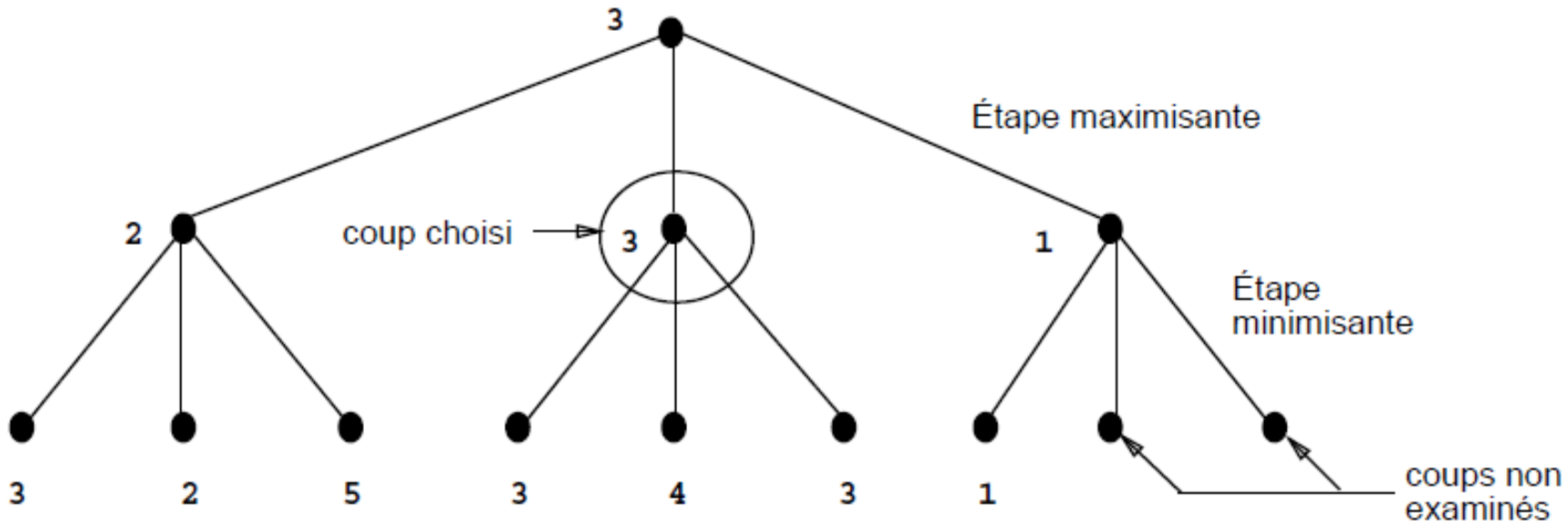
- Implante le principe de recherche minimax



Algorithme alpha-beta

-Implante également le principe minimax

-Examine moins de nœuds



Efficacité de l'alpha-beta

- L'alpha-béta est d'autant plus efficace que l'arbre est bien ordonné.
- Si l'arbre est bien ordonné, un alpha-béta n'examinera que $\sqrt{n}$ feuilles sur un arbre contenant n feuilles

Iterative deepening

- Première évaluation en profondeur n
- On utilise les informations obtenus sur les feuilles et les nœuds de l'arbre pour ordonner l'arbre dans la recherche en profondeur n+1
- Permet un meilleur contrôle du temps

