Universidad Simón Bolívar
Departamento de Computación y Tecnología de la Información
CI5438 - Inteligencia Artificial I
Trimestre Abril - Julio 2021
Prof. Carlos Infante
Amin Arriaga 16-10072, David Segura 13-11341, Wilfredo Graterol 15-10639

Scheduling Problem as CNF

July 22, 2021

1 Formulación

Sea $\mathcal{E}$ el conjunto de equipos, $\mathcal{F}$ el conjunto de fechas disponibles y $\mathcal{H}$ el conjunto de horarios disponibles tal que $E = |\mathcal{E}|$, $F = |\mathcal{F}|$ y $H = |\mathcal{H}|$. Entonces, las variables de nuestro CNF se identifican de la forma (e_i, e_j, h, f) donde $e_i, e_j \in \mathcal{E}$ con $e_i \neq e_j$, $h \in \mathcal{H}$ y $f \in \mathcal{F}$ e indican que el equipo e_i juega como local en contra del equipo e_j como visitante en la fecha f y hora h. Por lo tanto, nuestro CNF tiene $E(E-1)FH$ variables, y las enumeramos como[1]:

$$ENUM(e_i, e_j, h_k, f_l) = i - (i > j) * j - (j > i)(j-1) + E * j + E(E-1) * k + E(E-1)H * l$$

donde

$0 \leq i, j < E$, $0 \leq l < H$ y $0 \leq k < F$

1.1 Restricciones y Clausuras

- Todos los participantes deben jugar dos veces con cada uno de los otros participantes, una como "visitantes" y la otra como "locales".

$$(\forall e_i, e_j \in \mathcal{E} \mid e_i \neq e_j : (\exists f \in \mathcal{F}, h \in \mathcal{H} \mid \ : (e_i, e_j, f, h))\ )$$

Con un total de $E(E-1)$ clausuras tal que cada una esta conformada por FH átomos.

- Dos juegos no pueden ocurrir al mismo tiempo.

$$\begin{array}{l} (\forall f \in \mathcal{F}, h \in \mathcal{H}, e_i, e_j \in \mathcal{E} \ \mid e_i \neq e_j : \\ \quad (\forall e'_m, e'_n \in \mathcal{E} \mid e'_m \neq e'_n \wedge (m > i \vee (m = i \wedge n > j) : \\ \quad\quad \neg(e_i, e_j, h, f) \vee \neg(e'_m, e'_n, h, f) \\ \quad) \\ ) \end{array}$$

Con un total de $\frac{1}{2}E(E-1)(E(E-1)-1)FH$ clausuras con 2 átomos cada una.

- Un participante puede jugar a lo sumo una vez por día.

$$\begin{array}{l} (\forall e, e'_i, e'_j \in \mathcal{E}, f \in \mathcal{F}, h, h' \in \mathcal{H} \ \mid e \neq e'_i \wedge e \neq e'_j \wedge i < j \wedge h \neq h' : \\ \quad (\neg(e, e'_i, f, h) \vee \neg(e, e'_j, f, h')) \wedge \ (\neg(e'_i, e, f, h) \vee \neg(e'_j, e, f, h')) \\ ) \wedge \\ (\forall e, e'_i, e'_j \in \mathcal{E}, f \in \mathcal{F}, h, h' \in \mathcal{H} \ \mid e \neq e'_i \wedge e \neq e'_j \wedge e'_i \neq e'_j \wedge h \neq h' : \end{array}$$

[1]Los booleanos son tratados como 1 para *True* y 0 para *False*

$\neg(e, e_i', f, h) \vee \neg(e_j', e, f, h')$
$) \wedge$
$(\forall e_i, e_j \in \mathcal{E}, f \in \mathcal{F}, h_m, h_n \in \mathcal{H} \mid i < j \wedge m < n :$
$(\neg(e_i, e_j, h_m, f) \vee \neg(e_i, e_j, h_n, f)) \wedge \ (\neg(e_j, e_i, h_m, f) \vee \neg(e_j, e_i, h_n, f)) \wedge$
$(\neg(e_i, e_j, h_m, f) \vee \neg(e_j, e_i, h_n, f)) \wedge \ (\neg(e_j, e_i, h_m, f) \vee \neg(e_i, e_j, h_n, f))$
$)$

Con un total de $E(E-1)(E-2)FH(H-1) + E(E-1)(E-2)FH(H-1) + E(E-1)FH(H-1)$ clausuras con 2 átomos cada una.

