

07/12/2020

Joel Farré Cortés (78103400T)

Joel Aumedes Serrano (48051307Y)

Escola politècnica superior – universitat de lleida

Pràctica 1 –

Pac-Man Search

Grau en Enginyeria Informàtica

**Introducció.**

Per a la realització d’aquesta pràctica es proposava, mitjançant els diversos algorismes de cerca informada i no informada vists durant les hores de classe, resoldre diverses qüestions en el clàssic i popular joc del Pac-Man.

**Descripció de les decisions preses en la implementació de l’algoritme A\*.**

Degut a la semblança entre l’A\* i l’algoritme UCS implementat a classe, veiem que l’A\* és el mateix codi amb l’adició de l’heurística. Per a garantir que l’algoritme sigui òptim, i assumint que la cerca es en graf tal com diu l’enunciat, necessitem una heurística consistent. La tècnica emprada per a transforma una heurística admissible a consistent es la tècnica del *path-max*. Hem optat per aplicar el *path-max* ja que en garanteix una heurística consistent a partir d’una heurística admissible ja que si es admissible i consistent al aplicar el *path-max* seguirà essent consistent.

path\_max\_fn = n.cost  
 path\_max\_gnhn = ns.cost+heuristic(ns.state, problem)  
 path\_max = max(path\_max\_fn, path\_max\_gnhn)  
 fringe.push(ns, path\_max)

**Descripció de les heurístiques proposades i de les decisions preses.**

- *Corners Problem*

Per a definir la heurística *Corners Problem* el que fem es la suma de la distància de Manhattan des del Pac-Man a cada un dels *corners* no visitats. Aquesta heurística és admissible ja que si solament queda una cantonada per visitar, la heurística com a molt serà la distància del Pac-Man a aquesta cantonada i mai es sobreestimarà el cost. Si tenim més d’una cantonada el Pac-Man tampoc ho sobreestimarà perquè la suma de les distàncies no sobreestimarà el cost del camí d’anar a la cantonada més propera i a les restants.

for corner in corners: # Corners no visitats  
 xy1 = position  
 xy2 = corner  
 total += abs(xy1[0] - xy2[0]) + abs(xy1[1] - xy2[1])  
 return total

- *Food Search Problem*

Dins del *Food Search Problem* disposem de dues heurístiques: una de consistent i una de no consistent i no admissible però molt més òptima i ràpida que la consistent.

La consistent, semblant al C*orners Problem*, retorna la suma de les distàncies als punts per on hem de passar encara, es a dir, les boles de menjar que el Pac-Man encara no s’ha menjat. Al igual que en el problema anterior, la demostració de consistència d’aquesta heurística es fa a partir de si queda una bola o més d’una. Si queda una bola és consistent ja que es el camí més proper i si en queden més d’una serà el camí més proper més el camí cap a les altres.

Aquesta heurística, en el mapa petit *Tricky Search*, donava un camí bastant òptim i no tardava massa. Al mapa gran *Big Search*, en canvi, la solució tardava més de 15 segons en trobar-se, expandia gairebé 50,000 nodes i la solució mostrada era òbviament lluny de ser òptima. Al veure això vam pensar que podia causar aquests resultats tan dolents. Observant el camí que feia el Pac-Man i tenint en compte el que calculava l’heurística vam descobrir l’error. Aquest error no es donava al *Tricky Search* però si al *Big Search* ja que al *Tricky Search* el Pac-Man comença a un costat del mapa però al *Big Search* comença al mig. Això feia que el Pac-Man contínuament: «menjo una bola i em desplaço cap a l’Est», «ara com que he menjat una bola a l’Est, l’heurística dóna millor resultat anar cap a l’Oest ja que estaré mes prop de la resta de nodes», aleshores menjarà dues boles de l’Oest i es repetirà el mateix però ara amb l’Est. Observant aquest comportament vam crear la nova heurística per a evitar aquest comportament. Aquest comportament el que acabava fent anant-se movent d’Est a Oest era augmentar la distància entre les boles de menjar. Per tant vam afegir la distància de Manhattan entre les boles de menjar restants, cada bola amb cada bola. Això feia que el Pac-Man també intentés disminuir aquesta dada i ho feia menjant-se primer totes les boles de l’Oest i després totes les de l’Est. Aquest canvi, tot i que causa que l’heurística no sigui ni admissible ni consistent genera un increment massiu de l’eficiència, trobant la solució al *Big Search* en menys de 1 segon, expandint solament 300 nodes i amb una solució que requereix 700 passos menys que l’heurística original.

**Opcional: Búsqueda subòptima**