# Pràctica 1: Cerca informada

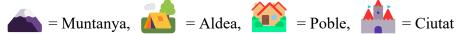
### Introducció

En aquesta pràctica es vol avaluar els coneixements sobre problemes de cerca explicats a classe. Concretament, la programació i anàlisi de diversos algoritmes de cerca informada i heurístiques. A continuació, s'explicarà el funcionament del problema a resoldre, les tasques a realitzar, el contingut a entregar i els criteris establerts per l'avaluació.

# Descripció del problema

Un comerciant ambulant medieval vol viatjar a la capital per reabastir-se. Es tracta d'un viatge de múltiples dies en que haurà de creuar una gran part del reialme. La regió a considerar està representada pel mapa inferior en forma de matriu de 10\*10 caselles. El viatge comença a la casella situada més al nord-oest (0,0) i acaba a la casella de la capital (9,9). Cada dia el comerciant es pot desplaçar a una casella adjunta en horitzontal o vertical (no diagonals) i té un cost de 5 monedes d'or. Excepció a això són les caselles amb muntanyes, que no són accessibles. Si la casella destí té una població, el comerciant guanyarà unes monedes d'or venent objectes residuals. En particular, 1 moneda si és una aldea, 3 monedes si és un poble i 4,5 monedes si és una ciutat (incloen-hi la capital). Aquestes vendes no evitaran que hagi de gastar les 5 monedes diàries, però reduiran les pèrdues. L'objectiu del comerciant és arribar a la capital havent gastat la mínima quantitat possible de diners.











### **Tasques**

Es demana el següent:

- Formalitzar el problema com cerca a un espai d'estats. Això inclou la descripció de què és un estat (amb els seus atributs), estat inicial, estat final, operadors, solució i cost a optimitzar. No més d'un paràgraf per cadascun.
- Definició de 3 heurístiques ben diferenciades (no tenen per què ser les 3 millors, però han de ser ben diferents) per intentar trobar el camí/camins més econòmics des de l'estat inicial al final. Poden ser heurístiques que només considerin la distància, només els diners o la combinació dels dos. No són vàlides heurístiques que sempre retornin el mateix número o números aleatoris. A més, s'afegeixen els següents requisits:
  - o Almenys una heurística ha de contenir alguna cosa més que les mètriques de distància estàndards (e.g., euclidiana o Manhattan).
  - o Per a cada heurística, indicar si és o no admissible respecte als diners, i per què.
  - o Almenys una heurística ha de ser admissible respecte als diners.
- Programació d'un codi capaç de resoldre el problema emprant els algoritmes de cerca heurística Best-first i A\* i les heurístiques definides.
- Prova dels algoritmes de cerca implementats amb les heurístiques definides. Concretament, avaluar la resolució del problema en els següents dos escenaris:
  - o Emprant el mapa 10x10 presentat a la descripció del problema.
  - o Emprant un mapa 5x5 dissenyat per vosaltres. Els estats inicial i final s'han de col·locar en cantonades oposades del taulell i s'ha de fer servir almenys una vegada cada tipus de cel·la (buit, aldea, poble i ciutat). Ha de ser diferent del mapa 10x10 de la descripció del problema (e.g., no pot ser un retall ni una versió a escala o rotada).
- Per a cada prova, s'ha d'indicar la següent informació:
  - o Solució (camí) que s'ha trobat, indicant el temps i cost associats.
  - O Quantitat d'estats tractats per l'algorisme de cerca (és a dir, el nombre d'iteracions de cerca que ha fet).
  - O Si la solució trobada és òptima o no respecte als diners.
  - o Anàlisi de com ha influenciat cada heurística, algoritme i mapa als estats tractats i la solució trobada.
- Indicar si l'algoritme *hill climbing* hagués trobat una solució per a cada heurística que heu dissenyat, tant al mapa 10x10 com al 5x5 dissenyat per vosaltres. No cal implementar l'algorisme, només justificar la resposta.

### **Facilitats**

Per alinear el treball de l'alumnat amb el que es considera més important de la pràctica, es donen les següents facilitats:

- Codi base en Java ja estructurat per la implementació demanada, incloent-hi comentaris amb la paraula "TODO" per indicar on és necessari afegir codi. Els alumnes poden crear mètodes i classes (aquestes últimes en nous fitxers) perquè el seu nou codi estigui ben estructurat.
- S'indica que la solució òptima al problema definit al mapa 10\*10 té un cost de 33,0. Qualsevol solució amb un cost superior (e.g., 33,5) és subòptima.

• S'indica que una implementació correcta de BestFirst o A\* hauria d'executar-se de manera quasi-instantània al mapa 10\*10. En altres paraules, si el programa porta 5 segons o més executant-se, el més probable és que hi hagi un error (e.g., un bucle infinit).

## Entrega

Es tracta d'una pràctica **individual**. Cada alumne haurà de **realitzar i entregar** a la tasca del Moodle el següent contingut:

- Informe responent les questions definides a la secció de Tasques.
- Codi Java amb la implementació dels algoritmes (Best-first i A\*) i les heurístiques. No s'ha d'incloure tot el codi del programa, només les parts indicades. Això sí, el codi d'aquestes parts ha d'estar al complet. Si s'utilitza alguna funció auxiliar o s'hereta d'una classe que es defineix en un altre fitxer, aquest també s'ha d'incloure.

Aquest contingut s'ha d'agrupar en un fitxer comprimit i tindre per nom «P1\_[Nom]\_[Cognoms].zip», substituint «[Nom]» i «[Cognoms]» pels corresponents a l'alumne. Per exemple: «PAC1 Benet ManzanaresSalor.zip».

#### Avaluació

S'avaluarà:

- Les respostes donades a l'informe.
- Correcta i clara implementació dels algoritmes de cerca demanats i les heurístiques definides.
- Respostes a l'entrevista amb el professor a la classe de laboratori el dia d'entrega.

Les pràctiques similars o idèntiques obtindran una qualificació de 0.