Apprentissage par renforcement RL

- Les fondamentaux cognitifs et comportementaux

- Upper Confidence Bound Algorithm

Learning by doing

- Agent intelligent
- Actions
- Exploration
- Iterations
- Strategie

Agent intelligent apprend à partir **d'expériences successives.**
A chaque iteration il doit choisir un comportement pour trouver la **meilleure solution.**
Il decouvre les regles en faisant les actions.

Apprentissage classique

Behavioral & Cognitive Sciences

1. Renforcement classique
2. Renforcement instrumental
3. Theorie du language
4. Apprentissage social par observation

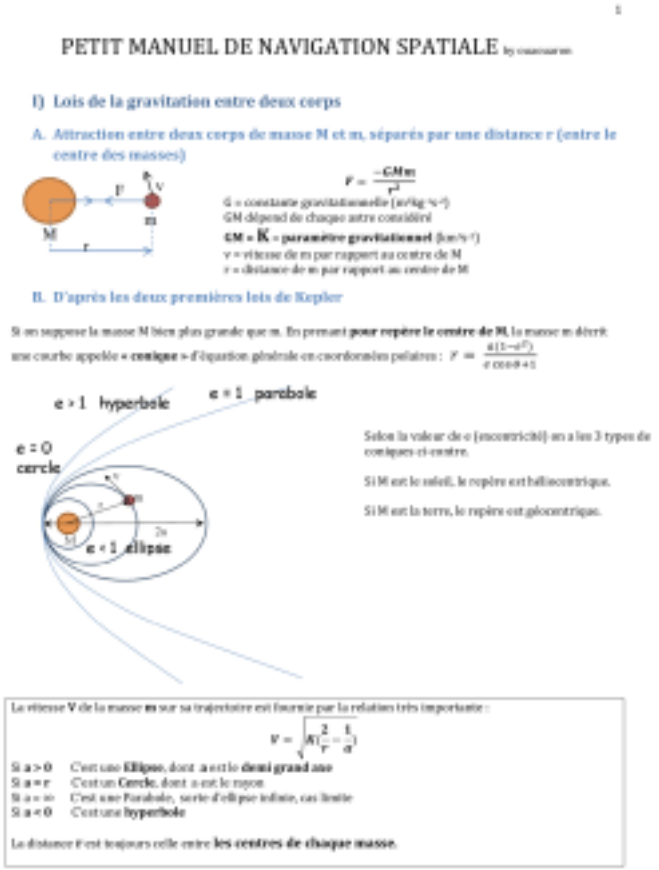
Pavlow, Skinner, Chomsky, Bandura

Comment a-t-il appris?

```
function passurletapis (chien) {
y = (7x1 + 3x2- 2x3 +log x4)/chien;
return y;
}
```

Comment j'ai appris à marcher?

Données :
Rayon de la Terre : R ≈ 6400 km,
masse de la Terre : M



PETIT MANUEL DE NAVIGATION SPATIALE

I) Lois de la gravitation entre deux corps

A. Attraction entre deux corps de masse M et m, séparés par une distance r (entre le centre des masses)

$$F = \frac{-GMm}{r^2}$$

G = constante gravitationnelle ($m^3 kg^{-1} s^{-2}$)
GM dépend de chaque autre considéré
GM = K = paramètre gravitationnel ($km^3 s^{-2}$)
v = vitesse de m par rapport au centre de M
r = distance de m par rapport au centre de M

B. D'après les deux premières lois de Kepler

Si on suppose la masse M bien plus grande que m. En prenant **pour repère le centre de M**, la masse m décrit une courbe appelée « **conique** » d'équation générale en coordonnées polaires : $r = \frac{a(1-e^2)}{e\cos\theta + 1}$

e > 1 hyperbole
e = 1 parabole
e = 0 cercle
e < 1 ellipse

Selon la valeur de e (excentricité) on a les 3 types de coniques ci-contre.

Si M est le soleil, le repère est héliocentrique.

Si M est la terre, le repère est géocentrique.

