Exploration et adversité

IA – Chapitre 5

Master IRAD - Université d'Orléans - Mars 2013

(Bref) historique

- Les jeux constituent l'une des branches initiales de l'IA
- Premier sujet d'intérêt : les échecs
- Konrad Zuse, Claude Shannon, Alan Turing...
- Dans certains jeux, les algorithmes actuels surpassent les humains

Exemple : deep blue

- Projet développé à partir de 1985
- Deux matches : 1996 (4-2) et 1997 (2,5-3,5)
- Mensurations : 700kg, 2m (1996) ; 1,4 T, 1,80m (1997)
- Performances : de 100 à 300 millions de coups par seconde, jusqu'à 12 demi-coups de profondeur en moyenne.
- L'ordinateur a ensuite été démantelé et le projet arrêté

Exemple : deep blue

- Théorie des jeux
- Plusieurs participants (ou adversaires ou agents)
- Coopération et/ou rivalité
- En IA :
  - A la base : jeux alternés, déterministes, à somme nulle
  - Notion de gain
  - Solutions (et exploration) complexes
  - Contrainte de temps (en général)
  - Solution optimale non atteignable

De quoi dispose-t-on ?

- Etat initial
- Fonction successeur : retourne (coup,état)
  $\rightarrow$ notion de coup légal
- Test de terminaison ($\rightarrow$ états terminaux)
- Fonction d'utilité (ou objectif ou de gain)
  - associe un score à chaque e.t.
  - généralement associé au décompte des scores spécifique au jeu
- Arbre de jeu (état initial + suite de coups légaux)

- Les joueurs jouent à tour de rôle
- Un coup = l'enchaînement de deux 1/2 coups (ou tours)
- Le score (utilité), à chaque niveau est donné pour un joueur
- Joueur MAX (objectif = score max) / joueur MIN (objectif = contrer MAX, donc obtenir un score min)
- Variante : Les deux joueurs cherchent à maximiser, mais l'un en positif, l'autre en négatif (négamax).

Algorithme du minimax

- Stratégie optimale, reposant sur une exploration complète
- Idée : trouver à chaque étape, le coup assurant un gain maximum (c'est à dire limitant les possibilités de gain de l'adversaire)
- Calcul récursif de l'arbre
- Remontée, depuis chaque feuille, des gains min et max (alternativement)

Valeur MINIMAX

- On donne un score à chaque nœud
- Ce score dépend du joueur cherchant à optimiser son score à ce niveau (c'est à dire le joueur ayant joué le coup).
- Joueur MAX : cherche à maximiser ; MIN : cherche à minimiser.
- VALEUR-MINIMAX(n) =
  UTILITE(n) si n état terminal
  $max_{s \in Successeurs(n)} VALEURMINIMAX(s)$ si n nœud MAX
  $min_{s \in Successeurs(n)} VALEURMINIMAX(s)$ si n nœud MIN
- On remonte la valeur minimax depuis les feuilles
- On obtient, à la racine la *décision minimax*

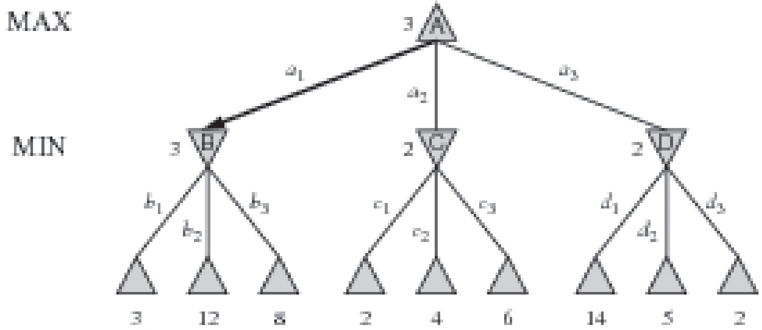
Algorithme de minimax

```
fonction DECISION_MINIMAX(etat)
  v <- VALEUR_MAX(etat)
  retourner l'action dans SUCCESSEURS_ETAT(etat)
    ayant la valeur v

fonction VALEUR_MAX(etat)
  si TEST_TERMINAL(etat) alors retourner UTILITE(etat)
  v <- -∞
  pour a,s dans SUCCESSEURS_ETAT(etat) faire
    v <- MAX(v, VALEUR_MIN(s))
  retourner v

fonction VALEUR_MIN(etat)
  si TEST_TERMINAL(etat) alors retourner UTILITE(etat)
  v <- ∞
  pour a,s dans SUCCESSEURS_ETAT(etat) faire
    v <- MIN(v, VALEUR_MAX(s))
  retourner v
```

