PRÁCTICA 2 – TÉCNICAS DE LOS SISTEMAS INTELIGENTES

SATISFACCIÓN DE SOLUCIONES CON MINIZINC

Antonio José Blánquez Pérez 3ºCSI
Grupo 2 – Viernes

EJERCICIO 1

- Puzzle Cripto-aritmético. El siguiente problema plantea un problema criptoaritmético, de forma que cada letra codifica un único dígito (es decir, un número entero en [0,9]) y cada dígito está asignado a una única letra. Se pide encontrar una asignación de dígitos a letras que satisfaga la siguiente suma (una posible traducción del alemán es "Prueba a fondo tus fortalezas"):

```
  TESTE
+ FESTE
+ DEINE
========
KRAFTE
```

Para resolver este ejercicio tan solo se ha planteado cada letra como una variable de 0 a 9, es decir números de una sola cifra.

Las restricción es simple: la suma de cada conjunto de letras debe ser igual a KRAFTE pasando cada cojunto de letras a decimal multiplicando cada letra según su posición por 10,100,1000... Por ejemplo ETE sería E+T*10+E*100. El resultado obtenido es el siguiente y se puede comprobar que es factible:

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

EJERCICIO 2

- Encontrar un número X de 10 dígitos que satisfaga que el primer dígito de X representa el número de 0s en X, el segundo dígito de X representa el número de 1s en X, etc…

Este ejercicio también es muy simple si utilizamos la función forall de globals. Para representar X tan solo es necesario un array de 0..9, y para que represente lo dicho utilizamos la función forall de tal manera que cada posición i del array(que representa cada dígito i de X) sea igual a las apariciones de i en X, lo cual se implementa a través de la función count.

Obtenemos la solución que se plantea como ejemplo en el enunciado, lo cual añadido a que es fácil de comprobar nos confirma que se ha obtenido una solución factible.

EJERCICIO 3

- Encontrar una asignación de horarios que satisfaga las siguientes condiciones:
a. Existe una única aula, disponible entre las 9:00 y las 15:00.
b. El aula sólo puede estar ocupada por un único profesor/a al mismo tiempo.
c. Cada profesor/a debe impartir una clase de 1h de duración.
d. Cada profesor/a tiene las siguientes restricciones de horarios:

Profesor	Horario disponible
Prof-1	11:00 – 15:00
Prof-2	11:00 – 13:00
Prof-3	10:00 – 14:00
Prof-4	10:00 – 13:00
Prof-5	11:00 – 13:00
Prof-6	09:00 – 15:00

Para este ejercicio se ha decidido representar la información a través de variables de tal manera que cada variable represente a un profesor y obtenga valores entre la hora a la que empieza su horario y una hora antes de la hora a la que acaba, por lo que esta variable representará la hora a la que empieza a dar clase, por ejemplo Prof-1 tendría una variable entre 11 y 14(con lo que se cumple la condición d).

Al ser valores enteros y ocupar las clases una hora se cumplen implícitamente las condiciones a y c en la representación del problema, además si no se repite ningún valor(condicion b) obtendremos una asignación factible, por lo que tan solo se necesita usar la función alldifferent de globals utilizando como entrada un vector con las 6 variables de los profesores.

Obtenemos una solución factible en la que todos los profesores tienen una hora de clase distinta y dentro de su horario disponible, por orden: Prof-6, Prof-4, Prof-5, Prof-2, Prof-3 y Prof-1.

EJERCICIO 4

- Encontrar una asignación de horarios que satisfaga las siguientes condiciones:
a. Existen cuatro aulas, disponibles entre las 9:00 y las 13:00.
b. Cada aula sólo puede estar ocupada por un único profesor/a al mismo tiempo.
c. Existen 3 asignaturas (IA, TSI y FBD), que se imparten en periodos de 1h semanal.
d. Los estudiantes se dividen en 4 grupos (G1, G2, G3 y G4).
e. Cada grupo recibe docencia de una única asignatura en cada momento.
f. Cada profesor da docencia a los grupos definidos en la siguiente tabla.
g. Cada profesor imparte docencia de un único grupo/asignatura en cada momento.
h. Cada profesor tiene las restricciones de horarios definidas en la siguiente tabla.

