APELLIDOS: PL:

NOMBRE: DNI:

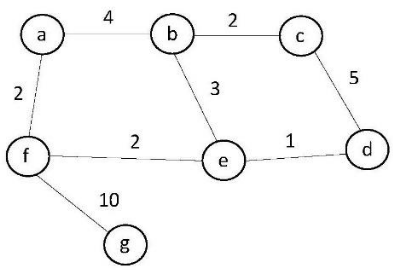
ESCUELA DE INGENIERÍA INFORMÁTICA

SISTEMAS INTELIGENTES

Examen Final de Teoría. Martes 14 de enero de 2020.

I.- Búsqueda

Se trata de resolver con búsqueda en espacios de estados el siguiente problema. Tenemos un grafo que representa conexiones entre ciudades, como por ejemplo el siguiente:



Una conexión entre dos ciudades expresa que se puede ir de una a la otra con el coste que indica el arco, pero también que se puede ver una ciudad desde la otra (esto sin coste alguno). Si dos ciudades no están conectadas con un arco, no se puede ir de una a la otra directamente, ni se puede ver una desde la otra. Una ciudad se ve desde sí misma, por supuesto.

El objetivo es calcular una ruta que parta de una ciudad concreta, en el ejemplo anterior es la ciudad a, que pase por una serie de ciudades, pero solamente una vez por cada una de ellas (y no tiene que pasar por todas las ciudades del grafo) y que termine de nuevo en la ciudad de partida, de forma que todas las ciudades del grafo puedan ser vistas desde alguna de las ciudades de la ruta. Además, la ruta debe ser de coste mínimo.

En el ejemplo anterior hay varias soluciones, la óptima es la ruta (a,b,e,f,a) con coste 11. En este problema es posible que no exista solución alguna, dependiendo de cómo sean las conexiones del grafo.

En este ejercicio se pide lo siguiente:

1.- [0,5 puntos] Indicar otra solución al problema del grafo anterior que sea distinta a (a,b,e,f,a) y su coste. Indicar también un ciclo que empiece y termine en la ciudad a y que no sea solución del problema. Dar también un grafo de conexiones distinto al anterior para el que el problema no tenga solución.

RESP: La ruta (a,b,c,d,e,f,a) es otra solución del problema con coste 16, por lo tanto no óptima. La ruta (a,b,a) no es solución del problema ya que no permite ver a las ciudades d, c y g. El grafo a – b – d – e no tiene solución ya que después del recorrido (a,b,d), aunque todas las ciudades están vistas, no se puede regresar a la ciudad a a través de ciudades no visitadas.

2.- [2 puntos] Modelar el problema para resolverlo con búsqueda en espacios de estados. Es decir, definir el espacio de búsqueda (estados, reglas, costes, estado inicial, estados objetivos). Dibujar una parte significativa del espacio de búsqueda para la instancia del problema definida por el grafo anterior.

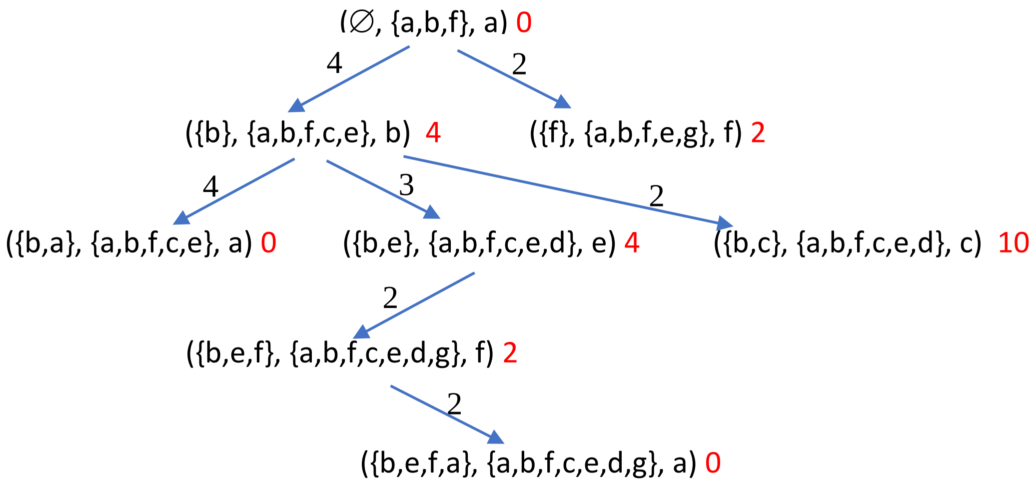
RESP: En cada estado hay que tener registradas las ciudades que ya han sido vistas y las que ya fueron visitadas, así como la ciudad actual. Con esta representación, el subproblema que representa un estado es: ir desde la ciudad actual a la ciudad de partida pasando por ciudades no visitadas, de forma que se puedan ver todas las ciudades no vistas, y con el menor coste posible.