Tables de transposition

- Stocker les positions déjà examinées

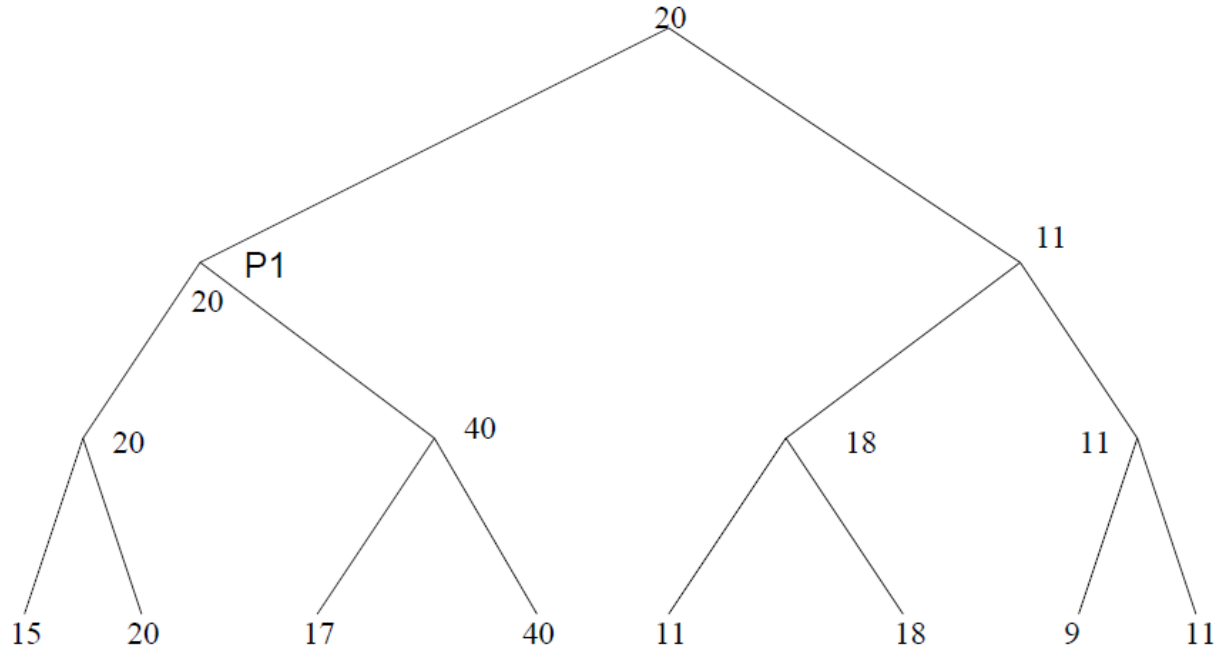
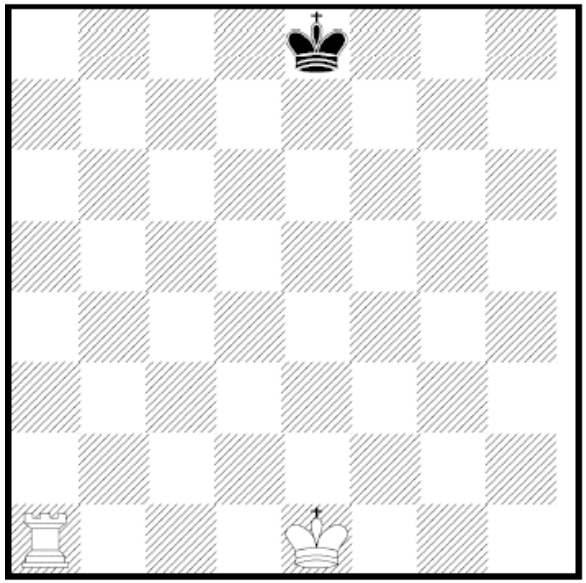


Figure 15.5 – Exemple d'arbre minimax

position	évaluation	meilleur coup	distance à la position terminale
P_1	20	Gauche	2

Tables de hachage



```
10101011110110110101010110101101  ⊕
11010110101001001010001011010110  ⊕
00100101010110110100100010101000  =
-------------------------------------
01011000001001001011111111010011
```

Pièce	couleur	position	valeur de hash-coding (choisie aléatoirement)
Tour	blanc	a1	10101011110110110101010110101101
Tour	blanc	a2	01000101001011001110100111101010
…	…	…	…
Roi	blanc	e1	11010110101001001010001011010110
…	…	…	…
Roi	noir	e8	00100101010110110100100010101000
…	…	…	…

Tables de transposition

- Eviter les erreurs: deux positions différentes qui ont le même index -> choisir des nombres suffisamment grands
- Traiter les collisions: deux positions différentes qui ont le même index

