Jeux & grand espace de recherche

M1 IDD 2019–2020 *Représentation des connaissances et raisonnement*

Stéphane Airiau

LAMSADE

DAUPHINE
UNIVERSITÉ PARIS

Quand on n'a pas le temps de visiter tout l'arbre

1 S'arrêter à une profondeur donnée
   - **`CutOff-Test`**`(s, profondeur)` va retourner vrai lorsqu'on atteint une certaine profondeur ou si l'état `s` est un état final
   - ⇨ il faut déterminer la profondeur seuil pour bien utiliser le temps.
   - lorsqu'on atteint cette profondeur, on utilise une fonction d'évaluation qui estime la qualité de l'état.

2 Autre solution : tant qu'il reste du temps, exécuter `Iterative-Deepening-Search` à la profondeur maximale courante, on utilise une fonction d'évaluation.

- Attention :
   - il ne faudrait pas s'arrêter à des états où l'évaluation a des risques de changer drastiquement
     ex : si on compte les pièces au échecs, évaluer juste avant une capture peut être trompeur.
   - l'effet d'horizon peut nous tromper (on peut imaginer une situation dans laquelle on est sûr de perdre, mais on peut repousser l'inévitable en faisant des actions qui ne vont rien changer)

Simulation de Monte Carlo

- Monte Carlo : nom générique utilisé lorsqu'on effectue des simulations pour obtenir des statistiques
- ici, on veut estimer la valeur des états

⇨ ensuite, il suffira de choisir le coup qui mène au meilleur état.

- *idée :* au lieu de visiter tout un sous arbre pour connaître la valeur d'un état
  on va simplement jouer "quelques parties" complètes
  ⇨ on fabrique des statistiques sur ce noeud

Comment choisir sa machine à sous ? UCB

- Vous êtes seul dans le casino, vous avez un bon nombres de jetons
- Toutes les machines ne sont pas identiques et certaines donnent de meilleurs gains.
- comment choisir la machine à sous ?
- explorer les machines, faire des statistiques
- exploiter ce qu'on a appris pour gagner un maximum

Ce problème est connu sous le nom de "bandits".

- application pour tester des traitements
- choisir une publicité pour un utilisateur en ligne (bandit contextuel, on peut chercher en plus des utilisateurs similaires pour aider la décision

⇨ cours d'apprentissage par renforcement (M2 IASD)

Une technique : constuire un intervalle de confiance des gains.

Utiliser un intervalle de confiance

- Q_a estime la valeur moyenne de chaque machine.
- idée : essayer de mesurer l'incertitude sur la valeur Q(a) qu'on estime
- but : on calcule une borne *supérieure* probable de la valeur Q(a) qu'on estime
- La borne *supérieure* est calculée par l'expression

$$Q_t(a) + c\sqrt{\frac{\log t}{N_t(a)}}$$

- plus la borne est élevée, plus on considère a comme prometteuse !

On va donc choisir l'action $\operatorname{argmax}_a \left[Q_t(a) + c\sqrt{\frac{\log t}{N_t(a)}} \right]$

⇨ On explore les actions les plus prometteuses
⇨ meilleure utilisation de nos jetons !

Utiliser un intervalle de confiance

a est prometteuse quand

- je ne connais pas bien a
- je connais a et j'ai eu de bons résultats avec

$$Q_t(a) + c\sqrt{\frac{\log t}{N_t(a)}}$$

- $N_t(a)$ est le nombre de fois où l'action a a été choisie
- c est une constante qui determine la mesure de confiance
- t est le nombre total d'itérations

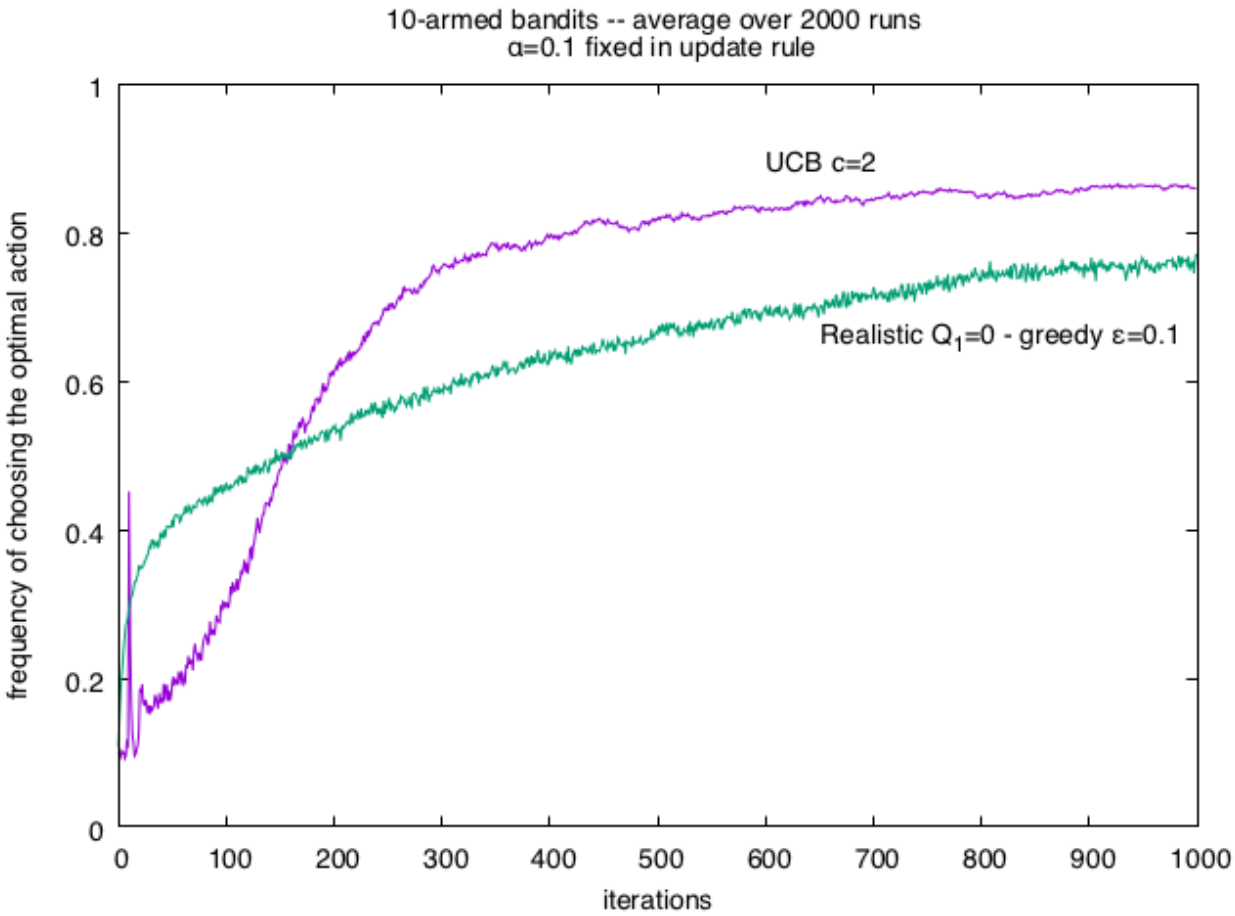
⇨ si a est choisie ⇨ l'incertitude baisse (donc la borne aussi)

⇨ si a n'est pas choisie, t croît, et l'incertitude croît.