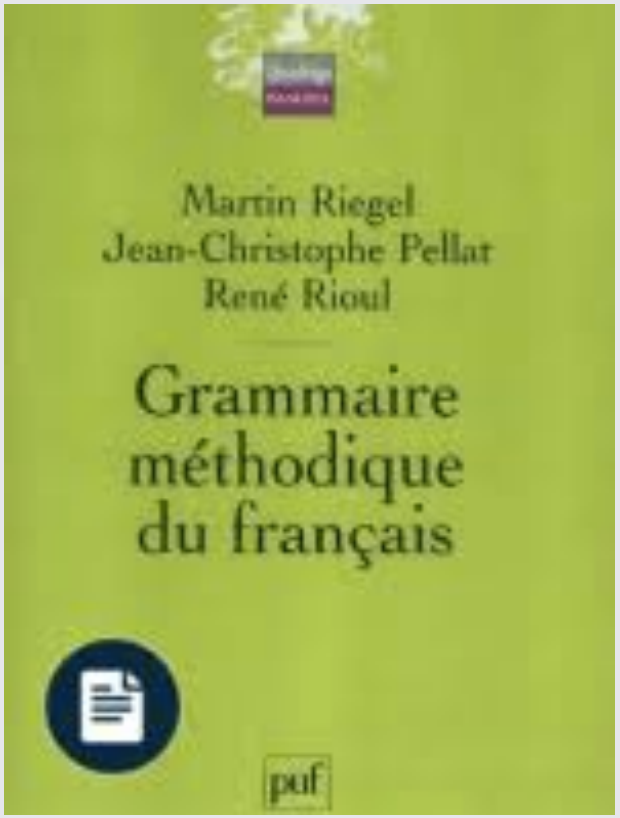
La vitesse **V** de la masse **m** sur sa trajectoire est fournie par la relation très importante :

$$V = \sqrt{K\left(\frac{2}{r} - \frac{1}{a}\right)}$$

Si a > 0 C'est une **Ellipse**, dont a est le **demi grand axe**
Si a = r C'est un **Cercle**, dont a est le rayon
Si a = ∞ C'est une Parabole, sorte d'ellipse infinie, cas limite
Si a < 0 C'est une **hyperbole**

La distance **r** est toujours celle entre **les centres de chaque masse.**

Comment j’ai appris à parler?



Les „synonimes" du RL?

- Apprentissage on - line
- Apprentissage interactif
- Learning by doing
- Programmation dynamique

Supervised

$y = f(x1, x2, x3)$

Non supervised

$f(x1, x2, x3)$

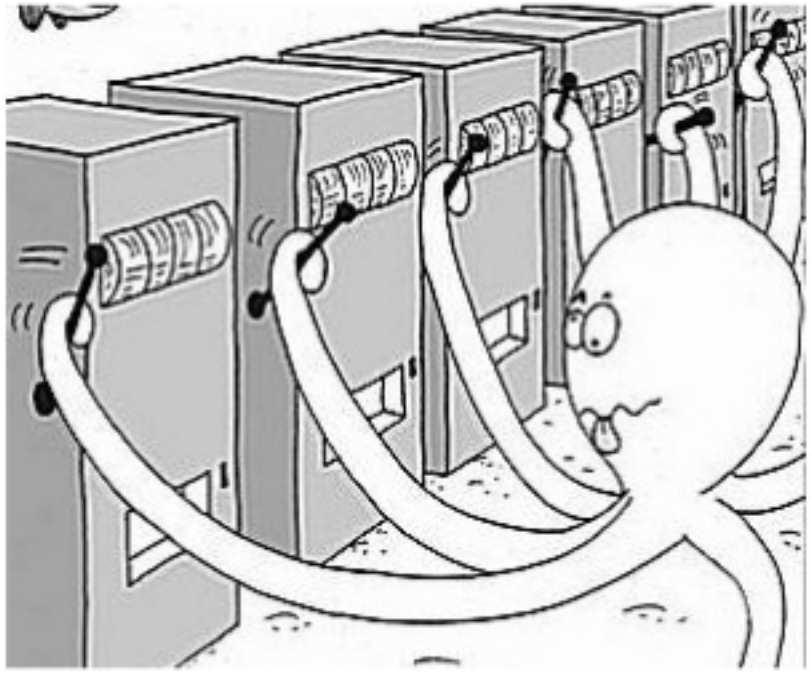
By reinforcement

$y(A) = f(x1, x2, x3)$
$y(B) = f(x1, x2, x3)$

Par exemple ?

