Grado en Ingeniería en Informática

Departamento de Informática
Universidad Carlos III de Madrid

Artificial Intelligence
Partial exam

February 2019

General instructions

- Time assigned to the exam is **1 hour 40 minutes**
- Teachers will not answer any question about the exam
- You cannot leave the classroom during the exam, unless you have finished it
- Exams cannot be answered using a pencil

Exercise 1 (3 points)

Consider the following rules and facts, where the variable x stands for a patient:

R1: IF abdominal_pain(x) **AND** vomit(x) **THEN** stomach_virus(x)

R2: IF runny_nose(x) **THEN** cold(x)

R3: IF cold(x) **THEN** contagious(x)

R4: IF abdominal_pain(x) **AND** fatigue(x) **THEN** hepatitis(x)

R5: IF stomach_virus(x) **THEN** contagious(x)

R6: IF hepatitis(x) **THEN** contagious(x)

R7: IF contagious(x) **AND** dangerous(x) **THEN** isolated(x)

R8: IF hepatitis(x) **THEN** dangerous(x)

1. (2 points) Simulate the execution of the production system using a forward chaining method. Decide which conflict resolution strategy to use. For each cycle, show clearly the conflict set, the selected rule, and the resulting working memory. The initial working memory is:

WM_0 = {fatigue(dave), abdominal_pain(dave), vomit(mary), abdominal_pain(mary), runny_nose(mary)}

2. (1 point) What can be derived from the execution of the production system? Explain your answer in detail.

Exercise 2 (7 points)

You are hired by Mediterránea S.A. to program scientific Autonomous Underwater Vehicles (AUV). The first task you are given consists of generating plans for moving an AUV from one position to another one in a grid, such that it can take a measurement at the end position. See Figure 1 for an example of a grid, the initial position and orientation of the AUV and the end position. As most scientific AUVs, this AUV has a torpedo shape with a tail.

The possible orientations are: north, north-east, east, south-east, south, south-west, west, and north-west. At each time step, the AUV can only move to the next position in the direction of the orientation (with a cost of 2), the next position to the left of the current orientation (with a cost of 3) and the next position to the right of the current orientation (with a cost of 3). In the two last cases, the orientation changes to the next orientation to the left or right, respectively. Figure 2 shows examples of possible movements when orientation is north (a) or north-east (b). The rest of movements is defined similarly.

- (2 points) Define the problem space as formally as you can. Use a production system when possible

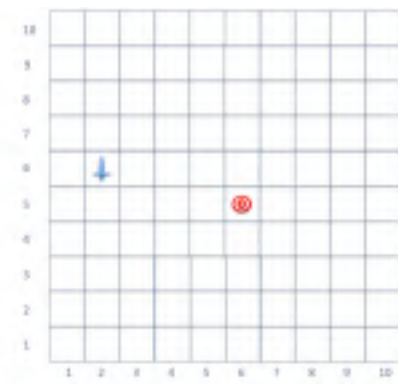
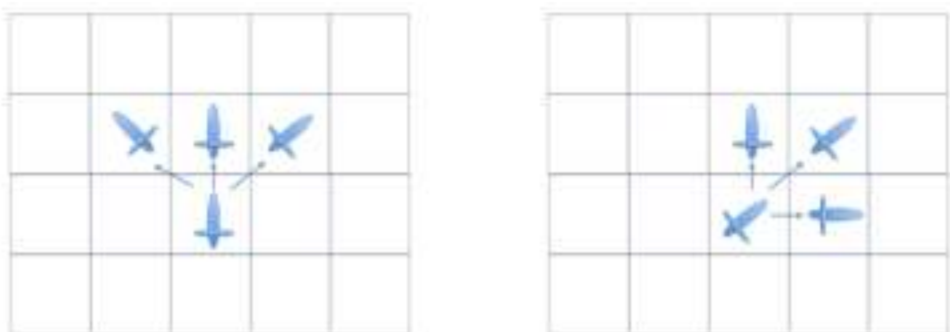


Figura 1: Example of initial position of the AUV (2,6), initial orientation (north) and destination position (6,5).



(a) (b)

Figura 2: (a) Possible movements when orientation is north. (b) Possible movements when orientation is north-east.

- (0.5 points) Define the initial state and goal state of Figure 1 according to the previous definition
- (1 point) Show the first five nodes of a search tree when using Depth-first search, Breadth-first search, and Dijkstra. Provide the necessary information to correctly interpret your solution such as open list, nodes generated and expanded, or order of generation/expansion. Check for repeated states when appropriate and prune them whenever possible. Define an order to generate successors and use the same order in all searches (e.g. left-straight-right)
- (1 point) Define a heuristic function to be used by A*
- (1.5 points) Show the first five nodes of a search tree when using Hill-climbing and A* with the previous heuristic function.
- (1 point) Which of the above algorithms finds the optimal solution? Which one guarantees the optimal solution? Explain your answers.