- Un participante no puede jugar de *visitante* en dos días consecutivos, ni de *local* dos días seguidos.

$(\forall e, e_i', e_j' \in \mathcal{E}, f_k, f_{k+1} \in \mathcal{F}, h, h' \in \mathcal{H} \mid e \neq e_i' \wedge e \neq e_j' \wedge e_i' \neq e_j' :$
$(\neg(e, e_i', h, f_k) \vee \neg(e, e_j', h', f_{k+1})) \wedge \ (\neg(e_i', e, h, f_k) \vee \neg(e_j', e, h', f_{k+1}))$
$)$

Con un total de $2E(E-1)(E-2)(F-1)H^2$ de clausuras con 2 átomos cada una.

- Todos los juegos deben empezar en horas "en punto" (por ejemplo, las 13:00:00 es una hora válida pero las 13:30:00 no). Esta restricci'on no forma parte de las condiciones para las clausuras.
- Todos los juegos deben ocurrir entre una fecha inicial y una fecha final especificadas. Pueden ocurrir juegos en dichas fechas. Esta restricci'on no forma parte de las condiciones para las clausuras.
- Todos los juegos deben ocurrir entre un rango de horas especificado, el cuál será fijo para todos los días del torneo. Esta restricci'on no forma parte de las condiciones para las clausuras.
- A efectos prácticos, todos los juegos tienen una duración de dos horas. Esta restricci'on no forma parte de las condiciones para las clausuras.

Estas última 4 condiciones definen los valores de $\mathcal{F}$ y $\mathcal{H}$. Usando la restricción de las fechas, identificamos las fechas en orden cronológico en el rango $[0, F)$ siendo F también el número de días entre las fechas inicial y final. Análogamente, usando las restricciones relacionadas con las horas, podemos calcular cuantos partidos se pueden realizar al día y en que horarios específicos, luego podemos identificar cada bloque del horario en orden cronológico en el rango $[0, H)$, donde H también es el número de partidos que se pueden realizar al día.

En total se tienen $O(E^3FH(E+H))$ clausuras, por lo tanto, el factor principal en la complejidad del problema es precisamente el número de equipos, seguido por el número de horas por día.

2 Formulación V2

Sea $\mathcal{E}$ el conjunto de equipos, $\mathcal{F}$ el conjunto de fechas disponibles y $\mathcal{H}$ el conjunto de horarios disponibles tal que $E = |\mathcal{E}|$, $F = |\mathcal{F}|$ y $H = |\mathcal{H}|$. Entonces, definimos dos tipos de variables:

2.1 Variables de tipo 1

Las variables de tipo 1 se definen como (e_i, e_j, h, f) donde $e_i, e_j \in \mathcal{E}$ con $e_i \neq e_j$, $h \in \mathcal{H}$ y $f \in \mathcal{F}$ e indican que el equipo e_i juega como local en contra del equipo e_j como visitante en la fecha f y hora h. Por lo tanto, hay $E(E-1)FH$ variables de tipo 1, y las enumeramos como[2]

$$ENUM_1(e_i, e_j, h_k, f_l) = i - (i > j) * j - (j > i)(j-1) + E * j + E(E-1) * k + E(E-1)H * l$$

donde

[2] Los booleanos son tratados como 1 para *True* y 0 para *False*

$0 \leq i, j < E$, $0 \leq k < F$ y $0 \leq l < H$

2.2 Variables de tipo 2

Las variables de tipo 2 son de la forma (e, t, h, f) donde $e_{\in}\mathcal{E}$, $f \in \mathcal{F}$, $t \in \{v, l\}$ y $h \in \mathcal{H}$ e indican que el equipo e juega como t (local o visitante) en la fecha f y hora h. Por lo tanto, hay $2EFH$ variables de tipo 2, y las enumeramos como:

$ENUM_2(e_i, t, h_j, f_k) = E(E-1)FH + 1 + i + E * (t == v) + 2E * j\ + 2EH * k$

donde

$0 \leq i < E$, $0 \leq k < F$ y $0 \leq l < H$

así, nuestro CNF tiene un total de $E(E+1)FH$ variables.

