El Problema de la Planificación del Calendario de Torneos Deportivos

Nelson Daniel Herrera Rey-1841270
Esteban Lopez Mazuera-1844880
Lina Duque-1841877

May 28, 2023

1 Introducción

El problema de la Planificación del Calendario de Torneos Deportivos (CalDep) consiste en encontrar un calendario válido para un torneo de tipo tttiv, considerando las distancias entre las ciudades sedes de los equipos. El objetivo es minimizar la suma total de los costos de las giras, donde el costo de una gira se calcula como la suma de las distancias entre las ciudades visitadas en esa gira.

En este caso, se proporciona una matriz de distancias D de dimensión n x n, donde D[i, j] representa la distancia entre la ciudad sede del equipo i y la ciudad sede del equipo j. También se especifican los valores de Mín y Máx que establecen los límites para el tamaño de las giras y permanencias. La salida esperada es una matriz Cal de dimensión 2(n - 1) x n, que representa un calendario válido.

2 El modelo

2.1 Parámetros

Los parametro que recibe el modelo de minizinc son:

- n: Número entero positivo n que representa el total de equipos del torneo. El numero de equipos debe ser de minimo 2 y debe ser par.
- min: Numero entero positivo n que representa la permanencia mínima que deben tener los equipos. Debe ser mayor a 0 y menor a lo que se defina en max.
- max: Numero entero positivo n que representa la permanencia máxima que deben tener los equipos. Debe ser mayor a 0 y mayor a lo que se defina en min.
- d: Matriz D de dimensión n×n. Cuando el equipo i juega de visitante ante el equipo j debe desplazarse una distancia d_{ij} . Esto representa la distancia entre ciudades de los equipos.
- Internamente, sin ser pasado como dato de entrada, se calcula el número de fechas que tiene el torneo. Esto se hace con la fórmula 2(n −1) puesto que cada equipo juega dos veces contra cada equipo distinto a si mismos.

2.2 Variables y dominios

- Cal: Es una matriz de tamaño $2 \times (n-1) \times n$ que representa el calendario. Cada elemento de la matriz $\texttt{Cal}[i,j]$ indica el equipo contra el cual el equipo j juega en la fecha i. Si $\texttt{Cal}[i,j]$ es positivo, significa que el equipo j juega como local contra el equipo $\texttt{Cal}[i,j]$. Si $\texttt{Cal}[i,j]$ es negativo, significa que el equipo j juega como visitante contra el equipo $|\texttt{Cal}[i,j]|$.
- Values: Es un conjunto de valores que representa los equipos existentes, que va desde 1 hasta n.

2.3 Restricciones

- Restricciones de no negatividad para n, min y max; así como que solo se permita ingresar un n (número de equipos) par.
- Restricción de correspondencia local-visitante: Esta restricción asegura que si un equipo j juega como local contra un equipo k en la fecha i, entonces el equipo k debe jugar como visitante contra el equipo j en la misma fecha. Esto se expresa mediante la siguiente fórmula lógica: $\texttt{Cal}[i,j] = k \rightarrow \texttt{Cal}[i,k] = -j$ y $\texttt{Cal}[i,k] = -j \rightarrow \texttt{Cal}[i,j] = k$.
- Restricción de equipos válidos en el calendario: Esta restricción garantiza que todos los valores en el calendario sean equipos válidos. Es decir, para cada fecha i y equipo j, el valor absoluto de $\texttt{Cal}[i,j]$ debe estar dentro del conjunto de valores de equipos existentes. Esto se expresa mediante la fórmula: $\text{abs}(\texttt{Cal}[i,j]) \in \texttt{values}$.
- Restricción de juegos diferentes en fechas consecutivas: Esta restricción asegura que no haya juegos repetidos en fechas consecutivas. Para cada fecha i, y cada equipo j, se verifica que el juego en la fecha i contra el equipo j sea diferente al juego en la fecha $i+1$ contra el mismo equipo j. Esto se expresa mediante la fórmula: $\text{abs}(\texttt{Cal}[i,j]) \neq \text{abs}(\texttt{Cal}[i+1,j])$.
- Restricción de permanencia máxima: Para cada equipo p, y para cada intervalo de días consecutivos de longitud max, se asegura que la suma de los días en los que el equipo juega como local sea menor o igual a max, y la suma de los días en los que juega como visitante también sea menor o igual a max. Esto garantiza que ningún equipo permanezca en una ubicación o realice un recorrido mayor a max días consecutivos.
- Restricción de permanencia máxima: Para cada equipo p, y para cada intervalo de días consecutivos de longitud min, se asegura que la suma de los días en los que el equipo juega como local sea mayor o igual a min, y la suma de los días en los que juega como visitante también sea mayor o igual a min. Esto garantiza que ningún equipo permanezca en una ubicación o realice un recorrido menor a min días consecutivos.
- Restricciones de localidad y visitante equitativos: Esta restricción garantiza que en cada fecha, la mitad de los equipos jueguen como locales y la otra mitad jueguen como visitantes. Se utiliza la suma acumulada en cada fila del calendario para contar la cantidad de valores positivos (locales) y la cantidad de valores negativos (visitantes) y se verifica que sean iguales a $\frac{n}{2}$. Esto se expresa mediante las fórmulas: $\sum_{j=1}^{n}(\texttt{if Cal}[i,j] > 0 \texttt{ then } 1 \texttt{ else } 0 \texttt{ endif}) = \frac{n}{2}$ y $\sum_{j=1}^{n}(\texttt{if Cal}[i,j] < 0 \texttt{ then } 1 \texttt{ else } 0 \texttt{ endif}) = \frac{n}{2}$.

