Tutorial 4: Sports Tournament

Three (e) universities organize a sports tournament. Each university has four (q) teams. The tournament is divided in six (l) rounds. At each round, two teams compete for each of the six (d) possible sports.

The planning of the tournament must respect the following constraints:

- a team only plays once during each round;
- a team must not play against another team of the same university;
- a team must not play twice the same sport;
- a team must not play twice against the same team;
- a team must play exactly against three ($\frac{l}{e-1}$) teams of each of the two ($e-1$) other universities (only if the division $\frac{l}{e-1}$ has a null remainder and if the number of teams q is greater or equal than $\frac{l}{e-1}$).

Solve this problem with FaCiLe.

CAUTION: in this tutorial (and almost all others...), it is forbidden to use anonymous numerical constants other than 0, 1 or 2 after the definitions of the parameters $e = 3$, $q = 4$, $l = 6$ and $d = 6$.

The solution can be presented as in the following array (the first university has teams numbered 1, 2, 3 et 4, the second has teams 5, 6, 7, 8...), where each row corresponds to a round:

rounds	sports					
	1	2	3	4	5	6
1	[1,5]	[2,6]	[3,9]	[4,10]	[7,11]	[8,12]
2	[7,12]	[8,11]	[1,6]	[2,5]	[3,10]	[4,9]
3			...			

To obtain results efficiently, you must break some symmetries among equivalent solutions with auxiliary constraints:

- game `[1,5]` is equivalent to game `[5,1]`;
- the order of the rounds has no significance, therefore the first team of the first game of all rounds can be arbitrarily ordered.

Too help developing the constraint program:

- Start with *few* constraints.
- First select the variable with the smallest size during the search.
- You cannot directly state constraints on variables tuples. A workaround is to *encode* a couple with a single integer variable according to the following bijection: let $x \in [0, n-1]$ and $y \in [0, m-1]$, $f(x, y) = x * m + y$ (and reciprocally, $f^{-1}(z) = (\frac{z}{m}, z \bmod m)$). The same kind of transformation is used to encode a $n \times m$ matrix with a nm vector: `flat[i*m+j] = mat[i][j]` and `mat[k/m][k%m] = flat[k]`.
- For the last constraint, the same encoding principle can be used to represent the couples of variables (team, opposing university) for each game with a **single** integer variable. The cardinal of each possible value can be then precisely set with a global cardinality constraint (`Gcc.cstr` with level `Gcc.Medium`).