Práctica 1: Satisfacción de restricciones

Curso 2023-24
Tercer Curso de Ingeniería Informática

José Antonio Zamora Reyes

Técnicas de los Sistemas Inteligentes

UNIVERSIDAD
DE GRANADA

Ejercicio 1 : Problema de coloreado de mapas

Tras realizar el problema de optimización de restricciones , al ejecutarlo, podemos observar que el costo mínimo que se puede conseguir satisfaciendo la restricción de que cada país tenga un color distinto al de sus vecinos, es 2900.

En cuanto a los resultados de (COP) , podemos conocer el número de soluciones mostrando las estadísticas , el número de soluciones que muestran las estadisticas es 7 , esto **no** quiere decir que existan 7 soluciones que proporcionen un coste igual a 2900 . Estas 7 soluciones ,son básicamente las soluciones por las cuales la ejecución ha pasado hasta conseguir una solución que consiga el coste mínimo , por tanto el número de soluciones que realmente se encuentra tras la ejecución es **1** , la cual ofrece el coste mínimo.

SECUENCIA DE COSTES : 3500→3400→3350→3300→3100→3000→**2900**

```
COSTE TOTAL : 3500
Argentina : Rojo 250
Bolivia : Naranja 350
Brasil : Verde 100
Chile : Verde 100
Colombia : Naranja 350
Ecuador : Verde 100
Guayana_Francesa : Naranja 350
Guayana : Naranja 350
Paraguay : Azul 450
Peru : Rojo 250
Surinam : Rojo 250
Uruguay : Naranja 350
Venezuela : Rojo 250
----------
COSTE TOTAL : 3400
Argentina : Naranja 350
Bolivia : Rojo 250
Brasil : Verde 100
Chile : Verde 100
Colombia : Rojo 250
Ecuador : Verde 100
Guayana_Francesa : Rojo 250
Guayana : Rojo 250
Paraguay : Azul 450
Peru : Naranja 350
Surinam : Naranja 350
Uruguay : Rojo 250
Venezuela : Naranja 350
```

```
COSTE TOTAL : 3350
Argentina : Azul 450
Bolivia : Naranja 350
Brasil : Rojo 250
Chile : Rojo 250
Colombia : Naranja 350
Ecuador : Rojo 250
Guayana_Francesa : Naranja 350
Guayana : Naranja 350
Paraguay : Verde 100
Peru : Verde 100
Surinam : Verde 100
Uruguay : Naranja 350
Venezuela : Verde 100
----------
COSTE TOTAL : 3300
Argentina : Azul 450
Bolivia : Verde 100
Brasil : Rojo 250
Chile : Rojo 250
Colombia : Naranja 350
Ecuador : Verde 100
Guayana_Francesa : Naranja 350
Guayana : Naranja 350
Paraguay : Naranja 350
Peru : Azul 450
Surinam : Verde 100
Uruguay : Verde 100
Venezuela : Verde 100
```

```
COSTE TOTAL : 3100
Argentina : Azul 450
Bolivia : Verde 100
Brasil : Rojo 250
Chile : Rojo 250
Colombia : Verde 100
Ecuador : Rojo 250
Guayana_Francesa : Verde 100
Guayana : Verde 100
Paraguay : Naranja 350
Peru : Naranja 350
Surinam : Naranja 350
Uruguay : Verde 100
Venezuela : Naranja 350
----------
COSTE TOTAL : 3000
Argentina : Azul 450
Bolivia : Rojo 250
Brasil : Naranja 350
Chile : Verde 100
Colombia : Rojo 250
Ecuador : Verde 100
Guayana_Francesa : Rojo 250
Guayana : Rojo 250
Paraguay : Verde 100
Peru : Azul 450
Surinam : Verde 100
Uruguay : Rojo 250
Venezuela : Verde 100
```

```
COSTE TOTAL : 2900
Argentina : Naranja 350
Bolivia : Verde 100
Brasil : Azul 450
Chile : Rojo 250
Colombia : Rojo 250
Ecuador : Verde 100
Guayana_Francesa : Rojo 250
Guayana : Rojo 250
Paraguay : Rojo 250
Peru : Naranja 350
Surinam : Verde 100
Uruguay : Verde 100
Venezuela : Verde 100
```

Sabiendo el coste mínimo que se puede conseguir , procedemos a plantearlo como un problema de satisfacción de restricciones (CSP) , para lograrlo debemos hacer los siguientes pasos:

1º→ Incluimos en el código la siguiente línea: **"constraint costo_total = 2900;** De esta manera indicamos la restricción de que el coste de las soluciones debe ser igual a 2900 y podremos saber cual es el número de soluciones que proporcionan un coste mínimo.