2.3 Restricciones y Clausuras

- Todos los participantes deben jugar dos veces con cada uno de los otros participantes, una como "visitantes" y la otra como "locales".

  $(\forall e_i, e_j \in \mathcal{E} \mid e_i \neq e_j : (\exists f \in \mathcal{F}, h \in \mathcal{H} \mid\ : (e_i, e_j, f, h))\ )$

  Con un total de $E(E-1)$ clausuras tal que cada una está conformada por FH átomos. Esta restricción fue la causante de tener que usar dos tipos de variables, pues si sólo tuviéramos las de tipo 2, entonces la restricción quedaría de la forma:

  $(\forall e_i, e_j \in \mathcal{E} \mid e_i \neq e_j :$
  $\quad (\exists f \in \mathcal{F}, h \in \mathcal{H} \mid\ :$
  $\quad\quad (e_i, l, f, h) \wedge (e_j, v, f, h)$
  $\quad)$
  $)$

  Notemos que hay conjunciones dentro de las cláusuras, lo cual hace que ya no sea CNF, por lo tanto, usando la propiedad distirbutiva entre conjunciones y disyunciones, reescribimos la restricción anterior como

  $(\forall e_i, e_j \in \mathcal{E} \mid e_i \neq e_j :$
  $\quad (\forall A \in \mathcal{P}(\mathcal{F} \times \mathcal{H}) \mid A \neq \emptyset \wedge A \neq \mathcal{F} \times \mathcal{H} :$
  $\quad\quad (\exists (f, h) \in A \mid\ : (e_i, l, f, h)) \vee$
  $\quad\quad (\exists (f', h') \in (\mathcal{F} \times \mathcal{H} - A)\ \mid\ : (e_j, v, f', h'))$
  $\quad)$
  $)$

  Dándonos un total de $E(E-1)(2^{FH} - 2)$ clausuras con FH átomos cada una. Tener un número exponencial de clausuras hace que resolver el problema se vuelva inviable. Usando solo 8 días con 3 partidos diarios y 4 equipos nos daría un total de 201,326,568 clausuras solo para esta restricción.

- Equivalencia entre las variables de tipo 1 y de tipo 2. La variable (e_i, e_j, h, f) es `True` si y solo si $(e_i, l, h, f) \wedge (e_j, v, h, f)$ es `True`.

  $(\forall e_i, e_j \in \mathcal{E}, h \in \mathcal{H}, f \in \mathcal{F} \mid\ e_i \neq e_j :$
  $\quad (\neg(e_i, e_j, h, f) \vee (e_i, l, h, f)) \wedge$

$(\neg(e_i, e_j, h, f) \vee (e_j, v, h, f)) \wedge$
$(\neg(e_i, l, h, f) \vee \neg(e_j, v, h, f) \vee (e_i, e_j, h, f))$
$)$

Dándonos un total de $2E(E-1)HF$ clausuras de 2 átomos cada una más $E(E-1)HF$ clausuras de 3 átomos cada una.

- Dos juegos no pueden ocurrir al mismo tiempo.

  $(\forall e_i, e_j \in \mathcal{E}, f \in \mathcal{F}, h \in \mathcal{H} \mid i < j :$
  $(\neg(e_i, l, f, h) \vee \neg(e_j, l, f, h)) \wedge (\neg(e_i, v, f, h) \vee \neg(e_j, v, f, h)))$
  $)$

  Dándonos un total de $E(E-1)FH$ clausuras de 2 átomos cada una.

- Un participante puede jugar a lo sumo una vez por día.

  $(\forall e \in \mathcal{E}, f \in \mathcal{F}, h_i, h_j \in \mathcal{H} \mid i < j :$
  $(\neg(e, l, f, h_i) \vee \neg(e, l, f, h_j)) \wedge (\neg(e, l, f, h_i) \vee \neg(e, v, f, h_j)) \wedge$
  $(\neg(e, v, f, h_i) \vee \neg(e, l, f, h_j)) \wedge (\neg(e, v, f, h_i) \vee \neg(e, v, f, h_j))$
  $)$

  Dándonos un total de $2EFH(H-1)$ clausuras de 2 átomos cada una.

- Un participante no puede jugar de *visitante* en dos días consecutivos, ni de *local* dos días seguidos.