Amplitud → Encuentra la optima, pero la complejidad es exponencial

Profundidad → Para el orden dado encuentra la optima, y es lineal la complejidad.

Dijkstra → Encuentra la optima y es simple el algoritmo

High climbing : Para la h(n) encuentra solución, pero no podemos asegurar que sea optima.

A* → Encuentra la optima y la heuristica es admisible.

1.) Encadenamiento hacia delante. Resolución del conjunto conflicto es FIFO

	MT	CC	Resultado
0	fatigue(dave) abdominal_pain(dave) vomit(mary) abdominal_pain(mary) runny_nose(mary)	R1(x=mary) $\xrightarrow{\text{Ejecutar}}$ R2(x=mary) R4(x=dave)	stomach_virus(mary)
1	fatigue(dave) abdominal_pain(dave) vomit(mary) abdominal_pain(mary) runny_nose(mary) stomach_virus(mary)	R2(x=mary) $\xrightarrow{\text{Ejecutar}}$ R4(x=dave) R5(x=mary)	cold(mary)
2	fatigue(dave) abdominal_pain(dave) vomit(mary) abdominal_pain(mary) runny_nose(mary) stomach_virus(mary) cold(mary)	R4(x=dave) $\rightarrow$ R5(x=mary) R3(x=mary)	hepatitis(dave)

fatigue(dave) abdominal_pain(dave) vomit(mary) 3 abdominal_pain(mary) runny_nose(mary) stomach_virus(mary) cold(mary) hepatitis(dave)	R5(x=mary) $\xrightarrow{\text{Ejecutar}}$ R3(x=mary) R6(x=dave) R8(x=dave)	contagious(mary)
fatigue(dave) abdominal_pain(dave) vomit(mary) 4 abdominal_pain(mary) runny_nose(mary) stomach_virus(mary) cold(mary) hepatitis(dave) contagious(mary)	R3(x=mary) $\xrightarrow{\text{1. Ejecutar}}$ R6(x=dave) $\xrightarrow{\text{2. Ejecutar}}$ R8(x=dave)	contagious(mary) ya está en MT contagious(dave)
MT_4 $\cup$ contagious(dave)	R8(x=dave) $\xrightarrow{\text{Ejecutar}}$	dangerous(dave)
MT_5 $\cup$ dangerous(dave)	R7(x=dave) $\longrightarrow$	isolated(dave)

$MT_6 \cup$ isolated(dave)

2.)

1. AUV(x, y, h)
x: posición en horizontal $0 < x < 11$
y: posición en vertical $0 < y < 11$
h: horientación $h \in (n, ne, e, se, s, sw, w, nw)$

El estado inicial es: AUV(2,6, n)

El estado final es: AUV (6, 5, x) $x \in (n, ne, e, se, s, sw, w, nw)$

Operaciones: Izquierda, Recto y Derecha.

2. Orden expansión: Recto, Izq, Dch

Profundidad

Lista Abierta	Expandir	Sucesores
(2,6, n)	(2,6, n)	(2,7, n), (1,7, nw), (3,7, ne)
(2,7, n), (1,7, nw), (3,7, ne)	(3,7, ne)	(4,8, ne), (3,8, n), (4,7, e)
(2,7, n), (1,7, nw), (4,8, ne) (3,8, n), (4,7, e)	(4,7, e)	(5,7, e), (5,8, ne), (5,6, se)
(2,7, n), (1,7, nw), (4,8, ne) (3,8, n), (5,7, e), (5,8, ne) (5,6, se)	(5,6, se)	(6,5, se), (5,5, s) (6,6, e) Fin

Queda el 5° y ultimo.

nw n ne
w e
sw s se

Amplitud

Lista Abierta	Expandir	Sucesores
(2,6,n)	(2,6,n)	(2,7,n),(3,7,ne),(1,7,nw)
(2,7,n),(3,7,ne),(1,7,nw)	(2,7,n)	(2,8,n),(3,8,ne),(1,8,nw)
(3,7,ne),(1,7,nw),(2,8,n),(3,8,ne) (1,8,nw),(4,8,ne),(4,7,e),(3,8,n)	(3,7,ne)	(4,8,ne),(4,7,e),(3,8,n)
(1,7,nw),(2,8,n),(3,8,ne),(1,8,nw) (4,8,ne),(4,7,e),(3,8,n)	(1,7,nw)	(1,8,n)
(2,8,n)(3,8,ne),(1,8,nw),(4,8,ne),(4,7,e) (3,8,n),(1,8,n)	(2,8,n)	(3,8,n), ya está ~~(3,8,ne)~~, (1,8,nw)

