Projet Foot
Cours 7
2I013

Nicolas Baskiotis
nicolas.baskiotis@lip6.fr

Université Pierre et Marie Curie (UPMC)
Laboratoire d'Informatique de Paris 6 (LIP6)

S2 (2015-2016)

Plan

Tournoi 1v1

```
lob (nihaakey) : 40 (13,1,4) - (88,46)
psg (luluperet) : 38 (12,2,4) - (61,16)
DZPOWER (Raouf16) : 35 (11,2,5) - (52,39)
lalya1 (hmdd) : 27 (8,3,7) - (56,48)
team1 (ad50144124) : 25 (8,1,9) - (37,45)
Warrior (Asparodia) : 24 (7,3,8) - (38,75)
Diego Costa (jordanupmc) : 23 (7,2,9) - (99,55)
JSK (lounisAmazigh) : 19 (6,1,11) - (24,62)
team2 (Kabegami) : 16 (4,4,10) - (25,45)
team1 (3408247) : 13 (4,1,13) - (15,64)
```

Tournoi 2v2

```
DZPOWER (Raouf16) : 41 (12,5,1) - (49,5)
psg (luluperet) : 41 (12,5,1) - (50,15)
lob (nihaakey) : 36 (11,3,4) - (55,18)
Tremblez! (Asparodia) : 32 (9,5,4) - (48,24)
JSK (lounisAmazigh) : 23 (7,2,9) - (49,45)
lalya2 (hmdd) : 22 (6,4,8) - (39,27)
team1 (Kabegami) : 19 (5,4,9) - (31,32)
team2 (3408247) : 17 (2,11,5) - (21,6)
equipe2 (jordanupmc) : 17 (4,5,9) - (52,42)
team1 (ad50144124) : 0 (0,0,18) - (0,180)
```

Tournoi 4v4

```
DZPOWER (Raouf16) : 19 (6,1,1) - (41,9)
lalya4 (hmdd) : 18 (6,0,2) - (32,10)
team4 (3408247) : 16 (5,1,2) - (25,4)
lob (nihaakey) : 16 (5,1,2) - (20,11)
JSK (lounisAmazigh) : 11 (3,2,3) - (33,13)
Lel (Asparodia) : 11 (3,2,3) - (18,16)
team4 (Kabegami) : 7 (2,1,5) - (19,17)
team1 (ad50144124) : 6 (2,0,6) - (20,29)
mars (luluperet) : 0 (0,0,8) - (0,99)
```

Plan

Formalisation

Modélisation

- L'environnement : un ensemble d'observations $\mathcal{O} = \{O_1, \ldots, O_o\}$
- Les états : $\mathcal{S} = \{s_1, \ldots, s_s\}$ en nombre fini ou infini, dénombrable ou non;
- Les actions : $\mathcal{A} = \{a_1, \ldots, a_a\}$ idem;
- Les récompenses dans $\mathbb{R}$.

Une séquence de jeu $\mathcal{T}$

- c'est une séquence de triplets (observation,action,récompense) : $\{(o_{t_1}, a_{t_1}, r_{t_1}), (o_{t_2}, a_{t_2}, r_{t_2}), \ldots, (o_{t_T}, a_{t_T}, r_{t_T})\}$

$\Rightarrow$ équivalent au triplet (état,action,récompense) : $\{(s_{t_1}, a_{t_1}, r_{t_1}), (s_{t_2}, a_{t_2}, r_{t_2}), \ldots, (s_{t_T}, a_{t_T}, r_{t_T})\}$

- La récompense : $R = r_{t_1} + r_{t_2} + \ldots r_{t_T}$
- A noter : $s_i = \phi(o_i)$ l'état "vu" par l'agent est une transformation de l'observation.

Apprentissage par renforcement

Principe

- apprendre une *politique* pour réagir à l'environnement

$\Rightarrow$ une fonction $\pi(s)$ renvoyant pour chaque état une action ou une distribution de probabilité sur les actions

- Objectif : trouver la politique qui maximise l'espérance des récompenses

Notion fondamentale : fonctions (équivalentes)

- de valeur d'état
  $v_\pi(s) = \mathbb{E}_\pi(r_{t+1} + \gamma r_{t+2} + \gamma^3 r_{t+3} + \ldots + \gamma^T r_T | s_t = s)$ prédit le score d'un état
- de valeur d'action $Q_\pi(s,a) = r_t + \gamma \sum_{s'} P_\pi(s'|s) v_\pi(s)$ prédit le score d'une action entreprise dans un état donné.

