Pràctica de PDA, Constraints i SAT (2024-2025)

Informe de PdA: Constraints i SAT: MiniZinc

Cels Esteva Planas
DNI: 41679856S
ID d'estudiant: u1978909
Correu electrònic: u1978909@campus.udg.edu
GEINF
Universitat de Girona

Índex

1 ViewPoint

Comento que no he sabut com fer que t1fix per a seguidors em retorni el mateix 233497 espectadors.

1.1 Explicació del ViewPoint

Només hi ha un ViewPoint en el model. El ViewPoint consisteix d'una taula de Camp/Dia a on s'hi guarda un set dels equips que juguen aquell dia en aquell camp. El set només pot contenir 0 o 2 elements.

1.1.1 Variables Triades

- `array[1..ndays,1..nstadiums] of var set of 1..nteams: matches`
  - variable per a declarar els partits. És una matriu de dies * estadis a on s'hi guarden els partits. Que són una parella de equips guardats en un set.
- `array[1..ndays,1..nstadiums] of var 0..(max(tifosi)*2): totalTifosi;`
  - per cada partit, és calcula quants no podran veure el partit.
  - és una variable auxiliar, es podria fer una suma d'aquest directament per a tenir missedMatch
- `var minEspectadorsPerduts..maxEspectadorsPerduts: missedMatch`
  - és la suma del total que s'han perdut els partits.
- `array[1..nteams,1..(ndays-1)] of var 0..max(distancies): arrayDistancies`
  - matriu a on s'hi guarden les distàncies recorregudes segons el l'equip i el dia.
  - és una variable auxiliar, es podria fer directament la suma de les distàncies a kilometers.
- `var minDistance..maxDistance: kilometers`
  - suma total dels kilòmetres recorreguts.

1.1.2 Dominis de les variables

- `array[1..ndays,1..nstadiums] of var set of 1..nteams: matches`
  - el domini es entre 1..nteams, i cada entrada només pot tenir cardinalitat 0 o 2
- `array[1..ndays,1..nstadiums] of var 0..(max(tifosi)*2): totalTifosi`
  - el domini és entre 0..(max(tifosi)*2). No pot haver-hi menys de 0 espectadors que s'hagin perdut el partit per cada partit, i el màxim és el total d'espectadors dels dos equips sumat menys la capacitat del camp. Aquí només he calculat el màxim d'espectadors per dos.
- `var minEspectadorsPerduts..maxEspectadorsPerduts: missedMatch`
  - el domini va del mínim d'espectadors al màxim d'espectadors.

  - El màxim d'espectadors és calcula fent la suma per a totes les combinacions d'equips que es poden jugar i calculant per a cada combinació el màxim de d'espectadors dels equips menys la capacitat del camp mínima, tot amb arrays en ordre decreixent per als espectadors i creixient per a les capacitats per a poder aproximar millor el màxim.
  - el mínim es calcula agafant els espectadors de cada equip, sumant-los i restant-li la capacitat màxima.

- `array[1..nteams,1..(ndays-1)] of var 0..max(distancies): arrayDistancies`
  - el domini de l'arrey de les distancies recorregudes va de 0 a la distància màxima que es pot recorre.

- `var minDistance..maxDistance: kilometers`
  - el domini dels kilòmetres va del minDistance fins a maxDistance
  - minDistance és la suma de per a tots els estadis de les dues distancies més petites sense comptar anar a ell mateix (ja que hi ha dos equips que aniran al camp cada cop). Això és multiplica per nmatchesperstadium per a tenir el càlcul de la distància mínima de tots els equips.
  - maxDistance és la suma de per a tots els estadis de les dues distàncies més grans (ja que hi ha dos equips que aniran al camp cada cop) multiplicat per nmatchesperstadium.

1.1.3 Restriccions

Per a posar els valors a la taula, només a calgut fer un subset de les dades d'entrada. També aprofito per a forçar que els sets dels partits només puguin tenir 2 o 0 equips, i que per tant no hi hagi partits amb un nombre diferent d'equips.

```
constraint forall(i in 1..ndays, j in 1..nstadiums) (
  card(matches[i,j]) in {0,2} /\
  fixes[i,j] subset matches[i,j]
);
```

La restricció del partition_set força a que l'array de sets donat es reparteixi el set donat, sense repeticions, ni que en faltin. Això força a que cada dia juguin tots els equips. Com que la cardinalitat força a que només pugui haver-hi 2 o 0 equips, això força a que hi hagi en un dia nmatchesperday.

```
constraint forall(day in 1..ndays)(
  partition_set(matches[day,1..nstadiums],TEAMS)
);
```