Así un estado será una tripleta de la forma (Visitadas, Vistas, Actual). Utilizando la idea de la pista (a), el estado inicial será

(∅, Conjunto de ciudades que se ven desde la ciudad de partida, ciudad de partida)

Es decir, no registramos en el conjunto de ciudades visitadas a la ciudad de partida. Un estado es objetivo si estamos en la ciudad de partida y el conjunto Vistas incluye a todas las ciudades. Los sucesores de un estado vienen dados por las posibilidades de ir directamente desde la ciudad actual a una ciudad no visitada, el coste de cada regla es el de la conexión correspondiente y el nuevo estado es como el anterior cambiando la ciudad actual, añadiendo ésta a las Visitadas y añadiendo también las que se ven desde ésta a las Vistas.

Para el problema representado por el grafo de conexiones anterior, el espacio de búsqueda es el siguiente (una parte representativa)



3.- [1 punto] Proponer un heurístico para resolver el problema con el algoritmo A*, e indicar sus propiedades. Dar los valores de este heurístico para un conjunto significativo de los nodos del espacio anterior.

RESP: Se puede relajar la restricción de que "hay que ver las ciudades no vistas en el estado actual" a través de la ruta desde la ciudad actual a la ciudad de partida. En ese caso, la solución óptima del problema relajado es el camino más corto desde la ciudad actual a la de partida que no pase por las ciudades ya visitadas. Ese camino se calcula mediante el algoritmo de Dijsktra aplicado al "grafo residual" (el grafo inicial sin las ciudades visitadas). Para el espacio de búsqueda anterior, se indican los valores de este heurístico al lado de cada nodo, en rojo.

El heurístico es consistente por haber sido diseñado utilizando el método de la relajación del problema.

PISTAS: (a) No hay que confundir las ciudades visitadas con las ciudades vistas. Cuando llegamos a una ciudad desde otra la registramos como visitada. Cuando estamos en una ciudad vemos las adyacentes. (b) El algoritmo de Dijskstra puede ser de utilidad.