- avec le log, l'accroissement devient de plus en plus petit.

Peter Auer, Nicolò Cesa-Bianchi, Paul Fischer. (2002). Finite-time Analysis of the Multiarmed Bandit Problem, *Machine Learning*, 47, 235—256

Evaluation empirique



fréquence du choix de la meilleure machine

Comment choisir sa machine à sous ? UCB

$$\left[\bar{x_i} - \sqrt{\frac{2ln\,n}{n_i}},\ \bar{x_i} + \sqrt{\frac{2ln\,n}{n_i}}\right]$$

- $\bar{x_i}$: moyenne des gains de la machine à sous i
- n_i nombre de fois où on a utilisé la machine i
- n le nombre total de fois où on a joué

Choisir la machine i avec la meilleure borne.

- pour toutes les autres machines, l'intervalle augmente
- pour la machine i, $\bar{x_i}$ change et la taille de l'intervalle diminue

On va utiliser cette technique pour la recherche !

UCT

selection On utilise UCB pour choisir les coups

expansion Lorsqu'on arrive à un noeud avec des satistiques incomplètes

- on choisit un noeud suivant au hasard

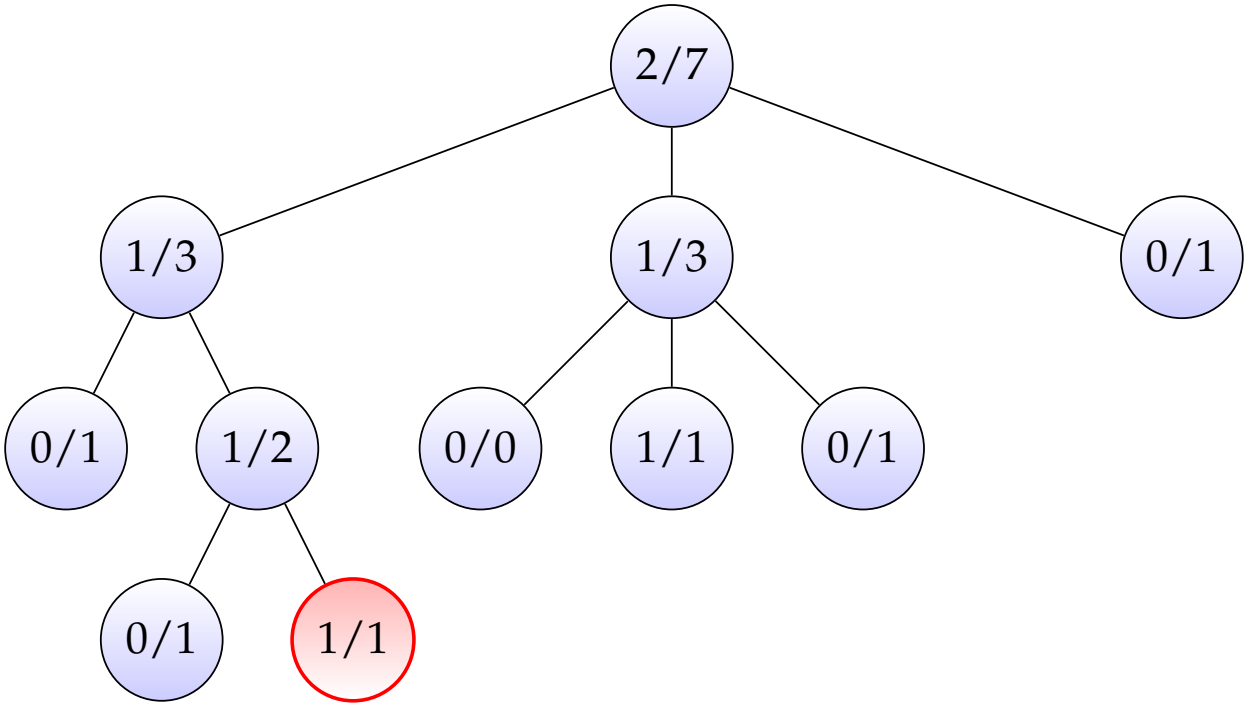
simulation On effectue une "partie complète" au hasard

on descend au hasard jusqu'à un état final

propagation on propage les résultats ⇨ on met à jour les statistiques pour les états parents

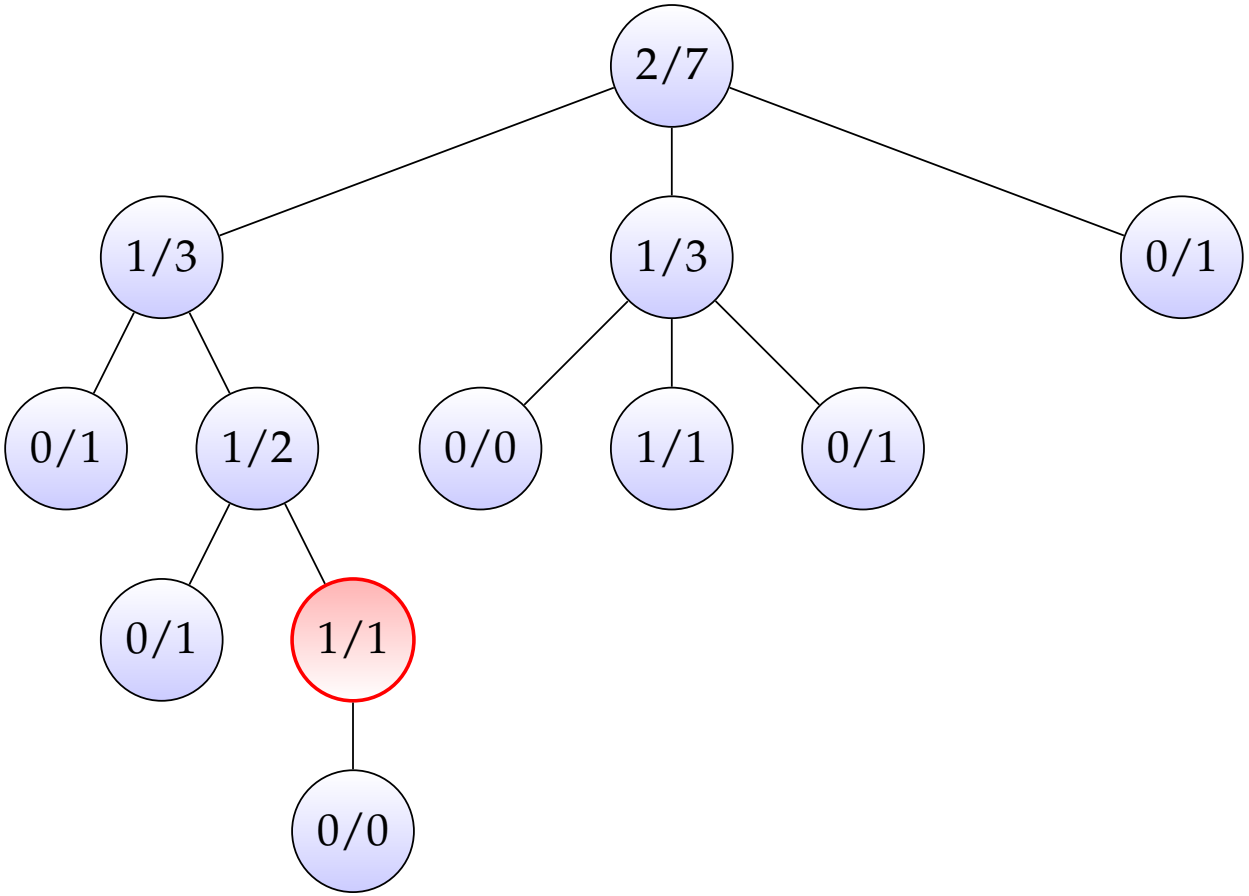
- On construit peu à peu l'arbre de recherche.
- On joue de façon guidée en conservant un bon équilibre exploration / exploitation