La següent restricció és igual que l'anterior però per estadis, així a cada estadi hi juguen tots els equips, però només un cop. Aconseguiex també que només hi hagi nmatchesperstadium.

```
constraint forall(estadi in 1..nstadiums)(
  partition_set(matches[1..ndays,estadi],TEAMS)
);
```

Per a calcular les restriccons dels espectadors es fa la suma de tots els espectadors perduts. Per a trobar tots els espectadors perduts es fa el maxim de la suma d'espectadors menys la capacitat comparat amb 0 per a no posar nombres negatius. Això es fa per a tots els partits jugats (i no jugats, però no sumarà res).

```
constraint forall(day in 1..ndays, estadi in 1..nstadiums) (
  totalTifosi[day,estadi] = max((sum(team in matches[day, estadi]) (
      tifosi[team])) - capacitats[estadi],0)
);

constraint missedMatch = sum(totalTifosi); %variable auxiliar
);
```

Les restriccions de les distàncies són molt costoses, ja que ha de fer molts de càlculs. És un bucle per a tots els equips, i per a tots els dies, a on es mira per cada equip a quin estadi a jugat en un dia i el següent per a trobar la distància recorreguda, i guardar-la per a després sumar-la.

```
constraint forall(team in 1..nteams, day in 1..(ndays-1)) (
  let {
    var int: stadium_day = sum(stadium in 1..nstadiums where team in
        matches[day, stadium]) (stadium),
    var int: stadium_next_day = sum(stadium in 1..nstadiums where team in
         matches[day+1, stadium]) (stadium)
  } in
  arrayDistancies[team, day] = distancies[stadium_day, stadium_next_day]
);

constraint kilometers = sum(arrayDistancies); %variable auxiliar
```

De les restriccions, només una és global: partition_set.

1.1.4 Upper bound i lower bound de la funció objectiu

La funció objectiu conté upper bounds i lower bounds depenent de que s'optimitzi.

- Espectadors:

```
int: maxEspectadorsPerduts = sum(t1 in 1..(nteams-1), t2 in (t1+1)..
    nteams) (
  max(drecreasingOrderedTifosi[t1] + drecreasingOrderedTifosi[t2] -
      increasingOrderedCapacitats[t1],0)
);

int: minEspectadorsPerduts = sum(t1 in 1..(nteams-1), t2 in (t1+1)..
    nteams) (
  max(tifosi[t1] + tifosi[t2] - max(capacitats),0)
);

var minEspectadorsPerduts..maxEspectadorsPerduts: missedMatch;
```

  - maxEspectadors: s'agafen els espectadors de cada parella d'equips i es busca a quin esta

- minEspectadors: s'agafen els espectadors de cada parella d'equips i es resten a la capacitat màxima de l'estadi al que aniran de forma que minimitzi l'upper bound. Així primer s'hagafen els que tenen la diferència més gran(entre tifosi i capacitat) per a maximitzar els espectadors.

- Kilometres:

```
int: maxDistance = sum(estadi in 1..nstadiums)(
  sum(playersPerMatch in 1..2)(
    reverse(sort([distancies[estadi, j] | j in 1..nstadiums where j
        != estadi]))[playersPerMatch]
  )
) * nmatchesperstadium;


int: minDistance = sum(estadi in 1..nstadiums)(
    sum(playersPerMatch in 1..2) (
      sort([distancies[estadi, j] | j in 1..nstadiums where j !=
          estadi])[playersPerMatch]
    )
) * nmatchesperstadium;

var minDistance..maxDistance: kilometers;
```

- minDistance és la suma de per a tots els estadis de les dues distancies més petites sense comptar anar a ell mateix (ja que hi ha dos equips que aniran al camp cada cop). Això és multiplica per nmatchesperstadium per a tenir el càlcul de la distància mínima de tots els equips.
- maxDistance és la suma de per a tots els estadis de les dues distàncies més grans (ja que hi ha dos equips que aniran al camp cada cop) multiplicat per nmatchesperstadium.

Per als que són una combinació, es fan servir els 2 upper bounds i lower bounds.

1.1.5 Restriccions implicades i simetries

De les restriccions implicades, quan es posa el cardinalitat a 2 o 0 i es posa la restricció de partition_set, aquesta ja porta implicada nmatchesperday i nmatchesperstadium i també, com que són sets, que en un dia i un estadi només hi pot haver un partit. De simetries no n'he trobat, però els sets ja treuen simetries de partit i-j, j-i.

1.1.6 Estratègies de cerca

Per a satisfy i espectadors no n'he posat perquè no he trobat que milloressin la cerca, per a km i espectadors/kilometres i kilometres/espectadors s'hi que n'he posat, perquè millorava força la cerca.

