Intelligence Artificielle

# Projet : La prise de décision

3 aspects :

* 1 produit : Aspirateur
* 1 société
* 1 organisme : Que choisir ?

La face informatique de l’aspirateur doit se faire en POO.  
L’aspirateur sera composé de composants avec des prix unitaires et des probabilités de tomber en panne.

# **Travail à faire :**

* **Codage de l’aspirateur**
* **Rapport :**
  + **Choix Société**
  + **Choix Organisme**

# **Chaque semaine :**

**Il faut envoyer un mail avec :**

* **Le mail universitaire**
* **CC**
* **Sujet : L3 IA**
* **Texte : expliquant le travail effectué dans la semaine**
* **Pièce jointe : état du code à l’instant t**

**La date limite de l’envoie du mail : Le Vendredi à 18H.  
Il faut changer l’expéditeur du mail chaque semaine.**

* Agent : Aspirateur avec capteurs, effecteurs, actions.
* Environnement

Système :

Environnement

Effecteur

Capteur

Agent

Agent réflexe : collecte des informations, identification, action

Agent Omniscient (information de tous l’environnement)  
 vs Agent rationnel (juste les informations de son propre environnement)

# Mesure de rationalité :

* Environnement :

Déterministe : pour une même action, le résultat est le même à chaque fois   
 vs Stochastique : le résultat est différent à chaque fois pour une même fonction

I I I  
Observable Partiellement observable inobservable

Episodique : pour une action faite, elle ne dépend pas du passé  
 vs Sériel : les actions dépendent du passé (le choix d’une action modifie les activités)

Discret (espace et temps )   
 vs Continu

Statique : évolutionque par les actions de l’agents  
 vs Dynamique : au cours du temps, l’environnement évolue indépendamment des actions de l’agent

Connu : l’agent connait les lois qui régissent  
 vs Inconnu

La prise de décision revient à évaluer la capacité de choix :

* Agent
* Extra-agent

Exemple : Environnement statique, épisodique, discret

B

A

Agent : Aspirateur : dans une pièce sale (état initiale)

* Arrêter : 0
* Aspirer : 1
* Vider sac : 1
* Gauche : 1
* Droite : 1

Pour maximiser la quantité de saleté aspirée maximisé : Aspirer, vider, aspirer, vider …  
Pour minimiser l’énergie (aspirer +1 → perte d’énergie -3) : Rien faire, arrêter.

Pour l’utilisateur, un aspirateur efficace est un aspirateur :

* A la fin, il n’y a plus de saleté
* N’utilise pas trop d’électricité.

: Aspirer, gauche, aspirer, droite, aspirer, éteindre.

< sal\* , rien >  
< sal\* , sal >  
< rien , sal\* >  
< sal , sal\* >

Espérance // utilité : 4 critères du choix :

* Complétude : choix parmi A et B, forcément une préférence

Perf(A) > pref(B)

Ou pref(A) < pref (B)

Ou pref (A) = pref (B)

* Transitivité :

Si pref(a) > pref (B) et si pref (B) > pref (C)

Alors Pref (A) > pref (C)

* Indépendance

Si pref(A) > pref (B), quel que soit p in [0,1] et quel que soit C une loterie

Pref( p A + (1-p) C) > pref ( p B + (1-p) C)

* Continuité : 3 choix A, B et C

Pref (A) > pref (B)

Pref (B) > pref (C)   
alors il existe p in [0, 1], pref (B) = pref (p A) = pref ( (1-p) C)

Choix rationnel ne fonctionne pas :

Exemple 1 :

|  |  |  |  |
| --- | --- | --- | --- |
| A | 1 000 € | A 100 % | E = 1 000 € |
| B | 1 500 € | A 90 % | E = 1 350 € |
| C | 1 000 € | A 10 % | E = 100 € |
| D | 1 500 € | A 9 % | E = 135 € |

Pref (A) > pref (B) et pref (D) > pref (C)

Or A = p.C + (1-p) Z et B = p.D + (1-p) Z, avec p in [0,1] et Z loterie.  
Ici la continuité n’est pas respectée.

Exemple 2 :

Cas d’une épidémie :

* Groupe 1 :
  + A : sauver 200 personnes sur 600
  + B : sauver les 600 avec une probabilité de 1/3
* Groupe 2 :
  + C : laisser mourir 400 personnes sur 600
  + D : 2/3 de chance de voir mourir 600 personnes

Résultat : Perf (A) > perf (B) et pref(D) > perf(C)  
or E(A) = E(B) = 200   
E(D) = E(C) et A = C et B = D

→ Aversion à la perte : optimisme, surévaluation de p faible et sousévaluation de p forte.