2.4 Función objetivo

- Total cost: Es una variable que calcula el costo total de todas las giras. Suma los costos de los desplazamientos de los equipos en cada fecha del calendario, teniendo en cuenta si el equipo juega como visitante o local y las distancias correspondientes en la matriz `d`.

3 Detalles importantes

Durante la implementación de esta solución, se encontró que uno de los aspectos más interesantes y desafiantes fue la incorporación de las restricciones de rango para las giras y las permanencias en el problema. En particular, puesto que la forma en que funcionan los bucles "for" en MiniZinc es significativamente diferente a la lógica a la que se está acostumbrado en la programación convencional que se realiza en otros cursos, se tuvo la necesidad de aprender a trabajar bajo esta lógica. En la mayoría de los lenguajes de programación, los bucles "for" se utilizan para iterar sobre una secuencia de elementos, incrementando o decrementando el valor de una variable de control en cada iteración. Sin embargo, en MiniZinc, los bucles "for" se utilizan principalmente para definir conjuntos y no para realizar iteraciones en el sentido tradicional.

Esto implicó pensar en cómo definir correctamente los conjuntos de índices y utilizar las restricciones específicas de MiniZinc para establecer los rangos y las relaciones entre las variables. Además, fue

necesario considerar la lógica del problema y transformarla en restricciones lógicas apropiadas que MiniZinc pudiera entender y resolver de manera eficiente.

4 Pruebas

En esta sección se muestran algunas de las pruebas realizadas, se hicieron 10 de estas, en las cuales se usaron matrices de distancia 4x4, cada una se corrio dos vecces cambiando unicamente su min y mix; los cuales son 1 y 3 respectivamente para su primera version de prueba, y 1 y 2 para su segunda prueba. Esto se hizo pues se notó que mientras la direncia entre el min y el max fuera mayor, el modelo tardaba un poco más en resolver el problema.

4.1 Entrada:

$$n = 4;$$

$$d = \begin{bmatrix} 0 & 745 & 665 & 929 \\ 745 & 0 & 80 & 337 \\ 665 & 80 & 0 & 380 \\ 929 & 337 & 380 & 0 \end{bmatrix}$$

$$\min = 1;$$
$$\max = 3;$$

Salida:

$$\begin{bmatrix} -4 & 3 & -2 & 1 \\ -2 & 1 & -4 & 3 \\ -3 & 4 & 1 & -2 \\ 2 & -1 & 4 & -3 \\ 4 & -3 & 2 & -1 \\ 3 & -4 & -1 & 2 \end{bmatrix}$$

El costo total de la solución, que corresponde a la suma de los costos de todas las giras es : 8276
El tiempo de respuesta fue: 1s 79msec.

4.2 Entrada:

$$n = 4;$$

$$d = \begin{bmatrix} 0 & 745 & 665 & 929 \\ 745 & 0 & 80 & 337 \\ 665 & 80 & 0 & 380 \\ 929 & 337 & 380 & 0 \end{bmatrix}$$

$$\min = 1;$$
$$\max = 2;$$

Salida:

$$\begin{bmatrix} -3 & -4 & 1 & 2 \\ 2 & -1 & 4 & -3 \\ 4 & 3 & -2 & -1 \\ -2 & 1 & -4 & 3 \\ -4 & -3 & 2 & 1 \\ 3 & 4 & -1 & -2 \end{bmatrix}$$

El costo total de la solución, que corresponde a la suma de los costos de todas las giras es : 10287
El tiempo de respuesta fue 717msec.

