Projecte 1: Navegació

Oscar Lostes Cazorla
Esteve Pineda Sánchez
Máximo Martínez Alcívar

31 de Març, 2019

Introducció

L'objectiu d'aquesta pràctica és resoldre un problema ben habitual: com arribar del punt A al B de la millor forma possible. A diari emprem eines com Google Maps per aquesta finalitat, i el que volem és fer una petita réplica aplicant els coneixements apresos a classe.

El nostre programa ens permet carregar la informació de la xarxa de metro (total o parcial) d'una ciutat, i trobar la ruta óptima entre dos punts de la mateixa. Establim diversos criteris d'optimitat com poden ser a arribar el més aviat possible, agafar el camí més curt, fer pocs transbords o passar per poques parades. Per a cada criteri, associem una heurística (una funció que fa una predicció optimista sobre el futur) que ens servirà a l'hora de calcular la ruta. La informació de la xarxa de metro, expressada en diversos fitxers de text, la convertirem en una representació en nodes, que contindran la informació de cada estació, i que esdevindran l'arbre a recórrer per A* Donades unes coordenades o estacions d'origen i destí, i tenint en compte els criteris anteriors, el programa aplicarà l'algoritme A* après a classe per a realitzar una cerca informada per tal d'obtenir la ruta desitjada.

Per tal d'arribar a aquest programa desitjat, anirem programant diverses funcions intermitges que després integrarem per a implementar l'algoritme A*, i que anirem explicant al llarg d'aquesta memòria.

- Expansió d'un node.
- Eliminació de cicles.
- Obtenir l'estació més propera donades unes coordenades.
- Establir la matriu de costos i el cost real d'un node.
- Establir heuristiques i funcions heurístiques globals.
- Inserció ordenada d'un node en una llista.
- Eliminació de camins redundants.

Per últim, tot l'anàlisi del programa descrit en aquesta memòria, així com els possibles exemples, imatges o figures, es bassen en en el mapa de la xarxa de metro de la ciutat francesa de Lyon, tot i que el programa és vàlid per a qualsevol xarxa de metro.

Treball previ a implementar A*

Exercici 2 - Expansió d'un node

Un primer pas per al nostre algoritme és expandir un node. Expandir vol dir que, donat un node, volem obtenir una llista amb els seus fills. Això, traduït a l'esquema de metro, ens diu a quines estacions podem anar des d'una estació donada, i per tant ens serveix per avançar per la xarxa. Cada node conté una llista amb els ID de les estacions adjacents. Si iterem sobre aquesta llista, podem construir-ne una nova amb els nodes que representen aquestes estacions, establint el node original com a pare, i acudint a la informació de la xarxa de metro per a obtenir les dades concretes de la estació (a partir de l'ID). Podem anar repetint aquesta expansió de forma cíclica, emprant els criteris adequats i filtrant-la amb les funcions pertinents, fins a formar una llista que acabarà esdevenint el camí òptim.

A la vegada, aprofitem per establir certs valors addicionals de cada node, i que veurem més endavant, com son el cost real d'arribar fins al node, el cost heurístic i la funció d'avaluació. Aquest valors tenen molta importància a l'algoritme, i per tant han de ser inclosos en tots els nodes.

Exercici 3 - Eliminar Cicles

Quan expandim un node, creem la llista amb totes les estacions adjacents. Això ens pot portar a situacions on, a la llista de fills, ens apareguin nodes ja expandits anteriorment. A aquest fenomen en diem cicle, i els cicles són contraproduents per al nostre algoritme, ja que el porta a entrar en bucles de camins (podem dir,que comença a fer voltes en cercles). És evident que un camí òptim no passa per aquest escenari, i per tant l'hem d'eliminar.

Tots els nodes contenen una llista amb els ID de les estacions pare. Amb aquesta informació podem saber quins nodes son un cicle en una llista de nodes. Si per cada node de la llista, iterem sobre el conjunt d'IDs de pares, podem eliminar de la lista aquelles estacions que formen part del conjunt, deixant així la llista neta de cicles.

Exercici 4 - Trobar l'estació més propera

Quan vulguem calcular la ruta entre dos punts, no sempre ens trobarem físicament a una estació de metro, així que una petita part del programa s'encarrega de trobar l'estació més propera a les coordenades indicades, a

partir de la qual farà els càlculs. Malauradament, el programa no ens indica com arribar a aquesta estació més propera; això ja depèn de l'usuari.