(2,8,n)(3,8,ne),(1,8,nw),(4,8,ne),(4,7,e),(3,8,n),(1,8,n),(3,8,n),(1,8,nw)

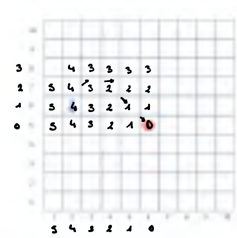
Dijkstra. Notación: (x,y,h,g) x: pos x, y: pos y, h: orientación, g: peso acumulado.

Recto 2
Izq 3
Dch 3

Lista abierta	Expandir	Sucesores.
(2,6,n,0)	(2,6,n,0)	(2,7,n,2),(3,7,nw,3)(1,7,ne,3)
(2,7,n,2)(3,7,nw,3)(1,7,ne,3)	(2,7,n,2)	(2,8,n,4),(3,8,nw,5),(1,7,ne,5)
(3,7,nw,3)(1,7,ne,3),(2,8,n,4), (3,8,n,5),(1,7,ne,5)	(3,7,nw,3)	(2,8,nw,5)(2,7,w,6) ~~(3,8,n,6)~~ ya está
(1,7,ne,3),(2,8,n,4),(3,8,n,5), (1,7,ne,5),(2,8,nw,5)(2,7,w,6)	(1,7,ne,3)	(2,8,ne,5),(1,8,n,6),(2,7,e,6)
(2,8,n,4),(3,8,n,5),(1,7,ne,5), (2,8,nw,5),(2,7,w,6)(2,8,ne,5), (1,8,ne,5),(1,8,n,6),(2,7,e,6)	(2,8,n,4)	(2,9,n,6),(3,9,nw,7)(1,9,ne,7)

(3,8,n,5),(1,7,ne,5),(2,8,nw,5),(2,7,w,6)(2,8,ne,5),(1,8,ne,5),(1,8,n,6),(2,7,e,6)
(2,9,n,6),(3,9,nw,7),(1,9,ne,7)

3.) $h(n) = \max(|x_f - x_0|, |y_f - y_0|)$ Distancia Mahattan.



Solution Exercise 1

We select the rule which has been added to the conflict set earlier (FIFO). We assume rules are only added to the conflict set (since it is monotonic reasoning - no falsifying in the RHS of rules). Thus, we can maintain the conflict set from the previous iteration, remove the executed rule and add the new ones that can be executed.

$$
\begin{aligned}
\mathrm{WM}_0 &= \{\text{fatigue(dave), abdominal\_pain(dave), vomit(mary), abdominal\_pain(mary), runny\_nose(mary)}\} \\
\mathrm{CS}_0 &= \{\text{R1(x=mary), R2(x=mary), R4(x=dave)}\} \\
\mathrm{WM}_1 &= \mathrm{WM}_0 \cup \{\text{stomach\_virus(mary)}\} \\
\mathrm{CS}_1 &= \{\text{R2(x=mary), R4(x=dave), R5(x=mary)}\} \\
\mathrm{WM}_2 &= \mathrm{WM}_1 \cup \{\text{cold(mary)}\} \\
\mathrm{CS}_2 &= \{\text{R4(x=dave), R5(x=mary), R3(x=mary)}\} \\
\mathrm{WM}_3 &= \mathrm{WM}_2 \cup \{\text{hepatitis(dave)}\} \\
\mathrm{CS}_3 &= \{\text{R5(x=mary), R3(x=mary), R6(x=dave), R8(x=dave)}\} \\
\mathrm{WM}_4 &= \mathrm{WM}_3 \cup \{\text{contagious(mary)}\} \\
\mathrm{CS}_4 &= \{\text{R6(x=dave), R8(x=dave)}\} \\
\mathrm{WM}_5 &= \mathrm{WM}_4 \cup \{\text{contagious(dave)}\} \\
\mathrm{CS}_5 &= \{\text{R8(x=dave)}\} \\
\mathrm{WM}_6 &= \mathrm{WM}_5 \cup \{\text{dangerous(dave)}\} \\
\mathrm{CS}_6 &= \{\text{R7(x=dave)}\} \\
\mathrm{WM}_7 &= \mathrm{WM}_6 \cup \{\text{isolated(dave)}\} \\
\mathrm{CS}_7 &= \{\emptyset\}
\end{aligned}
$$

Mary has a cold and an stomach virus, which is contagious. Dave has hepatitis, which is contagious and dangerous; he needs to be isolated.