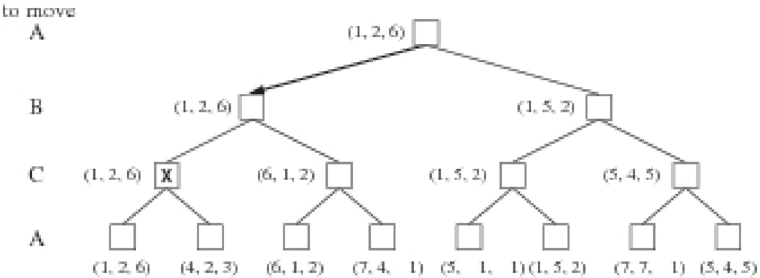
Exemple



Minimax et jeux multijoueurs

- Notre joueur doit maintenant tenir compte des intérêts de plusieurs adversaires
- hyp : Pas de stratégie d'alliance, chacun tente de maximiser son gain.
- on ne peut plus se contenter d'une information alternée min ou max
- Méthode : UTILITE retourne un vecteur
- Remarque : le cas à deux joueurs est une spécialisation de ce cas
  - vecteur à deux éléments + jeu à somme nulle $\rightarrow$ un seul élement
- Etat non terminal : on choisit le vecteur de l'état successeur conduisant au meilleur gain pour le joueur courant

Exemple


to move
A
B
C
A
(1, 2, 6)
(1, 2, 6)
(1, 5, 2)
(1, 2, 6)
X
(6, 1, 2)
(1, 5, 2)
(5, 4, 5)
(1, 2, 6)
(4, 2, 3)
(6, 1, 2)
(7, 4, 1)
(5, 1, 1)
(1, 5, 2)
(7, 7, 1)
(5, 4, 5)

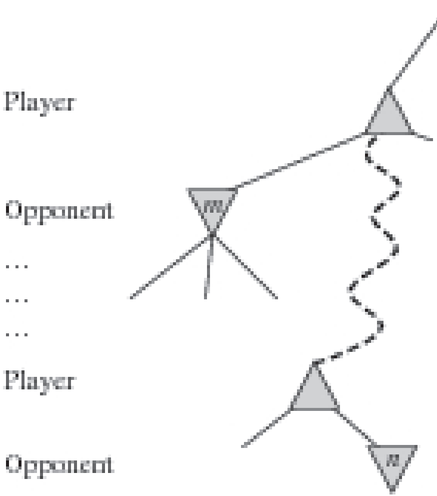
Bilan mimimax

- Exploration complète, DFS
- Coût (temps) en $O(b^m)$
- Coût espace en $O(bm)$ (tous successeurs en une fois) ou $O(m)$ (au besoin)
- Adapté à des problèmes (très) simples
- Peut-on éviter d'explorer certaines branches ?

Stratégie alpha-bêta

- Certains sous-arbres sont visiblement inintéressants pour le joueur courant
  - e.g. : on sait qu'il y existe une super opportunité pour l'adversaire...
- On décide de ne pas les explorer
- Technique d'élagage (ou *pruning*)
- Suppose d'avoir des certitudes sur l'intérêt d'une branche $\rightarrow$ génération d'au moins une feuille (DFS !)
- Evaluation de l'intérêt ?