2º→ Sustituimos en el código **"solve minimize costo_total;"** por **"solve satisfy;"**

Una vez realizados los cambios le indicamos a minizinc que nos muestre las estadísticas y podemos ver el número de soluciones que satisfacen el coste mínimo , las cuales son **4**.

Soluciones de CSP:

```
COSTE TOTAL : 2900
Argentina : Naranja 350
Bolivia : Rojo 250
Brasil : Azul 450
Chile : Verde 100
Colombia : Verde 100
Ecuador : Rojo 250
Guayana_Francesa : Verde 100
Guayana : Verde 100
Paraguay : Verde 100
Peru : Naranja 350
Surinam : Rojo 250
Uruguay : Rojo 250
Venezuela : Rojo 250
```

```
COSTE TOTAL : 2900
Argentina : Naranja 350
Bolivia : Verde 100
Brasil : Azul 450
Chile : Rojo 250
Colombia : Rojo 250
Ecuador : Verde 100
Guayana_Francesa : Rojo 250
Guayana : Rojo 250
Paraguay : Rojo 250
Peru : Naranja 350
Surinam : Verde 100
Uruguay : Verde 100
Venezuela : Verde 100
```

```
COSTE TOTAL : 2900
Argentina : Naranja 350
Bolivia : Verde 100
Brasil : Azul 450
Chile : Rojo 250
Colombia : Verde 100
Ecuador : Rojo 250
Guayana_Francesa : Verde 100
Guayana : Verde 100
Paraguay : Rojo 250
Peru : Naranja 350
Surinam : Rojo 250
Uruguay : Verde 100
Venezuela : Rojo 250
```

```
COSTE TOTAL : 2900
Argentina : Naranja 350
Bolivia : Rojo 250
Brasil : Azul 450
Chile : Verde 100
Colombia : Rojo 250
Ecuador : Verde 100
Guayana_Francesa : Rojo 250
Guayana : Rojo 250
Paraguay : Verde 100
Peru : Naranja 350
Surinam : Verde 100
Uruguay : Rojo 250
Venezuela : Verde 100
```

El mínimo número de colores para que el problema sea satisfacible es **4.**

Ejercicio 2 : Problema lógico

En este problema podemos obtener una única solución siendo esta la siguiente:

a = Y , b = X, c = W, d = Z

En este problema cabe la posibilidad de que salgan varias soluciones todo depende de como lo implementemos , para evitarlo he enfocado la implementa de la siguiente forma:

- Defino a ,b,c,d como variables enteras que pueden tomar el valor 0 ,1 ,2 ,3.
  Es como decir que a,b,c,d pueden tomar el valor de W,X,Y,Z

- Le he dado unos valores fijos a W,X,Y,Z→ W = 0
  X = 1
  Y = 2
  Z = 3

- Definido lo anterior procedo a aplicar las restricciones que indica el problema y también aplicó la restricción de que a ,b, c, d tiene que ser distintas.

Estableciendo valores fijos para W,X,Y,Z evitó que se den soluciones simétricas.

Ejercicio 3 : Problema de horarios

El número de soluciones válidas son **2.**

Se pueden dar la existencia de soluciones simétricas, dependiendo de la implementación, en mi caso no se producen soluciones simétricas.

Partimos de que tenemos 5 horas de tutoría (Una por cada día a tercera hora).
Cada una de estas horas está identificada internamente por un entero diferente , por lo tanto podrían ser tratadas como horas distintas y se podrían producir soluciones simétricas
Ejemplo:

Partimos de los siguientes identificadores como ejemplo:
H1 = 10 - H3 = 12 - H5 = 14
H2 = 11 - H4 = 13

Algunas soluciones simétricas:

10 - 11 - 12 - 13 - 14
11 - 10 - 12 - 13 - 14
12 - 10 - 11 - 13 - 14

Todas ellas en el horario producirían el mismo resultado aunque internamente sean distintas.