Solution exercise 2

1. Problem space:
   - set of states: each state is a tuple $\langle p, o \rangle$ where p is a position of the form (x, y), and o is an orientation. You should also include the map in each state in any form. Formally, if we use predicate logic, we would use the predicates: position(X,Y), orientation(O). The map can be represented in many different ways. As an example, we can define adjacency as: adjacent(X,Y,X1,Y1,O) to say that position (X,Y) is adjacent to (X1,Y1) in orientation O. We would also need adjacent-left(X,Y,X1,Y1,O1,O) and adjacent-right(X,Y,X1,Y1,O1,O) for the left and right movements.
   - set of actions/rules:

     R1. IF position(X,Y), orientation(O), adjacent(X1,Y1,X,Y,O)
     THEN position(X1,Y1)
     R2. IF position(X,Y), orientation(O), adjacent-left(X1,Y1,X,Y,O1,O)
     THEN position(X1,Y1), orientation(O1)
     R2. IF position(X,Y), orientation(O), adjacent-right(X1,Y1,X,Y,O1,O)
     THEN position(X1,Y1), orientation(O1)

2. Initial state:

   {position(2,6), orientation(north), adjacent(1,2,1,1,north), adjacent-left(2,2,1,1,north-east,north), ...}.

   Goal: {position(6,5)}

3. Algorithms
   - *Depth-first search*

     In order to shorten the description of states, we will use the notation $\langle 2, 6, N \rangle$ to refer to the fact that the AUV is at position (2,6), orientation north.

Open	Expanded	Generated
(2,6,N)	(2,6,N)	G1={(2,6,N), (2,7,N), (1,7,NW), (3,7,NE)}
(2,7,N), (1,7,NW), (3,7,NE)	(2,7,N)	G2=G1∪{(2,8,N), (1,8,NW), (3,8,NE)}
(2,8,N), (1,8,NW), (3,8,NE)	(2,8,N)	G3=G2∪{(2,9,N), (1,9,NW), (3,9,NE)}
(2,9,N), (1,9,NW), (3,9,NE)	(2,9,N)	G4=G3∪{(2,10,N), (1,10,NW), (3,10,NE)}
(2,10,N), (1,10,NW), (3,10,NE)	(2,10,N)	G5=G4

And it finds a dead end.

- *Breadth-first search*

Open	Expanded	Generated
(2,6,N)	(2,6,N)	G1={(2,6,N), (2,7,N), (1,7,NW), (3,7,NE)}
(2,7,N), (1,7,NW), (3,7,NE)	(2,7,N)	G2=G1∪{(2,8,N), (1,8,NW), (3,8,NE)}
(1,7,NW), (3,7,NE), (2,8,N), (1,8,NW), (3,8,NE)	(1,7,NW)	G3=G2∪{(1,8,N)}
(3,7,NE), (2,8,N), (1,8,NW), (3,8,NE), (1,8,N)	(3,7,NE)	G4=G3∪{(4,8,NE), (3,8,N), (4,7,E)}
(2,8,N), (1,8,NW), (3,8,NE), (1,8,N), (4,8,NE) (3,8,N), (4,7,E)	(2,8,N)	G5=G4∪{(2,9,N), (1,9,NW), (3,9,NE)}

- *Dijkstra*

  For nodes in open we will also show now their $g(n)$ value.

Open	Expanded	Generated
((2,6,N),0)	(2,6,N)	G1={(2,6,N), (2,7,N), (1,7,NW), (3,7,NE)}
((2,7,N),2), ((1,7,NW),3) ((3,7,NE),3)	(2,7,N)	G2=G1∪{(2,8,N), (1,8,NW), (3,8,NE)}
((1,7,NW),3) ((3,7,NE),3) ((2,8,N),4) ((1,8,NW),5) ((3,8,NE),5)	(1,7,NW)	G3=G2∪{(1,8,N)}
((3,7,NE),3) ((2,8,N),4) ((1,8,NW),5) ((3,8,NE),5) ((1,8,N),6)	(3,7,NE)	G4=G3∪{(4,8,NE), (3,8,N), (4,7,E)}
((2,8,N),4) ((1,8,NW),5) ((3,8,NE),5) ((1,8,N),6) ((4,8,NE),5) ((3,8,N),6) ((4,7,E),6)	(2,8,N)	G5=G4∪{(2,9,N), (1,9,NW), (3,9,NE)}