Profesor	Asignaturas	Horario restringido
Prof1	IA-G1, IA-G2, TSI-G1, TSI-G2	Ninguno
Prof2	FBD-G1, FBD-G2	10:00-11:00
Prof3	TSI-G3, TSI-G4, FBD-G3, FBD-G4	Ninguno
Prof4	IA-G3, IA-G4	09:00-10:00

Para la realización de este ejercicio se ha optado por codificar cada asignatura-grupo con un número de la siguiente manera:

	G1	G2	G3	G4
IA	1	2	3	4
TSI	5	6	7	8
FBD	9	10	11	12

De tal manera que, representando la solución como una matriz donde cada fila sea una hora(entre 9 y 12) y cada columna un aula(entre 1 y 4), ya cumplimos las dos primeras condiciones y, si nos aseguramos de que no se repita ningún número en la matriz, también la tercera. Al ser mayor el número de horas*aulas que el de las asignaturas-grupos a repartir, se ha rellenado la información restante con datos de manera que no afecte a los datos originales.

Ahora, para el resto de condiciones debemos de controlar dos cosas: que ningún profesor esté dando clase a más de un grupo al mismo tiempo y que ningún grupo tendo más de una clase a la vez. Ambas se solucionan de la misma manera, no se deben repetir ni profesores ni grupos en las filas, para lo cual han creado dos vectores, uno para el que en cada posición i está el número del profesor que da la asignatura-grupo i, y el otro el equivalente para el grupo de la asignatura-grupo. Así sencillamente con dos restricciones, una forall(filas)(all_different(profesores[columnas])) y la otra cambiando grupos por profesores tenemos resuelto el problema.

Para la última condición basta con añadir restricciones para que el profesor 2 no pueda dar clase a las 10 y que el 4 tampoco puede a las 9.

Tras la ejecución se ha obtenido la siguiente tabla:

	AULA 1	AULA 2	AULA 3	AULA 4
09:00 – 10:00		TSI-G3 (P3)	FBD-G2 (P2)	TSI-G1 (P1)
10:00 – 11:00		TSI-G4(P3)	IA-G3 (P4)	IA-G1 (P1)
11:00 – 12:00			FBD-G4 (P3)	TSI-G2(P1)
12:00 – 13:00	IA-G4 (P4)	FBD-G3(P3)	IA-G2 (P1)	FBD-G1(P2)

En ella se puede observar que se cumplen todas las restricciones antes planteadas, por lo que hemos obtenido una solución factible al problema planteado.

EJERCICIO 5

- Encontrar una asignación de horarios que satisfaga las siguientes condiciones:
a. Existen un aula, disponible en seis franjas consecutivas de 1h (por ejemplo, de 8:00 a 14:00) de lunes a viernes.
b. Existen nueve asignaturas (A1..A9). El número de horas semanales de cada asignatura se detalla en la siguiente tabla:

A1	A2	A3	A4	A5	A6	A7	A8	A9
4 hrs.	2 hrs	4hrs	4hrs	4hrs	2hrs	2hrs	2hrs	1hr

c. Las asignaturas {A1,A3,A4,A5,A8} deben impartirse en bloques de 2h consecutivas, mientras que el resto, es decir {A2,A6,A7,A9}, se imparten en bloques de 1h.

d. En cada día de la semana sólo se puede impartir, como máximo, UN bloque de cada asignatura.
e. El profesor/a de cada asignatura es el siguiente: Prof1={A1,A3}; Prof2={A4,A5}; Prof3={A6,A9}; Prof4={A2,A7,A8}.
f. Cada profesor sólo puede impartir un bloque de alguna de sus asignaturas cada día, excepto Prof4 (quien puede impartir más de una).
g. La cuarta franja horaria debe reservarse para el recreo; es decir, no asignar ninguna asignatura.

En este ejercicio se va a usar una representación de la solución igual a la del ejercicio anterior, una matriz en la que las filas serán las horas de inicio de las clases(8-13) y las columnas los dias de la semana. Las asignaturas se codificarán del 1 al 9, siendo 10 el recreo y los números i entre 10 y 19 la segunda hora de la asignatura i(estos últimos para facilitar las siguiente restricciones).

