| | |
|---|---|
| APELLIDOS: | PL: |
| NOMBRE: | DNI: |

**ESCUELA DE INGENIERÍA INFORMÁTICA**

**SISTEMAS INTELIGENTES**

**Primer Examen Parcial. 3 de noviembre de 2021.**

---

**I.** Se trata de resolver el problema de asignación de revisores a propuestas de proyectos de investigación (Reviewer Assignment Problem o RAP). En el RAP tenemos un conjunto de **R** revisores y otro conjunto de **P** propuestas de proyectos, cada proyecto debe ser revisado por un revisor, y un revisor no puede revisar más de **N** proyectos. Además, hay una matriz $\mathbf{C_{R\times P}}$ de conflictos; $\mathbf{C_{ij}}$ es el conflicto entre el revisor ***i*** y la propuesta ***j***. El objetivo es minimizar el conflicto total, calculado como la suma de los conflictos entre cada proyecto y el revisor asignado.

En el siguiente ejemplo, con **R**=3 revisores, **P**=4 proyectos, **N** = 2 y la matriz M de conflictos:

| | | | |
|---|---|---|---|
| 2 | 3 | **1** | **2** |
| 2 | 2 | 3 | 4 |
| **1** | **1** | 3 | 1 |

una solución óptima se obtiene asignando las dos primeras propuestas al revisor 3, y las dos siguientes al revisor 1 y ninguna al revisor 2. El coste de la solución es 5 (la suma de los conflictos en negrita). En este ejemplo, si el valor de **N** fuese 1 el problema no tendría solución.

En esta pregunta se pide lo siguiente:

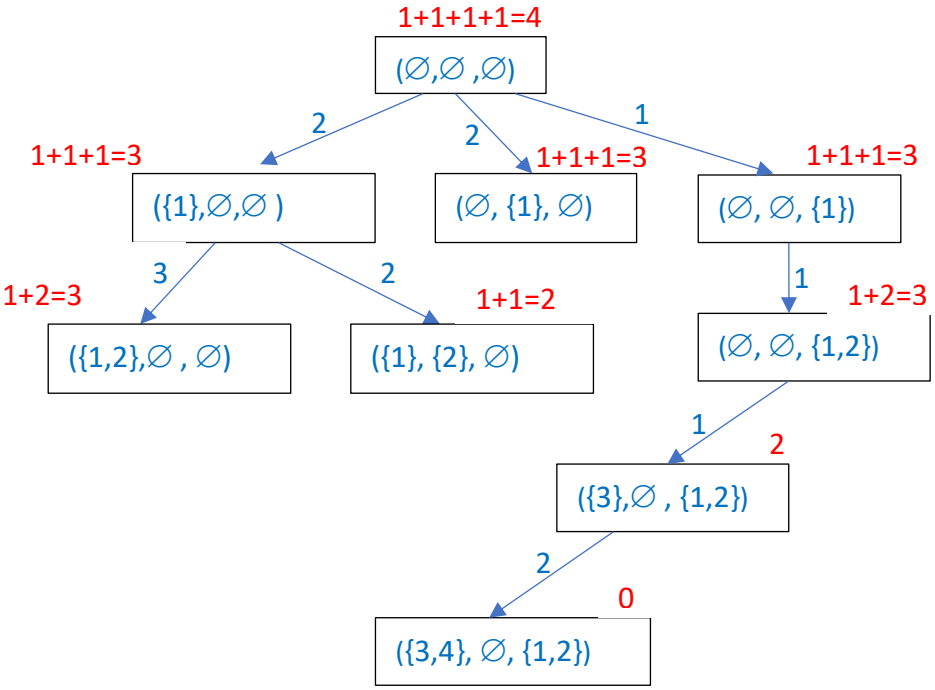
a) **[0,25 puntos]** Si la solución óptima anterior la representamos por (3, 3, 1, 1), dar la representación análoga para otras dos soluciones óptimas de esta instancia.

SOLUCION: (2, 3, 1, 3) y (1, 3, 1, 3)