Per al de KM he fet servir el següent:

```
solve :: int_search(arrayDistancies, first_fail, indomain_min) minimize
    kilometers;
```

Per a espectadors/kilometres he fet servir el següent:

```
int: maxKilometersRounded = ceil(pow(10, ceil(log(10, maxDistance))));

solve :: seq_search([
            int_search(totalTifosi, first_fail, indomain_min),
            int_search(arrayDistancies, first_fail, indomain_min)])
      minimize missedMatch*maxKilometersRounded+kilometers ;
```

La part de cerca fa servir 2, un per a trobar primer el mínim de tifosi, i l'altre per a l'arreyDistancies un cop ja ha trobat el mínim de tifosi.

Per a kilometres/espectadors he fet servir l'oposat a l'anterior.

```
int: maxEspectadorsRounded = ceil(pow(10, ceil(log(10,
    maxEspectadorsPerduts))));


solve :: seq_search([
            int_search(arrayDistancies, first_fail, indomain_min),
            int_search(totalTifosi, first_fail, indomain_min)
         ]) minimize kilometers*maxEspectadorsRounded + missedMatch;
```

En tots he fet servir first_fail, perquè he trobat que anava millor, i indomain_min per assignar la variable el valor més petit possible.

Per als dos últims calculo el màxim de nombres que pot tenir el segon valor a maximitzar i ho multiplico per al primer a maximitzar pe atenir les prioritats correctes. Això crea valors molt grans que alguns solvers no poden tenir en compte.

1.2 Execució del ViewPoint

El test t2fix dona unsat perquè hi ha un equip fora del rang del set 11 > (1..10) El primer valor són els KM i el segon els seguidors que s'han quedat a fora. i el tercer els ms . El timeout l'he posat als 6 MINUTS, per a poder fer més proves.

Satisfy	Chuffed	Gecode	OR Tools
t0	536 44000 - 225	510 46000 - 145	498 44000 - 273
t1	1151 224000 - 421	1302 277000 - 150	1150 196000 - 447
t1fix	1309 217993 - 367	1433 246198 - 144	1518 272627 - 338
t2	1991 427000 - 807	2255 407000 - 158	2142 395000 - 872
t2fix	UNSAT - 140	UNSAT - 254	UNSAT - 130
t3	T.O.	3268 485000 - 169	3066 408000 - 1723
t3fix	2949 400000	2789 471000 - 153	2738 391000 - 1564
t4	T.O.	6346 844000 - 167	5674 674500 - 3139
t4fix	5964 733500 - 3197	7273 812500 - 163	7349 773500 - 2765

Taula 1: Taula d'execucions trobant la primera solució.

ESPECTADORS	Chuffed	Gecode	OR Tools
t0	596 31000 - 298	574 31000 - 355	553 31000 - 341
t1	1412 212000 - T.O.	1380 187000 - T.O.	1386 176000 - 1631
t1fix	1462 207834 - T.O.	1440 207834 - 462	1462 207834 - 2208
t2	2047 424000 - T.O.	2304 377000 - T.O.	2627 373000 - 3972
t2fix	UNSAT	UNSAT	UNSAT
t3	T.O.	3990 412000 - T.O.	3945 340000 - T.O.
t3fix	4499 398000 - T.O.	2840 403000 - T.O.	4322 342000 - T.O.
t4	T.O.	6430 756000 - T.O.	7413 606500 - T.O.
t4fix	5711 686500 - T.O.	6777 758500 - T.O.	7316 609500 - T.O.

Taula 2: Taula d'execucions minimitzant per espectadors que no poden anar al partit.

KM	Chuffed	Gecode	OR Tools
t0	456 36000 - 249	T.O.	456 47000 - 424
t1	872 196000 - T.O.	T.O.	866 225000 - T.O.
t1fix	1185 239098 - 4m 2s	1253 260983 - T.O.	1185 239098 - 2991
t2	1518 377000 - T.O.	T.O.	1438 477000 - T.O.
t2fix	UNSAT	UNSAT	UNSAT
t3	T.O.	T.O.	2054 487000 - T.O.
t3fix	T.O.	T.O.	2090 347000 - T.O.
t4	T.O.	T.O.	3996 769000 - T.O.
t4fix	T.O.	T.O.	4319 790500 - T.O.

Taula 3: Taula d'execucions minimitzant per kilòmetres que cada equip ha de viatjar.

ESPECTADORS/KM	Chuffed	Gecode	OR Tools
t0	468 31000 T.O.	T.O.	456 31000 - 706.
t1	1167 212000 - T.O.	ERROR	866 176000 - T.O.
t1fix	T.O.	ERROR	1440 207834 - 5602
t2	T.O.	ERROR	1543 373000 - T.O.
t2fix	UNSAT	ERROR	UNSAT
t3	T.O.	ERROR	2493 340000 - T.O.
t3fix	T.O.	ERROR	2892 342000 - T.O.
t4	T.O.	ERROR	5583 606500 - T.O.
t4fix	T.O.	ERROR	5567 609500 - T.O.

