Universidad Politécnica de Madrid:
Escuela Técnica Superior de Ingenieros Informáticos

Inteligencia Artificial

"Aplicación para calcular el trayecto óptimo entre dos estaciones del metro de Milán"

Grupo número 4

17 de Diciembre de 2017

Indice

Introducción

El objeto de este proyecto es diseñar una aplicación para hallar el trayecto óptimo entre dos estaciones del metro de Milán (incluyendo las líneas que se indican en el plano anexo), teniendo en cuenta distintos parámetros que se explican en los apartados posteriores a este. Para el cálculo del mejor camino entre dos estaciones se utilizará el algoritmo A*, un algoritmos de búsqueda en la optimización de caminos de coste mínimo en grafos de decisión.

Componentes del grupo

Los miembros del grupo responsables de realizar este proyecto son los siguientes:

- Daniel Alejandro Lobo - 170500
- Diego Fernández Gutierrez - t110122
- Adrián García Ajo - v130319
- David Feeney - 170525
- Laura Jiménez Barranquero - w140200
- Adrián Fuentes Ampudia - w140164

Desarrollo de la práctica

Lenguaje y librerías utilizadas

Se optó por utilizar Java SE Development Kit 8 para este proyecto. Además se utilizaron librerías que se adjuntan junto al código fuente de este proyecto entregado en conjunto a esta memoria. Se pueden encontrar en la carpeta “lib” dentro del código fuente. Las librerías externas utilizadas son las siguientes:

- net.datastructures 4.0
- Java BDD repackaged

Herramienta gráfica

La herramienta gráfica que se decidió utilizar es AWT que es una interfaz de Java y viene con las librerías que cualquier desarrollador de Java con el Java SE Development Kit en su última versión puede acceder fácilmente y ejecutar el código. Se trata de una herramienta estándar y de gran adopción en la comunidad de desarrolladores Java.

Algoritmo utilizado

Se utilizó un algoritmo de búsqueda en la optimización de caminos de coste mínimo en grafos de decisión. El algoritmo utilizado fue el A*. Presentado por primera vez en 1968, es un algoritmo que está bastante estudiado y del que se puede obtener bastante información dada su popularidad. El algoritmo es una combinación entre búsquedas del tipo primero en anchura con primero en profundidad: mientras que h’(n) tiende primero en

profundidad, g(n) tiende primero en anchura. De este modo se cambia de camino de búsqueda cada vez que existen nodos más prometedores.

Particularmente, en su función de evaluación se pueden representar las distintas características y particularidades de cada estación (si tiene transbordo, en caso de que lo tenga la distancia que hay que recorrer y si es mediante escaleras mecanicas o no ya que influye en el tiempo de transbordo).

Características y particularidades de las estaciones

Para implementar el área de búsqueda hemos dividido el mapa del Metro de Milan en cuadrantes de modo que cada estación tenga sus propias coordenadas cartesianas, y sus estaciones adyacentes queden en cuadrantes a los que se puede acceder horizontal y verticalmente desde el cuadrante de la estación actual. De este modo evitaremos el salto de una línea a otra como consecuencia de un movimiento diagonal sin llegar a uno de los transbordos.

Para conseguir que solo se pueda establecer un movimiento vertical u horizontal de modo que las líneas encajen en los ejes cartesianos y además añadir lejanía entre estaciones, hemos añadido nodos vacíos que solo se tendrán en cuenta durante la ejecución del algoritmo, pero que, lógicamente no aparecerán en el trayecto resultado.

La implementación de los ejes cartesianos y los nodos en el código la hemos llevado a cabo mediante una matriz de 20x18 celdas donde se albergarán los nodos (tanto estaciones como nodos vacíos). Las celdas vacías (null) representan nodos no transitables.

Para obtener los tiempos entre estaciones se ha buscado toda la información en la página oficial del metro de Milán. Se han introducido los tiempos que se tarda de una estación a otra colindante, de manera que se ha considerado que la ida y la vuelta, en cuanto a tiempo se refiere, es igual, ya que las diferencias eran mínimas. Se ha añadido también un tiempo de trasbordo en las estaciones donde hay varias líneas.

Pasos dados para la ejecución

Incluir las siguientes librerías:

- net.datastructures 4.0
- Java BDD repackaged

en el Java Build Path del proyecto. Estas son las únicas dependencias que se necesitan por fuera del Java SE Development Kit. Las mismas están incluídas en el proyecto entregado, en la carpeta “lib”.

Dificultades encontradas

Una de las dificultades encontradas fue incluir los transbordos tanto en el mapa como en los tiempos para ir de una estación a otra. Incluirlos en el mapa no fue un gran desafío dado que es simplemente contar cada estación como un punto en el plano de coordenadas cartesianas.

La consideración para el tiempo transbordo en las estaciones se solucionó al considerar las estaciones de transbordo como dos, por ejemplo Loreto: Loreto-V (linea verde) y Loreto-R (linea roja). Además añadimos otras dos estaciones exclusiva para el trasbordo que son M1 y M2 que están conectadas a las estaciones colindantes de las líneas 1 y 2 respectivamente. Lo que sucede cuando se produce un transbordo de, por ejemplo, linea 2 a linea 1 en este punto, el recorrido saldria como Piola-transbordo-Lima. En el caso de que el trayecto se produjera solo en la línea 2 seria del estilo Piola-Loreto-V-Caiazzo.