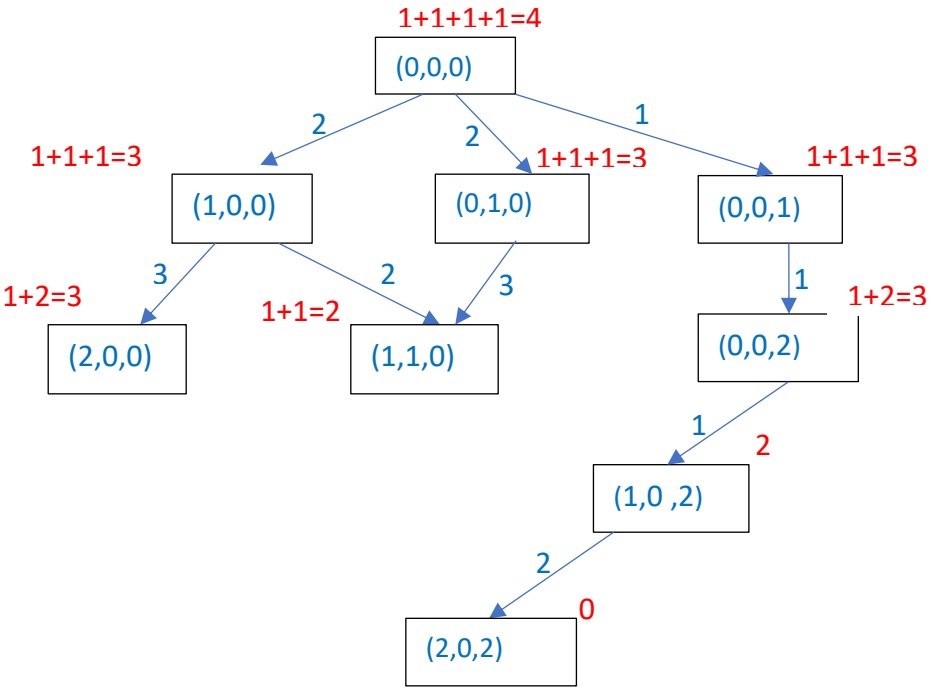
b) Resolver el problema con búsqueda en espacios de estados, concretamente:

b1) **[2,75 puntos]** Describir un espacio de búsqueda adecuado. En primer lugar, dar una descripción general (estados, reglas y costes), y luego dibujar una parte significativa del espacio de búsqueda para la instancia anterior.

SOLUCION: Una forma de representar los estados es mediante una tupla de la forma (C1, …, CR), donde Ci, 1<=i<=**R** es un conjunto de como mucho **N** proyectos asignados al revisor i. Ci ∩Cj = ∅, si i≠j. En el estado inicial Ci=∅, para todo i; y en los objetivos tendremos que C1∪…∪CR contiene los **P** proyectos. El espacio de búsqueda se puede generar asignado los proyectos por orden 1,…,**P** de forma que los sucesores de un estado en el que están asignados los proyectos 1,…,k-1 a revisores, se generan tantos sucesores como posibilidades haya de asignar el proyecto k a un revisor j que tenga menos de N revisores asignados. El coste de la regla correspondiente será el valor del conflicto Mjk. El espacio de búsqueda tiene forma de árbol.
Una parte representativa del espacio de búsqueda para la instancia anterior, en el que aparece representada la solución óptima anterior, es la siguiente:




Este esquema de representación de estados en realidad incluye más información de la estrictamente necesaria ya que cada estado refleja el camino para llegar desde el inicial hasta él. Dado que un estado solamente tiene que representar una situación en la que queda por resolver un subproblema, basta con que el estado represente este subproblema. Así, una representación alternativa puede ser una tupla de la forma (N1,…,N**R**), siendo $0 \leq Ni \leq$ **N** el número de proyectos asignados al revisor i, además la suma N1 +…+ N**R** = K ≤ **P** en cada estado es el número de proyectos asignados, y tomamos el convenio de que sean los K primeros, es decir 1,…,K. Si K=**N**, el estado es un objetivo, y en el estado inicial K=0. Así el subproblema que representa un estado consiste en asignar los proyectos restantes, K+1,…,**P**, a revisores, teniendo en cuenta que cada revisor i puede coger como mucho **N**-Ni de estos K proyectos. El subproblema es independiente de los proyectos concretos que tenga asignados cada revisor (solo importa el número Ni), con lo que el espacio de búsqueda es un árbol. Con esta interpretación, los sucesores de un estado vendrán dados por todas las posibles asignaciones del proyecto K+1 a un revisor i con Ni<**N** proyectos asignados. El coste de la regla correspondiente es Mij, con j=K+1.



b2) **[2,75 puntos]** Definir un heurístico para el algoritmo A* y el espacio anterior, mediante una relajación del problema. Dar los valores de este heurístico para el estado inicial y para alguno de los estados intermedios del espacio de búsqueda dibujado en la pregunta anterior.

SOLUCION: Con las dos interpretaciones anteriores del espacio de búsqueda, los estados representan los mismos subproblemas, por lo tanto, los heurísticos que definamos para uno sirven para el otro. Se puede relajar la restricción de que en un estado n aquellos revisores que tengan menos de **N** elementos, puedan tener cualquier número de elementos (los que ya tienen **N** se mantienen con estos elementos ya que en cualquier solución que encontremos a partir de este estado estos revisores tendrán los proyectos que ya contienen en el estado n). De esa forma el problema relajado se resuelve en tiempo polinomial sin más que asignar cada proyecto al revisor con el que tiene menor conflicto de entre los revisores que tienen menos de **N** proyectos. El coste de esta solución es el valor del heurístico para el estado n.
Los valores de este heurístico para los nodos de los espacios anteriores son los que se muestran en rojo.

c) Resolver el problema mediante un algoritmo genético, concretamente:

c1) **[0,75 puntos]** Proponer un esquema de codificación de cromosomas adecuado e indicar cómo se pueden generar los cromosomas iniciales.