Pour évaluer ses fonctions, besoin de simulations (beaucoup) !

Modèle Processus de décision de Markov (MDP)

Modèle d'automate probabiliste

Le jeu du point de vue de l'agent peut être modélisé par un MDP : un quadruplet $(\mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R})$

- un ensemble d'états $\mathcal{S}$, un ensemble d'actions $\mathcal{A}$ et de récompenses $\mathcal{R} : \mathcal{A} \times \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$.
- une fonction de transition : $\mathcal{P} : \mathcal{S} \times \mathcal{A} \times \mathcal{S} \rightarrow [0, 1],\; P(s, a, s') = Pr[s_{t+1} = s' | s_t = s, a_t = a]$

Une politique

$\pi : \mathcal{S} \rightarrow \mathcal{A},\; \pi(s) = a$ choix d'une action par état
ou $\pi : \mathcal{S} \times a \rightarrow \mathbb{R},\; \pi(s, a) = Pr[a_t = a | s_t = s]$ une distribution d'actions par état.

La fonction retour

$R_t = \sum_{k>0} \gamma^k r_{t+k}$ somme des récompenses.

Objectif : trouver la politique qui maximise la fonction de retour.

Q-value

On définit

- $V^{\pi}(s) = \mathbb{E}[R_t | s_t = s]$, la valeur d'un état s
- $Q^{\pi}(s, a) = \mathbb{E}[R_t | s_t = s, a_t = a]$ la valeur d'une action pour un état donné.
- $Q^{\pi}(s, a) = \sum_{s \in \mathcal{S}} \mathcal{P}(s, a, s')[\mathcal{R}(s, a, s') + \gamma V^{\pi}(s')]$

Equation de Bellman

- $V^{\pi}(s) = \sum_{a \in \mathcal{A}} \pi(s, a) \sum_{s' \in \mathcal{S}} \mathcal{P}(s, a, s')[\mathcal{R}(s, a, s') + \gamma V^{\pi}(s')]$
- Si on a connaissance du modèle, alors possible de résoudre par programmation dynamique.
- Dans notre cas, on ne connait pas $\mathcal{P}$.

Q-learning

Objectif : Apprendre une politique en estimant $Q(s,a)$

Algorithme

1. $\forall s, a\ Q(s,a) = 0$
2. Pour chaque scénario
3. Tant que l'état n'est pas final
4. Choisir l'action a_t en fonction de s_t
5. $Q(s_t, a_t) = Q(s_t, a_t) + \alpha(r_t + \gamma max_{a'} Q(s_{t+1}, a') - Q(s_t, a_t)]$

Fonction de choix

Détermine l'exploration ou l'exploitation de la stratégie :

- glouton : $argmax_a Q(s_t, a)$
- ϵ-glouton : glouton avec un probabilité ϵ, pris au hasard parmi les actions possibles avec une probabilité $1 - \epsilon$
- softmax : $Pr(a_s|s_t) = \frac{Q(s_t,a_t)}{\sum_a Q(s_t,a)}$
- Boltzman : $Pr(a_s|s_t) = \frac{e^{\frac{Q(s_t,a_t)}{\tau}}}{\sum_a e^{\frac{Q(s_t,a)}{\tau}}}$

Monte-Carlo control

Toutes les corrections sont faites à la fin d'un sénario

Algorithme

1. $\forall s, a \; Q(s, a) = 0$
2. Pour chaque scénario
3. Tant que l'état n'est pas final
4. Choisir l'action a_t en fonction de s_t
5. $R \rightarrow 0$
6. Pour $t = T - 1$ à 0 faire :
7. $R = \gamma R + r_t$
8. $Q(s_t, a_t) = Q(s_t, a_t) + \alpha(R - Q(s_t, a_t)]$