Taula 4: Taula d'execucions minimitzant per a espectadors i després kilòmetres.

KM/ESPECTADORS	Chuffed	Gecode	OR Tools
t0	456 31000	T.O.	456 31000 - 813
t1	T.O.	ERROR	866 176000 - T.O.
t1fix	1185 239098 - 3m 30s	ERROR	1185 239098 - T.O.
t2	T.O.	ERROR	1452 373000 - T.O.
t2fix	UNSAT	ERROR	UNSAT
t3	T.O.	ERROR	2101 340000 - T.O.
t3fix	T.O.	ERROR	2139 364000 - T.O.
t4	T.O.	ERROR	3813 631500 - T.O.
t4fix	T.O.	ERROR	4680 672000 - T.O.

Taula 5: Taula d'execucions minimitzant per a kilòmetres i després espectadors.

A Codi

A.1 Satisfy

```
include "globals.mzn";

%%%%%% llegir variables
int: nmachesperday;
int: nmatchesperstadium;
int: ndays;
int: nteams;
int: nstadiums;

set of int: NDAYS = 1..ndays;
set of int: TEAMS = 1..nteams;

int: matchesNotPlayedPerDay = ndays-nmachesperday;
int: matchesNotPlayedPerStadium = nstadiums-nmatchesperstadium;


array[1..ndays,1..nstadiums] of set of int: fixes;
%files ->dies;  estadis -> columnes
array[1..nstadiums,1..nstadiums] of int: distancies;
array[1..nteams] of int: tifosi;
array[1..nstadiums] of int: capacitats;


array[1..nteams] of int: increasingOrderedTifosi = sort(tifosi);
array[1..nteams] of int: drecreasingOrderedTifosi = reverse(sort(tifosi))
    ;
array[1..nstadiums] of int: increasingOrderedCapacitats = sort(capacitats
    );
array[1..nstadiums] of int: drecreasingOrderedCapacitats = reverse(sort(
    capacitats));

%Variables.
array[1..ndays,1..nstadiums] of var set of 1..nteams: matches; %Equip -
    Equip.

constraint forall(i in 1..ndays, j in 1..nstadiums) (
  card(matches[i,j]) in {0,2} /\
  fixes[i,j] subset matches[i,j]
);


%constraints
%a team can only play once per day:
constraint forall(day in 1..ndays)(
  partition_set(matches[day,1..nstadiums],TEAMS)
);

%a team can only play in one stadium
constraint forall(estadi in 1..nstadiums)(
```

```
  partition_set(matches[1..ndays,estadi],TEAMS)
);


solve satisfy;


function string: formatSet(var set of int: s) =
  if(card(fix(s)) == 0) then
    "     "
  else
  join("-", [show_int(2,i) | i in fix(s)])
  endif;


output
["\t:" ++ concat([" Es" ++ show_int(2,team) ++ "  |" | team in 1..
    nstadiums]) ++ "\n"]
++
["-" | i in 1..((8*nstadiums)+9)]
++ ["\n"] ++
["Dia   " ++ show_int(2,dia) ++ ": " ++ concat([formatSet(matches[dia,
    estadi]) ++ " | " | estadi in 1..nstadiums]) ++ "\n" | dia in 1..ndays
    ]
++
["-" | i in 1..((8*nstadiums)+9)]
++ ["\n"] ++

["KMs TOTALS: " ++ show(
  sum(team in 1..nteams, day in 1..(ndays-1)) (
    let {
      var int: stadium_day = sum(stadium in 1..nstadiums where team in
          matches[day, stadium]) (stadium),
      var int: stadium_next_day = sum(stadium in 1..nstadiums where team
          in matches[day+1, stadium]) (stadium)
    } in
    distancies[stadium_day, stadium_next_day]
  )
)]

++ ["\n"] ++

["SEGUIDORS FORA: " ++ show(
  sum(day in 1..ndays, estadi in 1..nstadiums) (
    max((sum(team in matches[day, estadi]) (tifosi[team])) - capacitats[
        estadi],0)
  ))
 ]
```

