Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI-5437 Inteligencia Artificial I
Trimestre Abril - Julio 2023

Informe Proyecto III

Estudiantes:

Br. Gamboa, Roberto

Br. Bandez, Jesús

Carnés:

16-10394

17-10046

Profesor

Carlos Infante

11 de julio de 2023

Índice

Introducción

El modelado de problemas y su posterior resolución son tareas esenciales en múltiples disciplinas, desde la informática hasta la matemática y la inteligencia artificial. Uno de los enfoques más efectivos y ampliamente utilizados para representar problemas de decisión es la forma normal conjuntiva (CNF, por sus siglas en inglés), la cual se basa en la lógica proposicional.

El objetivo principal de este proyecto es familiarizarse con el modelado de problemas en CNF y aprovechar las ventajas de los SAT solver, herramientas diseñadas específicamente para resolver problemas en CNF. En este caso, se utilizará el SAT solver Glucose para resolver el problema dado y encontrar soluciones válidas.

Una de las ventajas de los SAT solvers es su capacidad para manejar problemas de gran complejidad que son fáciles de representar utilizando lógica proposicional. El SAT solver examina todas las posibles combinaciones de valores para las variables en la fórmula CNF y determina si existe una asignación que satisfaga todas las cláusulas.

Sin embargo, la salida generada por un SAT solver generalmente se presenta en un formato que no es directamente legible para los seres humanos. Por lo tanto, otra parte crucial de este proyecto será aprender a traducir la salida del SAT solver a un formato comprensible y útil para el usuario final. En específico se traducirá la salida del SAT solver al formato iCalendar, el cual puede ser posteriormente importado por cualquier aplicación de calendario para crear recordatorios para cada evento.

Decisiones de diseño e implementación

Para realizar el modelo se utilizó una variable booleana con 4 subíndices que se puede representar por $P_{i,j,d,s}$. La variable $P_{i,j,d,s}$ es cierta cuando ocurre un partido en el que el equipo i juega como local contra el equipo j el día d en la hora s. Los equipos se representan por un numero de 1 a la cantidad total de equipos. De igual forma, los días se representan por un numero que va de 1 al cantidad total de días disponibles en el torneo. Los slots representan un bloque de horas en el que puede ocurrir un partido. Mínimo debe existir un slot y máximo pueden ocurrir 11 slots por día.

Se utilizaron las siguientes restricciones para garantizar el resultado del modelo:

Cada equipo debe jugar como local contra todos los otros equipos exactamente una vez

$$(\forall_{i,j} | i \neq j : \sum_{d} \sum_{s} P_{i,j,d,s} = 1)$$

Note que esta restricción implícitamente fuerza a todos los equipos a jugar como visitante contra todos los otros equipos

A lo mas un juego por día para cada equipo

$$(\forall_{i,d} | \sum_{j,j \neq i} \sum_{s} P_{i,j,d,s} + P_{j,i,d,s} <= 1)$$

No puede haber dos juegos al mismo tiempo

$$(\forall_{d,s} | \sum_{i} \sum_{j,j \neq i} P_{i,j,d,s} <= 1)$$

Un equipo no puede jugar como local dos días consecutivos

$$(\forall_{i,d} | \sum_{j,j \neq i} \sum_{s} P_{i,j,d,s} + P_{i,j,d+1,s} <= 1)$$

Un equipo no puede jugar como visitante dos días consecutivos

$$(\forall_{i,d} | \sum_{j,j \neq i} \sum_{s} P_{j,i,d,s} + P_{j,i,d+1,s} <= 1)$$

Este modelo se instancia con cada problema que se le pasa al programa con el .json y se convierte a formato CNF. Para convertir las restricciones a CNF se usaron los métodos mostrados en artículo web “Pseudo-Boolean and Cardinality Constraints”

Ejecución del proyecto

El proyecto fue realizado en el lenguaje de programación Python, en su versión 3.10.8. Para ejecutar el proyecto inicialmente se requiere instalar las librerías necesarias, especificadas en el archivo requirements.txt ubicado en el directorio raíz del proyecto, las mismas pueden instalarse mediante el comando

```
$ pip install -r requirements.txt
```

Posteriormente basta con acceder al directorio Solver y ejecutar el archivo main.py con el comando

```
$ python3 main.py datos.json
```

donde datos.json es un archivo .json con los datos del torneo que debe encontrarse dentro del mismo directorio Solver. Al ejecutar el archivo se crea en la misma ubicación un archivo .ics con el nombre del torneo, el mismo puede ser importado por cualquier aplicación de calendario para recordar las fechas de cada partido.

Tanto el programa que lee los datos del archivo .json como el que los traduce a forma CNF y, posteriormente, crea el archivo .ics con la solución son llamados desde donde se encuentran implementados en main.py. También puede encontrar instrucciones detalladas en el "Readme.md" del repositorio.

Resultados

Para medir el rendimiento de la implementación realizada se realizaron 9 pruebas de distintas dificultades, con cantidades de equipos y slots por día distintos en cada caso.

Los resultados obtenidos al ejecutar tales pruebas se resumen en la siguiente tabla:

Pruebas				
Dificultad	Cant. equipos	Duración del torneo (días)	Slots por día	Tiempo de ejecución (segundos)
Muy fácil	4	13	4	0,246
Fácil	6	40	3	6,481
Media	8	50	3	29,615
Media	10	50	3	54,628
Difícil	10	60	3	83,652
Muy difícil	12	60	3	166,339
Muy difícil	14	60	3	300,446
Ultra difícil	16	60	3	519,748
Extrema	18	60	3	-

De los resultados presentados en la tabla se pueden extraer las siguientes conclusiones:

- El tiempo de resolución para cada caso crece exponencialmente a medida que aumenta la dificultad (mayor cantidad de días, mayor cantidad de equipos y menor cantidad de slots por día)
- El crecimiento exponencial del tiempo de resolución se puede deber a las restricciones implementadas para asegurar que todas las condiciones del torneo se cumplan
- En la mayor dificultad, es tal la magnitud de los cálculos que se deben realizar para evaluar todas las posibles combinaciones de equipos y las restricciones, que el computador se ve incapaz de resolverlo, incluso luego de mas de 4 horas de ejecución
- Con las restricciones actuales es imposible resolver problemas con un gran número de equipos

Conclusión

A lo largo del proyecto, adquirimos un conocimientos sobre el modelado de problemas en CNF, lo que nos permitió representar el problema presentado de asignación de slots para los partidos en términos de lógica proposicional. Aprendimos a identificar y expresar las restricciones y condiciones del problema en forma de cláusulas y a transformar estas cláusulas en una fórmula CNF.

El uso del SAT solver Glucose fue una parte fundamental del proyecto, al suministrarle las fórmulas CNF a este pudimos observar cómo hallaba la solución fácilmente y encontraba una asignación válida para cada equipo.

En conclusión, este proyecto brindó una valiosa experiencia en el modelado de problemas en CNF y el uso de un SAT solver para resolverlos. Se aprendió a abordar problemas complejos y a utilizar herramientas especializadas para encontrar soluciones válidas y traducir la salida del solver a un formato más útil.