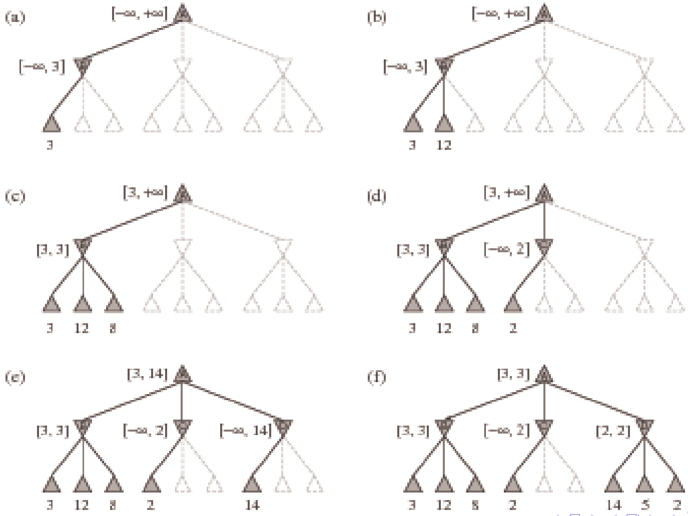
Principe général



Stratégie alpha-bêta : principe

- L'exploration se fait branche par branche (DFS)
- On connaît l'intérêt des branches déjà parcourues
- pour une branche en cours de parcours, on dispose d'une plage d'intérêt
  $[\beta, \alpha]$
  - $\alpha =$ meilleur choix pour joueur MAX
  - $\beta =$ meilleur choix pour joueur MIN
- Suivant le joueur du niveau de profondeur n, on met à jour, à partir des successeurs, α ou β.

Exemple


(a)
[−∞, +∞]
[−∞, 3]
3
(b)
[−∞, +∞]
[−∞, 3]
3
12
(c)
[3, +∞]
[3, 3]
3
12
8
(d)
[3, +∞]
[3, 3]
[−∞, 2]
3
12
8
2
(e)
[3, 14]
[3, 3]
[−∞, 2]
[−∞, 14]
3
12
8
2
14
(f)
[3, 3]
[3, 3]
[−∞, 2]
[2, 2]
3
12
8
2
14
5
2

Algorithme alpha-bêta

```
fonction RECHERCHE-ALPHA-BETA(etat)
  v ← VALEUR_MAX(etat,-∞,+∞)
  retourner l'action dans SUCCESSEURS (etat) de valeur v

fonction VALEUR_MAX(etat,α,β)
  si TEST_TERMINAL(etat) alors retourner UTILITE(etat)
  v ← −∞
  pour a,s dans SUCCESSEURS (etat) faire
    v ← MAX(v,VALEUR_MIN(s,α,β)
    si v ≥ β alors retourner v
    α ← MAX(α,v)
  retourner v

fonction VALEUR_MIN(etat,α,β)
  si TEST_TERMINAL(etat) alors retourner UTILITE(etat)
  v ← +∞
  pour a,s dans SUCCESSEURS (etat) faire
    v ← MIN(v,VALEUR_MAX(s,α,β)
    si v ≤ β alors retourner v
    β ← MIN(β,v)
  retourner v
```

Bilan alpha-bêta

- Efficacité dépend de l'ordre des successeurs
- Cout (temps) de l'ordre de $O(b^{3m/4})$
- Avec ordonnancement $O(b^{m/2})$
  $\rightarrow$ facteur de branchement effectif : $\sqrt{b}$!
- Amélioration possible : table de transposition
  - Souvent, plusieurs chemins mênent à un même état
    $\rightarrow$ le premier passage permet de faire l'élagage !
  - Mémorisation des états (les plus intéressants) dans une table de hachage

Des situations plus réalistes...

- Dans les jeux réels, les nœuds terminaux ne peuvent pas être atteints
- (Sauf lorsque l'on approche de la solution !)
- Il faut jouer le mieux possible en temps raisonnable
- Fonction d'évaluation
- Test d'arrêt

Fonction d'évaluation

- Fournit une estimation de l'utilité espérée pour une position donnée
- $\sim$ fonctions heuristiques
- De telles fonctions sont utilisées implicitement depuis très longtemps
  (Sur quelles bases ?)
- Caractéristiques
  - Les états terminaux doivent être ordonnés comme par la fonction d'utilité !
  - Temps de calcul maîtrisé
  - Pour les états non terminaux, l'estimation doit être satisfaisante

