

Intelligence Artificielle 1

Cours 7 - Récursivité

Steve Lévesque, Tous droits réservés © où applicables

Table des matières

- 1 Définition
- 2 Utilité de la récursivité
- 3 Exemple simple avec problématique
- 4 Exemple simple avec cas d'arrêt
- 5 Exemple - Factoriel

Définition

Une fonction récursive est une fonction qui s'appelle elle même.

Ce principe permet de résoudre un problème large en plusieurs petites instances du même problème par récursivité.

Sources :

<https://www.geeksforgeeks.org/recursive-functions/>

<https://www.geeksforgeeks.org/>

[what-is-memoization-a-complete-tutorial/](#)

Utilité de la récursivité

- 1. Résoudre des tâches complexes :** revient au principe fondamental de séparer un gros problèmes en petits sous-problèmes.
- 2. Diviser et régner :** Même principe que 1., mais il est possible de fusionner les résultats des sous-problèmes pour obtenir la solution finale.
- 3. Backtracking :** Permet le retour en arrière pour accomplir plus efficacement des problèmes si l'on doit utiliser des solutions antérieures (i.e. Sudoku).
- 4. Programmation dynamique (memorisation) :** Même principe que 2., mais on peut garder en mémoire des étapes répétées pour sauver du temps (i.e. Factoriel, Fibonacci, etc.).
- 5. Graphes et arbres :** Pour les problèmes qui touche les graphes et arbres, le principe 4. peut être appliqué (i.e. "Traveling Salesman Problem").

Exemple simple avec problématique

- Cet exemple cause quel problème ?

```
1  # Steve Levesque, All rights reserved
2  def recursive_hello():
3
4      print("Hello")
5      recursive_hello()
6
7
8  recursive_hello()
```

Exemple simple avec problématique

- Cet exemple cause quel problème ?
- Comment peut-on le régler ?

```
1  # Steve Levesque, All rights reserved
2  def recursive_hello():
3
4      print("Hello")
5      recursive_hello()
6
7
8  recursive_hello()
```

Exemple simple avec cas d'arrêt

```
1  # Steve Levesque, All rights reserved
2  def recursive_hello(iteration):
3      # Base case (stop case)
4      if iteration == 0:
5          print("iteration is equal 0, stopped")
6          return
7
8      print("Hello " + str(iteration))
9      recursive_hello(iteration - 1)
10
11
12 recursive_hello(5)
```

Exemple - Factoriel

```
1  # Steve Levesque, All rights reserved
2  def fact(n):
3      """
4       $n! = n * (n - 1)!$ 
5      s.a.  $n! = 1$  if  $n = 0$  or  $n = 1$ 
6
7      :param n: [int] number to apply on the factorial equation
8      :return: [int] result of the factorial equation
9      """
10     # Base cases (stop cases)
11     if n == 0:
12         return 1
13     if n == 1:
14         return 1
15
16     return n * fact(n - 1)
17
18 print(fact(0))
```


Exemple - Factoriel

Condition d'arrêt : $u_0 = 1$ et $u_1 = 1$

Récurrance : n entier et $n > 0$, $u_n = n * u_{n-1}$

$$u_0 = 1, u_1 = 1$$

$$u_2 = 2 * u_1 \implies 2 * 1 \implies 2$$

$$u_3 = 3 * u_2 \implies 3 * 2 \implies 6$$

Bibliographie

- <https://coin-or.github.io/pulp/main/index.html>