SOLUCION: Una codificación adecuada puede ser una permutación de los elementos de un (multi)conjunto en el que cada uno de los **R** revisores aparece **N** veces. En el caso anterior, el conjunto será {1,1,2,2,3,3} y tres cromosomas válidos son: (1 3 3 2 1 2) (2 1 1 2 3 3) y (3 3 1 1 2 2). Los cromosomas iniciales pueden generarse como permutaciones aleatorias de estos elementos.

c2) **[0,25 puntos]** Describir un operador de cruce adecuado para el esquema anterior.

SOLUCION: Se podría adaptar el operador OX, clásico para la codificación basada en permutaciones, pero para este tipo de codificación puede ir mejor un operador que consiste en seleccionar un conjunto de revisores de un padre y mantener su orden y posición en el hijo, y luego rellenar el resto de posiciones del hijo con el resto de revisores manteniendo su orden relativo en el segundo padre. Por ejemplo, el cruce de los dos primeros cromosomas de c1) si el conjunto de revisores elegidos es {3}, en el primer paso tendríamos: ( - 3 3 - - - ), las posiciones vacías se rellenan con los revisores 1 y 2 manteniendo el orden relativo de estos valores es decir 2 1 1 2 empezando por el principio. Así el hijo resultante sería ( 2 3 3 1 1 2 ).

c3) **[0,25 puntos]** Indicar cómo se obtiene una solución a partir de un cromosoma, y dar una forma de calcular el fitness.

SOLUCION: La decodificación se puede hacer, por ejemplo, asignando los proyectos, en el orden 1,…,**P** a los **P** primeros elementos del cromosoma. Por ejemplo, la decodificación del tercero de los cromosomas anteriores daría lugar a la solución óptima indicada en el enunciado.

II. **[1 punto]** Las dos reglas siguientes simulan la acción de coger con un robot un bloque que está encima de otro y la de depositar un bloque que tiene cogido un robot encima de otro, respectivalemte:

```
(: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)))))
```

Suponiendo que solamente hay un robot disponible en el entorno, se pide diseñar una acción que simule el resultado de las dos acciones anteriores, es decir que mueva un bloque que está encima de una pila de bloques a otra pila.

SOLUCION:

```
(:action unstack-stack
        :parameters (?h – hand ?b ?underb.before ?underb.after  - block)
        :precondition (and (empty ?h)(on ?b ?underb.before) (clear ?b) (clear ?underb.after ))
        :effect (and (not (on ?b ?underb.before)) (not  (clear ?underb.after))
                        (clear ?underb.before) (on ?b ?underb.after)))
```

III. Dada la siguiente base de reglas de producción:

| | |
|---|---|
| R1: H,F,E -> D<br>R2: F,C -> I<br>R3: D,G -> I<br>R4: H -> A<br>R5: F -> E<br>R6: I,A ->B<br>R7: G->F<br>R8: A,G->I<br>R9: A,H ->F | a) **[1 punto]** Aplicando encadenamiento hacia delante, describe el proceso de inferencia resultante suponiendo más prioritaria la regla con más condiciones en su antecedente (en caso de igualdad, se escogerá la regla con menor identificador). La base de hechos inicial es BH: {F, H} y la meta es **I**. ¿Se puede verificar esta meta?<br>b) **[1 punto]** Aplicando encadenamiento hacia atrás, describir el proceso de inferencia resultante suponiendo que la regla más prioritaria es la de identificador mayor, que la meta a obtener es **B** y que la base de hechos inicial es BH: {C, F} . ¿Se puede verificar esta meta? |

**Solución:**

a) Iteración 1: CC={R4,R5}, se selecciona R4 y BH={F,H,A}
Iteración 2: CC={R5,R9}, se selecciona R9 y BH={F,H,A}
Iteración 3: CC={R5}, se selecciona R5 y BH={F,H,A,E}
Iteración 4: CC={R1}, se selecciona R1 y BH={F,H,A,E,D}
Iteración 5: CC es vacío. No se puede continuar porque no quedan más reglas que aplicar. La meta no se puede verificar.

b)
CC={R6}
Para R6 se deben verificar sus antecedentes, I y A
Submeta I: CC={R2,R3,R8}, se escoge R8
Para R8 se debe verificar A y G
Submeta A: CC={R4}
Para R4 hay que verificar H
Submeta H: CC es vacío, no se puede verificar

Para R3 se debe verificar D y G

Submeta D: CC={R1}

Para R1 se debe verificar H y E (F ya está verificado)

Submeta H: CC es vacío, no se puede verificar

Para R2 sus antecedentes ya están en BH, por tanto -> BH={C,F,I}

Submeta A: CC={R4}

Para R4 hay que verificar H

Submeta H: CC es vacío, no se puede verificar

No se puede continuar y por tanto no se puede verificar la meta B.