Para controlar los profesores de cada asignatura se utiliza un vector como en el ejercicio anterior(siendo solo relevantes los profesores 1, 2 y 3, ya que el 4 no tiene restricciones), también las horas a la semana que se da cada asignatura y la longitud de cada bloque de cada una de estas se controlarán con vectores.

Ya representada toda la información necesario pasemos a explicar las restricciones: las condiciones a y e se cumplen implícitamente en la representación de la informacion, para la b haremos un recuento de las 9 asignaturas en toda la matriz y restringiremos que sea igual al número de horas dividido entre el número de bloques(ya que la segunda hora será el mismo número más 10) , para la c diremos que para cada posición de la matriz que tenga un valor para una asignatura con bloques de 2 horas la posición inmediatamente inferior sea este valor más 10, siendo necesario especificar que la hora anterior al recreo(10) y la última hora(13) deben ser asignaturas con bloque de una hora(como es lógico). Las últimas 3 condiciones son simples, haremos un all_different para las columnas de manera que no se repitan ni las asignaturas ni los profesores(haciendo uso del vector anteriormente mencionado) para las dos primeras, no teniendo que controlar las asignaturas de dos horas ya que ya les hemos impuesto valores diferentes a las dos horas, y para la última tan solo debemos especificar que la fila 11 sea igual a 10(recreo).

Podemos ver como el resultado es satisfactorio en la siguiente tabla y como se cumplen las condiciones a la perfección.

<table>
<tr><th></th><th>LUNES</th><th>MARTES</th><th>MIÉRCOLES</th><th>JUEVES</th><th>VIERNES</th></tr>
<tr><th>08:00 – 9:00</th><td rowspan="2">A3</td><td rowspan="2">A3</td><td>A6</td><td>A6</td><td>A2</td></tr>
<tr><th>09:00 – 10:00</th><td>A2</td><td rowspan="2">A1</td><td rowspan="2">A1</td></tr>
<tr><th>10:00 – 11:00</th><td>A9</td><td>A7</td><td>A7</td></tr>
<tr><th>11:00 – 12:00</th><td colspan="5">RECREO</td></tr>
<tr><th>12:00 – 13:00</th><td rowspan="2">A5</td><td rowspan="2">A8</td><td rowspan="2">A5</td><td rowspan="2">A4</td><td rowspan="2">A4</td></tr>
<tr><th>13:00 – 14:00</th></tr>
</table>

EJERCICIO 6

- Cinco personas de cinco regiones diferentes viven en las primeras cinco casas contiguas de una calle. Practican cinco profesiones distintas, y cada uno tiene un animal y una bebida favoritos, todos ello diferentes. Las casas están pintadas con diferentes colores. Ademas sabemos lo siguiente:
a. El vasco vive en la casa roja.
b. El catalán tiene un perro.
c. El gallego es un pintor.
d. El navarro bebe te.
e. El andaluz vive en la primera casa de la izquierda.
f. El de la casa verde bebe café.
g. La casa verde está al lado de la blanca y a su derecha.
h. El escultor cría caracoles.
i. El diplomático vive en la casa amarilla.
j. En la casa central se bebe leche.
k. La casa del andaluz está al lado de la azul.
l. El violinista bebe zumo.
m. El zorro está en una casa al lado de la del médico.
n. El caballo está en una casa al lado de la del diplomático.
Resolver el problema de forma que podamos responder: ¿dónde está la cebra y quién bebe agua?

Se ha establecido la siguiente codificación para la información:

	PERSONA	CASA	ANIMAL	TRABAJO	BEBIDA
1	Vasco	Roja	Perro	Pintor	Té
2	Catalán	Verde	Caracol	Escultor	Café
3	Gallego	Blanca	Zorro	Diplomático	Leche
4	Navarro	Amarilla	Caballo	Violinista	Zumo
5	Andaluz	Azul	Cebra	Médico	Agua

Todos los valores anteriores se han introducido como constantes y, para buscar la solución final, se ha creado un array por campo(persona, casa, animal, trabajo y bebida) con valores entre 1 y 5.