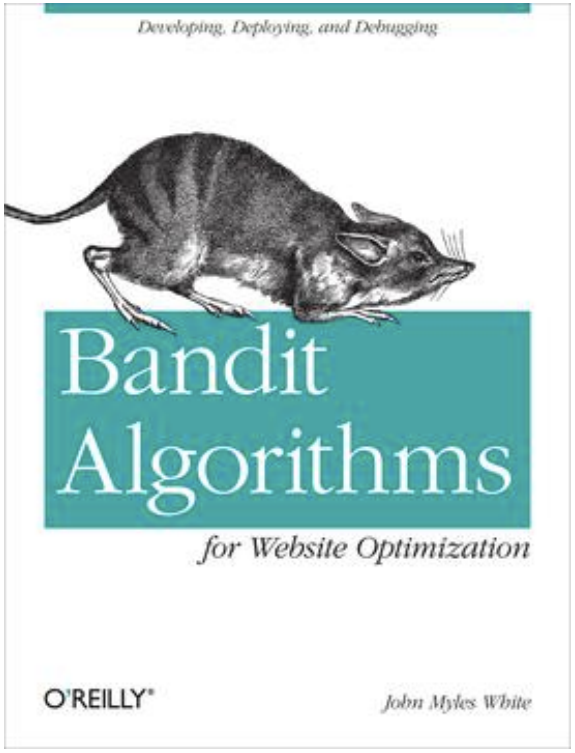
- Apprendre une machine à jouer au jeu: go, echecs
- Piloter un agent à travers un labyrinthe
- Apprendre un robot à marcher en terrain difficile,
- Apprendre une voiture autonome à conduire