4.3 Entrada:

$$n = 4;$$

$$d = \begin{bmatrix} 0 & 1000 & 2000 & 3000 \\ 1000 & 0 & 4000 & 5000 \\ 2000 & 4000 & 0 & 6000 \\ 3000 & 5000 & 6000 & 0 \end{bmatrix}$$

$$\min = 1;$$
$$\max = 3;$$

Salida:

$$\begin{bmatrix} -3 & -4 & 1 & 2 \\ -4 & -3 & 2 & 1 \\ 2 & -1 & -4 & 3 \\ 3 & 4 & -1 & -2 \\ 4 & 3 & -2 & -1 \\ -2 & 1 & 4 & -3 \end{bmatrix}$$

El costo total de la solución, que corresponde a la suma de los costos de todas las giras es : 55000
El tiempo de respuesta fue 1s 168msec.

4.4 Entrada:

$$n = 4;$$

$$d = \begin{bmatrix} 0 & 1000 & 2000 & 3000 \\ 1000 & 0 & 4000 & 5000 \\ 2000 & 4000 & 0 & 6000 \\ 3000 & 5000 & 6000 & 0 \end{bmatrix}$$

$$\min = 1;$$
$$\max = 2;$$

Salida:

$$\begin{bmatrix} -4 & -3 & 2 & 1 \\ 3 & 4 & -1 & -2 \\ 2 & -1 & -4 & 3 \\ -3 & -4 & 1 & 2 \\ 4 & 3 & -2 & -1 \\ -2 & 1 & 4 & -3 \end{bmatrix}$$

El costo total de la solución, que corresponde a la suma de los costos de todas las giras es : 69000
El tiempo de respuesta fue 726msec.

4.5 Entrada:

$$n = 4;$$

$$d = \begin{bmatrix} 0 & 10000 & 20000 & 30000 \\ 10000 & 0 & 40000 & 50000 \\ 20000 & 4000 & 0 & 60000 \\ 30000 & 50000 & 60000 & 0 \end{bmatrix}$$

$$\min = 1;$$
$$\max = 3;$$

Salida:

$$\begin{bmatrix} 4 & -3 & 2 & -1 \\ 2 & -1 & 4 & -3 \\ -3 & 4 & 1 & -2 \\ -2 & 1 & -4 & 3 \\ 3 & -4 & -1 & 2 \\ -4 & 3 & -2 & 1 \end{bmatrix}$$

El costo total de la solución, que corresponde a la suma de los costos de todas las giras es : 472000
El tiempo de respuesta fue 1s 102msec.

4.6 Entrada:

$$n = 4;$$

$$d = \begin{bmatrix} 0 & 10000 & 20000 & 30000 \\ 10000 & 0 & 40000 & 50000 \\ 20000 & 4000 & 0 & 60000 \\ 30000 & 50000 & 60000 & 0 \end{bmatrix}$$

$$\min = 1;$$
$$\max = 2;$$

Salida:

$$\begin{bmatrix} -4 & 3 & -2 & 1 \\ 2 & -1 & 4 & -3 \\ -3 & 4 & 1 & -2 \\ -2 & 1 & -4 & 3 \\ 3 & -4 & -1 & 2 \\ 4 & -3 & 2 & -1 \end{bmatrix}$$

El costo total de la solución, que corresponde a la suma de los costos de todas las giras es : 556000
El tiempo de respuesta fue 818msec.

4.7 Entrada:

$$n = 4;$$

$$d = \begin{bmatrix} 0 & 100000 & 200000 & 300000 \\ 100000 & 0 & 400000 & 500000 \\ 200000 & 40000 & 0 & 600000 \\ 300000 & 500000 & 600000 & 0 \end{bmatrix}$$

$$\min = 1;$$
$$\max = 3;$$

Salida:

$$\begin{bmatrix} 4 & -3 & 2 & -1 \\ 2 & -1 & 4 & -3 \\ -3 & 4 & 1 & -2 \\ -2 & 1 & -4 & 3 \\ 3 & -4 & -1 & 2 \\ -4 & 3 & -2 & 1 \end{bmatrix}$$

El costo total de la solución, que corresponde a la suma de los costos de todas las giras es : 4720000
El tiempo de respuesta fue 1s 130msec.

4.8 Entrada:

$$n = 4;$$

$$d = \begin{bmatrix} 0 & 100000 & 200000 & 300000 \\ 100000 & 0 & 400000 & 500000 \\ 200000 & 40000 & 0 & 600000 \\ 300000 & 500000 & 600000 & 0 \end{bmatrix}$$

$$\min = 1;$$
$$\max = 2;$$

Salida:

$$\begin{bmatrix} -4 & 3 & -2 & 1 \\ 2 & -1 & 4 & -3 \\ -3 & 4 & 1 & -2 \\ -2 & 1 & -4 & 3 \\ 3 & -4 & -1 & 2 \\ 4 & -3 & 2 & -1 \end{bmatrix}$$

El costo total de la solución, que corresponde a la suma de los costos de todas las giras es : 5560000
El tiempo de respuesta fue 692msec.

