Técnicas, Entornos y Aplicaciones de Inteligencia Artificial

Evaluación Práctica-3: CSPs - MiniZinc. 2022-2023.

Nombre: _____

1) **Subid a Poliformat todos los ficheros correspondientes a la práctica (habrá distintas versiones: versión original y cada uno de los ejercicios). Se puede subir un archivo .zip.**
2) **Contestad a las preguntas siguientes, rellenando los huecos con las respuestas. Se deben comentar los cambios realizados. Se debe partir de la práctica ya realizada.**
3) **La entrega fuera de plazo tendrá penalización en la nota.**
4) **La utilización de herramientas de mensajería está terminantemente prohibida.**

Tiempo: 75 minutos.

Nota: todos los ejercicios y apartados son incrementales. Es decir, el resultado del ejercicio 1 se utiliza como base para el ejercicio 2, el ejercicio 2 se usa como base para el ejercicio 3, y así sucesivamente.

---

1. **(2 *puntos, Tiempo estimado: 15')*** Utilizando vuestro modelo original, añadir la siguiente nueva información:

- Hay una operación más: O7, con una duración de 80 min.
- Hay dos cirujanos más: C4 y C5.
- Hay dos técnicos más: T6 y T7.

Indica las modificaciones realizadas:

```
int: Cirujanos = 5;
int: Tecnicos = 7;

int: Operaciones = 7;

Duracion = [60, 120, 180, 30, 45, 60, 80]; % Las duraciones de las operaciones
```

Indica qué operaciones realizan los cirujanos C3, C4 y C5 y los técnicos T6 y T7:

Persona	Operaciones
C3	O1
C4	O4
C5	NINGUNA
T6	O3, O6, O7
T7	O3, O4, O5

En el apartado c) original se producen dos cambios. Ahora, la operación O1 la debe realizar, al menos, el cirujano C2, C3 o C4 de forma exclusiva (xor). La operación O3 la debe realizar, al menos, el cirujano C1, C3 o C5 de forma exclusiva (xor).

Indica las modificaciones realizadas:

```
% O1 la debe realizar, al menos, el cirujano C2 o C3 o C4 de forma exclusiva (xor)
constraint (Asignacion[2,1] = 1) xor (Asignacion[3,1] = 1) xor (Asignacion[4,1] = 1);
```

```
% O3 la debe realizar, al menos, el cirujano C1 o C3 o C5 de forma exclusiva (xor)
constraint (Asignacion[1,3] = 1) xor (Asignacion[3,3] = 1) xor (Asignacion[5,3] = 1);
```

Indica qué operaciones realizan los cirujanos C3, C4 y C5 y el técnico T7:

Persona	Operaciones
C3	NINGUNA
C4	O5, O7
C5	O3
T7	NINGUNA

2. ***(3 puntos, Tiempo estimado: 15')*** Disponemos de dos técnicos más (T8 y T9). Se deben contemplar las siguientes restricciones:

- T8 y T9 son técnicos en prácticas, debiendo estar siempre supervisados por T1. Esto significa que si en una operación participa T8, también debe hacerlo T1. Análogamente ocurre con T9.
- Tanto T8 como T9 tienen que participar al menos en una operación.

Indica las modificaciones realizadas:

```
int: Tecnicos = 9;

% T8 y T9 son técnicos en prácticas
% T1 = Cirujanos+Anestesistas+Enfermeros+1
% T8 = Cirujanos+Anestesistas+Enfermeros+Tecnicos-1
% T9 = Cirujanos+Anestesistas+Enfermeros+Tecnicos
% Si en una operación participa T8, también debe hacerlo T1
constraint forall (o in 1..Operaciones)
((Asignacion[Cirujanos+Anestesistas+Enfermeros+Tecnicos-1,o] == 1) ->
(Asignacion[Cirujanos+Anestesistas+Enfermeros+1,o] == 1));

% Si en una operación participa T9, también debe hacerlo T1
constraint forall (o in 1..Operaciones)
((Asignacion[Cirujanos+Anestesistas+Enfermeros+Tecnicos,o] == 1) ->
(Asignacion[Cirujanos+Anestesistas+Enfermeros+1,o] == 1));

% T8 tiene que participar al menos en una operación.
constraint (sum (o in 1..Operaciones)
(Asignacion[Cirujanos+Anestesistas+Enfermeros+Tecnicos-1,o]) >= 1);

% T9 tiene que participar al menos en una operación.
constraint (sum (o in 1..Operaciones)
(Asignacion[Cirujanos+Anestesistas+Enfermeros+Tecnicos,o]) >= 1);
```