Alpha-beta à mémoire

```
Alpha-Béta(n, α, β);
si n est dans la table alors
    si n.bas >= β alors retourner n.bas;
    si n.haut <= α alors retourner n.haut;
    α ← max(α, n.bas);
    β ← min(β, n.haut);
si n est terminal alors retourner h(n);
si n est de type Max alors
    soit g ← −∞;
    soit j ← 1;
    soit (f_1 ... f_k) les fils de n;
    tant que (g < β et j ≤ k) faire
        g ← max(g,Alpha-Béta(f_j, α, β));
        α ← max(α, g);
        j ← (j + 1);
sinon
    soit g ← +∞;
    soit j ← 1;
    soit (f_1 ... f_k) les fils de n;
    tant que (g > α et j ≤ k) faire
        g ← min(g,Alpha-Béta(f_j, α, β));
        β ← min(β, g);
        j ← (j + 1);
si g <= α alors n.bas ← g;
si g >= β alors n.haut ← g;
si g > α et g < β alors
    n.bas ← g;
    n.haut ← g;
retourner g
```

Alpha-beta à fenêtre réduite

- Utiliser l'information retournée par l'évaluation précédente pour réduire la fenêtre de l'alpha-beta
- D'autant plus efficace que l'évaluation est stable
- Efficace avec l'alpha-béta à mémoire

Algorithme MTD(f)

Algorithme 15: MTD(f)

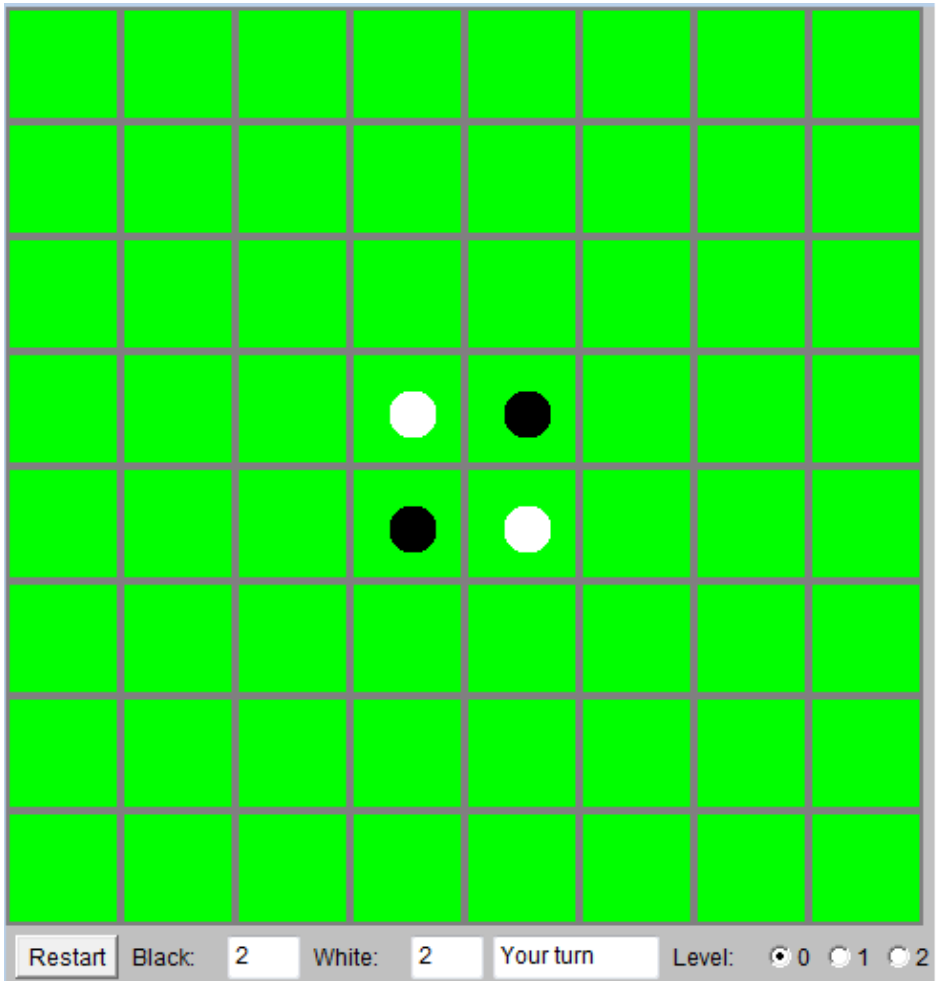
; ; ; $root$ est la racine de l'arbre;