La idea és senzilla: com disposem de les coordenades de totes les estacions, podem crear una llista amb la distància euclidiana INSERTAR FORMULA DISTANCIA entre el punt indicat i cadascuna de les estacions. Cada distància s'associa amb la ID de l'estació corresponent, i posteriorment aquesta llista s'ordena de menor a major distància. Finalment obtenim els ID de les estacions més properes (és a dir, els primers X elements de la llista. Quan parlem de les X primeres estacions de la llista ens referim a que podem tenir el cas de que tinguem més d'una estació exactament a la distància més propera, i per tant hem de recollir totes aquestes estacions.

Exercici 6 - Cost

En aquest exercici se'ns demana treballar sobre dos elements relacionats amb els costos: la taula de costos i el cost real d'un node.

Taula de costos

Segons la preferència escollida per a calcular el trajecte, crearem una taula de costos basant-nos en la matriu d'adjacència (que forma part de la definició de la xarxa de metro). Aquesta matriu ens proporciona el temps que es triga entre una estació i una altra adjacent. Segons cada preferència, la taula de costos ens queda de les següents formes (com es calcula el valor per a cada element de la matriu):

Adjacencia Si hi ha adjacència entre les estacions, l'element val 1; sinó val 0.

Temps Com la matriu original està expressada en temps, la copiem directament per la taula de costos, i cada element expressa el temps entre estacions.

Distància Disposem del temps entre estacions i de la velocitat mitjana de cada línia (també forma part de la definició de la xarxa), per tant podem establir la distància com INSERTAR FORMULA DISTANCIA CON VELOCIDAD MEDIA, i cada element expressa aquest mateix càlcul.

Transbords Si les dues estacions estan en línies diferents (i per tant hem de fer transbord per anar d'una a l'altra, l'element val 1; sinó val 0.

Parades Una mateixa estació a nivell de xarxa de metro pot tenir més d'una estació a nivell de codi, ja que per cada línea en una estació de metro es genera una estació diferent a nivell de codi. En aquest cas ens interessa el primer tipus d'estació. Si les dues estacions tenen noms diferents (el nom no inclou la línia, i per tant representa el primer concepte) l'element val 1; sinó val 0.

Cost real d'un node

Cada node guarda, entre d'altres la dada del cost real. Aquest numero indica, segons la preferència escollida, quan ens ha costat arribar a aquell node des del node que designem com a origen del trajecte. Això es pot reinterpretar com: COSTE REAL = COSTE REAL PADRE + COSTE TABLA COSTE[PADRE][SELF] Si partim de la taula de costos calculada a l'apartat anterior, i que el cost del node orígen és només el cost d'anar al següent node, podem calcular el cost real segons aquesta simple formula.

Exercici 7 - Heurístiques

Per poder determinar la funció de costos final, primer necessitem calcular la heurística adient segons la preferència que determini l'usuari. La heurística calculada, ha d'esdevenir admissible (mai ha de sobre-estimar el cost que falta per arribar a la solució). Segons la preferència que indiqui l'usuari, es tindran en compte una sèrie d'elements o no:

Mínim temps Per calcular l'heurística admissible depenent del temps, s'ha de tenir en compte si l'usuari es troba en una estació que comparteix línia amb l'estació de destí o no. En el millor dels casos, aquesta condició es complirà i per tant, aprofitant la fórmula de VELOCITAT = ESPAI / TEMPS, podem calcular el temps òptim (aprofitant la velocitat màxima de la ciutat (que podem obtenir del paràmetre city que conté la informació de la ciutat), així com la distància en línea recta calculada prèviament). En cas contrari, a aquest temps òptim calculat, li haurem de sumar el temps que trigaria un usuari en realitzar un transbord.

Distància mínima Com ja hem esmentat, per tal que l'heurística sigui admissible, la manera que tenim de calcular aquesta, tenint en compte que la prioritat de l'usuari és obtenir el camí més curt, és utilitzar la distància en línia recta que hi ha entre les estacions d'origen i destí (aquest serà el cas òptim).

Mínim nombre de transbords En aquest cas, per determinar l'heurística, s'ha de comprovar si l'usuari es troba en la mateixa línia que l'estació de destí, de ser així, el millor cas és que l'usuari no ha de fer cap transbord, mentre que si no es compleix aquesta condició, l'usuari com a mínim (i per tant aquesta serà l'heurística que definirem) haurà de fer un transbord.

Mínim nombre de parades Per determinar el millor cas en quant a nombre de parades, hi ha dues possibilitats per a que l'heurística esdevingui admissible:

1. L'estació d'origen sigui alhora la de destí (tot i que és una mica absurd)

2. L'estació de destí es trobi a una parada de distància respecte l'estació d'origen.