Appentissage actif avec PDA

function ~~PASSIVE~~ ACTIVE-ADP-AGENT(*percept*) returns an action

inputs: *percept*, a percept indicating the current state s' and reward signal r'

persistent: π, a fixed policy
mdp, an MDP with model P, rewards R, discount γ
U, a table of utilities, initially empty
N_{sa}, a table of frequencies for state–action pairs, initially zero
$N_{s'|sa}$, a table of outcome frequencies given state–action pairs, initially zero
s, a, the previous state and action, initially null

if s' is new **then** $U[s'] \leftarrow r'$; $R[s'] \leftarrow r'$
if s is not null **then**
  increment $N_{sa}[s,a]$ and $N_{s'|sa}[s',s,a]$
  for each t such that $N_{s'|sa}[t,s,a]$ is nonzero **do**
    $P(t|s,a) \leftarrow N_{s'|sa}[t,s,a] \,/\, N_{sa}[s,a]$
$U \leftarrow$ POLICY-EVALUATION(π, U, mdp)
if s'.TERMINAL? **then** $s, a \leftarrow$ **null else** $s, a \leftarrow s', \pi[s']$
return a

Value iteration

$$V(s) = R(s) + \max_a \gamma \sum_{s' \in S} P(s'|s,a)\, V(s')$$

$$\text{argmax}_{a \in A(s)} \sum_{s \in S} P(s|s',a)\, V(s)$$

Dilemme exploration vs. exploitation

- L'approche précédente est dite **vorace (gloutonne)**
  - elle met à jour le plan suivi par celui qui est optimal **maintenant**
  - en d'autres mots, **elle exploite le plus possible** l'information recueilli jusqu'à maintenant

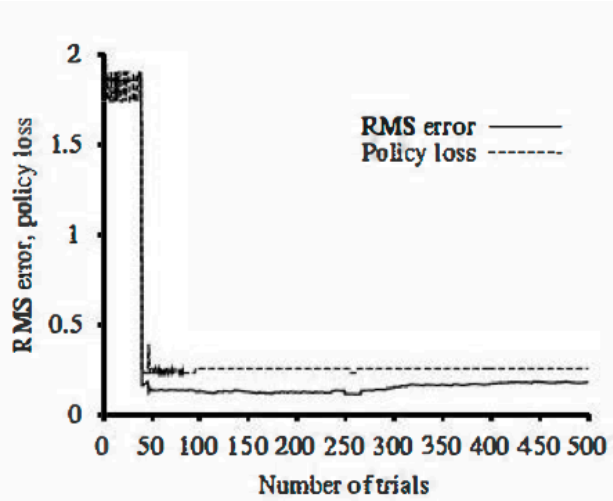
- Les approches voraces trouvent rarement le plan optimal
  - elles ne tiennent pas compte du fait que l'**information accumulée jusqu'à maintenant est partielle**
  - en d'autres mots, elles ne considèrent pas la possibilité d'**explorer l'environnement** plus longuement, pour amasser plus d'information sur celui-ci

- Un parallèle similaire existe entre le *hill-climbing* et le *simulated annealing* en recherche locale

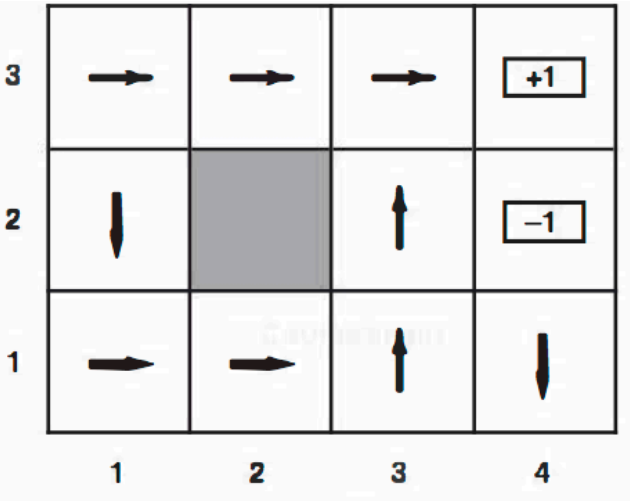
Dilemme exploration vs. exploitation

- Exemple: cas où l'action « ↑ » n'a jamais été exécutée à l'état (1,2)
- L'agent ne sait pas que ça mène à (1,3), qui mène à un chemin plus court!

Courbe d'apprentissage



Plan découvert



Dilemme exploration vs. exploitation

- **Trop exploiter** mène à des plans non optimaux
- **Trop explorer** ralentit l'apprentissage inutilement
- Trouver la balance optimale entre l'exploration et l'exploitation est un problème ouvert en général
- Des stratégies d'exploration/exploitation optimales existent seulement dans des cas très simples
  - voir le cas du *n-armed bandit* dans le livre, p. 841

Dilemme exploration vs. exploitation

- On se contente donc d'heuristiques en pratique

- Exemple: introduction d'une **fonction d'exploration** $f(u,n)$
  - cette fonction augmente artificiellement les récompenses futures d'actions inexplorées
- L'approche par PDA basée sur *value iteration* ferait les mises à jour

$$V'(s) = R(s) + \max \gamma\, f\Big(\Sigma_{s' \in S} P(s'|s,a)\ \ V(s'), N(s,a) \Big)$$

où $N(s,a)$ est le nombre de fois que l'action a a été choisie à l'état s
et

$$f(u,n) = \begin{cases} R^+ & \text{si } n < N_e \\ u & \text{sinon} \end{cases}$$

estimation optimiste de récompense future (hyper-paramètre)

- Garantit que a sera choisie dans s au moins N_e fois durant l'apprentissage