Computación Inteligente y Lenguaje Natural – 2018

# Búsqueda en Pacman

Oriol Valentín

## Introducción

En la primera parte de la asignatura usaremos en clásico juego Pacman, siguiendo un proyecto desarrollado por la Universidad de California en Berkeley.

El proyecto será desarrollado en Python y corregido en Linux. Sin embargo, se puede usar Windows o Mac para el desarrollo, si instala el entorno de trabajo adecuado.

Poeis usar IDLE, un editor para Python desarrollado en Python. Sino,

- En Linux puede usar Gedit o Emacs.
- En Windows puede usar Notepad++. Existe un plugin para Python
- En Mac puede usar Aquamacs.

U otro editor según preferencias. Lo más crítico es que el editor no confunda espacios y tabulaciones (indispensable si trabajamos en Python). Idealmente debería suportar tabulador y shift-tabulador para indentar o desindentar líneas o bloques de código.

## Introducción a Python

Con el objetivo de introducirse/repasar Python, seguir los tutoriales de `https://wiki.python.org/moin/BeginnersGuide/Programmers`. Concretamente, para aquellos que se inician en Python, se recomienda el tutorial en `http://www.afterhoursprogramming.com/tutorial/Python/Introduction/` Se recomienda seguir el tutorial y realizar los ejercicios propuestos. No debe entregarse nada relacionado con esta sección.

## Search in Pacman

En el enlace `http://inst.eecs.berkeley.edu/~cs188/pacman/search.html` encontraréis el código sobre el que tendréis que trabajar así como las instrucciones a seguir para completar la practica.

Deberéis trabajar sobre los ficheros `search.py` y `searchAgents.py` y responder a las preguntas de la 1 a la 6, las cuales consisten tanto en tareas a implementar como respuestas que deberéis dar sobre vuestra implementación. Siguiendo las preguntas de la 1 a la 6, deberéis implementar varios algoritmos de búsqueda: DFS, BFS, UC, A*, así como heurísticas y los métodos adicionales que se os pida.

Las preguntas 7 y 8 son optativas. Completarlas supondrá un 15% de puntuación adicional.

Podéis repasar los algoritmos con las transparencias vistas en clase de teoría. Tened claro como funciona el esquema general de búsqueda, y a partir de ahí, es importante que entendais bien como funcionan los distintos algoritmos.

## Requisitos y consejos

- Implementad los algoritmos de búsqueda usando siempre la verificación de nodos cerrados (closed nodes) usando la estructura de datos Python `set`.
  - Si un ha sido expandido (es decir, se calcularon sus sucesores), debe colocarse en closed.
  - Al expandir nodos, todos los que estén en closed, serán ignorados (es decir, nunca agregados a la lista de nodos por procesar.
  - En el caso de A*, un nodo nuevo debe agregarse si resulta en un estado nuevo, o el estado fue visto antes, y guardado en un nodo, pero con un peor valor f.
- Tened clara la diferencia entre nodos y estado.
- Los nodos son, en teoría, caminos, pero es mejor representarlos con un estado, su cost, la última acción ejecutada, más una referencia al nodo padre. Así podrá calcular el camino desde el estado inicial.
- En algunas distribuciones de Linux, puede que se necesite instalar el paquete python-tk, o algunos otros. La línea siguiente debería instalar todo lo requerido para ejecutarla:

  `sudo apt-get install python python-tk idle python-pmw python-imaging`

  incluido el editor IDLE.
- Tened siempre en cuenta las estructuras de datos que se están utilizando; su costo; el tiempo que demora haciendo las operaciones.

## Condiciones de la entrega

1. Las prácticas se harán en grupos de dos personas.
2. En la sesión de prácticas siguiente se realizará una entrevista a cada grupo, dirigiendo preguntas a cada miembro del equipo. Son preguntas sencillas que quién hizo la páactica podrá responder con facilidad. De no responder correctamente, podría suponer la suspensión de la práctica.
3. La entrega consistirá en un fichero comprimido en .zip o .tar con el formato P1_NIA1_NIA2, el cual debe contener:
   - `search.py`: Con vuestra implementación. Debéis comentar breve, claro y conciso vuestro código.
   - `searchAgents.py`: Con vuestra implementación. Debéis comentar breve, claro y conciso vuestro código.
   - `P1_NIA1_NIA2.pdf`, indicando
     - Nombres, NIA
     - Estado de la práctica:
       - ¿Qué funciona?
       - ¿Qué no funciona?
       - ¿Qué problemas encontraron?
       - Breve respuesta a aquellas preguntas que la requieran

   Importante no modificar el contenido ya dado en ningún archivo Python.
4. La fecha de entrega de la práctica es: Hasta la siguiente práctica.

## Comentarios adicionales

- Antes de empezar a implementar vuestro código, debeis estar seguros que entendéis lo que debéis hacer. Planear y diseñar vuestro código antes de programarlo resultará en un código más claro y os ahorrará tiempo.
- El plagio, tanto entre estudiantes como desde internet, y la colaboración de código entre equipos no será tolerada. Se puede discutir sobre la práctica, pero no muestren su código a otros equipos.
- Desarrolle incrementalmente y vaya probando tras cada cambio. Puede ser útil ir desarrollando un script o una función que pruebe el programa con diversas entradas y en diversos escenarios.

## Tips en Python

Python usa referencias, como Java. Por lo tanto, tras hacer las siguientes instrucciones

```
a=[1, 2, 3]
b=a
b[0]=10
print a
```

Veremos que se imprime

```
[10, 2, 3]
```

Esto es uno de los errores más frecuentes al usar Python en esta asignatura. Para copiar la lista a tendría que usarse un `slice`:

```
a=[1, 2, 3]
b=a[:]
b[0]=10
print a
```

que imprimiría

```
[1, 2, 3]
```