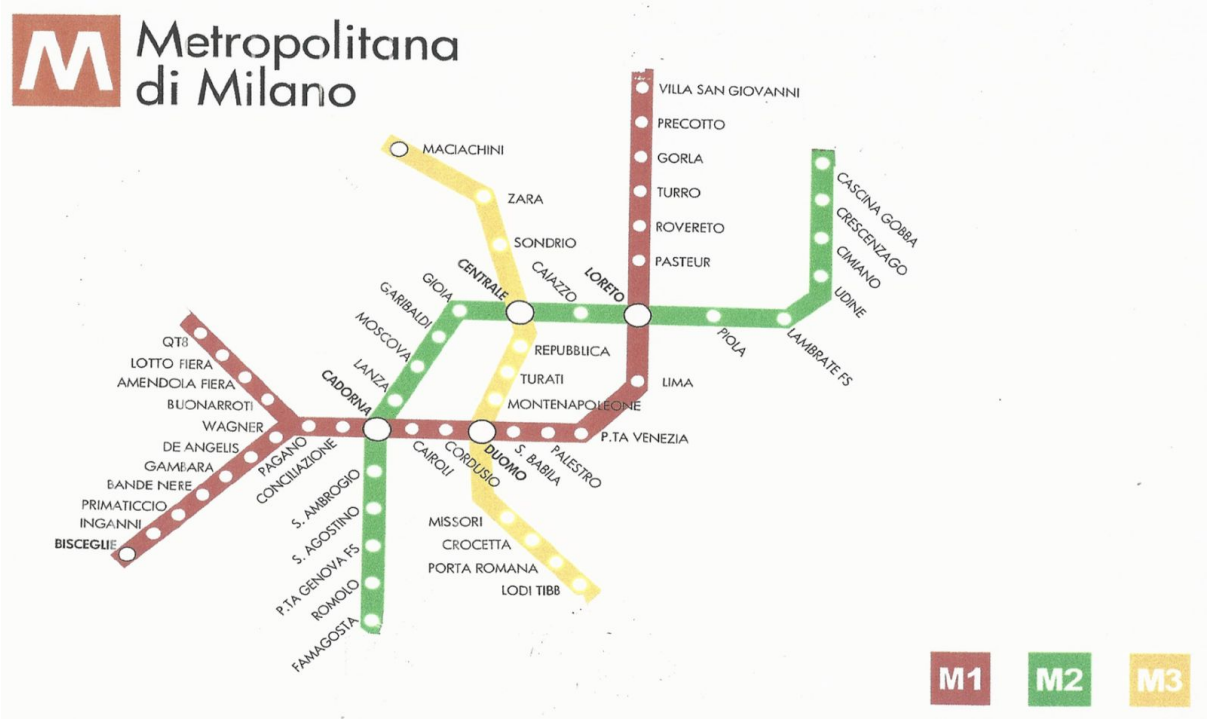
Otra de las dificultades encontradas fue la realización de la interfaz gráfica y conectar los datos en tiempos de ejecución con lo calculado por el algoritmo.

Se logró el objetivo planeado desde el comienzo del proyecto, mostrando resultados por la salida estándar de Eclipse y por la interfaz gráfica del usuario.

Bibliografía utilizada

- Rusell S. and Norving P. Inteligencia Artificial. Un enfoque moderno. Prentice Hall 1996.
- https://en.wikipedia.org/wiki/A*_search_algorithm
- https://en.wikipedia.org/wiki/Iterative_deepening_A*
- http://www.sciencedirect.com/science/article/pii/0004370295000179
- Apuntes teóricos de Inteligencia Artificial y material didáctico visto en clase.
- http://www.muoversi.milano.it/c/portal/layout?p_l_id=17110&p_v_l_s_g_id=0&idCtx=perc&tab=tpl
- http://net3.datastructures.net/download.html
- http://grepcode.com/snapshot/repo1.maven.org/maven2/de.fosd.typechef/javabdd_repackaged/1.0b2/

Anexo


Metropolitana di Milano
VILLA SAN GIOVANNI
PRECOTTO
GORLA
TURRO
ROVERETO
PASTEUR
MACIACHINI
ZARA
SONDRIO
CAIAZZO
LORETO
CENTRALE
GIOIA
GARIBALDI
MOSCOVA
LANZA
CADORNA
QT8
LOTTO FIERA
AMENDOLA FIERA
BUONARROTI
WAGNER
DE ANGELIS
GAMBARA
BANDE NERE
PRIMATICCIO
INGANNI
BISCEGLIE
PAGANO
CONCILIAZIONE
CAIROLI
CORDUSIO
DUOMO
S. BABILA
PALESTRO
P.TA VENEZIA
LIMA
REPUBBLICA
TURATI
MONTENAPOLEONE
PIOLA
LAMBRATE FS
CASCINA GOBBA
CRESCENZAGO
CIMIANO
UDINE
S. AMBROGIO
S. AGOSTINO
P.TA GENOVA FS
ROMOLO
FAMAGOSTA
MISSORI
CROCETTA
PORTA ROMANA
LODI TIBB
M1
M2
M3