Indica qué operaciones realizan los técnicos T1, T8 y T9, así como el cirujano C3:

Persona	Operaciones
T1	O1, O2
T8	O1
T9	O1
C3	NINGUNA

3. ***(2 puntos, Tiempo estimado: 10')*** Los dos técnicos en prácticas no pueden participar en la misma operación. Adicionalmente, T9 no puede participar en la operación O1.

Indica las modificaciones realizadas:

```
% T8 y T9 no pueden participar en la misma operación (son mutuamente excluyentes en cada
operación)
constraint (forall (o in 1..Operaciones) (not
(((Asignacion[Cirujanos+Anestesistas+Enfermeros+Tecnicos-1,o]) == 1) /\
((Asignacion[Cirujanos+Anestesistas+Enfermeros+Tecnicos,o]) == 1))));

% T9 no puede participar en la operación O1
constraint Asignacion[Cirujanos+Anestesistas+Enfermeros+Tecnicos,1] == 0;
```

Indica qué operaciones realizan los técnicos T1, T8 y T9, así como el cirujano C3:

Persona	Operaciones
T1	O1, O2, O3
T8	O1
T9	O2
C3	NINGUNA

4. ***(2 puntos, Tiempo estimado: 15')*** La utilización del técnico que supervisa a los técnicos en prácticas puede incrementar el número de técnicos por operación más de lo necesario (idealmente nos interesaría que hubiera solo dos técnicos por operación). Por tanto, se desea minimizar el número de técnicos en total que se necesitan en el total de operaciones.

NOTA: Si queremos almacenar un sumatorio de variables sobre varios índices basta con usar:
constraint (sum (a in 1..X, b in 1..Y) (matriz[a,b])) = variable_sumatorio;

Indica las modificaciones realizadas:

```
var int: TecnicosNecesarios; % Representa el número de técnicos necesarios

constraint (sum (t in
1+Cirujanos+Anestesistas+Enfermeros..Cirujanos+Anestesistas+Enfermeros+Tecnicos, o in
1..Operaciones) (Asignacion[t,o])) = TecnicosNecesarios;

solve minimize TecnicosNecesarios;
```

Indica qué operaciones realiza cada técnico y el número de técnicos necesarios en total (**NOTA:** se recomienda utilizar el solver "COIN-BC" en lugar de "Gecode"):

Persona	Operaciones
T1	O4, O6
T2	O1
T3	O1
T4	O5, O7
T5	O3
T6	O2, O7
T7	O2, O3, O5

T8	O6
T9	O4

TécnicosNecesarios: 14 (pues hay 7 operaciones)

5. ***(1 punto, Tiempo estimado: 10')*** Supongamos que se desean incluir más operaciones sin modificar la plantilla. Obviamente, llegará un momento en el que no todas las operaciones serán posibles y se desea considerar el beneficio de cada operación para realizar aquellas que ofrezcan mayor beneficio. Explica qué modificaciones serían necesarias para realizar una asignación de operaciones que conduzca al máximo beneficio. **Nota**: no es necesario implementar nada; basta con explicar las variables y restricciones de forma textual.

Habría que añadir una variable: Asignación[o] que será 0/1 en función de si la operación "o" se realiza o no

Habría que añadir un vector de entrada con los beneficios de cada operación

Habrá que añadir una variable BeneficioTotal que queremos maximizar

En todas las restricciones existentes habría que hacer una condición de forma que solo se contemplarían si Asignación[o]==1