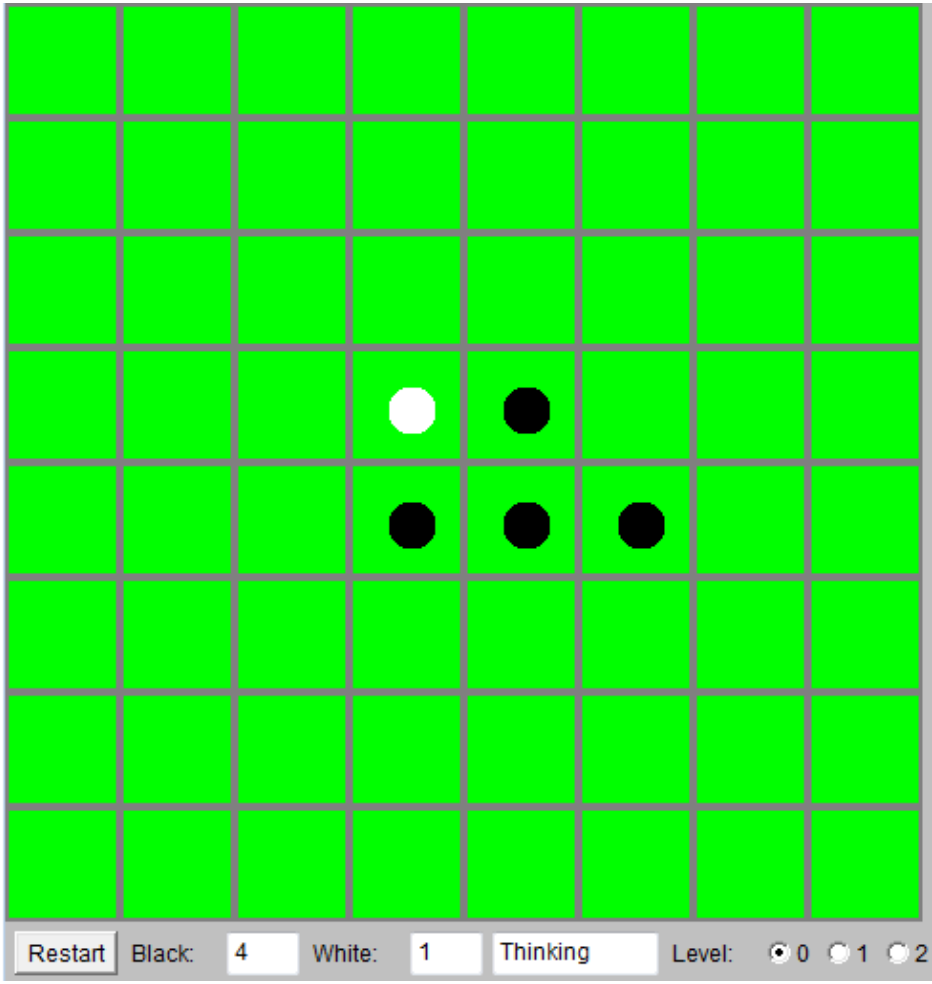
MTD($root$, f);
soit $g \leftarrow f$;
soit $h \leftarrow +\infty$;
soit $b \leftarrow -\infty$;
tant que $h > b$ **faire**
    si $g = b$ **alors** $\beta \leftarrow g + 1$ **sinon** $\beta \leftarrow g$;
    $g \leftarrow$Alpha-Béta($root, \beta - 1, \beta$);
    si $g < \beta$ **alors** $h \leftarrow g$ **sinon** $b \leftarrow g$
retourner g

Parallélisation d'un algorithme de recherche

- Problème majeur: choisir la granularité
  - Trop petite: trop d'allers-retours entre le processeur maître et les autres
  - Trop grande: les informations retournées par l'alpha-béta ne sont pas pleinement utilisées
- Aujourd'hui la totalité des bons programmes sont parallèles

Othello


Restart
Black: 2
White: 2
Your turn
Level: 0 1 2


Restart
Black: 4
White: 1
Thinking
Level: 0 1 2

Othello: les ouvertures

- Utilisation de table de transpositions
- Domaine peu développé
- Le meilleur programme (Logistello) utilise des techniques assez originales