  $(\forall e \in \mathcal{E}, f_i \in \mathcal{F}, h, h' \in \mathcal{H} \mid i < F - 1 :$
  $(\neg(e, l, f_i, h) \vee \neg(e, l, f_{i+1}, h')) \wedge (\neg(e, v, f_i, h) \vee \neg(e, v, f_{i+1}, h'))$
  $)$

  Dándonos un total de $2E(F-1)H^2$ clausuras de 2 átomos cada una.

- Todos los juegos deben empezar en horas "en punto" (por ejemplo, las 13:00:00 es una hora válida pero las 13:30:00 no). Esta restricci'on no forma parte de las condiciones para las clausuras.
- Todos los juegos deben ocurrir entre una fecha inicial y una fecha final especificadas. Pueden ocurrir juegos en dichas fechas. Esta restricci'on no forma parte de las condiciones para las clausuras.
- Todos los juegos deben ocurrir entre un rango de horas especificado, el cuál será fijo para todos los días del torneo. Esta restricci'on no forma parte de las condiciones para las clausuras.
- A efectos prácticos, todos los juegos tienen una duración de dos horas. Esta restricci'on no forma parte de las condiciones para las clausuras.

  Estas última 4 condiciones definen los valores de $\mathcal{F}$ y $\mathcal{H}$. Usando la restricción de las fechas, identificamos las fechas en orden cronológico en el rango $[0, F)$ siendo F también el número de días entre las fechas inicial y final. Análogamente, usando las restricciones relacionadas con las horas, podemos calcular cuantos partidos se pueden realizar al día y en que horarios específicos, luego podemos identificar cada bloque del horario en orden cronológico en el rango $[0, H)$, donde H también es el número de partidos que se pueden realizar al día.

En total se tienen $O(E * F * H * (E + H))$ clausuras, por lo tanto, los factores principales que determinan la complejidad del problema es el número de equipos y partidos diarios.

3 Estructura del repositorio

El repositorio del proyecto tiene la siguiente estructura de arbol:

```
proyecto-3-ci5437/
|--- benchmarks/...
|--- bin/...
|--- Makefile
|--- README.md
+--- src/
    |--- benchmarksGen.py
    |--- closuresGen.cpp
    |--- closuresGenV2.cpp
    |--- glucose/...
    |--- L_informe/...
    |--- main.py
    |--- verifyClosures.py
    +--- verifyClosuresV2.py
```

donde

- `benchmarks/` contiene los casos de prueba en formato JSON junto a sus soluciones (en caso de haberla) en archivos `.ics`.
- `bin/` contiene los archivos binarios.
- `glucose/` contiene toda la implementación del SAT Solver Glucose.
- `L_informe/` contiene los archivos de LaTeX necesarios para generar este informe.
- `benchmarksGen.py` es el generador de casos de prueba.
- `closuresGen.cpp` y `closuresGenV2.cpp` son el código fuente para el generador de archivos `.cnf` usando la formulación original y V2 respectivamente.
- `verifyClosures.cpp` y `verifyClosuresV2.cpp` son el código fuente para verificar que las clausuras codificadas no se repiten y su número coincide con el calculado teóricamente usando la formulación original y V2 respectivamente.
- `main.py` se encarga de leer el archivo JSON, generar su correspondiente archivo `.cnf`, ejecutar el SAT Solver y traducir la solución en un archivo `.ics`.

4 Compilación y ejecución

Los archivos que necesitan ser compilados son el generador de CNF y Glucose. Para el primero, debemos ejecutar lo siguiente mientras nos encontramos en el directorio raiz del repositorio:

`make closuresGen` ó `make closuresGenV2`

Esto nos creará el archivo binario para la generación de CNF dentro del directorio `bin/`. Mientras que para Glucose mantenemos el archivo binario almacenado en `bin/` ya que la implementación del SAT Solver no será modificada por nosotros, así que no es necesario compilarlo en cada ocasión. Luego, para la ejecución de `main.py` se debe seguir la siguiente sintaxis:

```
python main.py   JSON   CLOSURES_GEN   SAT_SOLVER   OUTPUT_ICS
```

donde

- `JSON` es el archivo que contiene el problema.

- `CLOSURES_GEN` es el binario generador de CNF.
- `SAT_SOLVER` es el binario para resolver el CNF.
- `OUTPUT_ICS` es el archivo `ics` donde se almacenará la solución.

