**Cuestiones**

**Cuestiones sobre guardado de grafos**

Añadir código Python y figuras cuando se juzgue necesario.

***Cuestión 1:***

**Describir qué se entiende por serializar un objeto Python.**

La serialización es un proceso muy usado para guardar objetos en un archivo o base de datos o para enviarlos a través de la red.

***Cuestión 2:***

**Json es otro formato de serialización de objetos. Comentar brevemente posibles diferencias entre pickle y json.**

Si no tiene ningún requisito de interoperabilidad (por ejemplo, solo va a utilizar los datos con Python) y el formato binario está bien, sería adecuado usar Pickle, que proporciona una serialización de objetos Python realmente rápida.

Si quiere tener interoperabilidad o tener un formato de texto que almacene sus datos, lo mejor sería usar JSON.

***Cuestión 3:***

**¿Qué ventajas e inconvenientes tendrían las funciones pickle sobre las construidas mediante el formato TFG? *Responder algo pertinente y no con lugares comunes.***

El formato TFG guarda los grafos de la forma explicada en el enunciado, primero escribiendo todos los nodos una separación con el símbolo “#”, después las ramas entre los nodos con el coste. Esto en grafos no dirigidos tendría un problema porque habría que guardar el doble de ramas ya que ambos nodos de la rama podrían utilizarla.

En pickle puede transformar un objeto complejo en una cadena de bytes y puede transformar la cadena de bytes en un objeto con la misma estructura interna. El destino más evidente de las cadenas de bytes mencionadas es escribirlas a un fichero, pero también se pueden enviar por una red o almacenarlas en una base de datos. No gestiona el tema de nombrar los objetos persistentes, ni el tema del acceso concurrente a los objetos persistentes.

**Cuestiones sobre Dijkstra**

***Cuestión 1:***

**¿Cuál es el coste teórico del algoritmo de Dijkstra? Justificar brevemente dicho coste**.

El algoritmo consiste en n-1 iteraciones como máximo. En cada iteración se añade un vértice al conjunto distinguido. En cada iteración se identifica el vértice con la menor etiqueta entre los que no están en Sk.

Además, se realizan una suma y una comparación para actualizar la etiqueta de cada uno de los vértices que no están en Sk.

Luego, en cada iteración se realizan a lo sumo 2(n-1) operaciones. Entonces tenemos:

Dado un grafo G = (V, R), donde V son los vértices y R las ramas.

El Algoritmo de Dijkstra realiza O(n2) operaciones (sumas y comparaciones) para determinar la longitud del camino más corto entre dos vértices de un grafo ponderado simple, conexo y no dirigido con n vértices.

El coste de todas las iteraciones es O(|R|log|R|), y dado que |R| = O(|V|²);

O(|V|) + O(|R|log|R|) =

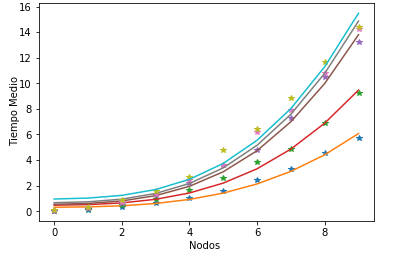
O(|V|) + O(|R|log|V|²) =

O(|V|) + O(|R|log|V |) = **O(|R|log|V|)**.

***Cuestión 2:***

**Expresar el coste de Dijkstra en función del número de nodos y el sparse factor ρ del grafo en cuestión. Para un número de nodos fijo adecuado, ¿cuál es el crecimiento del coste de Dijkstra en función de ρ?**

**Ilustrar este crecimiento ejecutando Dijkstra sobre listas de adyacencia de grafos con un número fijo de nodos y sparse factors 0.1, 0.3, 0.5, 0.7, 0.9 y midiendo los correspondientes tiempos de ejecución.**



Al aumentar el sparse factor se aumenta el número de ramas del grafo, por tanto, el tiempo de Dijkstra, que como hemos visto es O(|R|log|V|), aumentará cuantas más ramas haya. Cuantos más vértices V, es decir nodos, tenga el grafo más notable es la diferencia.

***Cuestión 3:***

**¿Cuál es el coste teórico del algoritmo de Dijkstra iterado para encontrar las distancias mínimas entre todos los vértices de un grafo?**

El coste teórico del algoritmo de Dijkstra iterado es el mismo que el Dijkstra simple pero multiplicado por la cantidad de vértices |V|.

Siendo el coste total el coste de iterar sobre un vértice multiplicado por el número de vértices totales. Es decir, |V|\*O(|R|log|V|) = **O(|V||R|log|V|).**

***Cuestión 4:***

**¿Cómo se podrían recuperar los caminos mínimos si se utiliza Dijkstra iterado?**

En el caso de Dijkstra iterado, se crearía una lista de previos donde se guardasen los padres de cada uno de los vértices. Recorriendo dicha lista se podrían recuperar los caminos.

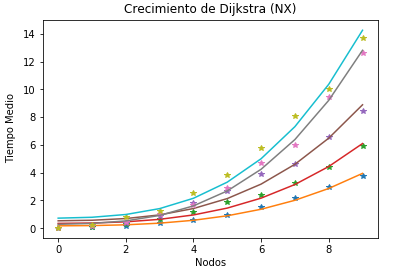
**Cuestiones sobre NetworkX**

Responder a las siguientes cuestiones incluyendo gráficas cuando sea necesario.

**Cuestión 1:**

**Show graphically the growth of the execution times of Dijkstra's algorthm as a fuction of the number of nodes and the sparsity factor using the NetworkX library.**

**Work with graphs with 100 nodes and sparse factors 0.1, 0.3, 0.5, 0.7, 0.9.**



**Cuestión 2:**

**Measure and show graphically the execution times of iterated Dijkstra to find the minimum distances between all the vertices of a graph using the NetworkX library and working on graphs with a fixed number of 25 nodes and sparse factors 0.1, 0.3, 0.5, 0.7, 0.9.**