Othello: les finales

- Othello est un jeu à durée connue (rare)
- On utilise un algorithme Cstar (variante d'un alpha-beta à fenêtre) avec une fonction d'évaluation exacte
- On déclenche la recherche exhaustive V/N/D plus de 20 coups avant la fin
- On déclenche la recherche exacte plus de 15 coups avant la fin

Othello: milieu de partie

- Valuation statique

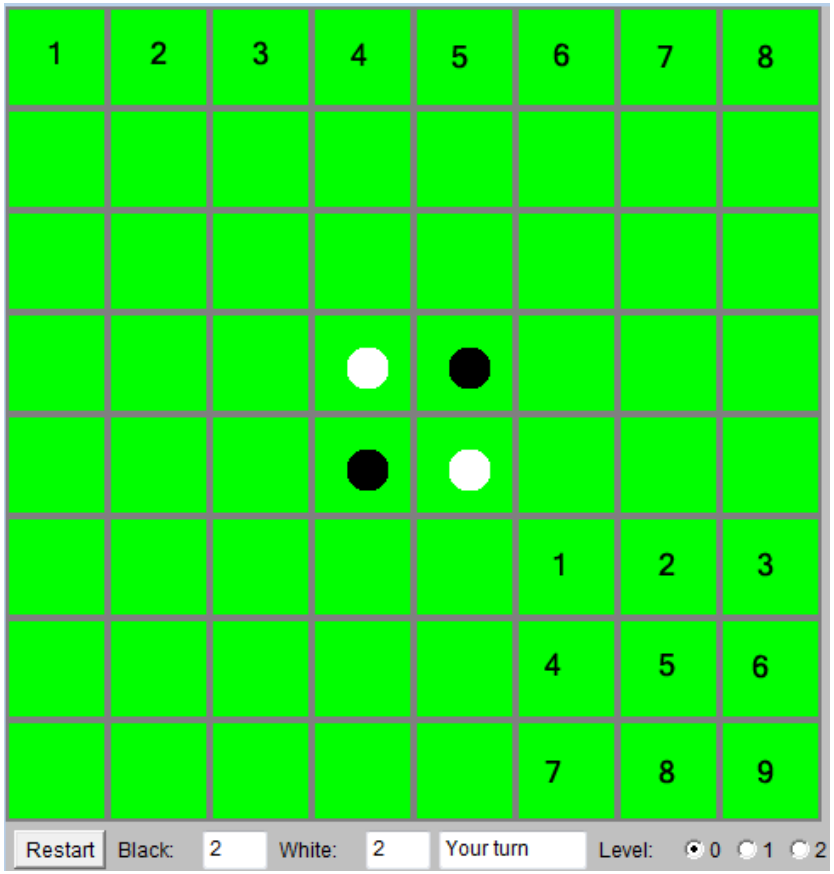
500	-150	30	10	10	30	-150	500
-150	-250	0	0	0	0	-250	-150
30	0	1	2	2	1	0	30
10	0	2	16	16	2	0	10
10	0	2	16	16	2	0	10
30	0	1	2	2	1	0	30
-150	-250	0	0	0	0	-250	-150
500	-150	30	10	10	30	-150	500

Othello: milieu de partie

- Evaluation dynamique:
  - Centrer au maximum les pions
  - Maximiser le nombre de coups jouables
  - Minimiser le nombre de coups jouables de l'adversaire

Othello: milieu de partie

- Système d'apprentissage utilisé par tous les programmes
- 2 bits/case (N/B/V)
- 1 bit (N ou B va jouer)
- Symétries
- Positions stockées sur:
  - 19 bits (coin)
  - 17 bits (bord)