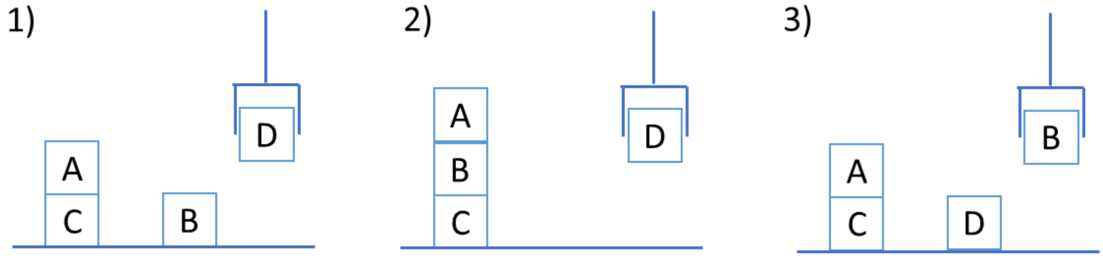
II.- Representación

4.- Dada la siguiente implementación en lenguaje STRIPS para el problema del mundo de bloques

```
(define (domain typed-blocksworld)
     (:requirements :typing)
     (:typing block hand)
     (:predicates
          (clear ?b - block)
          (on-table ?b - block)
          (empty ?h - hand)
          (holding ?h - hand ?b - block)
          (on ?b1 ?b2 - block))
     (:action pickup
          :parameters (?h - hand ?b - block)
          :precondition (and (clear ?b) (on-table ?b) (empty
     ?h))
          :effect (and (holding ?h ?b) (not (clear ?b))
                    (not (on-table ?b)) (not (empty ?h))))
     (:action putdown
          :parameters (?h - hand ?b - block)
          :precondition (holding ?h ?b)
          :effect (and (clear ?b) (empty ?h) (on-table ?b)
                    (not (holding ?h ?b))))
     (:action stack
          :parameters (?h - hand ?b ?underb - block)
          :precondition (and (clear ?underb) (holding ?h ?b))
          :effect (and (empty ?h) (clear ?b) (on ?b ?underb)
                    (not (clear ?underb)) (not (holding ?h
?b))))
     (:action unstack
          :parameters (?h - hand ?b ?underb - block)
          :precondition (and (on ?b ?underb) (clear ?b) (empty
     ?h))
          :effect (and (holding ?h ?b) (clear ?underb)
               (not (on ?b ?underb))(not (clear ?b))(not (empty
?h)))))


(define (problem typed-blocks1)
     (:domain typed-blocksworld)
     (:requirements :typing)
     (:objects H - hand A B C D - block)
     (:init (clear D) (on D A) (on A C) (on-table C) (holding H
B))
     (:goal (and (on D C) (on C B) (on B A))))
```

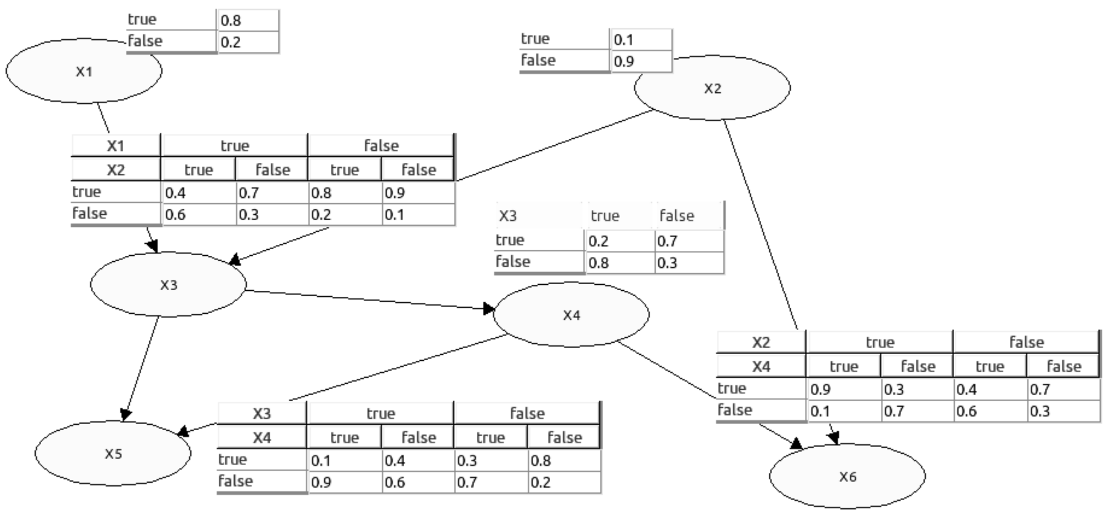
[0,4 puntos] Indica cuáles de las siguientes situaciones se pueden alcanzar desde el estado inicial aplicando dos acciones consecutivas y cuáles son estas acciones en cada caso:



Solución: Desde el estado inicial que es está constituido por la pila de bloques (de abajo a arriba) C, A y D teniendo el brazo sujeto el bloque B tenemos que:

1) Se puede alcanzar con (putdown H B) y a continuación (unstack H D A)
2) No se puede alcanzar
3) No se puede alcanzar

5.- Dada la siguiente red bayesiana



a) **[0.7 puntos]** Indica, utilizando el criterio de D-separación, el manto de Markov o la condición de Markov, si son ciertas o falsas las siguientes igualdades:

1) $P(X1, X2) = P(X1) \cdot P(X2)$
2) $P(X2|X3, X4, X5) = P(X2|X3, X4)$
3) $P(X1|X6) = P(X1)$

Solución:
1) Esta igualdad implica que X1 y X2 son independientes y lo son porque todos los caminos entre los dos nodos están bloqueados:

X1 -> X3 <- X2 bloqueado en X3
X1 -> X3 -> X4 -> X6 <- X2 bloqueado en X6
X1 -> X3 -> X5 <- X4 -> X6 <- X2 bloqueado en X5 y X6

2) Esta igualdad implica que X2 y X5 son condicionalmente independientes dados X3 y X4. El manto de Markov de X5 son precisamente X3 y X4 por lo que X5 es condicionalmente independiente de X2 (también de X1 y X6) dados X3 y X4.
3) Esta igualdad implica que X1 y X6 son marginalmente independientes. En este caso la igualdad no es cierta porque existe el camino X1 -> X3 -> X4 -> X6 que no está bloqueado.

b) **[0.7 puntos]** Calcula $P(\neg X3, X4, X6|X1, \neg X2)$

Solución: $P(\neg X3, X4, X6|X1, \neg X2) = \frac{P(X1, \neg X2, \neg X3, X4, X6)}{P(X1, \neg X2)}$

$$P(X1, \neg X2, \neg X3, X4, X6) = \sum_{X5} P(X1, \neg X2, \neg X3, X4, X5, X6)$$

$$= \sum_{X5} P(X1) \cdot P(\neg X2) \cdot P(\neg X3|X1, \neg X2) \cdot P(X4|\neg X3) \cdot P(X5|\neg X3, X4) \cdot P(X6|\neg X2, X4)$$

$$= (0.8 \cdot 0.9 \cdot 0.3 \cdot 0.7 \cdot 0.3 \cdot 0.4) + (0.8 \cdot 0.9 \cdot 0.3 \cdot 0.7 \cdot 0.7 \cdot 0.4)$$

$$= 0.018144 + 0.042336 = 0.06048$$

Dado que X1 y X2 son independientes $P(X1, \neg X2) = P(X1) \cdot P(\neg X2) = 0.8 \cdot 0.9 = 0.72$

$$P(\neg X3, X4, X6|X1, \neg X2) = \frac{0.06048}{0.72} = 0.084$$

c) **[0.7 puntos]** Utilizando los números aleatorios siguientes: 0.04, 0.54, 0.65, 0.17, 0.32, 0.83, 0.97, 0.58 0.09, 0.25; indica paso a paso cuál sería la muestra generada por el método de muestreo estocástico para calcular la probabilidad $P(X4, \neg X5|\neg X1, X6)$.

Solución: Primero hay que ordenar las variables en orden topológico. Uno posible es X1, X2, X3, X4, X5 y X6.

Con X1, como la probabilidad de *true* es 0.8 y el número aleatorio es 0.04 se queda X1=true.

Con X2, como la probabilidad de *true* es 0.1 y el número aleatorio es 0.54 se queda X2=false.

Con X3, como la probabilidad de *true* (dado X1=true, X2=false) es 0.7 y el número aleatorio es 0.65 se queda X3=true.
Con X4, como la probabilidad de *true* (dado X3=true) es 0.2 y el número aleatorio es 0.17 se queda X4=true.
Con X5, como la probabilidad de *true* (dado X3=true, X4=true) es 0.1 y el número aleatorio es 0.32 se queda X5=false.
Con X6, como la probabilidad de *true* (dado X2=false, X4=true) es 0.4 y el número aleatorio es 0.83 se queda X6=false.
Por tanto, la muestra generada es $\{X1, \neg X2, X3, X4, \neg X5, \neg X6\}$

d) **[0.5 punto]** Para la probabilidad del punto anterior, ¿cuántas muestras de las generadas por el método de muestreo estocástico se descartarían en general?

