Perception vs intelligence

Certains considèrent que l'apprentissage par ordinateur n'a rien à voir avec l'intelligence.

Mais quelque soit la définition qu'on donne à l'intelligence artificielle, peu de personnes se risquent à argumenter que jouer aux échecs/go n'est pas associé à une forme d'intelligence !

Perception vs intelligence

Or, aujourd'hui, on traite de plus en plus les problèmes types échecs/go avec des méthodes proches de l'apprentissage par ordinateur.

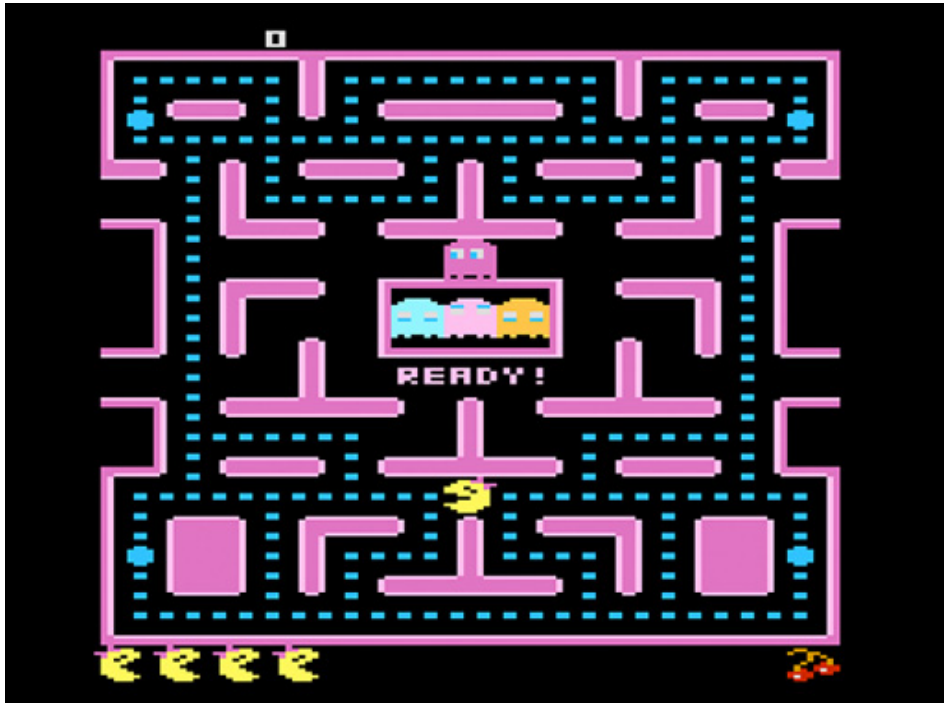
Reinforcement learning

Agent

observe son environnement
peut effectuer une action
l'action modifie l'environnement