UCT : exemple



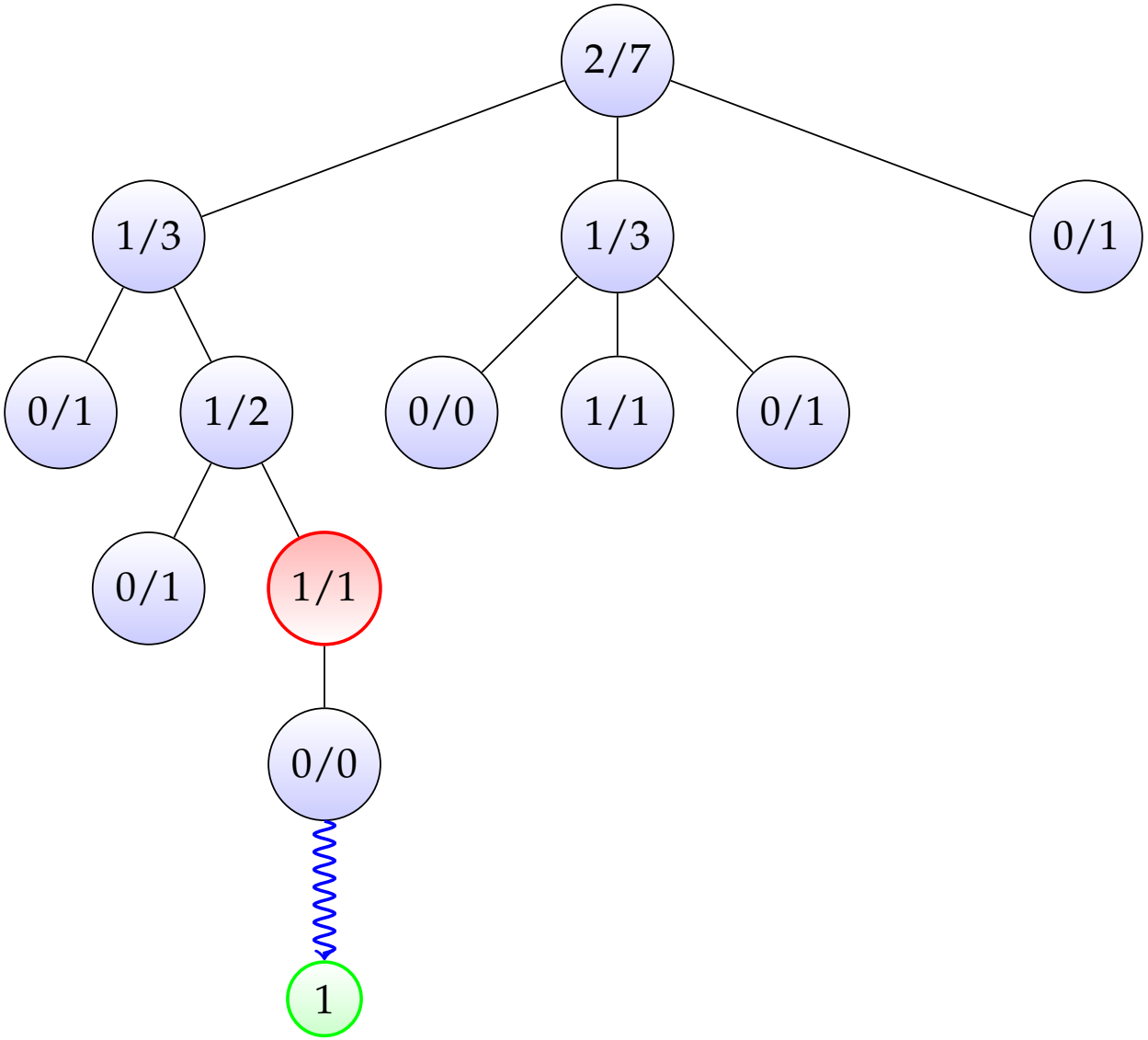
Selection : On choisit les actions avec UCB, dans l'exemple, cela nous mène au noeud rouge.
Le noeud en rouge n'a pas de fils avec des statistiques

