Trabajo Metaheurística – Downloader, Algoritmo de Dijkstra & Lógica Difusa

Alumno: Jorge de Andrés

Profesor: Alfonso López

Cuatrimestre: 1º

Curso: 4º

Asignatura: Investigación Operativa

Objetivo del Trabajo

El objetivo principal del trabajo es la implementación por mi cuenta del algoritmo de Dijkstra, de tal manera que una situación controlada como la búsqueda de ruta más corta en una red de nodos (con un principio y un final definidos, como cualquier grafo) pueda ser hecha.

Lenguaje de Programación

Java - Versión 8.181

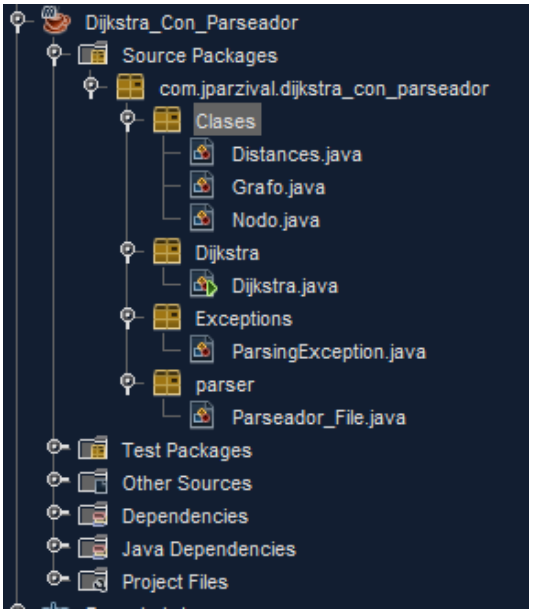
Python – Versión 3.7

R – Versión 3.5.1

Comentarios de Interés:

El código está totalmente y ampliamente comentado para su entendimiento, explicando en él mismo las bases del algoritmo de Dijkstra mientras que se codifican.

La estructura de carpetas es la siguiente:



Es importante comentar que tanto la clase Distances como la clase Grafo han sido implementadas para futuras actualizaciones del código, ya que la primera me servirá para el parseo automático del Json al correr el proyecto (cosa actualmente no implementada) y la segunda, como pone comentado en la misma, podrá ayudar a implementar una interfaz gráfica o similares en el futuro.

Problema concreto:

El problema está planteado de la forma siguiente:

Tenemos un nodo origen llamado “Salamanca”, y el Grafo apunta hacia un nodo llamado “Valencia” a donde queremos ir. El algoritmo será el encargado de mostrarnos como resultado final la ruta más corta para llegar a nuestro destino, que en este caso consiste en:

Salamanca – Madrid – Valencia

Otros nodos añadidos a la red son:

- Valladolid
- Zaragoza

Todo esto se puede ver bien en la imagen que dejo adjunta (Imagen_Dibujo_Grafo.jpg).

Aparte, como elemento adicional, he cogido las distancias entre los nodos de una API que me ha sido prestada. Esta API hace llamadas a una máquina utilizada como rutero. El paquete está dentro de la carpeta “Downloader”.

Un rutero es un software que permite el cálculo de rutas y distancias entre nodos de una forma óptima. En este caso, el rutero al que estoy haciendo llamadas tiene implementado el Algoritmo de Dijkstra para ello. Es decir, esa máquina está haciendo el Algoritmo de Dijkstra para obtener la ruta más corta entre mis nodos (utilizando sus nodos intermedios) y devolverme la distancia, y yo estoy haciendo Dijkstra para obtener la ruta más corta entre mis nodos. También hay que puntualizar que la máquina también utiliza el algoritmo de contracción de jerarquías, que sirve para mejorar la velocidad al encontrar el camino más corto.

De esta forma, el algoritmo en Python me está calculando las distancias entre ciudades, y obtengo un Json. Este json lo parseo con una librería de Python para poder obtener únicamente lo que me interesa, la distancia, y la guardo en un dictionary de Python, el que posteriormente paso a Json para la visualización clara de los datos (distancias.json).

Este archivo lo que estoy haciendo es meterlo dentro del proyecto Java y mostrarlo, y después en el proyecto Java ya implemento mi Dijkstra.

La información de este rutero que me ha sido prestado es de dominio público, ya que está subido a GitHub y es de la OSRM Foundation. Los dos links que dejo a continuación sirven para buscar información:

- http://project-osrm.org/
- https://github.com/Project-OSRM/osrm-backend

Finalmente, hay un pequeño módulo hecho en R donde defino una serie de funciones de lógica difusa para valorar lo “largo” que es el viaje.

Finalmente, hay un pequeño módulo hecho en R donde defino una serie de funciones de lógica difusa para valorar lo “largo” que es el viaje. En este módulo, tendremos que tener en cuenta que tenemos dos zonas con intersección, por lo que en esas zonas los usuarios tendrán diferencias entre si el viaje es de una manera o de otra.

Ejemplo:

Para 370 km, tenemos dos funciones afectadas. Calculamos de cada una de ellas sustituyendo:

MEDIA: (400 – 360) / (400-350) → 0.8

ALTA: (360-350) / (450-350) → 0.1

A la luz de estos resultados, podemos obtener que las posibilidades de que sea media son realmente superiores a las de que sea alta. Hay que tener en cuenta de que estos NO son probabilidades, son índices.

¿Qué son los elementos que estoy utilizando?

A continuación, voy a dar una serie de definiciones de los conceptos que estoy utilizando en el algoritmo:

- Algoritmo de Dijkstra

  El algoritmo de Dijkstra es un algoritmo que está dentro de los llamados "Algoritmos de Búsqueda". El objetivo de este algoritmo es el buscar la ruta más corta desde un nodo origen hasta un nodo fin.

  Tiene un gran problema en redes grandes, ya que, como se basa en sucesivas iteraciones de la red, en una red de grandes dimensiones este algoritmo quedará muy atrás respecto a otros. Por ello, algunas soluciones como la contracción de jerarquías son bienvenidas.

  Este algoritmo fue inventado por Edsger W. Dijkstra (Holanda, 1930).

  Como curiosidad, algunas personas dicen que sólo tardó 20 minutos en inventarlo.

- Rutero

  Un rutero, o Máquina de Rutas, es una máquina que posee un motor de alto rendimiento de búsqueda de rutas entre dos o más puntos.

  La fundación más grande que desarrolla ruteros es la OSRM Foundation (www.project-osrm.org), que posee su código abierto de tal manera que cualquier persona con unos conocimientos medios de informática y Linux (aunque se acepta en otras plataformas también) puede formar su propio rutero desde casa.

- JSON

  Un JSON (Java Script Object Notation), es un formato de archivación de datos que destaca por ser muy intuitivo, manejable y de fácil acceso.

  Actualmente, el formato JSON es un estándar en la Web y en los intercambios de datos de archivos.

- Algoritmos de Búsqueda

Los algoritmos de búsqueda son un conjunto de algoritmos que sirven para encontrar elementos dentro de una estructura de datos definida.

Todos estos algoritmos buscan dos cosas fundamentales:

  - Encontrar si el elemento está dentro de la estructura de datos (en nuestro caso, en Dijkstra, es el último elemento al que apuntamos, por lo que lo damos por hecho).
  - Hallar la posición o el camino a la posición en la que se encuentra dicho elemento.