Reinforcement learning

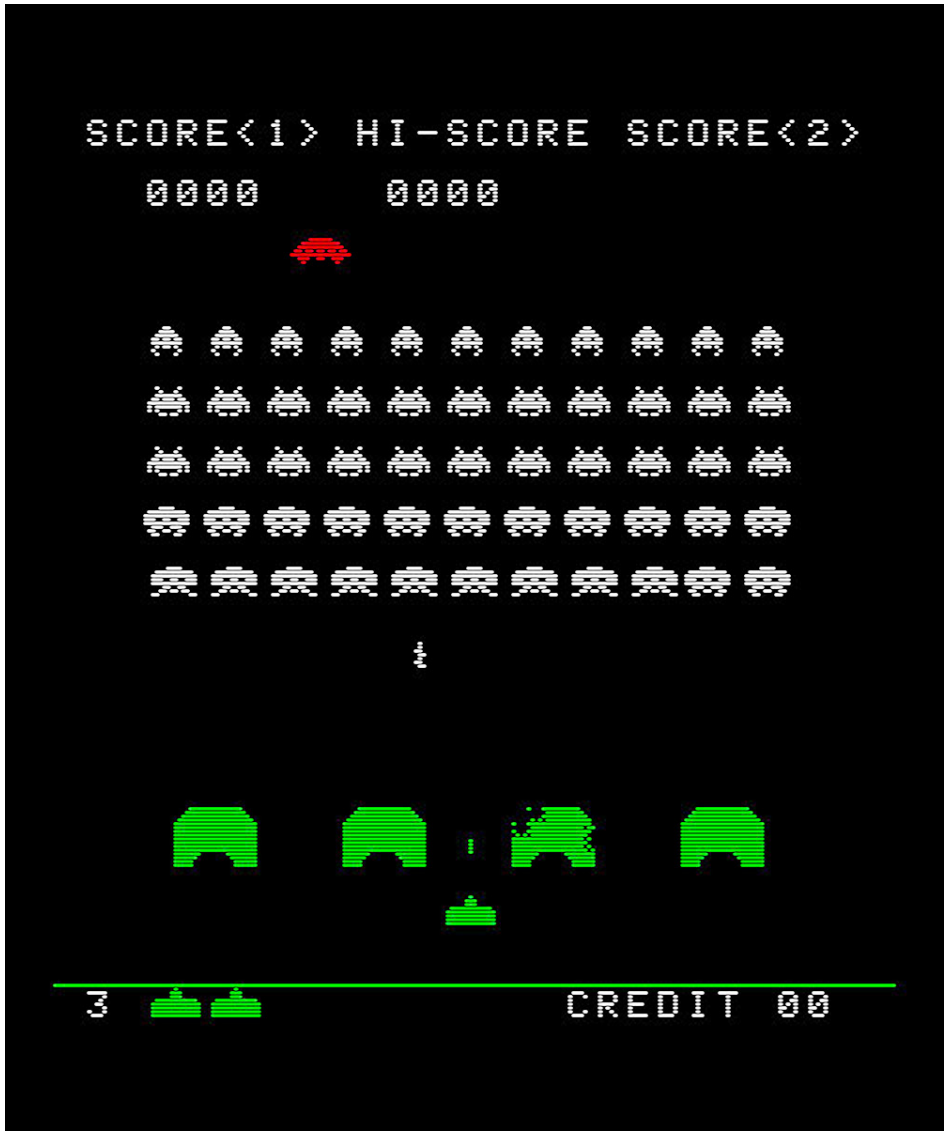
Exemple Atari



on voit ce qu'il se passe à l'écran
on peut appuyer sur A, B, haut, bas, droite, gauche ou ne rien faire
le jeu se déroule