UCT : exemple



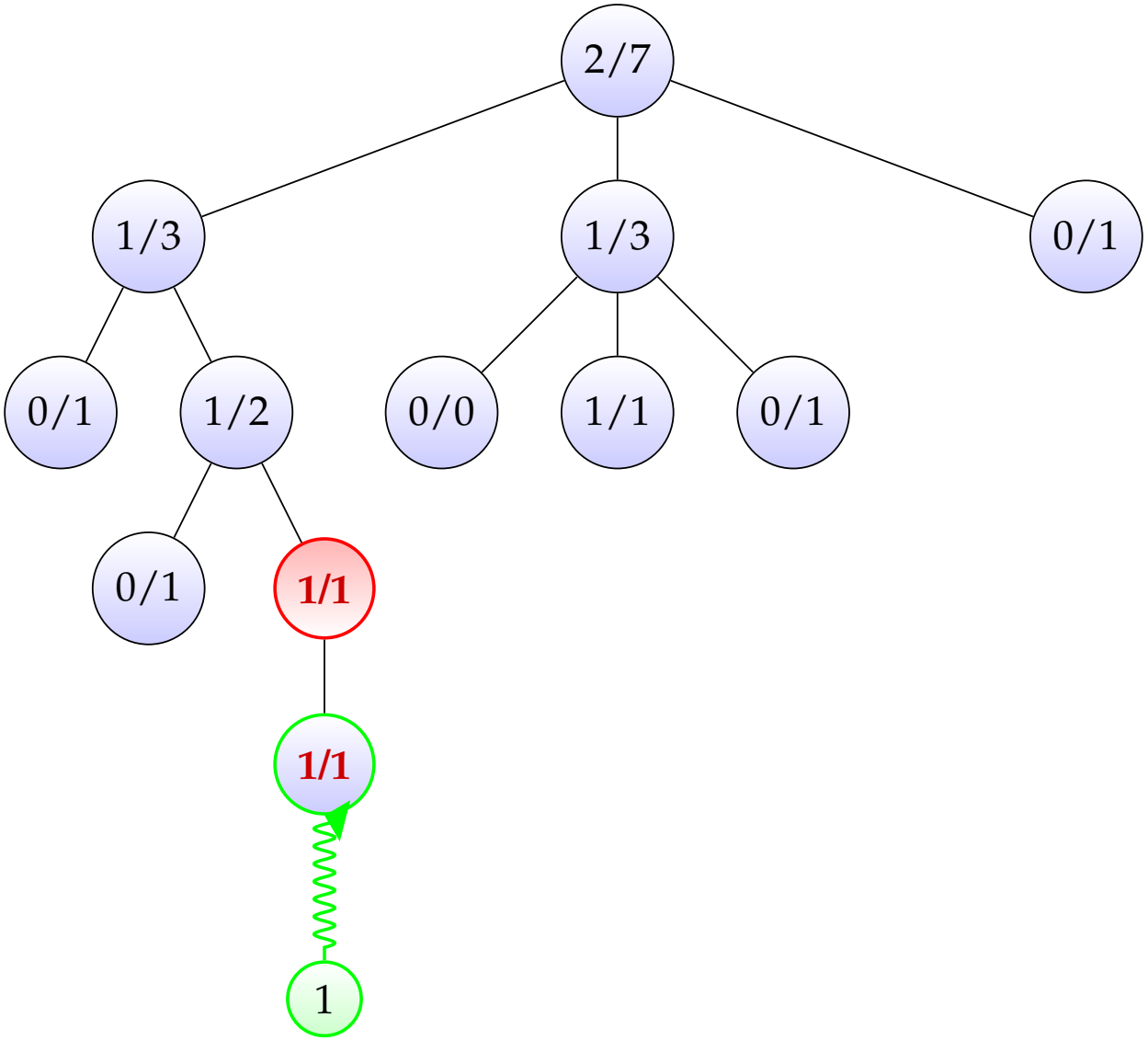
Expansion : on choisit un fils au hasard

UCT : exemple



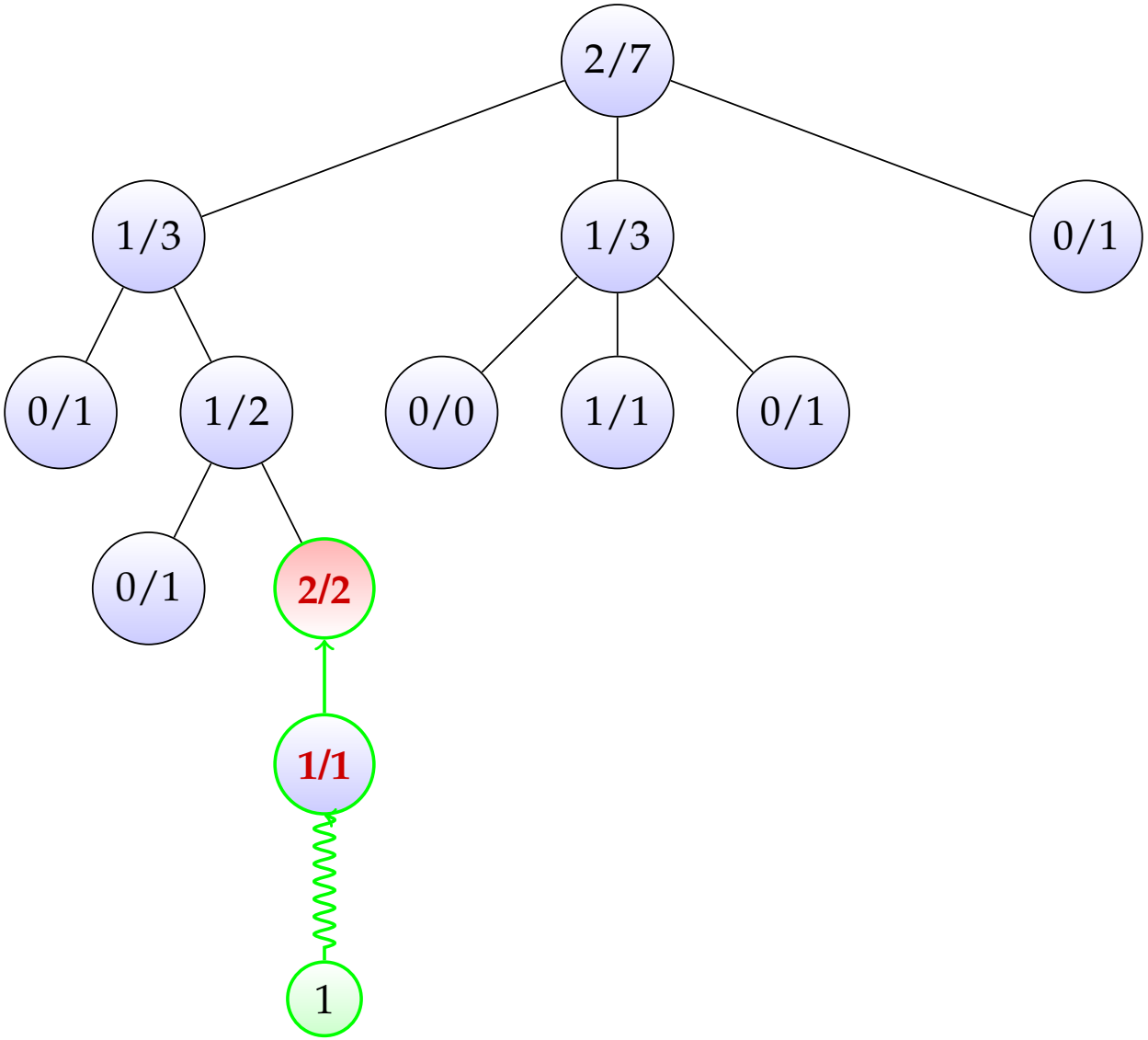
Simulation : on complète une partie au hasard, dans l'exemple on gagne