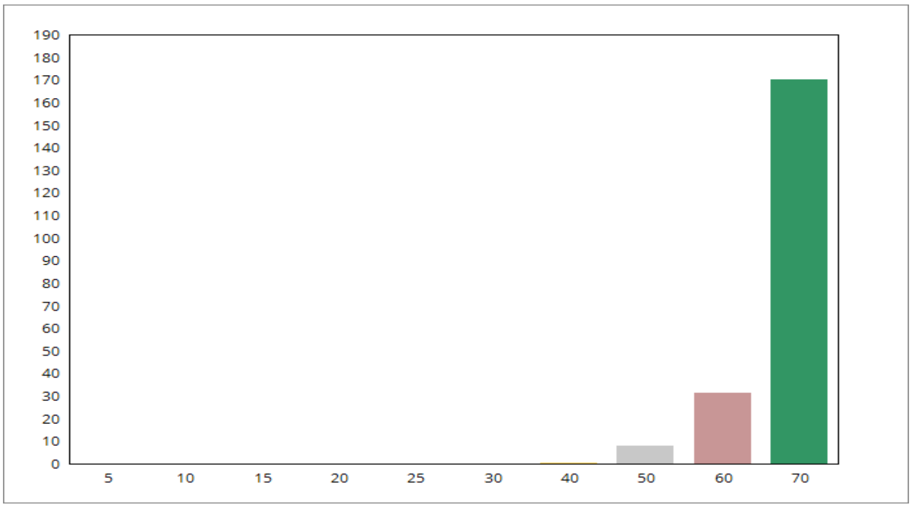
De igual manera que pasa con las horas de tutorías , puede pasar con las asignaturas que se imparten en bloques de dos horas . Tomando cada hora de este bloque como un entero que la identifique , se van a dar dos soluciones que son simétricas.

Para evitar simetrías la implementación se basa en un matriz[hora,dia] sobre la cual vamos a aplicar las restricciones , la restricciones sobre la matriz estarán basadas en las asignaturas que pueden o no estar en una determinada posición , por ejemplo si tenemos las 5 horas de tutorías , debemos de satisfacer la restricción de que en la tercera hora de cada dia se imparte tutoría independientemente de que esa hora de tutoría tenga internamente un identificador u otro , en mi caso basta con que se detecte que la tercera hora de cada dia valga 0 (Tengo asignado el valor 0 a todas las horas de tutorías) . En resumen y siguiendo con el ejemplo , cada hora de tutoría se toma como 0 y se detectan como que todas son iguales.

En el caso en el que no se dan soluciones simetrías la matriz garantiza que en una determinada posición , exista un valor que identifique a cualquier hora de una determinada asignatura, en el caso de no identificar a todas las horas de una misma asignatura con un mismo valor si se producirían soluciones simétricas.

Ejercicio 5: Problema de la mochila

Nº de objetos posibles	Número de objetos incluidos en la mochila	Suma de las prioridades de los objetos de la mochila	Peso final de la mochila (en kgs)	Runtime
5	4	360	4.5	0.043 s
10	6	440	4.5	0.045 s
15	9	690	4.8	0.044 s
20	9	690	4.8	0.045 s
25	11	810	4.6	0.078 s
30	11	860	4.6	0.072 s
40	14	1065	4.9	0.605 s
50	16	1230	4.7	8.111 s
60	17	1280	5.0	31. 554 s
70	17	1390	4.5	170 s



NOTA: En la gráfica de barras no podemos observar los primeros tiempos , esto se debe a que son milisegundos , mientras que los últimos tiempos son segundos.

Observando la tabla y la gráfica , podemos observar como los tiempos de ejecución aumentan exponencialmente conforme el número de objetos sobre el que trabajamos aumenta , esto lo convierte en un problema que no es escalable. Se puede observar como el tiempo de ejecución con solo 70 objetos se dispara ,esto indica que los tiempos aumentan de una manera exponencial , provocando así que con un gran número de objetos sea imposible conseguir una solución al problema.