Primera práctica de Programación con restricciones evaluación Tipo A (por curso)

Facultad de Informática
Profesor: Albert Rubio

marzo de 2021

1. Introducción

Esta práctica es **individual o por parejas** y tenéis que entregarla mediante el campus virtual. Las entregas por parejas serán realizadas por uno de los dos miembros. La pareja tiene que haber sido informada mediante la tarea "Elección de parejas de prácticas" del CV.

El enunciado incluye una serie de restricciones que la solución debe satisfacer. Vuestra solución debe cubrir tantas restricciones como sepáis codificar con MiniZinc. Tenéis que incluir también una serie de juegos de prueba que hayáis utilizado para poner a prueba vuestra solución. Además del código y de los juegos de prueba tenéis que incluir un documento corto que explique qué restricciones habéis codificado y cuanto tarda en encontrar una solución para los distintos ejemplos. Si habéis hecho la parte de optimización, indicad que soluciones encuentra y si encuentra el óptimo (indicando el tiempo que tarda en encontrar la primera solución y la óptima. En caso de que tarde mucho, indicad cuál es la mejor solución que ha encontrado y en cuanto tiempo. El documento puede incluir también información sobre distintas opciones de búsqueda que hayáis probado.

2. Problema de satisfacción

Considerad una fábrica que nunca para la producción. Por esta razón, es necesario cubrir cada día los tres turnos diarios asignando a cada uno de ellos los trabajadores que sean necesarios. Suponed que tenemos que hacer la planificación de turnos para D días consecutivos con los T trabajadores que tenemos y con $N1$, $N2$ y $N3$ trabajadores respectivamente en cada turno. La solución que se produce puede ser una de las dos siguientes (o las dos):

- que turno tiene asignado cada trabajador cada día o si libra,
- qué trabajadores hay en cada turno cada día.

Se valorará que uséis "constraint assert" cuando sea posible detectar que no existe solución antes de iniciar la búsqueda.

A continuación se detallan las restricciones que se deben cumplir (implementad tantas como sepáis codificar).

1. Cada turno tiene el número de trabajadores (N1, N2 o N3) que le corresponde.
2. Un trabajador solo puede estar en un turno cada día.
3. Dado un número MaxDT, garantizar que nadie trabaja MaxDT días consecutivos.
4. Dado un número MaxDL, garantizar que nadie tiene MaxDL días libres consecutivos.
5. Dado un número MinDT, garantizar que todos trabajan como mínimo MinDT en los D días.
6. Un trabajador no puede hacer el último turno de un día y el primero del día siguiente dos veces seguidas en cuatro días consecutivos.
7. Dada una serie de parejas de trabajadores afines y un número A, cada trabajador de un turno tiene que estar con al menos A trabajadores afines en ese turno.
8. Sea C el número de categorías. Dada la categoría de los trabajadores. En cada turno debe haber al menos un trabajador de cada categoría. Podéis generalizar esta restricción a que se indique exactamente (o como mínimo) cuántos de cada categoría tiene que tener cada turno y tenerlo distinto para cada uno de los tres turnos.

3. Problema de optimización

Podéis añadir más restricciones como las siguientes o, mejor aún, optimizar respecto a ellas. También podéis proponer nuevas restricciones.

1. Añadid la posibilidad de que los trabajadores pidan días que no quieren trabajar. Tened en cuenta que estos días no cuentan en la restricción de días consecutivos libres y que hay que descontarlos de MinDT. Convertid el problema en un problema de optimización, minimizando el número de veces que hacemos trabajar a alguien cuando no quiere. Pensad en formas de conseguir que los incumplimientos se distribuyan entre todos.
2. Añadid la posibilidad de que los trabajadores pidan qué turno no quieren (único o para cada día), si es que tienen alguna preferencia. Convertid el problema en un problema de optimización, minimizando el número de veces que incumplimos la petición. Pensad en formas de conseguir que los incumplimientos se distribuyan entre todos.