UCT : exemple



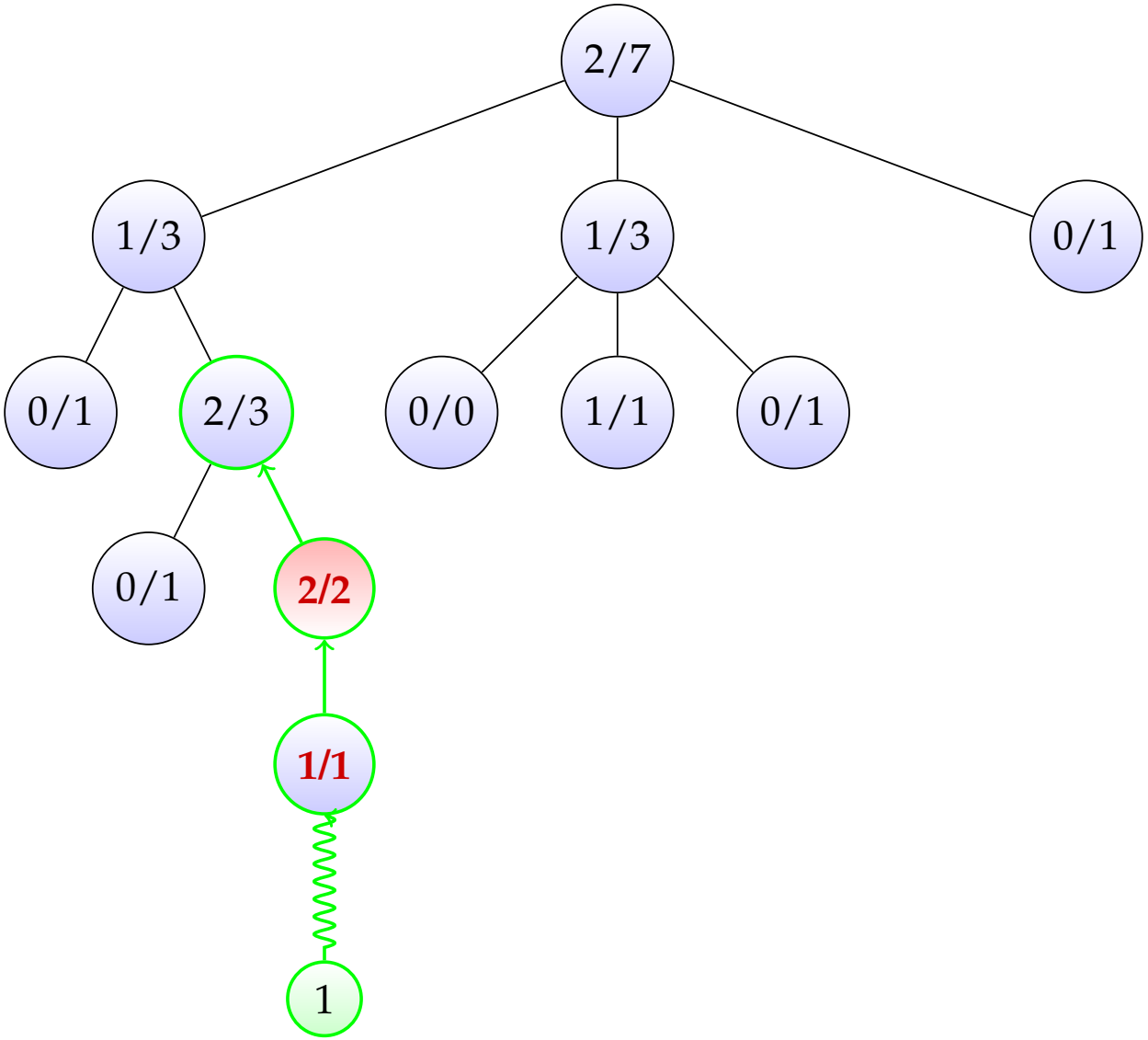
Retropropagation : on met à jour les statistiques

UCT : exemple



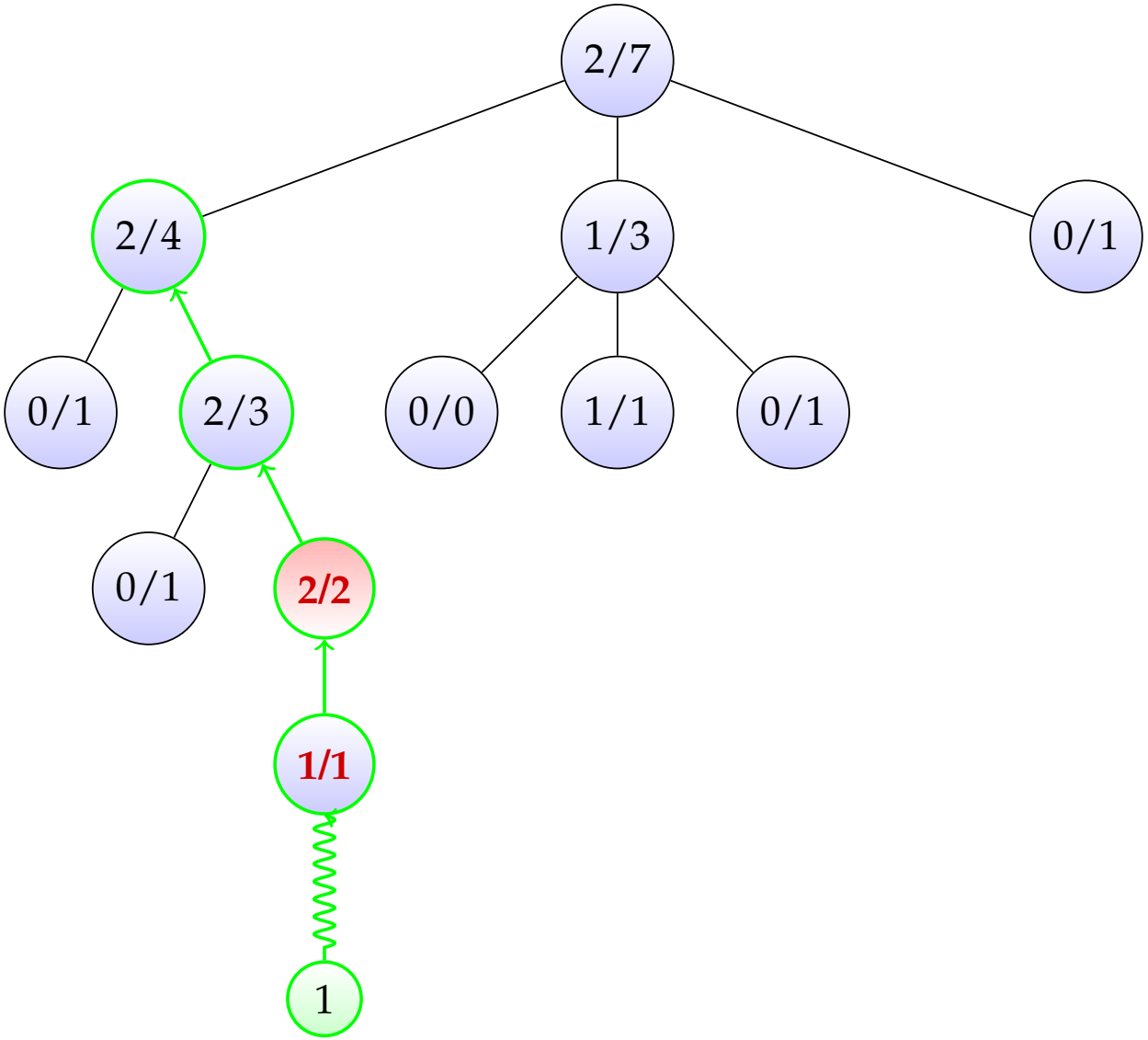
Retropropagation : on met à jour les statistiques

