Labos LI: Resolución de problemas con SAT

Si el penúltimo dígito de tu DNI es D, te toca resolver el problema número D/2 de la lista. En todos ellos, hay que crear un programa Prolog que: (1) genera un conjunto de cláusulas SAT para resolverlo, (2) hace una llamada al SAT solver `picosat`, y (3) a partir del modelo encontrado (si hay), presenta la solución al problema original (siguiendo el ejemplo ya visto del coloreado de grafos y del sudoku). Para cada uno de los problemas damos ejemplo(s) de entrada.

Problema 0: la planificación semanal de una acería. En una acería, cada semana (24 horas/dia * 7 días = 168h) hay que realizar un número de tareas, usando tres tipos de recursos: 3 plataformas, 2 hornos (movibles entre plataformas), y 1 refrigeradora (movible entre plataformas). Cada tarea usa una plataforma y consta de tres fases seguidas (inmediatamente): 1 hora de carga, X horas de horno, y 5 horas de enfriamiento. Es decir, lo que varía entre tareas es el tiempo de horno. Presenta el resultado como en `barcelogic.com/?page=resourceScheduling&lang=es`. Ejemplo: planifica las siguientes 20 tareas, minimizando el tiempo total, y sin que en ningún momento se usen más de 3 plataformas, o más de 2 hornos, o más de 1 refrigeradora: 9 tareas de 10 horas de horno, 5 de 12, 1 de 15, 2 de 16, y 3 de 22. Objetivo: encontrar la solución que menos horas necesita (168h es fácil, 150 también...).

Problema 1: planificación de ligas deportivas. La liga española de primera división de fútbol tiene 20 equipos, que juegan todos contra todos en 19 jornadas la primera vuelta, y otra vez en otras 19 jornadas en la segunda vuelta, en el mismo orden, pero en casa del otro. Hay que admitir los siguientes tipos de restricciones: –el equipo i no quiere jugar en casa la jornada j, –el equipo i sí quiere jugar en casa la jornada j, –en las jornadas i e i + 1 no se admiten repeticiones: dos partidos seguidos en casa, o dos partidos seguidos fuera, –el partido i-i (es decir, en casa del equipo i) no debe ser la jornada j –el partido i-i (es decir, en casa del equipo i) sí debe ser la jornada j. No se permitir'an tripeticiones, es decir, ningún equipo jugará tres jornadas seguidas en casa ni tres jornadas seguidas fuera. Recomendación: primero aborda un subproblema, por ejemplo, una sola vuelta, pocos equipos, etc. Extensión: a lo largo de la temporada ningún equipo tiene más de k repeticiones (esto puede hacer que el problema SAT sea duro para ligas de mas de 16 equipos). Extensión: hacer que la segunda vuelta no sea la simétrica de la primera, es decir, que cada jornada tenga su vuelta, pero donde no necesariamente la vuelta de la jornada 1 es la 20, etc.

Problema 2: planificación del horario semanal de la FIB: cinco días de 8 a 20h (12 horas diarias). En los datos de entrada se indican las listas de aulas, profesores, y cursos que hay. Todos ellos reciben un natural no nulo como identificador. Por cada curso se da su lista de asignaturas, cada una con su número de horas semanales de clase (máximo una al día), la lista de aulas adecuadas para ella, y la lista de profesores que la podrían impartir. Todas las sesiones de una misma asignatura deben impartirse en la misma aula por el mismo profesor. Por cada profesor se indican las horas (con un entero entre 1 y 60) en que no puede dar clase. Por supuesto, no puede haber más de una clase a la vez por profesor ni por aula. Cada curso ha de tener un horario compacto (sin horas libres entre dos clases el mismo día) y no más de seis horas de clase al día. Sesiones de un mismo curso no pueden solaparse en el tiempo. Ayuda: es mejor introducir diversos tipos de variables que relacionan dos entidades (profesor-asignatura, asignatura-hora, etc.) que tener variables que relacionan más de dos.

Problema 3: Rodear números con ciclos. Dada un cuadrícula (n filas, m columnas) donde algunas de sus casillas contienen un entero entre 0 y 3, se trata de dibujar un conjunto de ciclos de manera que, dada una casilla con un número k, exactamente k de los 4 lados de la casilla formen parte de un ciclo. Los ciclos no pueden compartir ni lados ni vértices.

Ayuda: proporcionamos un predicado displaySol(M), que genera un archivo PostScript (ver los ejemplos rodear.ps y rodear.pdf) que representa la solución que el SAT solver ha encontrado. Dicho predicado asume que el problema se ha modelado con variables `h-I-J`,con $1 \leq I \leq n+1, 1 \leq J \leq m$, y `v-I-J` con $1 \leq I \leq n, 1 \leq J \leq m+1$. La variable `h-I-J` indica si el segmento horizontal correspondiente a la I-ésima fila y la J-ésima columna forma parte de algún ciclo y similarmente con `v-I-J` para los segmentos verticales. Podéis modificar el predicado si lo consideráis conveniente. Extension: intenta conseguir el minimo numero de ciclos (ayuda: mediante una secuencia de llamadas al sat solver).

Problema 4: Hay que hacer un solver para el juego de Flow (Flow Free) que existe sobre Android, iPhone, Ipad... Proporcionamos un predicado displaySol(M) que genera un archivo PostScript (ver los

ejemplos flow.ps y flow.pdf) que representa la solución que el SAT solver ha encontrado. Extension: genera problemas de flow más grandes, con más colores. Extension: cuenta cuántas soluciones distintas hay (ayuda: mediante una secuencia de llamadas al sat solver). Extension: intenta conseguir que el camino más largo sea lo más corto posible (ayuda: mediante una secuencia de llamadas al sat solver).