Solución: Las muestras generadas que no sirven o se descartan para estimar la probabilidad son las que no son compatibles con la evidencia, en el caso anterior $\neg X1, X6$. Por tanto, en generar se descartarían $1 - P(\neg X1, X6) = 0.8779$, o casi el 88% de las muestras.

III.- Aprendizaje

6.- [0.75 puntos] Se utiliza un SVM con margen blando (considerando C=1.5) para predecir si se le concede un préstamo o no a un cliente a partir de sus ingresos anuales, su edad y el importe del préstamo. Resolviendo el problema de optimización se obtienen 3 vectores soporte (sv_1, sv_2, sv_3), sv_1 y sv_2 asociados a la clase *concede préstamo* (+1) y sv_3 asociado a la clase *no concede préstamo* (-1). **Responde razonadamente** si los coeficientes $\alpha_1, \alpha_2, \alpha_3$ pueden tomar los valores indicados en a) o b).

a) $\alpha_1 = 1, \alpha_2 = 1, \alpha_3 = 2$
b) $\alpha_1 = 0.5, \alpha_2 = 1, \alpha_3 = 0.5$

RESP: Las restricciones que deben cumplir los α_i son:

$$(1)\ 0 \le \alpha_i \ \le C$$

$$(2) \sum_{i \in SV} \alpha_i\, y_i = 0$$

La combinación propuesta es a) no cumple (1) y la propuesta en b) no cumple (2).

7.- [0.5 puntos] Considera un problema de clasificación binaria. Para resolverlo disponemos de un conjunto de entrenamiento con 30 ejemplos de la clase positiva y 10 de la clase negativa. Si utilizamos como conjunto de test el conjunto de entrenamiento, **responde razonadamente** cuál será el porcentaje de acierto que obtenemos al utilizar 40-NN.

RESP: Dado que el valor de k es superior al número de ejemplos de entrenamiento, para clasificar cada ejemplo se consideran el resto como vecinos, por lo que la clase mayoritaria entre los k-vecinos siempre será la clase mayoritaria del conjunto de entrenamiento. Así que acertará en 30 de los 40 ejemplos, es decir, en un 75% de las veces.

8.- [0.75 puntos] Si tenemos un conjunto de entrenamiento con 150 ejemplos y realizamos validación cruzada con 10 cajas para entrenar un método de aprendizaje,

a) ¿Cuántos entrenamientos se realizan? 10
b) ¿Con cuántos ejemplos entrenamos el modelo en cada entrenamiento? 15*9=135
c) ¿Cuántos ejemplos utilizamos para validar en cada entrenamiento de la validación cruzada? 15

d) ¿Cuántas veces utilizamos cada ejemplo para entrenar en todo el proceso de validación cruzada? 9

9.- [0.75 puntos] Supongamos que tenemos que construir un árbol de decisión para predecir una variable binaria C. Tenemos 100 ejemplos de entrenamiento, de los cuales 30 son de la clase positiva y 70 de la negativa. Se selecciona el atributo A como raíz, se ramifica el conjunto según los valores de A en dos ramas, dando lugar a un hijo con 18 ejemplos positivos y 22 negativos y otro con 12 ejemplos positivos y 48 negativos.

a) Calcula H(C/A).

H(C/A)=P(A=a1)H(C/A=a1)+ P(A=a2)H(C/A=a2)=

$$(\frac{40}{100}\left(-\frac{18}{40}\log_2\frac{18}{40}-\frac{22}{40}\log_2\frac{22}{40}\right)+\frac{60}{100}\left(-\frac{12}{60}\log_2\frac{48}{60}-\frac{12}{60}\log_2\frac{48}{60}\right)=0.82$$

b) Calcula el error obtenido tras ramificar utilizando el atributo A.

Opciones consideradas correctas:

Error(A=a1)=18/40

Error(A=a2)=12/48

Error =(0.45+0.25)/2=0.35

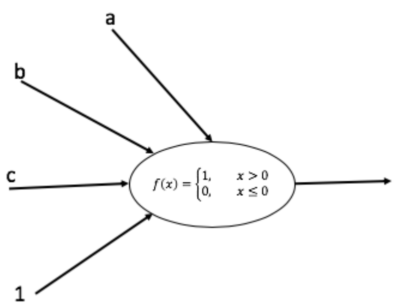
Error=30/100

10.- [0.75 puntos] Para qué sirve la **tasa de aprendizaje** η en una red neuronal?