UCT : exemple



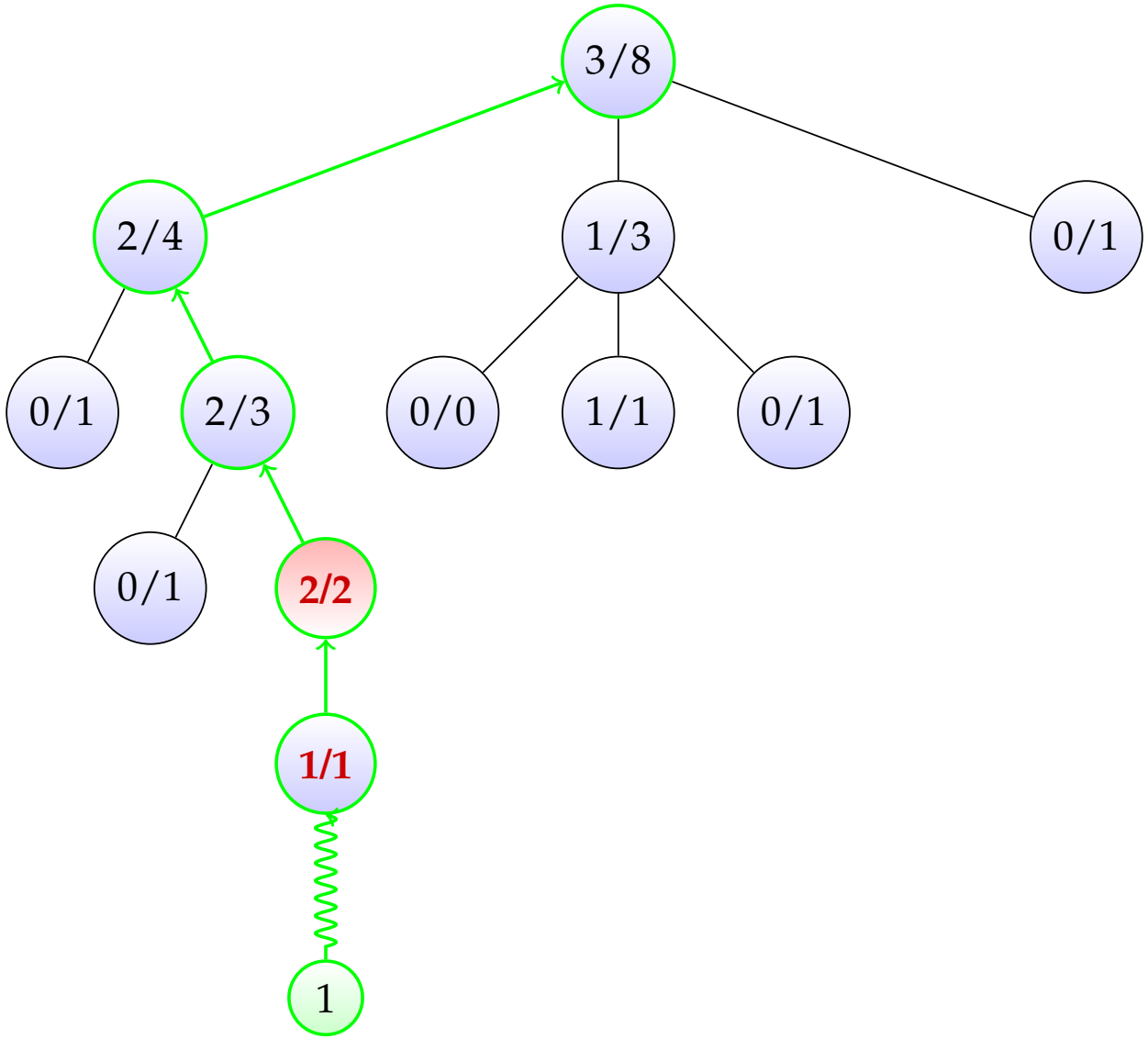
Retropropagation : on met à jour les statistiques

UCT : exemple



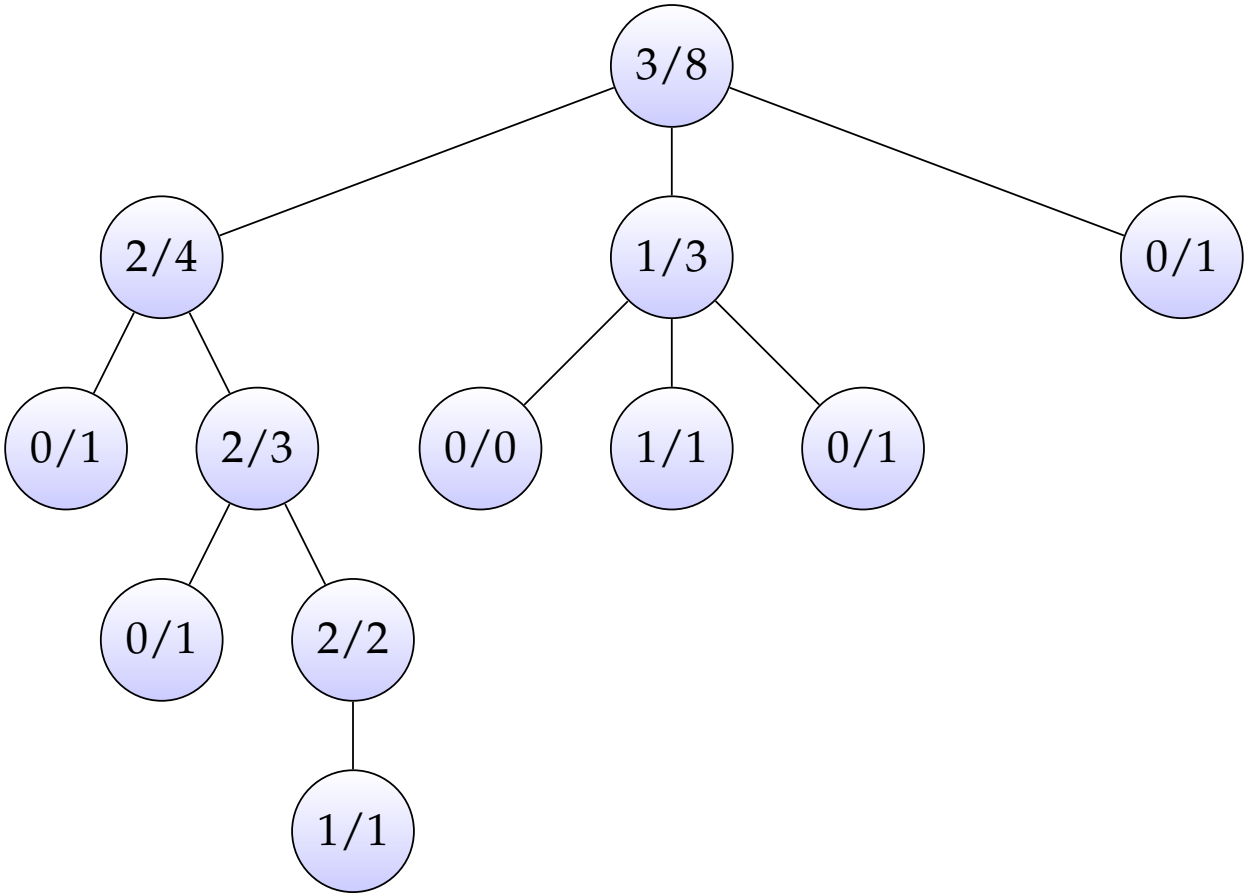
Retropropagation : on met à jour les statistiques

