Escuela Técnica Superior de Ingeniería Informática y Telecomunicaciones

Práctica 2: Ejercicios Minizinc

Juan Emilio Martínez Manjón
77024428-G
juanemartinez@correo.ugr.es
Grupo 1 Lunes 17:30h

2019-2020

1 Ejercicio 1

Para representar la solución de este ejercicio, se ha usado un array de 10 valores, donde cada posición equivale a una de las letras de la operación TESTE + FESTE + DEINE = KRAFTE. Para darle el valor númerico correcto hemos usado 4 variables y una restricción.

Cada variable se encarga de representar numericamente cada miembro de la operación. Para calcular dicho miembro haremos lo siguiente:
Letra de las unidades + letra de las decenas * 10 + letra de las centenas * 100 + letra de las unidades de millar * 1000.

Finalmente añadiremos la restricción de que la suma de los miembros da como resultado la solución.

```
"T" = 3
"E" = 0
"S" = 8
"F" = 6
"D" = 5
"I" = 9
"N" = 7
"K" = 1
"R" = 4
"A" = 2
```

Esta es la solución que obtenemos al ejecutar el programa, donde vemos que número tiene asignada cada letra. En términos del problema, esto equivaldría a la operación 30830 + 60830 + 50970 que daría 142630, equivalente a la combinación de letras KRAFTE.

2 Ejercicio 2

Para representar la solución de este ejercicio, se ha usado un array de 10 valores, donde cada posición i corresponde con la posición i de nuestro número X.

La única restricción es que el numero de veces que aparezca cada elemento de nuestro array X sea igual al valor en esa posición del array. Tenemos que hacer que sea igual a la posición i+1 porque i empieza en 0, y los indices empiezan a contar a partir de 1.

```
6210001000
```

Esta es la solución de nuestro problema, donde vemos que efectivamente hay 6 ceros, 2 unos, 1 dos y 1 seis.

3 Ejercicio 3

Para representar la solución de este ejercicio se va a utilizar un array que va a contener la asignación en intervalos de tiempo de la solución, es decir cada posición i va a contener el intervalo en que va a impartir clase el profesor i. Si para el profesor i obtenemos el valor 1, quiere decir que impartirá clase de las 9:00 a las 10:00.

Vamos a utilizar un vector de restricciones, donde cada indice tendrá un dominio diferente, dependiendo de las horas disponibles del profesor relacionado con dicho índice.
Si ponemos que los valores que puede adoptar el índice 1 son 3..6, quiere decir que el profesor 1 esta disponible de las 11:00 a las 15:00.

La clave de este ejercicio es que le asignemos a cada posición i de la asignación solución, la correspondiente posición i de restricciones (para asegurarnos de que ese profesor obtenga un valor dentro de su horario disponible). Y sobre todo que hagamos que todos los valores de la asignación sean diferentes entre ellos. Ya que dos profesores no pueden dar clase en la misma franja horaria.

```
6
4
5
2
3
1
```

Esta sería una solución a nuestro problema, donde el profesor 1 daría clase de 14:00 a 15:00, el profesor 2 de 12:00 a 13:00, y así sucesivamente.

4 Ejercicio 4

Para codificar este ejercicio, he decidido dividir cada franja horaria (9:00-10:00, 10:00-11:00...) en una matriz que contenga lo siguiente por cada fila:

-1º Un numero correspondiente al profesor que imparte la clase (entre 1 y 4).

-2º La asignatura que imparte el profesor en ese momento (entre 1 y 3). Si aparece el valor 5 es que en esa franja horaria, el profesor no va a dar clase.

-3º El grupo al que el profesor va a impartir la asignatura (entre 1 y 4).

-4º El aula donde el profesor va a impartir la clase (entre 1 y 4).

Cada matriz tendrá de nombre el límite inferior de la franja horaria. Cuando pone NueveAM se refiere a la franja de las 9:00 a las 10:00. Las dos primeras matrices solo tienen 3 filas en vez de 4, porque en la tabla que se adjunta en el enunciado, nos dicen que de 9:00 a 10:00 el profesor 4 no puede dar clase. Y de 10:00 a 11:00 no puede dar clase el profesor 2.

Ahora para cada matriz, añadiremos las restricciones necesarias para que los valores de profesores, asignaturas y grupos se adapten a lo explicado en el enunciado. Incluidas las relaciones entre dichos valores.

Cuando tengamos todas las restricciones de todas las matrices que actúan en nuestro problema, nos centramos en escribir las restricciones entre matrices.