Fonction d'évaluation : principe

- Description de l'état : attributs (*features*)
- Première approche : classification
  - Etats $\rightarrow$ classes (ou catégories) $=$ indexation
  - Informations statistiques par classe (eg % de victoire, de défaite, de nul)
  - Valeur espérée $=$ somme pondérée (pondération $=$ score associé)
  - Risque : de nombreuses catégories
- Une approche plus réaliste
  - Score $=$ fonction linéaire pondérée des attributs
  - $EVAL(s) = w_1 f_1(s) + w_2 f_2(s) + \ldots + w_n f_n(s)) = \Sigma_{i=1}^{n} w_i f_i(s)$

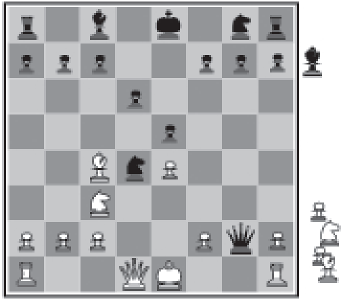
Fonction d'évaluation

- Dans cette dernière formule, on suppose l'indépendance des attributs
- Exemple aux échecs : w_i points associés à chaque pièce de type i
- Il est fréquent que la pondération (et plus généralement la fonction d'évaluation) dépende de l'avancement de la partie, ou qu'il y ait des interactions entre les attributs (exemple : coupe franche au tarot)
- Introduction de combinaisons non linéaires
- Comment pondérer ?
  - Expérience ?
  - Apprentissage ?

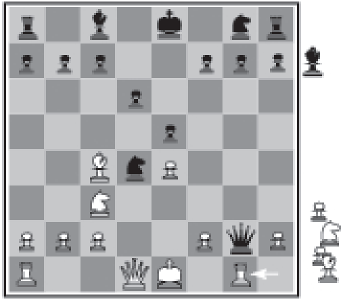
Test d'arrêt

- Reprise de l'approche alpha-bêta, en remplaçant le test d'état terminal par un test de cut off
- suppose de gérer la profondeur courante et de fixer la profondeur d de cut off
- Autre approche : exploration itérative (IDS)
  - Meilleure solution atteinte lorsque le temps est écoulé $\rightarrow$ plus robuste que l'estimation de d $\rightarrow$ risque de passer à côté d'un coup important

Exemple : échecs



(a) White to move

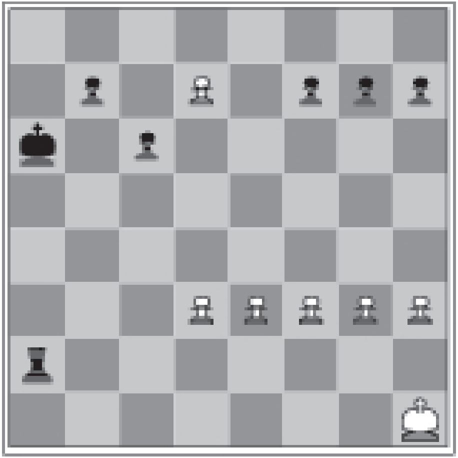


(b) White to move

Améliorations

- Recherche d'états stables (cut off)
  $\rightarrow$ éloignées de possibilités de renversement de valeur
- eg échecs : états ne débouchant pas sur une prise par l'adversaire
- Recherche de stabilité (*quiescence search*)
- Autre problème : effet d'horizon
  - Une action importante de l'adversaire est hors de portée de l'exploration
  - Solutions (partielles) :
    - Augmenter la profondeur (puissance brute)
    - Extensions singulières : n'approfondir que sous les coups clairement meilleurs (limite b)
    - de même, n'approfondir que les coups instables (au delà de la profondeur standard d'exploration).