Othello

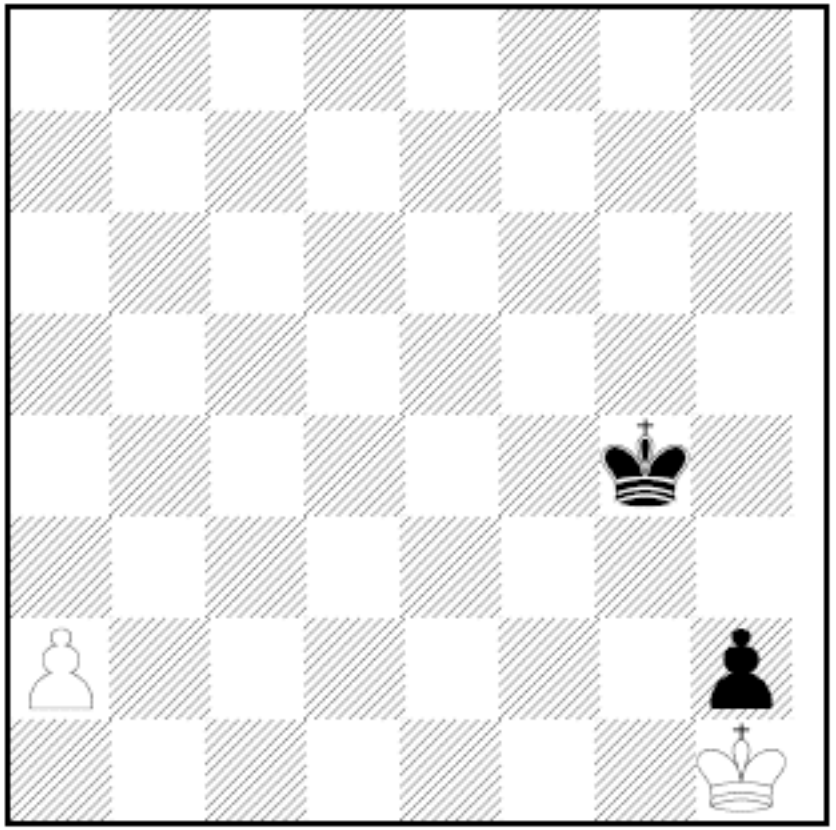
- Les programmes sont aujourd'hui beaucoup plus forts que les meilleurs joueurs humains
  - 1997: Logistello bat Takeshi Murakami 6-0
- Les développements nouveaux ont quasiment cessés.

Les échecs: ouvertures

- Domaine largement étudié au cours des siècles
- Enorme bibliographie
- Certains programmes, comme Hydra, préfèrent renoncer partiellement à une bibliothèque d'ouverture
- Métier à part entière consistant à préparer des lignes spécifiques