La primera restricción es clara, y es que no pueden repetirse número en cada array, ya que, por ejemplo, el vasco no puede vivir a la vez en la casa roja y la casa verde a la vez.

Con esto ya obtenemos una codificación con la que abordar el problema, ya que basta con añadir las rectricciones para que las posiciones de cada array sean iguales según los criterios que nos pide. Por ejemplo, el vasco vive en la casa roja, persona[vasco(1)] = casa[roja(1)]. Posteriormente bastará con comparar los valores de los vectores para saber qué enlaza con qué, además, de paso, cada número representará las posiciones de las casas(necesario para las restricciones del tipo: x está al lado de y), representando el 1 todos los hechos de la casa más a la izquierda y 5 todos los hechos de la casa más a la derecha.

Ahora, para responder a la pregunta, habrá que interpretar la solución obtenida, que en este caso es la siguiente:

Personas: [3, 4, 5, 2, 1]
Casas: [3, 5, 4, 1, 2]
Animales: [4, 3, 1, 2, 5]
Trabajos: [5, 3, 1, 4, 2]
Bebidas: [2, 5, 3, 4, 1]

De esto podemos sacar la siguiente información:

(1) La casa de más a la izquierda es de color amarillo y en ella vive el diplomático andaluz, que tiene un zorro y bebe agua.
(2) La casa a la izquierda de la central es de color azul y en ella vive el médico navarro, que tiene un caballo y bebe té.
(3) La casa central es de color rojo y en ella vive el escultor vasco, que tiene caracoles y bebe leche.
(4) La casa a la derecha de la central es de color blanco y en ella vive el violinista catalán, que tiene un perro y bebe zumo.
(5) La casa de más a la derecha es de color verde y en ella vive el pintor gallego, que tiene una cebra y bebe café.

Como vemos, se cumplen todas las restricciones y no se repiten elementos. Hecho todo esto ya se puede responder que la cebra está en la casa verde y que quien bebe agua es el andaluz.

EJERCICIO 7

- En la tabla adjunta aparece la información necesaria para llevar a cabo la construcción de una casa. En la primera columna aparecen los indentificadores de las tareas necesarias para construirla. En la segunda columna aparece la descripción de cada una de estas. En la tercera columna se muestra la duración (en días) de cada una de las tareas. Y la cuarta columna muestra la relación de precedencia entre tareas: por ejemplo, si el contenido de la celda correspondiente a la tarea "F" es "C,D" se debe interpretar como que "las tareas C y D deben finalizarse antes que comience la tarea "F". Se pide encontrar una asignación de tiempos de inicio a estas tareas de forma que se pueda construir la casa en el menor tiempo posible, asumiendo que cada tarea la realiza un único trabajador y que se disponen de tantos trabajadores como se necesiten.
Nota: Este problema tiene una parte de satisfacción (respetar la precedencia de las tareas) y una parte de optimización (minimizar la duración total). Para resolverlo se recomienda comenzar por la parte de satisfacción y guardar en una variable el "coste" de la solución (es decir, el tiempo requerido para terminar todas las tareas), y tras esto resolver la parte de optimización usando la sentencia: solve minimize <variable>

Tarea	**Descripción**	**Duración**	**Predecesoras**
A	**Levantar muros**	**7**	**Ninguna**
B	**Carpintería de tejado**	**3**	**A**
C	**Poner tejado**	**1**	**B**
D	**Instalación eléctrica**	**8**	**A**
E	**Pintado fachada**	**2**	**C,D**
F	**Ventanas**	**1**	**C,D**
G	**Jardín**	**1**	**C,D**
H	**Techado**	**3**	**A**
I	**Pintado interior**	**2**	**F,H**