Por ejemplo si nos encontramos en el directorio raiz del repositorio, una posible ejecución puede ser

```
python src/main.py benchmarks/benchmark_e=4.json \
  bin/closuresGenV2.out \
  bin/glucose \
  benchmarks/benchmark_e=4.ics
```

4.1 Generador de Casos de Prueba

Para generar casos de prueba de forma automática puede ejecutar el script `benchmarksGen.py` siguiendo la sintaxis:

```
python benchmarksGen.py TEAMS DAYS DAILY_MATCHES
```

Esto creará un archivo JSON siguiendo la sintaxis de los torneos dado en el README tal que el nombre del torneo es *POMAC*, la fecha de inicio es *1999-11-28*, la fecha final es igual a la inicial mas `DAYS` días, la hora inicial es a las *00:00:00*, la hora final es igual a la inciial más 2 veces `DAILY_MATCHES` horas y los equipos serán desde *T0* hasta *TN* donde *N* es igual a `TEAMS`. El nombre del archivo será `benchmark_E<TEAMS>H<DAILY_MATCHES>F<DAYS>.json`. Por ejemplo si se ejecuta:

```
python benchmarksGen.py 15 55 7
```

Se obtendrá el archivo `benchmark_E15H7F55.json`

5 Resultados y Conlusiones

Los casos de prueba se definen por el número de equipos que participarán, el número de días que durará el torneo y el número máximo de partidos diarios. Decidimos probar tanto la formulación V1 como V2 y comparar sus desempeños. En caso de que alguno de estos dos no haya terminado un caso de prueba, se coloca una nota explicando la razón por la que se canceló su ejecución. En la *Figura 1* se encuentra el resumen de ejecución de cada caso de prueba.

No se podría decir a partir de los casos de pruebas cual versión de la formulación es mejor, ya que no hay una que tenga un mejor rendimiento de forma constante. Sin embargo, se puede notar que si la holgura (diferencia entre el número de días por el número de enfrentamientos diarios menos el número de partidos necesarios, es decir, $E * (E - 1)$) es muy ajustada, V1 tiende a tener un mejor rendimiento, mientras que al aumentar la holgura, V2 mejora considerablemente, llegando a tardar pocos segundos para casos de prueba grande. También podemos notar que a partir de cierto tamaño en los casos de prueba, V1 se vuelve inviable debido al uso de memoria. Esto se debe a la cantidad de casos de prueba que genera. Por ejemplo, para $(19, 90, 9)$, V1 genera alrededor de 23 millones de clausuras, mientras que V2 sólo genera 565 mil. Suponiendo que cada clausura ocupe 64 bytes (por poner un número), entonces las 23M de clausuras ocuparían más de 1GB de memoria. Para el caso $(20, 110, 10)$ ni siquiera vale la pena hacer el cálculo. Sin embargo, V2 puede llegar a ser muy ineficiente cuando la holgura es muy ajustada, como se nota en los casos de prueba donde no termina.

Se puede concluir que si el caso de prueba es lo suficientemente pequeño (menos de 19 equipos) y la holgura es poca, entonces se debe usar V1, en caso contrario V2.

# TEAMS	# DAYS	# DAILY MATCHES	SECONDS BY V1	SECONDS BY V2	NOTE
2	2	1	0.0019	0.00191	
3	6	1	0.011	0.0097	
4	6	2	0.0206	0.0183	
5	10	2	0.0372	0.1251	
6	11	3	0.04329	0.08993	
7	15	3	0.13106	0.16529	
8	15	4	4.92013	14.94994	
9	20	4	1.0293	0.29077	
10	20	5	341.09118	----	Se espero 1h por V2
11	24	5	54.71695	----	Se espero 15m por V2
12	28	6	20.43231	6.77356	
13	32	6	7.23963	----	Se espero 15m por V2
14	45	7	17.19769	8.47556	
15	55	7	26.01891	2.40701	
16	62	8	48.62187	5.07059	
17	70	8	69.97276	73.92808	
18	77	9	111.65029	12.24569	
19	90	9	----	5.27383	V1 excedio consumo de memoria
20	110	10	----	12.52614	V1 excedio consumo de memoria

Figura 1. Resultados de los casos de prueba aplicados a ambas versiones de la Formulación.