A.2 Espectadors

```
include "globals.mzn";

%%%%% llegir variables
int: nmachesperday;
int: nmatchesperstadium;
int: ndays;
int: nteams;
int: nstadiums;

set of int: NDAYS = 1..ndays;
set of int: TEAMS = 1..nteams;

int: matchesNotPlayedPerDay = ndays-nmachesperday;
int: matchesNotPlayedPerStadium = nstadiums-nmatchesperstadium;


array[1..ndays,1..nstadiums] of set of int: fixes;
%files ->dies;  estadis -> columnes
array[1..nstadiums,1..nstadiums] of int: distancies;
array[1..nteams] of int: tifosi;
array[1..nstadiums] of int: capacitats;


array[1..nteams] of int: increasingOrderedTifosi = sort(tifosi);
array[1..nteams] of int: drecreasingOrderedTifosi = reverse(sort(tifosi))
    ;
array[1..nstadiums] of int: increasingOrderedCapacitats = sort(capacitats
    );
array[1..nstadiums] of int: drecreasingOrderedCapacitats = reverse(sort(
    capacitats));

%Variables.
array[1..ndays,1..nstadiums] of var set of 1..nteams: matches; %Equip -
    Equip.

constraint forall(i in 1..ndays, j in 1..nstadiums) (
  card(matches[i,j]) in {0,2} /\
  fixes[i,j] subset matches[i,j]
);


%constraints
%a team can only play once per day:
constraint forall(day in 1..ndays)(
  partition_set(matches[day,1..nstadiums],TEAMS)
);

%a team can only play in one stadium
constraint forall(estadi in 1..nstadiums)(
  partition_set(matches[1..ndays,estadi],TEAMS)
);
```

```
48
49
50
51
52
53
54  int: maxEspectadorsPerduts = sum(t1 in 1..(nteams-1), t2 in (t1+1)..
        nteams) (
55    max(drecreasingOrderedTifosi[t1] + drecreasingOrderedTifosi[t2] -
          increasingOrderedCapacitats[t1],0)
56  );
57
58  int: minEspectadorsPerduts = sum(t1 in 1..(nteams-1), t2 in (t1+1)..
        nteams) (
59    max(tifosi[t1] + tifosi[t2] - max(capacitats),0)
60  );
61
62  var minEspectadorsPerduts..maxEspectadorsPerduts: missedMatch;
63
64  array[1..ndays,1..nstadiums] of var 0..(max(tifosi)*2): totalTifosi;
65
66  constraint forall(day in 1..ndays, estadi in 1..nstadiums) (
67    totalTifosi[day,estadi] = max((sum(team in matches[day, estadi]) (
          tifosi[team])) - capacitats[estadi],0)
68  );
69
70  constraint missedMatch = sum(totalTifosi);
71
72  solve minimize missedMatch;
73
74
75
76
77  function string: formatSet(var set of int: s) =
78    if(card(fix(s)) == 0) then
79      "     "
80    else
81    join("-", [show_int(2,i) | i in fix(s)])
82    endif;
83
84  output
85  ["\t:" ++ concat([" Es" ++ show_int(2,team) ++ "  |" | team in 1..
        nstadiums]) ++ "\n"]
86  ++
87  ["-" | i in 1..((8*nstadiums)+9)]
88  ++ ["\n"] ++
89  ["Dia   " ++ show_int(2,dia) ++ ": " ++ concat([formatSet(matches[dia,
        estadi]) ++ " | " | estadi in 1..nstadiums]) ++ "\n" | dia in 1..ndays
        ]
90  ++
91  ["-" | i in 1..((8*nstadiums)+9)]
92  ++ ["\n"] ++
93
```

```
["KMs TOTALS: " ++ show(
  sum(team in 1..nteams, day in 1..(ndays-1)) (
    let {
      var int: stadium_day = sum(stadium in 1..nstadiums where team in
          matches[day, stadium]) (stadium),
      var int: stadium_next_day = sum(stadium in 1..nstadiums where team
          in matches[day+1, stadium]) (stadium)
    } in
    distances[stadium_day, stadium_next_day]
  )
)]

++ ["\n"] ++
["SEGUIDORS FORA: " ++ show(missedMatch)];
```