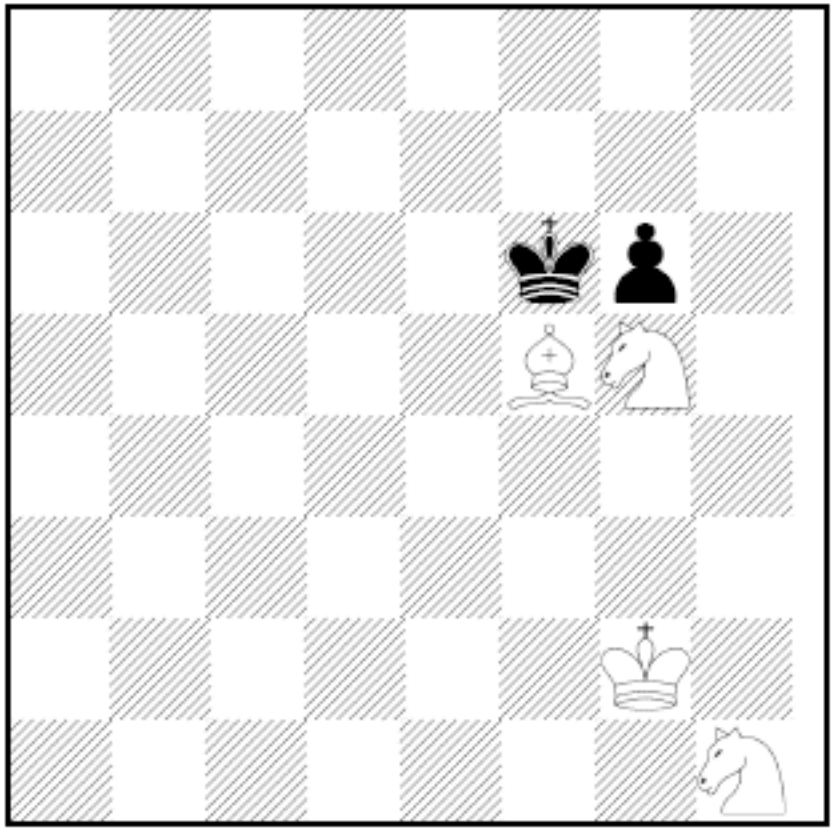
Les échecs: finales

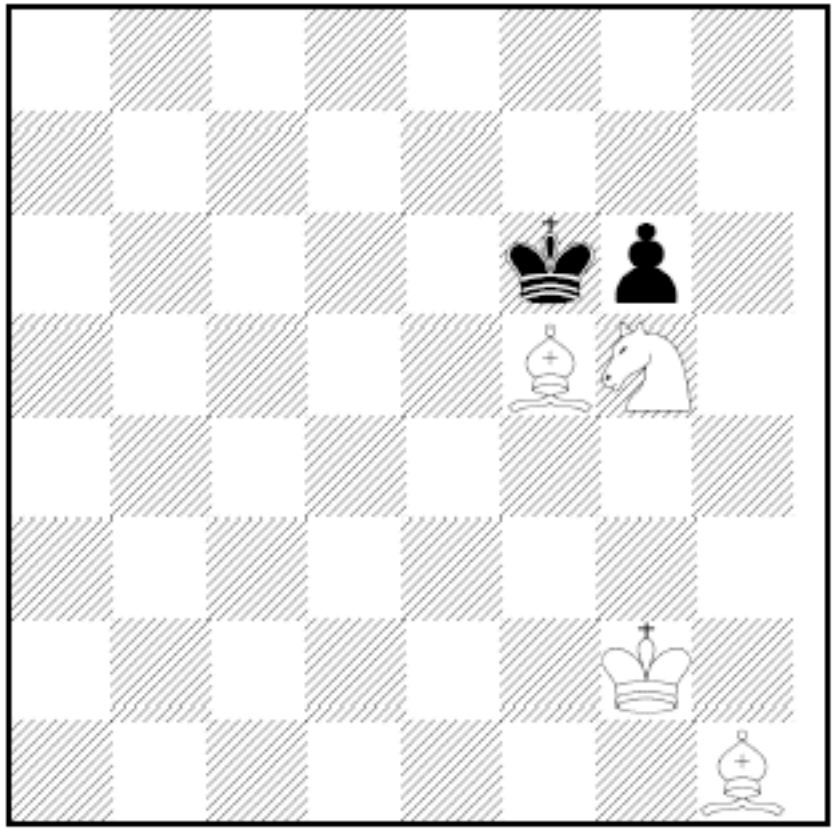
- Problème de l'effet horizon
- Il faudrait examiner la position en profondeur 11 pour voir que a3 est gagnant et Rxh2 perdant