Multi – armed bandit problem



source: Microsoft Research

Website Optimisation

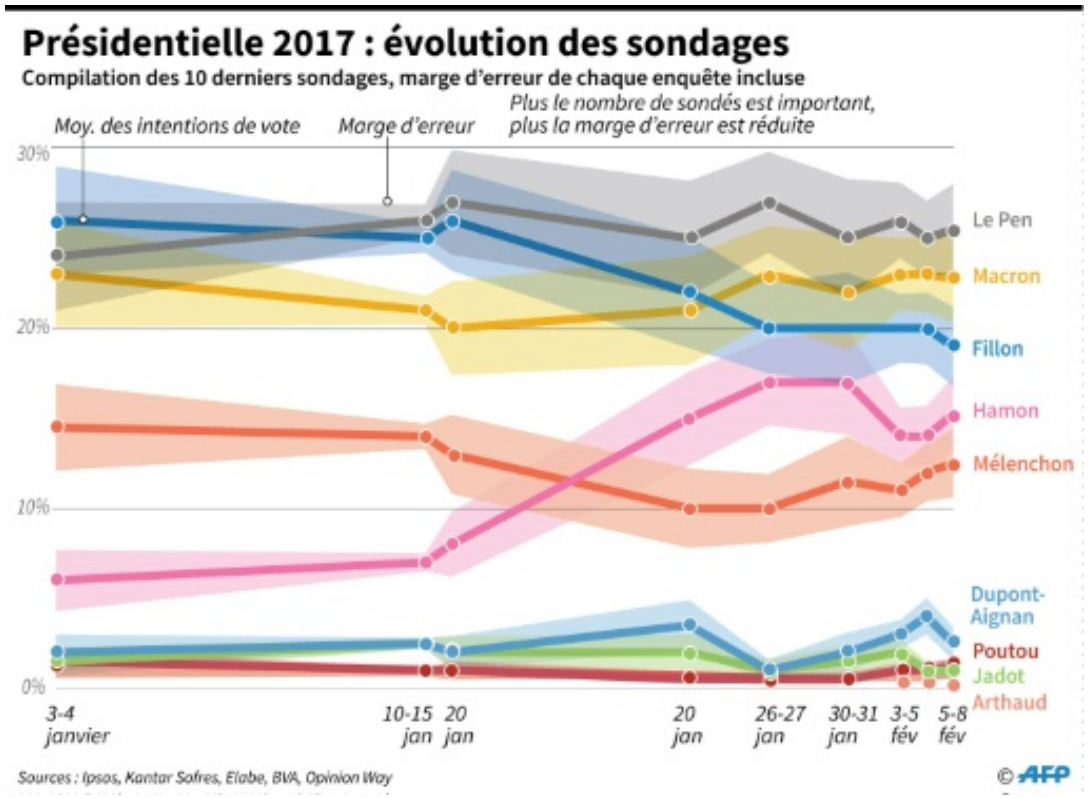


Solutions

- Classic Sampling
- Home made algorithm

- Upper Confidence Bound Algorithm
- Thompson Sampling

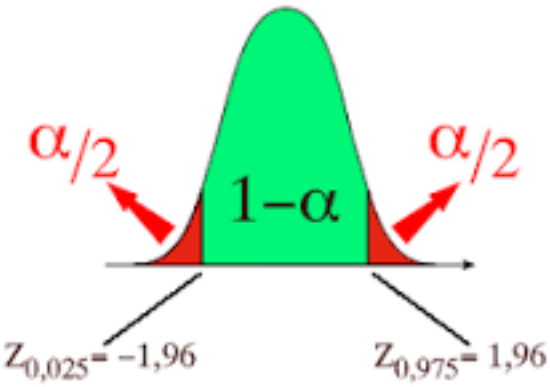
Confidence Interval



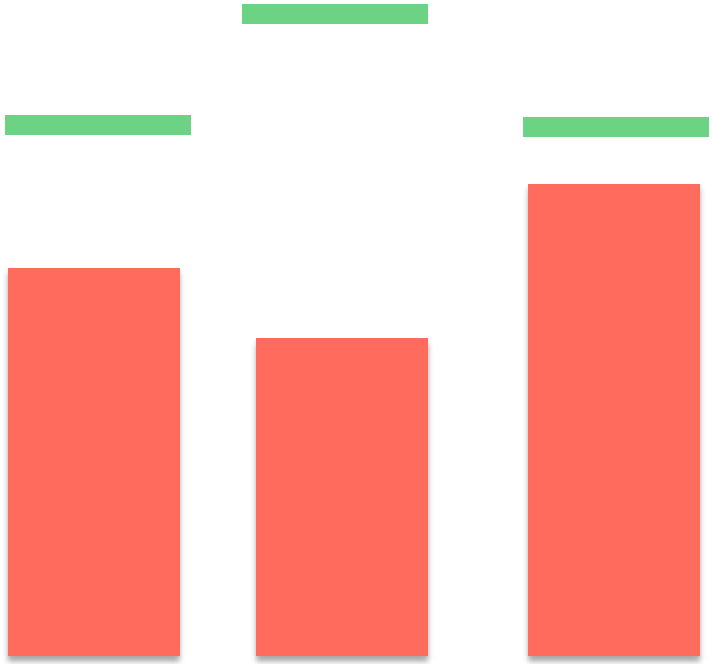
Confidence Interval

Intervalle de confiance de la moyenne lorsque la variance de la population est connue. La moyenne d'un échantillon (Mx), étant une variable aléatoire, est rarement égale à la moyenne réelle de la population (μ) dont l'échantillon est issu. Elle s'en rapproche d'autant plus que la taille de l'échantillon (n) est grande.

Intervalles de confiance
webapps.fundp.ac.be/umdb/biostats/?q=book/export/html/217



Upper Confidence Bound



Calculer Delta (interval)

Step 1. At each round n, we consider two numbers for each ad i:

- $N_i(n)$ - the number of times the ad i was selected up to round n,
- $R_i(n)$ - the sum of rewards of the ad i up to round n.

Step 2. From these two numbers we compute:

- the average reward of ad i up to round n

$$\bar{r}_i(n) = \frac{R_i(n)}{N_i(n)}$$

- the confidence interval $[\bar{r}_i(n) - \Delta_i(n), \bar{r}_i(n) + \Delta_i(n)]$ at round n with

$$\Delta_i(n) = \sqrt{\frac{3}{2}\frac{\log(n)}{N_i(n)}}$$

Step 3. We select the ad i that has the maximum UCB $\bar{r}_i(n) + \Delta_i(n)$.

Pseudo-code 1

```
//////////////    classic random sampling ///////////////////

// variables
int number_ads = 3;  // number of action types
int iterations = 1000;  // number of iterations,
int[] nombre_affichages = new int[actions]; // number of choices for each action type
int[] nombre_clicks = new int[actions]; // number of rewards for each action type
int[] taux_clicks = new int[actions];  // average reward for each action type
Dataset simulation = read('fichier');

// algo

for (int iter = 0; iter < iterations; iter ++) {
    int ad_index = random(0, actions);
    int result = simulation.get(iter, ad_index);
    nombre_affichages[ad_index]++;
    nombre_clicks[ad_index] =+ result;
    average[ad_index] = nombre_clicks[ad_index] / nombre_affichages[ad_index];
}


// affichage de resultat
 afficher(nombre_affichages);
 afficher(nombre_clicks);
 afficher(taux_clicks);
```

Pseudo-code 2

```
////////////  Upper Confidence Bound ////////////////////////

// variables supplementaires
int[] delta = new int [actions]; // interval de confiance
int[] ucb  = new int [actions]; // average + interval
int best_ucb = 0 ;

// algo
for (int iter = 0; iter < iterations; iter ++) {
    int best_ad = random(0, actions);

    // re-choisir une publicite d'apres le meilleur ucb
    for (int index = 0; index < actions; index ++){
        delta[index] = formule_magique();
        ucb[index] = taux_clicks[index] + delta [index];
        if (ucb[index] > best_ucb) {
            best_ucb = ucb[index];
            best_ad = index;
        }
    }

    int result = simulation_dataset.get(iter, ad_index);
    nombre_affichages[ad_index]++;
    nombre_clicks[ad_index] =+ result;
    taux_clicks[ad_index] = nombre_clicks[ad_index] / nombre_affichages[ad_index];
}



// affichage de resultat
```