A.3 Kilometres

```
include "globals.mzn";

%%%%% llegir variables
int: nmachesperday;
int: nmatchesperstadium;
int: ndays;
int: nteams;
int: nstadiums;

set of int: NDAYS = 1..ndays;
set of int: TEAMS = 1..nteams;

int: matchesNotPlayedPerDay = ndays-nmachesperday;
int: matchesNotPlayedPerStadium = nstadiums-nmatchesperstadium;


array[1..ndays,1..nstadiums] of set of int: fixes;
%files ->dies;  estadis -> columnes
array[1..nstadiums,1..nstadiums] of int: distancies;
array[1..nteams] of int: tifosi;
array[1..nstadiums] of int: capacitats;


array[1..nteams] of int: increasingOrderedTifosi = sort(tifosi);
array[1..nteams] of int: drecreasingOrderedTifosi = reverse(sort(tifosi))
    ;
array[1..nstadiums] of int: increasingOrderedCapacitats = sort(capacitats
    );
array[1..nstadiums] of int: drecreasingOrderedCapacitats = reverse(sort(
    capacitats));

%Variables.
array[1..ndays,1..nstadiums] of var set of 1..nteams: matches; %Equip -
    Equip.

constraint forall(i in 1..ndays, j in 1..nstadiums) (
  card(matches[i,j]) in {0,2} /\
  fixes[i,j] subset matches[i,j]
);


%constraints
%a team can only play once per day:
constraint forall(day in 1..ndays)(
  partition_set(matches[day,1..nstadiums],TEAMS)
);

%a team can only play in one stadium
constraint forall(estadi in 1..nstadiums)(
  partition_set(matches[1..ndays,estadi],TEAMS)
);
```

```
48
49
50
51
52
53
54
55
56  int: maxDistance = sum(estadi in 1..nstadiums)(
57    sum(playersPerMatch in 1..2)(
58      reverse(sort([distancies[estadi, j] | j in 1..nstadiums where j !=
            estadi]))[playersPerMatch]
59    )
60  ) * nmatchesperstadium;
61
62
63  int: minDistance = sum(estadi in 1..nstadiums)(
64      sum(playersPerMatch in 1..2) (
65        sort([distancies[estadi, j] | j in 1..nstadiums where j != estadi])
              [playersPerMatch]
66      )
67  ) * nmatchesperstadium;
68
69  var minDistance..maxDistance: kilometers;
70
71  array[1..nteams,1..(ndays-1)] of var 0..max(distancies): arrayDistancies;
72
73  constraint forall(team in 1..nteams, day in 1..(ndays-1)) (
74    let {
75      var int: stadium_day = sum(stadium in 1..nstadiums where team in
            matches[day, stadium]) (stadium),
76      var int: stadium_next_day = sum(stadium in 1..nstadiums where team in
             matches[day+1, stadium]) (stadium)
77    } in
78    arrayDistancies[team, day] = distancies[stadium_day, stadium_next_day]
79  );
80
81  constraint kilometers = sum(arrayDistancies);
82
83  solve :: int_search(arrayDistancies, first_fail, indomain_min) minimize
        kilometers;
84
85
86
87  function string: formatSet(var set of int: s) =
88    if(card(fix(s)) == 0) then
89      "     "
90    else
91    join("-", [show_int(2,i) | i in fix(s)])
92    endif;
93
94
95  output
```

```
96  ["\t:" ++ concat([" Es" ++ show_int(2,team) ++ "  |" | team in 1..
        nstadiums]) ++ "\n"]
97  ++
98  ["-" | i in 1..((8*nstadiums)+9)]
99  ++ ["\n"] ++
100 ["Dia   " ++ show_int(2,dia) ++ ": " ++ concat([formatSet(matches[dia,
        estadi]) ++ " | " | estadi in 1..nstadiums]) ++ "\n" | dia in 1..ndays
        ]
101 ++
102 ["-" | i in 1..((8*nstadiums)+9)]
103 ++ ["\n"] ++
104 ["KMs TOTALS: " ++ show(kilometers)]
105 ++ ["\n"] ++
106
107 ["SEGUIDORS FORA: " ++ show(
108   sum(day in 1..ndays, estadi in 1..nstadiums) (
109     max((sum(team in matches[day, estadi]) (tifosi[team])) - capacitats[
            estadi],0)
110   ))
111  ]
```

