# Examen d'IA : Applications

J.P. Forest - P. Loubière - H. Senoussi

25 mars 2019, durée : 3h

La clarté et la précision de la rédaction seront prises en compte dans l'évaluation.
Ordinateur sous exammanager obligatoire.
Tous documents électroniques autorisés, à déposer avant l'examen dans le répertoire `ia_documents`.
Autres machines (ordinateurs, téléphones, montres connectées et calculatrices) interdites.

Aucune question ne pourra être posée durant l'examen. En cas de doute concernant le sujet, vous expliquerez vos hypothèses.

**Pour chaque question vos résultats ET votre code source devront être déposés dans le répertoire `ia_rendu`.**

**Certains algorithmes sont joints à titre d'aide.**
**Attention si vous souhaitez les utiliser, ce sont des fichiers Python "à trous" dont il faut compléter/modifier le code.**

## 1 Apprentissage par renforcement

On considère un robot qui peut occuper 6 positions que nous notons $P_1, P_2, P_3, P_4, P_5, P_6$. Les déplacements possibles sont les suivants :

1. $P_1 \rightarrow P_3$
2. $P_2 \rightarrow P_3$
3. $P_3 \rightarrow P_4$
4. $P_5 \rightarrow P_4$
5. $P_4 \rightarrow P_6$

Les règles du déplacement sont les suivantes :

1. Lorsque le robot tente un déplacement interdit il est sanctionné et il ne bouge pas.
2. Lorsqu'il atteint la position $P_6$ il reçoit de la nourriture.
3. Tous les mouvements licites coûtent une certaine quantité d'énergie.

Questions :

1. Donnez le processus de décision markovien décrivant ce problème.
2. Écrire un programme Python (ou autre langage) permettant d'appliquer le Q-learning pour trouver une stratégie optimale.
   — On essaiera plusieurs valeurs des paramètres de l'algorithme et on veillera à ce que le programme fasse de l'exploration et de l'exploitation.
3. Décrivez brièvement les effets des changements des valeurs des paramètres.

# 2 Optimisation

## 2.1 Optimisation à variables discrètes : La TDB

1. Appliquer une des métaheuristiques que **vous avez implémenté** pour résoudre le problème de la TDB (Tournée Des Bars) afin de trouver le trajet minimum du problème décrit par le fichier `tdb.txt` de l'archive `data.tgz`.
   Pour ce problème, dérivé du TSP, on considère que le trajet (en mètres) s'effectue à pieds et que la consommation dans chacun des N bars ajoute une pénalité cumulative sur chaque arête entre deux bars, selon :

   $$distance_{trajet}(bar_i, bar_{i+1}) = (1 + i * 0.01) * distance_{euclidienne}(bar_i, bar_{i+1}), i \in \{1..N\}$$

   Dans le calcul du trajet, on considère également que l'on doit revenir au premier bar pour récupérer son/sa vélo/trotinette/pull oublié...
   Pour la génération de voisin, vous utiliserez la permutation 2-opt :

   `deux-opt([1,2,3,4,5,6], 1, 4)` donne `[1,5,4,3,2,6]`.

   Retourner le résultat (*longueur du chemin ; liste des villes*) dans un fichier `tsp1.res`, ainsi que votre code source.

## 2.2 Optimisation à variables continues : PSO

1. Appliquer **votre algorithme de PSO** pour trouver l'optimum (minimum) de la fonction eq. (1) :

   $$f(X) = 100(x_1^2 - x_2)^2 + (x_1 - 1)^2 + (x_3 - 1)^2 + 90(x_3^2 - x_4)^2 + 10.1\left[(x_2 - 1)^2 + (x_4 - 1)^2\right] + 19.8(x_2 - 1)(x_4 - 1), \quad (1)$$

   pour $X = (x_1, ..., x_4)^T$, avec $x_i \in [-100, 100]$.
   Rendre le résultat $(f(X);\ X)$ dans un fichier `pso1.res`, ainsi que votre code source d'autre part.
2. La version FIPS de PSO définit une nouvelle équation de la vitesse ; cf Eq. (2).

   $$v_{i,j}^{t+1} = \chi \left( v_{i,j}^t + \sum_{k=1}^{N_i} \frac{random(0, \phi)(pbest_{k,j}^t - x_{i,j}^t)}{N_i} \right). \quad (2)$$

avec $\chi = 0.729844$, $\phi = 4.1$, $pbest^t_{n,j}$ est la meilleure position personnelle de la particule $x_k$ et $N_i$, la taille du voisinage de la particule $i$ (ici, on supposera que le voisinage est l'essaim tout entier).

Modifier votre algorithme PSO de sorte que l'équation de la vitesse soit celle de FIPS.

Rendre le résultat $(f(X);\ X)$ d'autre part, dans un fichier `pso2.res`, ainsi que votre code source.