Reinforcement learning

état de l'art



Reinforcement learning

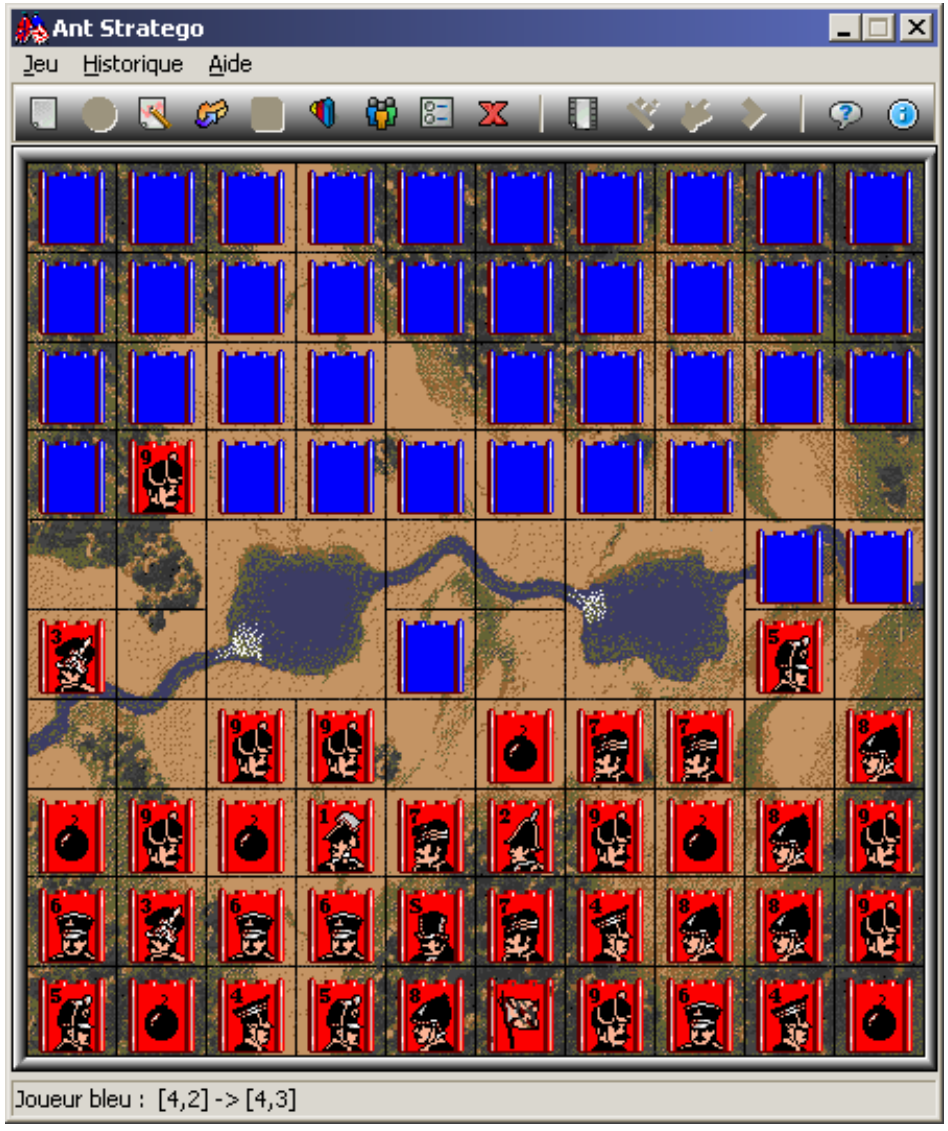
état de l'art

Reinforcement learning

état de l'art

Reinforcement learning

état de l'art



Reinforcement learning

Le but de l'apprentissage est de calculer une politique dans le but de récolter un score maximal.
Une fois calculée, cette politique est utilisée pour interagir avec l'environnement.

s_t l'état à l'instant t
a_t l'action à l'instant t
une politique c'est par exemple $Q(s_t, a_t)$ qui estime $Q^*(s_t, a_t)$ le score total qu'on peut obtenir en commençant par faire a_t depuis s_t

Reinforcement learning

$Q^*(s_t, a_t)$ le score total qu'on peut obtenir en commençant par faire a_t depuis s_t
c'est le score élémentaire reçu par le mouvement s_t vers s_{t+1} qu'on note $r(s_t, a_t, s_{t+1})$
plus le score total qu'on peut obtenir depuis s_{t+1}

Équation fondammentale de Q^*

$$Q^*(s_t, a_t) = r(s_t, a_t, s_{t+1}) + \gamma \max_a Q^*(s_{t+1}, a)$$

Reinforcement learning

ex : le plus court chemin vu comme du renforcement

la plus petite distance entre A et C est plus petite
que la distance élémentaire entre A et B
plus la plus petite distance entre B et C

$$D(A,C) = \min_{B \text{ voisin de A}} d(A,B) + D(B,C)$$

Reinforcement learning

Équation fondammentale de Q^*

$$Q^*(s_t, a_t) = r(s_t, a_t, s_{t+1}) + \gamma \max_a Q^*(s_{t+1}, a)$$

Cas fini

résoudre ce programme linéaire

$$\min_Q \sum_{s,a} Q(s, a)$$

$$\forall s, a, s', a', \quad Q(s, a) \geq r(s, a, s') + \gamma Q(s', a')$$

permet de trouver Q^*

Reinforcement learning cas infini

impossible d'utiliser

$$\min_Q \sum_{s,a} Q(s,a)$$

$$\forall s, a, s', a', \quad Q(s,a) \geq r(s,a,s') + \gamma Q(s',a')$$

Rightarrow

- choisir une famille $Q(s,a,\theta)$ paramétré par θ
- explorer pour trouver s, a, s', r
- sur des packets de s, a, s', r, optimiser θ de sorte que

$$Q(s,a,\theta) \approx r(s_t, a_t, s_{t+1}) + \gamma \max_{a'} Q(s', a', \theta)$$

Deep reinforcement learning

choisir un réseau de neurone $Q(s, a, \theta)$
optimiser θ par SGD avec l'objectif

$$(r + \gamma \max_{a'} Q(s', a', \theta) - Q(s, a, \theta))^2$$

tout en explorant pour trouver des quadruplets s, a, s', r

Deep reinforcement learning

en pratique **seul** deepmind est capable d'obtenir des modèles performants à Starcraft ou Stratego ! ! !