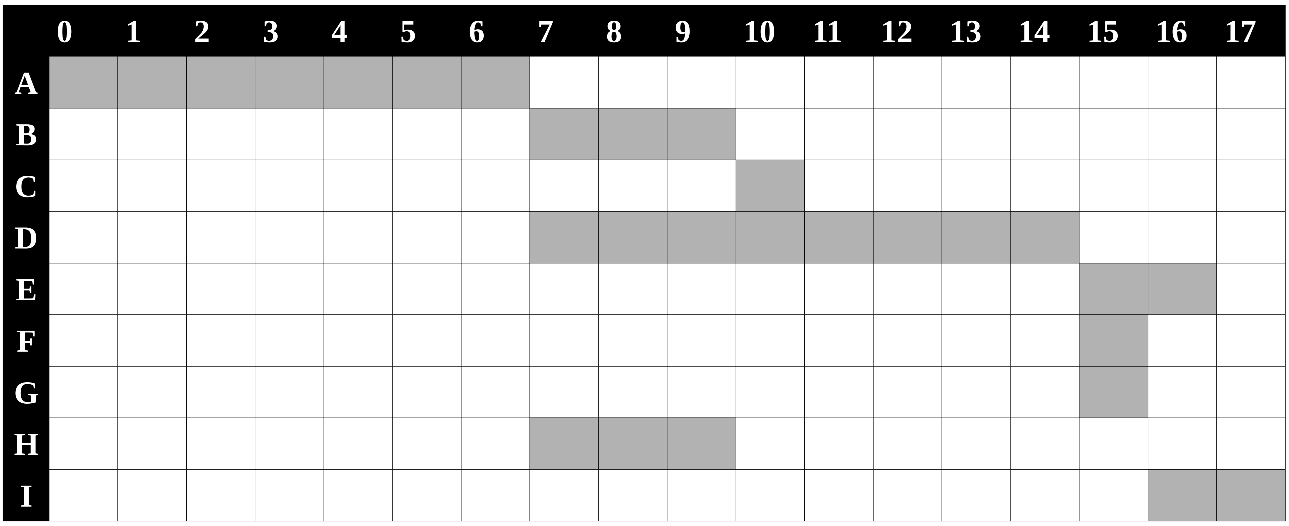
En este ejercicio se ha representado la entrada de datos de la siguiente manera: cada letra tendrá asignada un número, empezando A con 1 hasta I con 9, y las duraciones estarán en un array donde la posición i tenga el valor de la duración de la tarea i. De esta manera, podremos crear otros dos vectores, inicio y fin, donde se establecerá las restricción de que fin – inicio = duración de manera sencilla.

Al no tener restricciones de paralelismo ya que tenemos todos los trabajadores que sean necesarios en cada momento, tan solo es necesario marcar como 0 el inicio de la tarea A que es la única que no tiene predecesoras y restringir que el inicio de cada tarea sea posterior a cada fin de sus tareas predecesoras. Recalcar que es necesario establecer que el instante de inicio sea mayor o igual que el final y no mayor estricto ya que, como se puede ver en la tabla donde se ha representado la respuesta, si una tarea acaba en el instante 7 la siguiente tarea puede comenzar en ese mismo instante y no tiene que esperar al 8; de otra manera se perdería una unidad de tiempo.

A partir de ahí solo es necesario minimizar el tiempo en el que se realizan todas las tareas, lo cual se puede hacer con los datos que ya hemos establecido, ya que basta con minimizar el mayor valor del vector fin que se puede pasar a Minizinc con la orden solve minimize(max(fin)).

Minizinc devuelve el siguiente resultado que paso a representar gráficamente.

T inicio: [0, 7, 10, 7, 15, 15, 15, 7, 16]
Tiempo total: 18



En la solución se puede ver que se cumplen las restricciones y el tiempo parece razonable que sea el mímimo posible, por lo que daremos como satisfactible y mínima la solución.