4.9 Entrada:

$$n = 4;$$

$$d = \begin{bmatrix} 0 & 1000000 & 2000000 & 3000000 \\ 1000000 & 0 & 4000000 & 5000000 \\ 2000000 & 4000000 & 0 & 6000000 \\ 3000000 & 5000000 & 6000000 & 0 \end{bmatrix}$$

$$\min = 1;$$
$$\max = 3;$$

Salida:

$$\begin{bmatrix} -3 & -4 & 1 & 2 \\ -4 & -3 & 2 & 1 \\ 2 & -1 & -4 & 3 \\ 3 & 4 & -1 & -2 \\ 4 & 3 & -2 & -1 \\ -2 & 1 & 4 & -3 \end{bmatrix}$$

El costo total de la solución, que corresponde a la suma de los costos de todas las giras es : 55000000
El tiempo de respuesta fue 1s 141.

4.10 Entrada:

$$n = 4;$$

$$d = \begin{bmatrix} 0 & 1000000 & 2000000 & 3000000 \\ 1000000 & 0 & 4000000 & 5000000 \\ 2000000 & 4000000 & 0 & 6000000 \\ 3000000 & 5000000 & 6000000 & 0 \end{bmatrix}$$

$$\min = 1;$$
$$\max = 2;$$

Salida:

$$\begin{bmatrix} -4 & -3 & 2 & 1 \\ 3 & 4 & -1 & -2 \\ 2 & -1 & -4 & 3 \\ -3 & -4 & 1 & 2 \\ 4 & 3 & -2 & -1 \\ -2 & 1 & 4 & -3 \end{bmatrix}$$

El costo total de la solución, que corresponde a la suma de los costos de todas las giras es : 69000000
El tiempo de respuesta fue 841.

5 Análisis de resultados

Las pruebas 1 y 2 usan la misma matriz de distancias de entrada, sin embargo, en la primera se establecen min=1 y max=3; y en la segunda min=2 y max=3. En base a esto, se puede evidenciar que el modelo obtiene una respuesta mas rapida en la segunda prueba, puesto que el modelo mas restringido

en min y max, lo cual hace que deba hacer menos comprobaciones para encontrar la solución, y esta puede ser la razon de porqé sucede esto. Esto mismo se puede evidenciar en las siguientes pruebas, pues en todas se reduce el tiempo de respuesta del modelo, al reducir el rango entre min y max. Esto se puede apreciar mejor en el siguiente grafico, en el cual se ve claramente la relacion entre la diferencia entre el min y max; y el tiempo de respuesta.

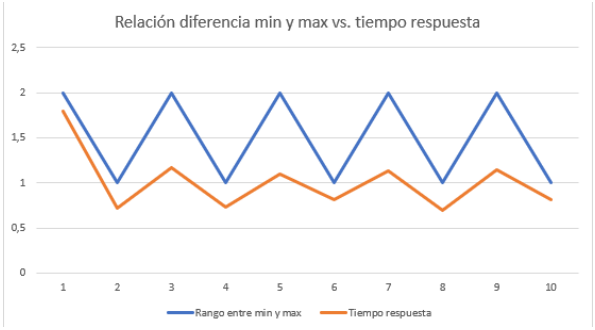

Figure 1: Relación rango entre min y max vs. tiempo

Tambien se pudo observar que para instancias de entrada con matrices de distancia 6x6 el modelo no fue capaz de encontrar una solución, pues este duraba ejecutándose por horas sin hallar una solución válida.

6 Video interfaz gráfica

Video demostración GUI

7 Conclusiones

El tiempo de respuesta del modelo para resolver las instancias del problema con matrices de distancia 4x4 fue razonable en la mayoría de los casos. Aunque hubo variaciones en los tiempos de respuesta, en general el modelo pudo generar las soluciones en un tiempo aceptable. Se evaluaron dos conjuntos de parámetros diferentes, uno con un rango mínimo y máximo de 1 a 3, y otro con un rango de 1 a 2. Se observó que el tiempo de respuesta fue menor mientras mas restringido fuera este rango.

En el caso de matrices de entrada 6x6 el modelo no logra terminar y se queda corriendo por tiempos superiores a la hora, se cree que esto puede ser normal puesto a la complejidad exponencial que caracterizan los algoritmos que dan solución a este tipo de problemas. En general, el modelo muestra capacidad para resolver El Problema de la Planificación del Calendario de Torneos Deportivos en instancias pequeñas, generando soluciones consistentes en un tiempo razonable.