UCT : exemple



Retropropagation : on met à jour les statistiques

UCT : exemple



Fini, on peut recommencer ![a]

a. Le point de départ de l'exemple n'est peut être pas un état atteignable avec UCT

Monte Carlo Tree Search

- UCT (Upper Confidence Tree) est une variante d'une famille plus large appelée Monte Carlo Tree Search.
- il existe d'autres variantes pour gérer l'exploration et l'exploitation

⇨ ces méthodes sont à l'origine d'un saut de performance pour le jeu de go dans les années 2006–2008
avant, des variantes de α-β étaient utilisées

jouer les parties aléatoire n'est peut être pas très informatif !

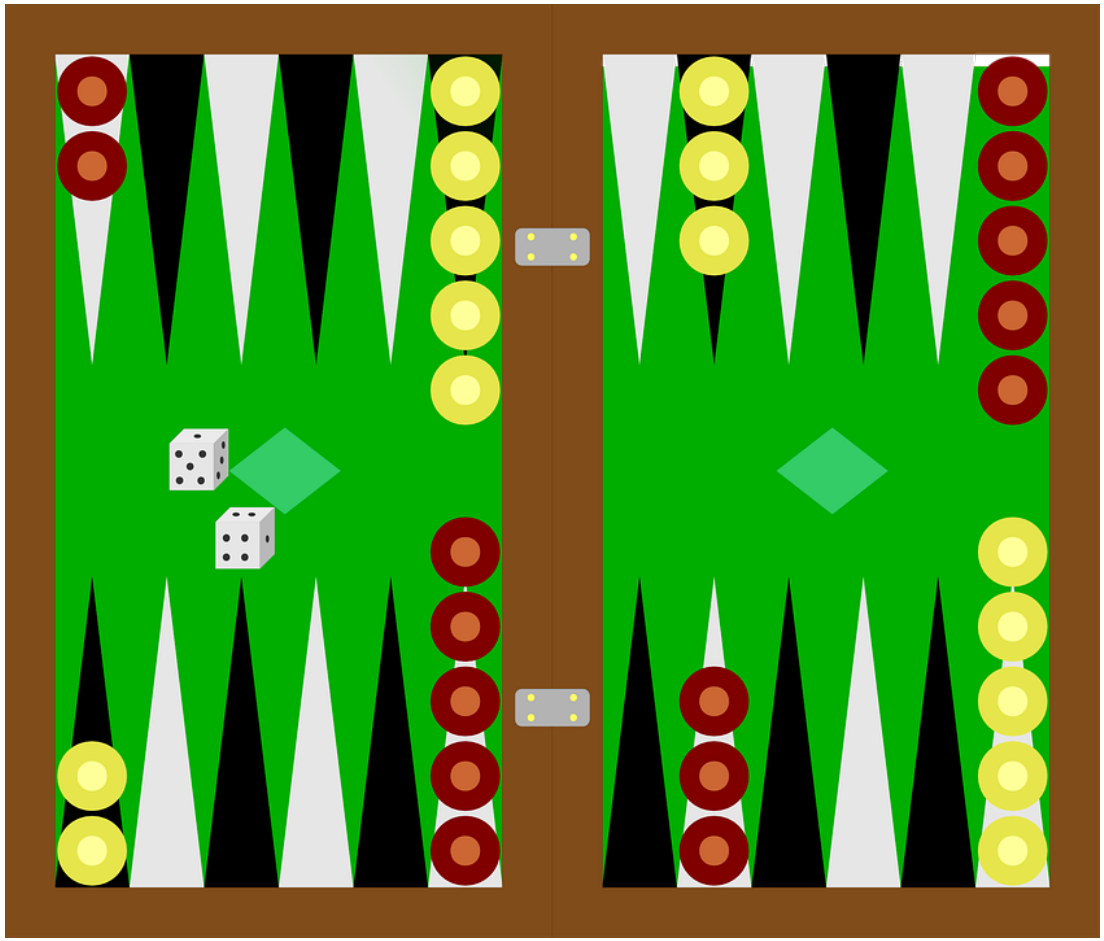
On verra à la fin du cours comment fonctionne alphago, le programme champion du monde de DeepMind.


Kocsis, Levente and Szepesvári, Csaba. Bandit based monte-carlo planning. In *Machine Learning : ECML 2006* pp. 282–293. Springer, 2006.


Silver et al. Mastering the game of go with deep neural networks and tree search. *Nature*, 2016.