Exemple : échecs



Black to move

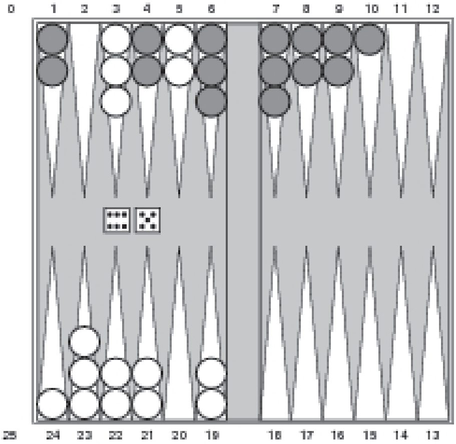
Elagage en avant

- Idée : choisir délibérément de ne pas explorer certains sous-arbres
- Avantage : gain de temps
- Inconvénient : risque d'erreurs grossières
- Un joueur humain n'explore, en général, qu'une partie des coups possibles
  $\rightarrow$ Pourquoi ? Sur quelle base ?
- Avec la machine : coups équivalents ou symétriques... (des choses plus subtiles ?)

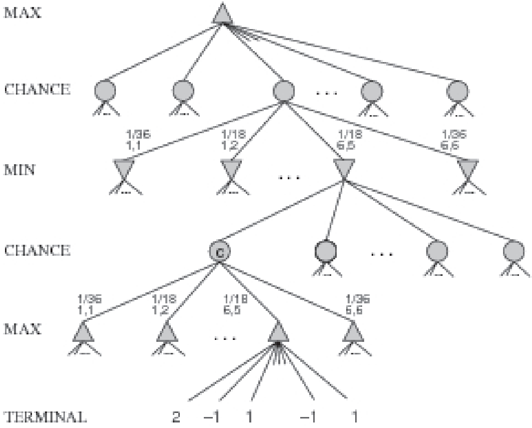
Ce qui change...

- Les possibilités de l'adversaire (et soi-même pour les coups futurs) ne sont pas déterministes
- Impossible de construire un arbre de type minimax !
- Il faut ajouter des niveaux intermédiaires : nœuds de hasard
    - Liste de tirages possibles
    - probabilité associée

Exemple : backgammon



Arbre associé



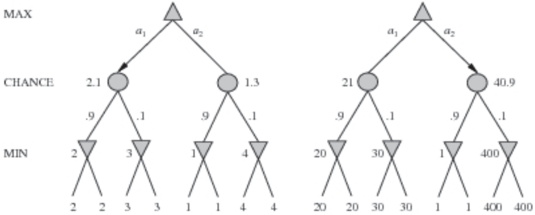
Hasard et prise de décision

- L'objectif est inchangé : aller vers la meilleure position
- Mais on ne dispose pas de valeurs minimax fixes...
- $\rightarrow$ valeur minimax *espérée* (expectiminimax)
- EXPECTIMINIMAX(n)=
  UTILITE(n) si n est un nœud terminal
  $max_{s \in Successeurs(n)}$EXPECTIMINIMAX(s) si n nœud MAX
  $min_{s \in Successeurs(n)}$EXPECTIMINIMAX(s) si n nœud MIN
  $\Sigma_{s \in Successeurs(n)}$P(s).EXPECTIMINIMAX(s) si n nœud HASARD
- Complexité ? $O(b^m n^m)$

Evaluation des positions

- Atteindre les feuilles est en général impossible (sauf fin de partie)
- Comme précédemment, recours à une fonction d'évaluation
- Du choix de la fonction d'évaluation...
  - L'échelle des valeurs peut avoir une influence sur le choix, et donc sur le risque pris
  - Fct éval = transformation linéaire positive de la probabilité de victoire

Importance de la fonction d'évaluation



Jeux de cartes

- Le joueur sait que tout est fixé dès le départ...
- Mais il ne sait pas comment !
- Diffère beaucoup des jeux de hasard
  - On ne peut pas, dès le début, fonder son raisonnement sur une espérance moyenne (sur la clairvoyance)
  - La connaissance du jeu évolue au cours de la partie
- Apparition d'un espace de croyance

Petit retour sur deep blue

- 30 processeur RS/6000 + 480 proc. VLSI dédiés
- en moyenne 126 millions de nœuds / sec
- jusqu'à 30 milliards de positions / coup, prof. 14
- exploration alpha-bêta itérative en profondeur
- table de transpositions
- extensions singulières : jusqu'à prof 40
- fct évaluation : plus de 8000 attributs
- bibliothèque d'ouvertures (env 4000 positions)
- BD de 700 000 partie de haut niveau
- BD de toutes les fins de parties résolues à cinq pièces + bcp à 6 pièces...