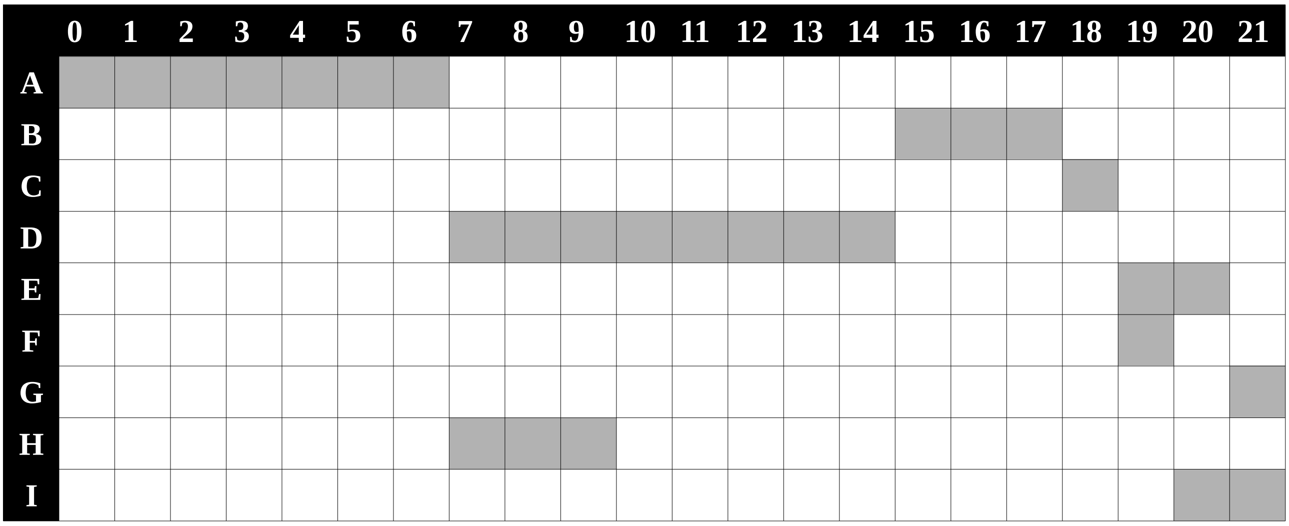
EJERCICIO 8

- Resolver el ejercicio anterior suponiendo que disponemos de tres trabajadores (que en cada momento sólo puede estar dedicados a una única tarea) y que las tareas requieren los siguientes números de trabajadores:

A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**
2	**3**	**2**	**2**	**1**	**2**	**1**	**1**	**2**

Este ejercicio se resuelve exactamente igual que el anterior, añadiendo solo la condición de que tenemos solo 3 trabajadores y cada tarea requiere los trabajadores mecionados en la tabla. Para estos casos existe una restricción global llamada cumulative que, pasandole los vectores de inicio y duración usado en el ejercicio anterior, además de un vector con el número de trabajadores que necesita cada tarea y el número máximo de trabajadores de los que disponemos, hace exactamente lo que necesitamos. No obstante es necesario añadir un límite a los valores de inicio y fin para que no se desborden al usar esta función.

El resultado obtenido, que se puede observar que cumple las restricciones, es el siguiente:



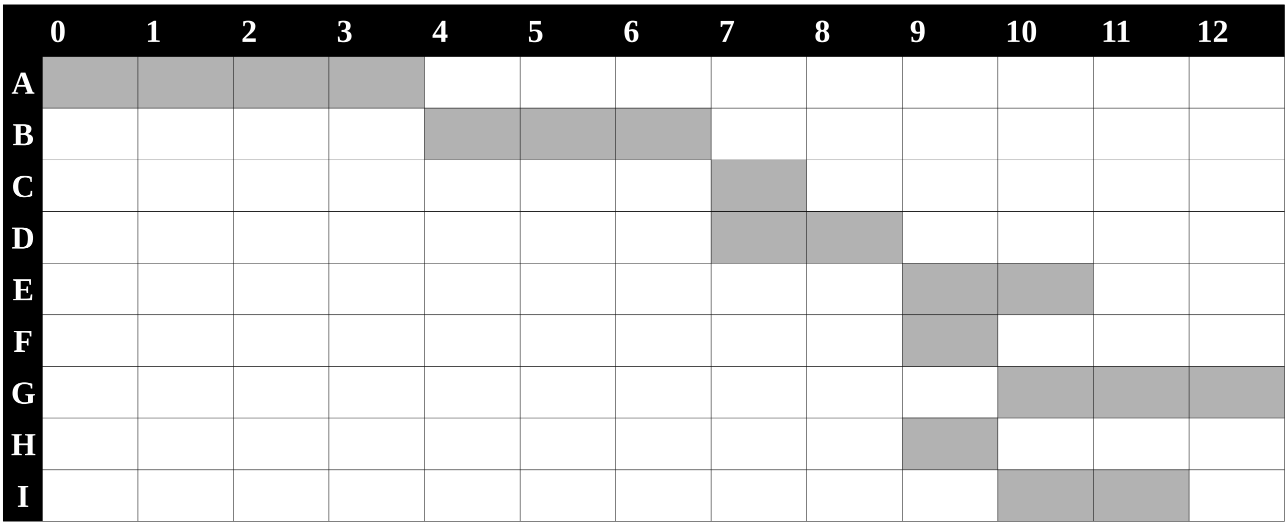
EJERCICIO 9

- Resolver el ejercicio anterior suponiendo ahora que cada tarea la realiza un único trabajador y que los tres son capaces de realizar cualquier tarea, pero el tiempo que cada trabajador requiere para finalizarla es distinto. La duración de cada tarea dependiendo del trabajador que la realiza se muestra en la siguiente tabla.

	A	**B**	**C**	**D**	**E**	**F**	**G**	**H**	**I**
Tr1	**4**	**3**	**3**	**2**	**4**	**3**	**1**	**1**	**2**
Tr2	**7**	**5**	**1**	**5**	**2**	**2**	**2**	**3**	**3**
Tr3	**10**	**7**	**4**	**8**	**6**	**1**	**3**	**5**	**4**

Este ejercicio también está basado en el ejercicio 7, añadiendo la restricción de que cada tarea empiece después de las anteriores o, si empieza antes de que otra tarea anterior termine, que no la haga el mismo trabajador, de manera que comprobamos que un trabajador no pueda hacer dos tareas a la vez.

Para gestionar las distintas duraciones de cada tarea se ha hecho una matriz con la tabla del enunciado y se ha restringido que la duración de cada tarea sea la de su trabajador asignado(lo cual se guarda en otro vector). Al minimizar el mayor valor de fin, se asigna cada trabajador de la mejor manera posible, podemos ver el resultado en la siguiente tabla y ver que es factible y que consigue un tiempo muy corto.



EJERCICIO 10

- Un turista desea llenar su mochila con varios objetos para hacer un viaje. El peso máximo que aguanta la mochila son 275kg, y cada objeto tiene distinto grado de preferencia para el turista (la preferencia se mide en una escala de 0 a 200, donde los números más altos representan los objetos más deseados por el turista). En la tabla siguiente se muestran los objetos, su peso y su preferencia. Encontrar el conjunto de objetos cuya suma de preferencias sea máxima sin exceder el peso máximo que aguanta la mochila.

Objeto	**Peso (Kg)**	**Preferencia**
Mapa	**9**	**150**
Compás	**13**	**35**
Agua	**153**	**200**
Sandwich	**50**	**160**
Azúcar	**15**	**60**
Lata	**68**	**45**
Plátano	**27**	**60**
Manzana	**39**	**40**
Queso	**23**	**30**
Cerveza	**52**	**10**
Protector Solar	**11**	**70**
Cámara	**32**	**30**

Para resolver este último problema se ha representado la información creando un vector de pesos, un vector de preferencias y un vector x de booleanos que será nuestra solución, echando el objeto en la mochila si la posición de este vector tiene un 1 y dejando fuera los objetos cuya posición tiene un 0, ya que es la forma más cómoda de manejar qué objeto echamos y cual no, pudiendo acceder a los pesos y a las preferencias tan solo con un indice común. Los objetos se han codificado tal cual están en la tabla de arriba a abajo y de 1 hasta 12, de tal manera que la primera posición de x representa el mapa y la última(12) la cámara.

Sólo ha sido necesario definir la restricción del peso, es decir, que la suma de todos los peos de objetos cuya posición sea igual a 1 en x no supere el peso máximo de 275. Para resolver el ejercicio, vamos a indicarle a Minizinc que debe maximizar la suma de las preferencias de los objetos cuya posición sea 1 en x.

Como mejor solución hemos obtenido, con un peso de 274kg y una suma de preferencias de 705, que debe de meter en la mochila el mapa, el compás, el agua, el sandwich, el azúcar, el queso y el protector solar.