### Modélisation :

**1 classe : Monde :**

* 2 pièces : A et B
* Sait où est l’aspirateur
* Connait l’état statique des pièces

**1 classe : Aspirateur :**

* Pas de capteurs
* Pas de mémoire
* Actions :
  + Aspirer
  + Droite
  + Gauche
  + Arrêt

Initialisation :

* Placer la poussière
* Placer l’aspirateur
* Initialiser l’horloge : temps = tmax

Tant que Temps > 0 faire :

Agent choisit 1 action

Temps ← Temps – 1

Monde met à jour les infos

Fin

* Renvoie performance Agent

# Prise de décisioN

Cours 20/01/16

1. Il faut des choix : un ensemble de choix possible C1, …, Cn
2. Il faut un contexte (qui permet de filtrer les choix adaptés au contexte) : on obtient un sous-ensemble {C1, …, Cp} appartenant à { C1, …, Cn}
3. Sélection dans le sous-ensemble filtré par un critère de sélection quantifiable (un but)  
   Quantifiable   
     
   évaluation   
     
   Maximisation / minimisation : Rappel : max (v1, …, vp) = - min (v1, …, vp)

**Application à l’aspirateur debilum** :

Il doit choisir une action :

* Aspirer (prendre quelque chose)
* Aller à gauche (déplacement)
* Aller à droite (déplacement)
* Extinction
* Ne rien faire Facultatif

Contexte : Tout est identique (rien de distinguable) ≡ 1 seul contexte  
→ filtrage = identité  
Les actions sélectionnables = A = {aspirer, gauche, droite}

Le module de décision : (suit les étapes où l’on a tous les choix et le filtrage)  
+ fonction d’évaluation  
 1 sous-ensemble « restreint » des « meilleurs actions » (au sens de la fonction d’évaluation).  
Stratégie du choix parmi les ex-aequo :

* Déterministe : dépendance de la structure de données ; l’objectif est de minimiser le coût
* Stochastique : random
* Agent envoie sa décision au monde
* Le monde met à jour l’état du monde (informations de chaque pièce) + posAgent
* Le monde renvoie1 feedback à l’agent :
  + L’agent le prend en compte → Apprentissage
  + L’agent ne le prend pas en compte

L’apprentissage : comment apprendre ?

* Modification de la probabilité de sélection d’une action.

C(i) ; p(i) : le feedback modifie les probabilité des choix possibles, comment ?   
à chaque choix possible est associée une valeur : C1, C2, C3, C4, C5 :Au départ ils sont tous égaux ( = 1).  
Après le filtrage, reste : C1, C3, C5   
 e1, e3, e5→ etot = e1 + e3 + e5

Roue de la fortune :  
p = choix d’une valeur entre 0 et etot

Si p < e1 alors C1 Sinon si p < e1 + e3 alors C3Sinon C5

Si le feedback est positif alors on augmente l’évaluation des choix fait (le bénéfice de l’élu = perte du peuple) ; sinon on diminue l’évaluation des choix fait (perte de l’élu + bénéfice réparti sur le peuple).

Coefficient d’apprentissage : 0 ≤ λ ≤ 1 → λ(t)

C(1) e(1) – λ/p  
… ……  
C(M) e(p) – λ/p

C1; e1 C3; e3 C5; e5

Bénéfice (e1) : e1 + λ, e2, e3 – λ/2, e4, e5 – λ/2  
Sélection : C1, C2, C5  
Même procédure : C5 (perte) : e1 + λ + λ/2, e2 + λ/2, e3 – λ/2, e4, e5 – λ – λ/2

Autre méthode de choix : exploration / exploitation (c’est la clé du problème).

L’objectif du système est de tirer un nombre pour choisir exploration vs exploitation

Si exploitation, on prend le « meilleur »  
sinon roue de la fortune sur les autres

C’est ce qu’on appelle une stratégie ε – glouton ou « e-greedy » (en général ε ~ 0.8 ou 0.9).

**Aspirateur, minimum debilum :**

1 contexte c’est chouette, mais ça coûte (les capteurs ont un coût et une fiabilité) :

* Durée de vie
* Information correcte.

Représentation des informations pour choisir des actions : les règles : Condition → Action

Exemple : les systèmes de production (ancien Système Expert) avec des applications en médecine, en chimie, exploration minière, automobile … Ce sont des systèmes de connaissances.

Omniscient : il voit le monde tel qu’il est   
Contexte :

* 4 mondes possibles
  + Propre propre
  + Propre sale
  + Sale propre
  + Sale, sale
* 2 posAgent : A et B

→ il y a donc 8 contextes différents

Monde : avec l’agent dans une des pièces (qu’elle soit propre ou pas)

B

A

**Aspiratum automaton** :

* Si < propre, propre > et < pos = A >, alors arrêt
* Si < propre, propre > et < pos = B >, alors arrêt
* Si < propre, sale > et < pos = A >, alors droite
* Si < propre, sale > et < pos = B >, alors gauche
* … (8 règles)

Le contexte est la partie gauche de la règle. Le filtrage défini les règles « déclenchables ». Cette méthode s’appelle le matching ou l’unification.

Cas d’un monde avec n pièces, chaque pièce a 2 valeurs possible (propre, sale) : pour créer vaccum-automaton, il faut **2n x n règles**. →Trop de règles : il faut donc :

* Factoriser → symbole générique et variable

< propre, propre > , pos = \* alors arrêt

< propre, \* >, pos = A alors droite

< \*, \* > , pos = \* alors arrêt

* ou faire de l’apprentissage
  + En généralisation
  + En spécialisation

< propre, propre > , pos = A, alors arrêt

< propre, propre >, pos = B, alors arrêt

* Généralisation : < propre, propre >, pos = A ou B (univers = \*), alors arrêt
* Spécialisation :  
  Si < \*, \* >, pos = \*, alors arrêt,

Et si l’environnement envoie : < sale, sale >, pos = A

On va chercher les règles qui correspondent à ce contexte :

* + On choisit la seule règle
  + Et le monde n’est pas content (retour négatif)

Au lieu de ça on se spécialise :

< sale, sale >, pos = A, alors pas arrêt