A.4 Espectadors i Kilometres

```
include "globals.mzn";

%%%%% llegir variables
int: nmachesperday;
int: nmatchesperstadium;
int: ndays;
int: nteams;
int: nstadiums;

set of int: NDAYS = 1..ndays;
set of int: TEAMS = 1..nteams;

int: matchesNotPlayedPerDay = ndays-nmachesperday;
int: matchesNotPlayedPerStadium = nstadiums-nmatchesperstadium;


array[1..ndays,1..nstadiums] of set of int: fixes;
%files ->dies;  estadis -> columnes
array[1..nstadiums,1..nstadiums] of int: distancies;
array[1..nteams] of int: tifosi;
array[1..nstadiums] of int: capacitats;


array[1..nteams] of int: increasingOrderedTifosi = sort(tifosi);
array[1..nteams] of int: drecreasingOrderedTifosi = reverse(sort(tifosi))
    ;
array[1..nstadiums] of int: increasingOrderedCapacitats = sort(capacitats
    );
array[1..nstadiums] of int: drecreasingOrderedCapacitats = reverse(sort(
    capacitats));

%totes les parelles d'equips. Es sumen de m s segudors a menys seguidors
     i es resten als camps amb menys visitants a m s visitants.
int: maxEspectadorsPerduts = sum(t1 in 1..(nteams-1), t2 in (t1+1)..
    nteams) (
  max(drecreasingOrderedTifosi[t1] + drecreasingOrderedTifosi[t2] -
      increasingOrderedCapacitats[t1],0)
);

int: minEspectadorsPerduts = sum(t1 in 1..(nteams-1), t2 in (t1+1)..
    nteams) (
  max(tifosi[t1] + tifosi[t2] - max(capacitats),0)
);

var minEspectadorsPerduts..maxEspectadorsPerduts: missedMatch;


int: maxDistance = sum(estadi in 1..nstadiums)(
  sum(playersPerMatch in 1..2)(
    reverse(sort([distancies[estadi, j] | j in 1..nstadiums where j !=
        estadi]))[playersPerMatch]
```

```
  )
) * nmatchesperstadium;


int: minDistance = sum(estadi in 1..nstadiums)(
    sum(playersPerMatch in 1..2) (
      sort([distancies[estadi, j] | j in 1..nstadiums where j != estadi])
          [playersPerMatch]
    )
) * nmatchesperstadium;

var minDistance..maxDistance: kilometers;

%Variables.
array[1..ndays,1..nstadiums] of var set of 1..nteams: matches; %Equip -
    Equip.

constraint forall(i in 1..ndays, j in 1..nstadiums) (
  card(matches[i,j]) in {0,2} /\
  fixes[i,j] subset matches[i,j]
);


%constraints
%a team can only play once per day:
constraint forall(day in 1..ndays)(
  partition_set(matches[day,1..nstadiums],TEAMS)
);

%a team can only play in one stadium
constraint forall(estadi in 1..nstadiums)(
  partition_set(matches[1..ndays,estadi],TEAMS)
);


array[1..ndays,1..nstadiums] of var 0..(max(tifosi)*2): totalTifosi;

constraint forall(day in 1..ndays, estadi in 1..nstadiums) (
  totalTifosi[day,estadi] = max((sum(team in matches[day, estadi]) (
      tifosi[team])) - capacitats[estadi],0)
);

constraint missedMatch = sum(totalTifosi);


array[1..nteams,1..(ndays-1)] of var 0..max(distancies): arrayDistancies;

constraint forall(team in 1..nteams, day in 1..(ndays-1)) (
  let {
    var int: stadium_day = sum(stadium in 1..nstadiums where team in
        matches[day, stadium]) (stadium),
    var int: stadium_next_day = sum(stadium in 1..nstadiums where team in
         matches[day+1, stadium]) (stadium)
```

```
  } in
  arrayDistancies[team, day] = distancies[stadium_day, stadium_next_day]
);

constraint kilometers = sum(arrayDistancies);

int: maxKilometersRounded = ceil(pow(10, ceil(log(10, maxDistance))));

solve :: seq_search([
             int_search(totalTifosi, first_fail, indomain_min),
             int_search(arrayDistancies, first_fail, indomain_min)])
      minimize missedMatch*maxKilometersRounded+kilometers ;



function string: formatSet(var set of int: s) =
  if(card(fix(s)) == 0) then
    "     "
  else
  join("-", [show_int(2,i) | i in fix(s)])
  endif;


output
["\t:" ++ concat([" Es" ++ show_int(2,team) ++ "  |" | team in 1..
    nstadiums]) ++ "\n"]
++
["-" | i in 1..((8*nstadiums)+9)]
++ ["\n"] ++
["Dia   " ++ show_int(2,dia) ++ ": " ++ concat([formatSet(matches[dia,
    estadi]) ++ " | " | estadi in 1..nstadiums]) ++ "\n" | dia in 1..ndays
    ]
++
["-" | i in 1..((8*nstadiums)+9)]
++ ["\n"] ++
["KMs TOTALS: " ++ show(kilometers)]
++ ["\n"] ++
["SEGUIDORS FORA: " ++ show(missedMatch)]
```