Escribe una combinación de pesos que permita aprender la función ***(a AND b) OR d*** mediante una red neuronal con las siguientes características.

Nota1: No existe una solución única.

Nota2: **Solo se piden los pesos finales, no que se entrene la red**



Sirve cualquier combinación de pesos tal que $a\omega_a + b\omega_b + c\omega_c + \omega_d \le 0$ cuando a, b y c tomen los valores sombreados en azul en la tabla adjunta y $a\omega_a + b\omega_b + c\omega_c + \omega_d > 0$ para el resto de casos. Por ejemplo $\omega_a = 1, \omega_b = 2,\ \omega_c = 3,\ \omega_d = -2.5$

a	b	c	$(\boldsymbol{a} \wedge \boldsymbol{b}) \vee \boldsymbol{c}$
1	1	1	1
1	1	0	1
1	0	1	1
1	0	0	0
0	1	1	1
0	1	0	0
0	0	1	1
0	0	0	0

$P(x_1, \dots, x_i)=\sum_{x_{i+1},\dots,x_n} P(x_1, \dots, x_i, x_{i+1}, \dots, x_n)= \sum_{x_{i+1}} P(x_1, \dots, x_i, x_{i+1})$		
$P(x \mid y) \equiv P(X = x \mid Y = y) = \frac{P(x,y)}{P(y)}$	$P(X,Y) = P(X \mid Y)P(Y) = P(Y \mid X)P(X)$	
$P(Y)=\sum_{i=1}^{m} P(Y \mid x_i)P(x_i))$	$P(X_1 = x_1, \dots, X_n = x_n) = \prod_{i=1}^{n} P(X_i = x_i \mid X_{i+1} = x_{i+1}, \dots, X_n = x_n)$	
$P(X \mid Y) = P(X)$	$P(X \mid Y, K) = P(X \mid K)$	$P(X \mid Y, K) \neq P(X \mid K)$
$P(A \mid MB(A), B) = P\big(A \mid MB(A)\big)$	$P(x_1, \dots, x_n) = \prod_{i=1}^{n} P\big(X_i = x_i \mid Padres(X_i)\big)$	
	$P(X \mid Padres(X)) = P(X \mid Padres(X), NoDescendientes(X))$	
$E(\boldsymbol{w}) = \frac{1}{2}\sum_{d}\sum_{k}(y_{kd} - o_{kd})^2$	$\delta_k = o_k(1 - o_k)(y_k - o_k)$ $\delta_h = o_h(1 - o_h)\sum_k w_{kh}\delta_k$	$\Delta w_i = \eta(y_d - \boldsymbol{w}\boldsymbol{x}_d)x_{di}$ $g(x) = \frac{1}{1 - e^{-x}}$
$dist(\vec{x}, \vec{y}) = \left(\sum_{i} \lvert x_i - y_i \rvert^p\right)^{1/p}$	$E(\boldsymbol{w}) = \frac{1}{2}(y_d - \boldsymbol{w}\boldsymbol{x}_d)^2$	$g(\cdot) = \begin{cases} 1 \text{ si } \boldsymbol{wx}>0 \\ -1 \text{ en otro caso} \end{cases}$
$H(X) = -\sum_{i=1}^{k} p_i \log_2 p_i$	$H(X \mid Y) = \sum_{y \in Y} p(y)H(X \mid Y = y)$	$K\big(x_i, x_j\big) = \phi(x_i)\phi\big(x_j\big)$ $K(\boldsymbol{x}, \boldsymbol{y}) = (\boldsymbol{xy} + 1)^d$
$b = \frac{1}{\lvert VS \rvert}\sum_{i \in VS} y_i - \boldsymbol{w}\boldsymbol{x_i}$	$\boldsymbol{w} = \sum_{i \in VS} \alpha_i y_i \boldsymbol{x_i}$	$d = \frac{2}{\lVert \boldsymbol{w} \rVert}$