Jeux avec un élément de chance

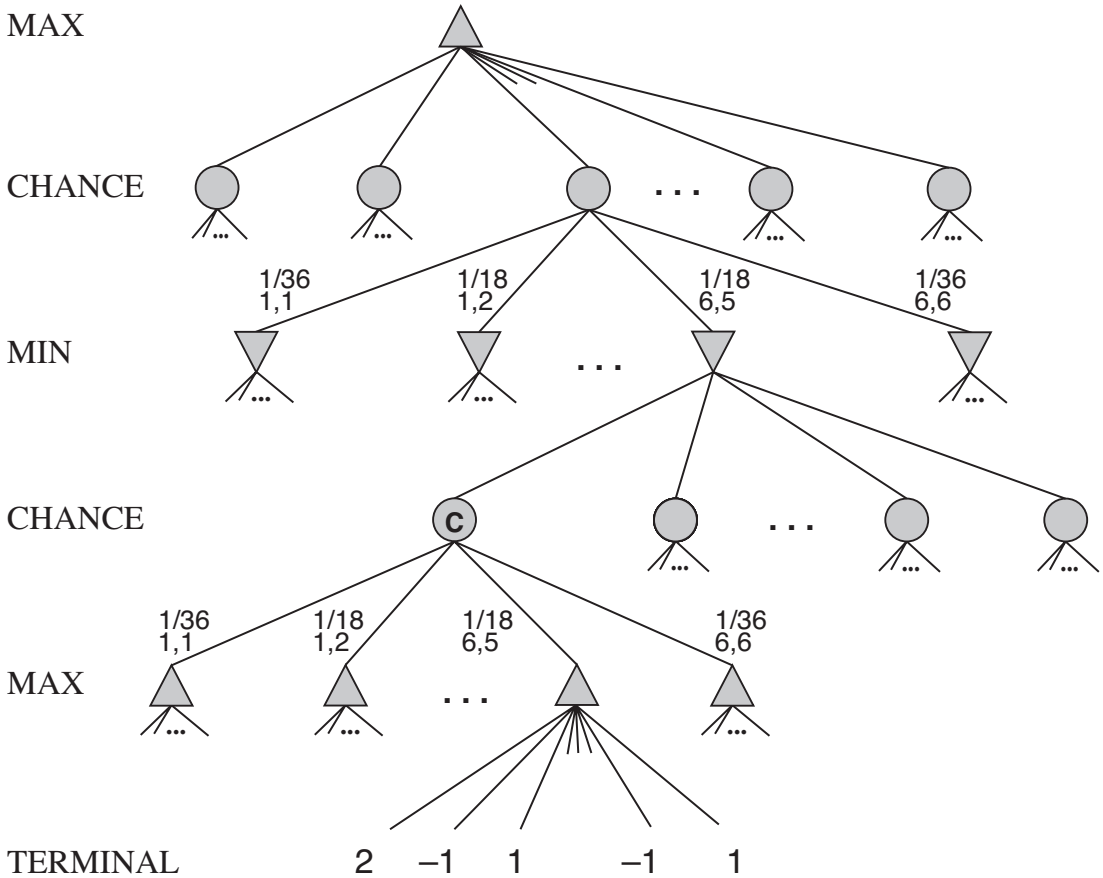
Exemple : backgammon



- Un joueur déplace ses jetons dans le sens trigonométrique, l'autre dans le sens contraire.
- on lance deux dés, on peut déplacer le jeton soit dans un espace qui contient un jeton adverse (on le mange), vide, des jetons alliés

Noeuds de chance

Il faut donc intégrer des noeuds de chance à un arbre de jeu.



Minimax en valeurs espérées

EXPECTEDMINIMAX(s) =

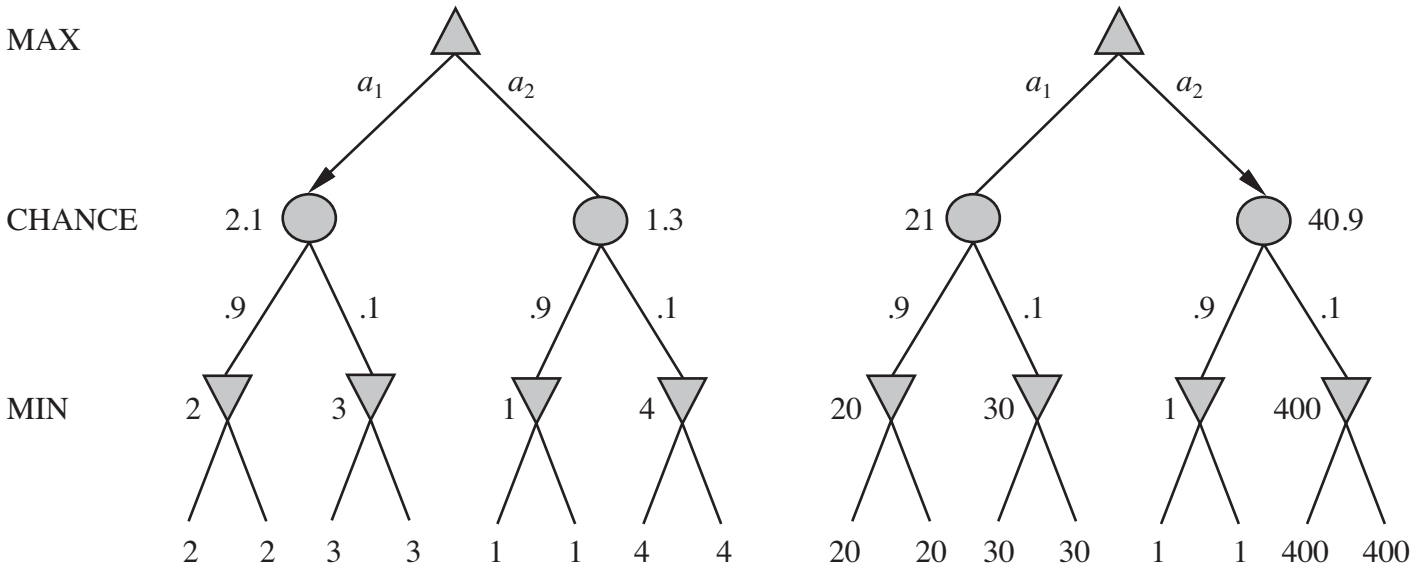
utilité(s) si s est un état final

$\max_{a \in actions(s)}$ EXPECTEDMINIMAX (resultat(s,a)) si MAX joue

$\min_{a \in actions(s)}$ EXPECTEDMINIMAX (resultat(s,a)) si MIN joue

$\sum_{r} \mathbb{P}(r)$ EXPECTEDMINIMAX(resultat(s,r)) si MIN joue

Fonction d'évaluation : attention



L'échelle pour les valeurs joue un rôle important pour la stratégie !

Complexité

- b est le facteur de branchement pour les actions des joueurs
- m la profondeur maximale de l'arbre de jeu
- n le nombre de possibilités différentes pour les noeuds de chance (un dé : 6 possibilités, deux dés 21 possibilités)

EXPECTEDMINIMAX s'exécute en $\mathcal{O}(b^m n^m)$.

Pour backgammon, regarder à plus de trois coups sera déjà pas mal !

Autres types de jeux

On ne les abordera pas en cours, mais ils ont fait l'objet de recherche en IA (pour beaucoup, y compris le poker, la machine joue maintenant mieux que l'humain).

- jeux où toute l'information n'est pas observable (*observation imparfaite*)
  exemple : jeux de cartes (on ne connait pas les cartes de l'adversaire)
  autre exemple : bataille navale
- jeux avec un élément de chance et avec information imparfaite
  poker, scrabble, bridge, guerre nucléaire

Le jeu de poker a été "résolu" ces dernières années

- DeepStack bat des joueurs professionnels au "heads-up no-limit Texas hold'em" (décembre 2016)
- Libratus écrase 4 joueurs professionnels au "heads-up, no-limit Texas Hold'em" (décembre 2017) 20 jours, 120,000 mains, 200 000$ à se partager !
- Pluribus (Facebook's AI lab and Carnegie Mellon University) "six-person no-limit Texas Hold 'em poker" (juillet 2019)