Petit retour sur deep blue

- Le “tour de force” matériel est un peu limite...
- Les progrès des heuristiques d'élagage rendent un simple PC très performant
- eg : coup nul (on laisse l'adversaire enchaîner deux coups)
- eg : élagage des absurdités

Les autres jeux

- Dames, Othello : niveau champions (Chinook, Logistello)
- Backgammon, Bridge : idem (TD-Gammon, Bridge Baron)
- Go : le niveau a beaucoup progressé récemment. (victoires sur jeux 9*9, moins souvent sur 19*19) Pb : profondeur d'exploration. Pistes : reconnaissance de formes, stratégies "locales"..

Aller plus loin...

- Minimax : évaluation et optimalité
  - S'il existe de forts écarts de valeur dans les évaluations des fils d'un noeud, faut-il en tenir compte ?
  - Utilisation de probabilités
  - coût ?
- Utilisation de méta raisonnement (intérêt d'une exploration) pour adapter à la volée l'algorithme d'exploration
- Utilisation d'un raisonnement orienté par les buts (priorités) + combinaison avec exploration avec fonction d'évaluation

Théorie des jeux

- Etude mathématique des situations impliquant plusieurs agents
- notion de gain / profit pour chaque agent
- Choix de stratégie
- Application à de nombreux problèmes réels (optimisation, négociation...)

Eléments

- Joueurs
- Actions
- Matrice de gains
- Exemple : pair / impair (E gagne la somme si paire, O sinon)

	O : un	O : deux
E :un	E=2 ; O=-2	E=-3 ; O=3
E :deux	E=-3 ; O=3	E=4 ; O=-4

Autre exemple : dilemne du prisonnier

- Une bande est arrêtée et interrogée après un cambriolage...
- Score = durée d'enfermement

	A. témoigne	A. refuse
B. témoigne	A=-5 ; B=-5	A=-10 ; B=0
B. refuse	A=0 ; B=-10	A=-1 ; B=-1

Vocabulaire

- Stratégie
  - Stratégie pure : un seul choix pour chaque configuration (déterministe)
  - Stratégie mixte : plusieurs choix + probabilité (eg : [p : a ; (1-p) : b])
- Solution : profil stratégique pour lequel chaque joueur adopte une stratégie rationnelle
  - rationnel ?

Concepts

- Stratégie dominante
  - Fortement dominante : s' domine fortement s (pour un joueur j) si son résultat est meilleur que celui de s' $\forall$ les stratégies adoptées par les autres joueurs.
  - Faiblement dominante : s meilleure dans au moins un cas et pas pire dans les autres.
- Stratégie Pareto - optimale
  - Pause culturelle : **Vilfredo Pareto (1848-1923)** Sociologue et économiste italien
  - Pause culturelle (bis) : **loi (empirique) de Pareto** 20% des causes sont à l'origine de 80% des effets
  - Stratégie P.O. : il n'existe aucun autre résultat que tous les joueurs préféreraient
  - Stratégie P. - dominé : tous les joueurs préfèrent un autre résultat

Concepts (2)

- Equilibre de stratégies
  - Aucun joueur ne peut adopter une meilleure stratégie (sachant que les autres n'en changent pas)
  - Equ. de stratégies dominantes : équilibre où chaque joueur a une stratégie dominante (i.e. "gagnant - gagnant !")
- Equilibre de Nash
  - Pause culturelle : **John F. Nash (1928)** Prix Nobel 1994 d'économie. Un (bon) film (assez romancé) lui est consacré ("Un homme d'exception").
  - Tout jeu possède un point d'équilibre (même en l'absence de stratégie dominante)
- Jeux de coordination : Les joueurs ont besoin de communiquer (pour maximiser leurs gains)
- Jeux à somme nulle : la somme des gain est nulle dans chaque case de la matrice.