A.5 Kilometres i Espectadors

```
include "globals.mzn";

%%%%% llegir variables
int: nmachesperday;
int: nmatchesperstadium;
int: ndays;
int: nteams;
int: nstadiums;

set of int: NDAYS = 1..ndays;
set of int: TEAMS = 1..nteams;

int: matchesNotPlayedPerDay = ndays-nmachesperday;
int: matchesNotPlayedPerStadium = nstadiums-nmatchesperstadium;


array[1..ndays,1..nstadiums] of set of int: fixes;
%files ->dies;  estadis -> columnes
array[1..nstadiums,1..nstadiums] of int: distancies;
array[1..nteams] of int: tifosi;
array[1..nstadiums] of int: capacitats;


array[1..nteams] of int: increasingOrderedTifosi = sort(tifosi);
array[1..nteams] of int: drecreasingOrderedTifosi = reverse(sort(tifosi))
    ;
array[1..nstadiums] of int: increasingOrderedCapacitats = sort(capacitats
    );
array[1..nstadiums] of int: drecreasingOrderedCapacitats = reverse(sort(
    capacitats));

%totes les parelles d'equips. Es sumen de m s segudors a menys seguidors
     i es resten als camps amb menys visitants a m s visitants.
int: maxEspectadorsPerduts = sum(t1 in 1..(nteams-1), t2 in (t1+1)..
    nteams) (
  max(drecreasingOrderedTifosi[t1] + drecreasingOrderedTifosi[t2] -
      increasingOrderedCapacitats[t1],0)
);

int: minEspectadorsPerduts = sum(t1 in 1..(nteams-1), t2 in (t1+1)..
    nteams) (
  max(tifosi[t1] + tifosi[t2] - max(capacitats),0)
);

var minEspectadorsPerduts..maxEspectadorsPerduts: missedMatch;


int: maxDistance = sum(estadi in 1..nstadiums)(
  sum(playersPerMatch in 1..2)(
    reverse(sort([distancies[estadi, j] | j in 1..nstadiums where j !=
        estadi]))[playersPerMatch]
```

```
  )
) * nmatchesperstadium;


int: minDistance = sum(estadi in 1..nstadiums)(
    sum(playersPerMatch in 1..2) (
      sort([distancies[estadi, j] | j in 1..nstadiums where j != estadi])
          [playersPerMatch]
    )
) * nmatchesperstadium;

var minDistance..maxDistance: kilometers;

%Variables.
array[1..ndays,1..nstadiums] of var set of 1..nteams: matches; %Equip -
    Equip.

constraint forall(i in 1..ndays, j in 1..nstadiums) (
  card(matches[i,j]) in {0,2} /\
  fixes[i,j] subset matches[i,j]
);


%constraints
%a team can only play once per day:
constraint forall(day in 1..ndays)(
  partition_set(matches[day,1..nstadiums],TEAMS)
);

%a team can only play in one stadium
constraint forall(estadi in 1..nstadiums)(
  partition_set(matches[1..ndays,estadi],TEAMS)
);


array[1..ndays,1..nstadiums] of var 0..(max(tifosi)*2): totalTifosi;

constraint forall(day in 1..ndays, estadi in 1..nstadiums) (
  totalTifosi[day,estadi] = max((sum(team in matches[day, estadi]) (
      tifosi[team])) - capacitats[estadi],0)
);

constraint missedMatch = sum(totalTifosi);


array[1..nteams,1..(ndays-1)] of var 0..max(distancies): arrayDistancies;

constraint forall(team in 1..nteams, day in 1..(ndays-1)) (
  let {
    var int: stadium_day = sum(stadium in 1..nstadiums where team in
        matches[day, stadium]) (stadium),
    var int: stadium_next_day = sum(stadium in 1..nstadiums where team in
         matches[day+1, stadium]) (stadium)
```

```
92    } in
93    arrayDistancies[team, day] = distancies[stadium_day, stadium_next_day]
94  );
95
96  constraint kilometers = sum(arrayDistancies);
97
98  int: maxEspectadorsRounded = ceil(pow(10, ceil(log(10,
        maxEspectadorsPerduts))));
99
100
101 solve :: seq_search([
102               int_search(arrayDistancies, first_fail, indomain_min),
103               int_search(totalTifosi, first_fail, indomain_min)
104           ]) minimize kilometers*maxEspectadorsRounded + missedMatch;
105
106
107
108 function string: formatSet(var set of int: s) =
109   if(card(fix(s)) == 0) then
110     "     "
111   else
112   join("-", [show_int(2,i) | i in fix(s)])
113   endif;
114
115
116 output
117 ["\t:" ++ concat([" Es" ++ show_int(2,team) ++ "  |" | team in 1..
        nstadiums]) ++ "\n"]
118 ++
119 ["-" | i in 1..((8*nstadiums)+9)]
120 ++ ["\n"] ++
121 ["Dia   " ++ show_int(2,dia) ++ ": " ++ concat([formatSet(matches[dia,
        estadi]) ++ " | " | estadi in 1..nstadiums]) ++ "\n" | dia in 1..ndays
        ]
122 ++
123 ["-" | i in 1..((8*nstadiums)+9)]
124 ++ ["\n"] ++
125 ["KMs TOTALS: " ++ show(kilometers)]
126 ++ ["\n"] ++
127 ["SEGUIDORS FORA: " ++ show(missedMatch)]
```