Echecs: finales

- A droite Fxg6 gagne, mais perd à gauche



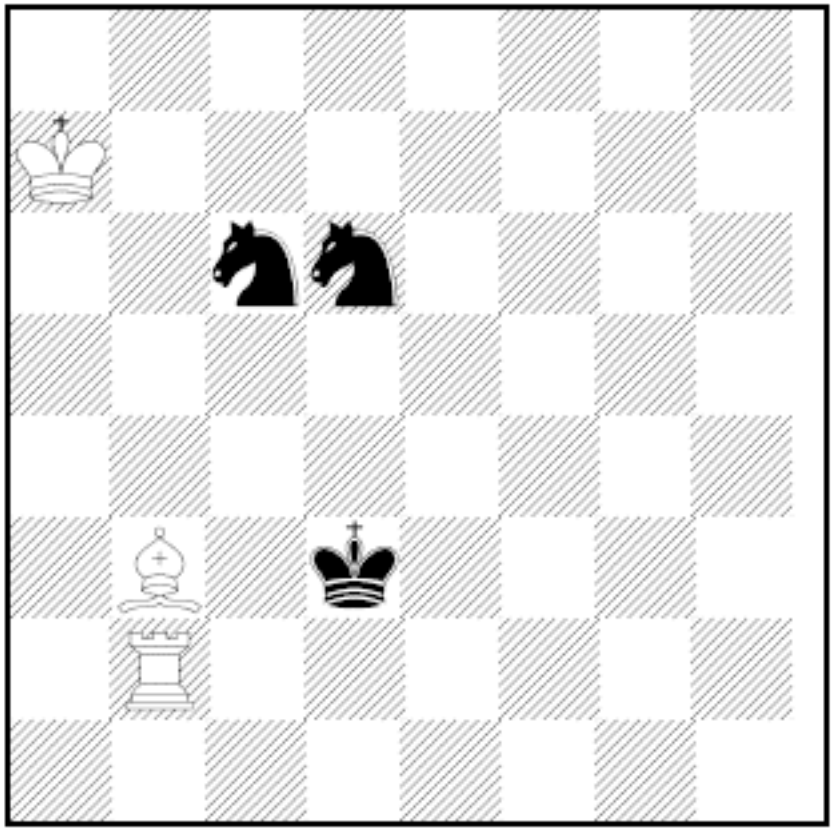


Echecs: finales

- L'analyse rétrograde consiste à partir d'une position de mat et à rechercher toutes les positions menant mecaniquement à celle-ci.
- En appliquant récursivement ce mécanisme, on peut construire de grandes bases de données contenant des solutions exactes.
- Toutes les finales à 5 pièces sont faites

Les échecs: finales

- Larry Stiller (1991): nécessite 223 coups en jouant parfaitement pour gagner



Les échecs: fonction d'évaluation

- Partie statique: la valeur des pièces
  - Dame: 9
  - Tour: 5
  - Cavalier: 3
  - Fou: 3
  - Pion: 1

Les échecs: fonction d'évaluation

- Mettre son Roi en sécurité
- Conserver la paire de Fous
- Dominer le centre
- Garder un maximum de mobilité
- Centraliser les Cavaliers
- Placer le Fou sur des diagonales ouvertes
- Placer la Tour sur des colonnes ouvertes
- Doubler la Tour sur les colonnes ouvertes
- Conserver une bonne structure de pions

Les échecs: quiescence

- Faire une recherche spécifique depuis chaque feuille n'incluant que les prises permettant d'atteindre une position « calme »
- Permet d'éviter l'instabilité de la fonction d'évaluation

Echecs: Elagage de futilité

- Au lieu d'élaguer sur les positions strictement inférieures, on élague sur les positions inférieures ou supérieures de pourcent
- Accélère la recherche au détriment de l'optimalité

Le coup meurtrier

- Si un coup est meurtrier (il provoque un effondrement brutal de l'évaluation) dans une position donnée, il est possible qu'il soit efficace dans d'autres positions
- On mémorise ces coups dans des tables spécifiques

Extension sélective de recherche

- Dans un alpha-béta normal, la profondeur est fixé à l'avance
- Dans un système d'extension sélective, la profondeur dépend de l'intérêt de la branche.
- Permet d'explorer plus en profondeur les branches contenant des coups complexes

Heuristique du coup nul

- Si on doit évaluer un nœud en profondeur n, on essaie d'abord de l'évaluer en profondeur n-2 en supposant que l'adversaire passe son tour
- Si un avantage clair n'est pas acquis, le nœud est élagué
- Stratégie apparemment risqué mais extrêmement efficace

Les échecs

- La recherche a toujours avancé par à-coup
- Les innovations restent généralement « secrètes » pendant un certain temps pour des raisons commerciales
- Le meilleur programme actuel (Rybka) a une force estimée de plus de 3000 ELO
- Un humain a aujourd'hui fort peu de chance de battre un programme

Le backgammon

- Examiné de bonne heure, en particulier par Hans Berliner
- En 1980, BKG bat le champion du monde humain
- Les meilleurs programmes sont aujourd'hui probablement meilleurs que les meilleurs humains

Les checkers

- Dames américaines 8x8
- Chinook: meilleur programme développé par l'équipe de Jonathan Schaeffer
- En 1992, Chinook bat le champion du monde humain Marion Tinsley +4-2=34
- Depuis le programme a continué sa progression
- Aujourd'hui les checkers sont un jeu « presque » résolu

Go

- Longtemps considéré comme inaccessible aux ordinateurs
- Facteur de branchement trop important
- Notion d'influence difficile à traduire informatiquement
- Actuellement MOGO tournant sur un cluster de 200 processeurs à une force d'environ 1 dan amateur

Le bridge

- Il est difficile d'évaluer le niveau d'un programme de façon objective.
- Le meilleur programme actuel (GIB/Matthew Ginsberg) est moins fort que les meilleurs humains, mais est meilleur que la majorité des joueurs de club
- Il utilise des méthodes de force brute (Monte-Carlo)