Finalmente, a nosotros nos interesa que se maximice el numero de clases que se impartan a lo largo del día. Y por lo tanto, que se minimice el numero de veces que el profesor no de clase. Es por esto que vamos a minimizar el numero de veces que aparezca el 5 como asignatura a lo largo del día. Para ello crearemos un array para cada franja horaria, y cada vez que nos encontremos un 5 pondremos un 1 en esa posición del array, en caso contrario pondremos un 0. Resolvemos el problema para que la aparición de 5 sea mínima entre todas las matrices.

```
De las 9:00 a las 10:00
1 2 2 3
2 5 1 2
3 3 4 1
-------------------------
De las 10:00 a las 11:00
1 2 1 3
3 3 3 2
4 1 4 1
```

(a)

```
De las 11:00 a las 12:00
1 1 1 4
2 3 2 3
3 2 3 2
4 5 4 1
-------------------------
De las 12:00 a las 13:00
1 1 2 4
2 3 1 3
3 2 4 2
4 1 3 1
```

(b)

Si observamos los valores del resultado anterior, veremos que todas las restricciones se cumplen.

5 Ejercicio 5

Para codificar este ejercicio, he decidido dividir cada dia (Lunes,Martes...) en una matriz que contenga lo siguiente por cada fila:

-1º Un numero correspondiente a la franja horaria (entre 1 y 6). Donde 1 corresponde a 8:00-9:00, 2 corresponde a 9:00-10:00...

-2º El profesor que esta impartiendo clase en ese momento (entre 1 y 4). En la 4ª franja siempre aparece 9, ese numero simboliza el recreo, ya que está fuera del dominio de profesores.

-3º La asignatura que se esta impartiendo (entre 1 y 9). En el caso del recreo aparecerá siempre la asignatura 0, que está fuera del dominio.

Ahora para cada matriz, aplicaremos las restricciones necesarias para que los valores de profesores y asignaturas se adapten a lo explicado en el enunciado del ejercicio.

Finalmente vamos a restringir que la suma de horas de cada asignatura a lo largo de la semana, sea igual al valor que nos piden en la tabla del enunciado. Para conseguir esto tenemos que crear restricciones que actúen sobre todas las matrices al mismo tiempo.

```
Lunes
1 1 3
2 1 3
3 3 6
4 9 0
5 2 4
6 2 4
```

Figure 1

```
Martes
1 3 6
2 4 8
3 4 8
4 9 0
5 2 5
6 2 5
```

Figure 2

```
Miercoles
1 4 7
2 2 5
3 2 5
4 9 0
5 1 3
6 1 3
```

Figure 3

```
Jueves
1 3 9
2 4 7
3 4 2
4 9 0
5 1 1
6 1 1
```

Figure 4

```
Viernes
1 2 4
2 2 4
3 4 2
4 9 0
5 1 1
6 1 1
```

Figure 5

Si observamos los valores del resultado anterior, veremos que todas las restricciones se cumplen.

6 Ejercicio 6

Este ejercicio lo resolveremos tratando la combinación de varios arrays, como una matriz. Donde cada array simboliza una de las categorías a representar.

Y cada una de las asignaciones finales corresponde a la lectura de una "columna" de dicha "matriz".

El objetivo de este ejercicio es restringir los valores de los arrays dependiendo de lo que nos digan en cada apartado del enunciado.

Finalmente hacemos que todos los valores sean diferentes entre ellos, ya que cada asignación es única.

```
Regiones:     1 2 3 4 5
Profesiones: 2 4 1 5 3
Color Casas: 1 3 2 5 4
Bebidas:      3 4 2 1 5
Animales:     3 2 1 5 4
Numero Casa: 3 4 5 2 1
```

Observando el resultado y teniendo delante lo que significa cada número, podemos llegar a la conclusión de que:

La cebra es el animal 1. Por lo que podemos decir que está en la casa 5 con el pintor Gallego.

El agua es la bebida 5. Por lo que podemos decir que el que bebe agua es el diplomático Andaluz.

7 Ejercicio 7

La solución del problema la meteremos en un array "asignaciones". En este array, pondremos en cada posición i, el instante de tiempo en que comienza la tarea i. Ejemplo -> Si i == 0, tarea == A.

Luego declararemos una matriz, donde indicaremos cuáles son las tareas predecesoras de cada una de las tareas.

En el array de tiempos totales, pondremos el instante de tiempo en que terminaría cada tarea. La tarea A terminaría en el instante 7 ya que no tiene predecesores y dura 7 dias.

Para cada tarea, le asignamos su tiempo de fin, como el tiempo máximo de fin de sus predecesores más el tiempo que tarda dicha tarea. Ahora, simplemente, para saber el tiempo en que comienza, le restamos a los tiempos de fin de cada tarea, su tiempo de duración. Así nos quedamos solamente con el tiempo de inicio, el cual meteremos en el array de asignaciones.

Lo que queremos es que la casa se construya en el menor tiempo posible, por lo que vamos a minimizar el tiempo de fin de la última tarea que se realice.

```
[0 - 7]
[7 - 10]
[10 - 11]
[7 - 15]
[15 - 17]
[15 - 16]
[15 - 16]
[7 - 10]
[16 - 18]
```

Podemos ver que la casa va a tardar como mínimo 18 días en construirse usando los valores de la tabla del enunciado.