4. Heuristic: the following are admissible heuristics. You are supposed to provide a formal definition.

   - if we relax orientation, the task simplifies to computing paths in graphs with diagonal moves. The Manhattan distance with costs would not be admissible here, since the cost of a combination of a horizontal and a vertical move will not be less than the cost of a diagonal move. The unit-cost version of the Manhattan distance would be admissible. Another possible admissible heuristic would be the Euclidean distance, though it is not very informative for this task. In case of diagonal moves and a cost of $\sqrt{2}$ for those moves, there is the octile heuristic:

$$h(n) = \sqrt{2} \times \min\{d_x, d_y\} + \max\{d_x, d_y\} - \min\{d_x, d_y\} = \max\{d_x, d_y\} + (\sqrt{2} - 1) \times \min\{d_x, d_y\}$$

     where x_g and y_g reflect the goal position, $d_x = |x_n - x_g|$ and $d_y = |y_n - y_g|$.
     Taking into account the costs of the problem, a modified octile heuristic would be:

$$h(n) = 2 \times \min\{d_x, d_y\} + 3 \times (\max\{d_x, d_y\} - \min\{d_x, d_y\}) = 3 \times \max\{d_x, d_y\} - \min\{d_x, d_y\}$$

   - if we relax position, we are left with number of moves related to orientation we have to perform to arrive to the goal. The problem for this task is that usually we do not have any constraint on the orientation at the goal. We can have a simple heuristic that returns 2 (least cost for any action) if we are not in the goal state, or 0 otherwise. If we knew the orientation at the goal, a more complex heuristic would compute where the goal is with respect to the current position and compute how many orientation actions we have to perform to arrive to the goal orientation. Again, we would multiply that value by 2. So, if we call o_n and o_g the orientations of the current node and the goal, respectively, and $d(o_n, o_g)$ the distance in changes in orientation between both:

$$h(n) = 2d(o_n, o_g)$$

     As examples of the distance function, $d(N, E) = 2$ and $d(N, S) = 4$. This heuristic is less informed than the previous one.

5. Heuristic algorithms

   - *Hill climbing*

     For nodes in open we will also show now their $h(n)$ value. We will use the modified octile heuristic:

Open	Expanded	Generated
((2,6,N),11)	(2,6,N)	G1={(2,6,N), (2,7,N), (1,7,NW), (3,7,NE)}
((2,7,N),10) ((1,7,NW),13) ((3,7,NE),7)	(3,7,NE)	G2=G1∪{(4,8,NE), (3,8,N), (4,7,E)}
((4,8,NE),10) ((3,8,N),6) ((4,7,E),4)	(4,7,E)	G3=G2∪{(5,7,E), (5,8,NE), (5,6,SE)}
((5,7,E),5) ((5,8,NE),5) ((5,6,SE),2)	(5,6,SE)	G4=G3∪{(6,5,SE), (6,6,E), (5,5,S)}

     And it finds a goal state, (6,5,SE).

   - A*

     For nodes in open we will also show now their $f(n) = g(n) + h(n)$ value. We will use the same heuristic as before. In case of a tie in the value of $f(n)$, we prefer nodes that are closer to the goal (less $h(n)$).

Open	Expanded	Generated
((2,6,N),0+11)	(2,6,N)	G1={(2,6,N), (2,7,N), (1,7,NW), (3,7,NE)}
((2,7,N),2+10) ((1,7,NW),3+13) ((3,7,NE),3+7)	(3,7,NE)	G2=G1∪{(4,8,NE), (3,8,N), (4,7,E)}
((2,7,N),2+10) ((1,7,NW),3+13) ((4,8,NE),5+10) ((3,8,N),6+6) ((4,7,E),6+4)	(4,7,E)	G3=G2∪{(5,7,E), (5,8,NE), (5,6,SE)}
((2,7,N),2+10) ((1,7,NW),3+13) ((4,8,NE),5+10) ((3,8,N),6+6) ((5,7,E),8+5) ((5,8,NE),9+5) ((5,6,SE),9+2)	(5,6,SE)	G4=G3∪{(6,5,SE), (6,6,E), (5,5,S)}

     And a goal state has been found, (6,5,SE). Since it is the best node in OPEN, it finishes.

6. Given that the task has actions with different costs, the only algorithms that can guarantee to find an optimal solution are Dijkstra and A*. In this case, Hill-climbing would also find the optimal solution given that there are no local minima. Breadth-first would also find the optimal solution, even if costs are not unitary, given that the best solution in length corresponds to the best solution in cost. Depth-first does not find a solution, even if we use backtracking, since it would find the solution with a higher cost than the optimal one.