8 Ejercicio 8

Este ejercicio se resuelve exactamente igual que el anterior, con la diferencia de que a algunas de las tareas les añadiremos nuevos predecesores dependiendo de los trabajadores asignados a cada una. Las nuevas asignaciones de predecesores se atienen a lo siguiente:

Los predecesores de la tarea A se mantienen, ya que no se ejecuta al mismo tiempo que ninguna otra y le podemos asignar perfectamente 2 de los 3 trabajadores de los que disponemos.

Los predecesores de la tarea B serian la A,D,H. La primera porque así lo indica la tabla y los otros dos por lo siguiente:
- Como la tarea B necesita 3 trabajadores
- La tarea D necesita 2 trabajadores
- La tarea H necesita 1 trabajador
- Las tres tareas comienzan a la vez por lo que le daremos prioridad a que comiencen a la vez la D y H (porque la suma de trabajadores necesarios seria 3). Y ya cuando terminen esas dos dejaremos que comience la B, que necesita la totalidad de trabajadores para realizarse.

Los predecesores de la tarea C se mantienen, ya que no se ejecuta al mismo tiempo que ninguna otra y le podemos asignar perfectamente 2 de los 3 trabajadores de los que disponemos.

Los predecesores de la tarea D se mantienen, ya que comienza al mismo tiempo que la B y la H, pero la eligiremos para que se ejecute al mismo tiempo que la H, debido a su numero de trabajadores necesarios. Por lo que solo tendrá que esperar al predecesor de la tabla.

Los predecesores de la tarea E se mantienen, ya que comienza al mismo tiempo que la F y la G, pero la eligiremos para que se ejecute al mismo tiempo que la F, debido a su numero de trabajadores necesarios. Por lo que solo tendrá que esperar a los predecesores de la tabla.

Los predecesores de la tarea F se mantienen, ya que comienza al mismo tiempo que la E y la G, pero la eligiremos para que se ejecute al mismo tiempo que la E, debido a su numero de trabajadores necesarios. Por lo que solo tendrá que esperar a los predecesores de la tabla.

Los predecesores de la tarea G serian la C,D,F. La primera porque así lo indica la tabla y los otros dos por lo siguiente:

- Como la tarea G necesita 1 trabajador
- La tarea F necesita 2 trabajadores
- Las tareas G,E,F comienzan a la vez por lo que le daremos prioridad a que comiencen a la vez la E y F (porque la suma de trabajadores necesarios seria 3). Y ya cuando termine la F (que es la que dura menos) dejaremos que comience la G, que necesita uno de los trabajadores que se han quedado libres.

Los predecesores de la tarea H se mantienen, ya que comienza al mismo tiempo que la B y la D, pero la eligiremos para que se ejecute al mismo tiempo que la D, debido a su numero de trabajadores necesarios. Por lo que solo tendrá que esperar a los predecesores de la tabla.

Los predecesores de la tarea I se mantienen, ya que no se ejecuta al mismo tiempo que ninguna otra y le podemos asignar perfectamente 2 de los 3 trabajadores de los que disponemos.

```
[0 - 7]
[15 - 18]
[18 - 19]
[7 - 15]
[19 - 21]
[19 - 20]
[20 - 21]
[7 - 10]
[20 - 22]
```

Podemos ver que la casa va a tardar como mínimo 22 días en construirse usando los valores de la tabla del enunciado.

9 Ejercicio 9

En este ejercicio mantendremos la estructura de asignaciones y predecesores del ejercicio 7. Además crearemos un array de tiempos donde le asignaremos a cada tarea uno de los tiempos de los que disponemos en la tabla de tiempos del enunciado.

El tiempo de cada tarea tiene que ser uno de los tres que vienen en la tabla. Y tenemos que asegurarnos de que el trabajador que elijamos sea diferente al trabajador de las que empiecen al mismo tiempo que dicha tarea. Las que empiezan al mismo tiempo son las que tienen los mismos predecesores. Esto lo hacemos comprobando que los indices sean diferentes.

Lo que queremos es que la casa se construya en el menor tiempo posible, por lo que vamos a minimizar el tiempo de fin de la ultima tarea que se realice.

```
[0 - 4]
[4 - 7]
[7 - 8]
[4 - 9]
[9 - 11]
[9 - 10]
[9 - 10]
[4 - 9]
[10 - 12]
```

Podemos ver que la casa va a tardar como mínimo 12 días en construirse usando los valores de la tabla del enunciado.

10 Ejercicio 10

Usaremos tres arrays para resolver este ejercicio: Uno para sacar la solución, otro para almacenar los pesos que nos vamos a quedar, y otro para almacenar las preferencias que nos quedamos.

Las única restricción que vamos a tomar es que la suma de pesos sea menor o igual a 275. Finalmente resolveremos el ejercicio indicando que maximice la suma de preferencias.

```
1 1 1 1 1 0 0 0 1 0 1 0
Peso alcanzado: 705
274
```

Los objetos que nos vamos a llevar son: Mapa, compás, agua, sandwich, azúcar, queso y protector solar.