CR TP2 Programmation par contraintes

Les régates

Julien LETOILE, Romain HUBERT

le 08/03/2021

Table des matières

I. Réponses rédigées

Question 4.1

```
getData(TailleEquipes, NbEquipes, CapaBateaux, NbBateaux,
NbConf):-
    TailleEquipes = [](7, 6, 5, 5, 5, 4, 4, 4, 4, 4, 4, 4, 4, 3,
3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2),
    NbEquipes #= 29,
    CapaBateaux = [](10, 10, 9, 8, 8, 8, 8, 8, 8, 7, 6, 4, 4),
    NbBateaux #= 13,
    NbConf #= 7.
```

Test:

> getData(T, N, C, Nb, NbConf).
>
> T = [](5, 5, 2, 1)
> N = 4
> C = [](7, 6, 5)
> Nb = 3
> NbConf = 3

Question 4.2

```
defineVars(T, NbEquipes, NbConf, NbBateaux):-
    dim(T, [NbEquipes, NbConf]),
    T :: 1..NbBateaux.
```

Test:

> solve(T).

T = [](*[](423{1 .. 3}, _438{1 .. 3}, _453{1 .. 3}), [](468{1 .. 3}, 483{1 .. 3}, _498{1 .. 3}), [](513{1 .. 3}, 528{1 .. 3}, _543{1 .. 3}), [](558{1 .. 3}, _573{1 .. 3}, _588{1 .. 3}))

Question 4.3

```
getVarList(T, L):-
    term_variables(T, L).
```

Question 4.4

```
solve(T):-
    getData(TailleEquipes, NbEquipes, CapaBateaux, NbBateaux,
NbConf),
    defineVars(T, NbEquipes, NbConf, NbBateaux),
    pasMemeBateaux(T, NbEquipes, NbConf),
    pasMemePartenaires(T, NbEquipes, NbConf),
    capaBateaux(T, TailleEquipes, NbEquipes, CapaBateaux,
NbBateaux, NbConf),
    getVarListAlt(T, Liste),
    labeling(Liste).
```

Test:

solve(T).

T = [](*[](1, 1, 1), [](1, 1, 1), [](1, 1, 1), [](1, 1, 1))
Yes (0.00s cpu, solution 1, maybe more) ? ;

T = [](*[](1, 1, 1), [](1, 1, 1), [](1, 1, 1), [](1, 1, 2))
Yes (0.00s cpu, solution 2, maybe more) ? ;

T = [](*[](1, 1, 1), [](1, 1, 1), [](1, 1, 1), [](1, 1, 3))
Yes (0.00s cpu, solution 3, maybe more) ?

Question 4.5

```
pasMemeBateaux(T, NbEquipes, NbConf):-
    (
        /* Pour chaque Equipe */
        for(Equipe, 1, NbEquipes),
        param(T, NbConf)
    do
        (
            for(I, 1, NbConf),
            param(T, Equipe, NbConf)
        do
            (
                for(J, I+1, NbConf),
                param(T),
                param(I),
                param(Equipe)
            do
                X is T[Equipe, I],
                Y is T[Equipe, J],
                X #\= Y
            )
        )
    ).
```

Test:

```
T = []([](1, 2, 3), [](1, 2, 3), [](1, 2, 3), [](1, 2, 3))
Yes (0.00s cpu, solution 1, maybe more) ? ;

T = []([](1, 2, 3), [](1, 2, 3), [](1, 2, 3), [](1, 3, 2))
Yes (0.00s cpu, solution 2, maybe more) ? ;

T = []([](1, 2, 3), [](1, 2, 3), [](1, 2, 3), [](2, 1, 3))
Yes (0.00s cpu, solution 3, maybe more) ? ;
```

Question 4.6

```
pasMemePartenaires(T, NbEquipes, NbConf):-
    (
        for(Equipe1, 1, NbEquipes),
        param(T, NbEquipes, NbConf)
    do
        (
            for(Equipe2, Equipe1 + 1, NbEquipes),
            param(T, Equipe1, NbConf)
        do
            (
                for(Conf1, 1, NbConf),
                param(T, Equipe1, Equipe2, NbConf)
            do
                (
                    for(Conf2, Conf1+1, NbConf),
                    param(T, Equipe1, Equipe2, Conf1)
                do
                    /* Equipes de la première confrontation */
                    E1_C1 is T[Equipe1, Conf1],
                    E2_C1 is T[Equipe2, Conf1],

                    /* Equipes de la deuxième confrontation */
                    E1_C2 is T[Equipe1, Conf2],
                    E2_C2 is T[Equipe2, Conf2],

                    /* Si deux équies sont ensemble dans une
confrontation alors elles ne le sont pas lors de l'autre
confrontation */
                    (E1_C1 #= E2_C1) => (E1_C2 #\= E2_C2),
                    (E1_C2 #= E2_C2) => (E1_C1 #\= E2_C1)
                )
            )
        )
    ).
```

Test:

```
solve(T)

T = [](1, 2, 3), [](1, 3, 2), [](2, 1, 3), [](2, 3, 1))
Yes (0.00s cpu, solution 1, maybe more) ? ;

T = [](1, 2, 3), [](1, 3, 2), [](2, 1, 3), [](3, 1, 2))
Yes (0.00s cpu, solution 2, maybe more) ? ;

T = [](1, 2, 3), [](1, 3, 2), [](2, 1, 3), [](3, 2, 1))
Yes (0.00s cpu, solution 3, maybe more) ? ;

T = [](1, 2, 3), [](1, 3, 2), [](2, 3, 1), [](2, 1, 3))
Yes (0.00s cpu, solution 4, maybe more) ?
```

Question 4.7

```
capaBateaux(T, TailleEquipes, NbEquipes, CapaBateaux, NbBateaux,
NbConf):-
    (
        % Pour chaque confrontation
        for(NumConf, 1, NbConf),
        param(T, TailleEquipes, NbEquipes, CapaBateaux,
NbBateaux)
    do
        (
            for(NumBateau, 1, NbBateaux),
            param(T, TailleEquipes, NbEquipes, CapaBateaux,
NumConf)
        do
            Capa #=< CapaBateaux[NumBateau],
            (
                for(NumEquipe, 1, NbEquipes),
                fromto(0, C, NC, Capa),
                param(T, TailleEquipes, NumBateau, NumConf)
            do
                NB is T[NumEquipe, NumConf],
                NC #= C + TailleEquipes[NumEquipe] * (NB #=
NumBateau)
            )
```

```
            )
        ).
```

Test:

```
T = _(_(1, 2, 3), _(2, 3, 1), _(3, 1, 2), _(3, 2, 1))
Yes (0.00s cpu, solution 1, maybe more) ? ;

T = _(_(1, 2, 3), _(3, 1, 2), _(2, 3, 1), _(1, 3, 2))
Yes (0.00s cpu, solution 2, maybe more) ? ;

T = _(_(1, 3, 2), _(2, 1, 3), _(3, 2, 1), _(3, 1, 2))
Yes (0.00s cpu, solution 3, maybe more) ? ;

T = _(_(1, 3, 2), _(3, 2, 1), _(2, 1, 3), _(1, 2, 3))
Yes (0.00s cpu, solution 4, maybe more) ? ;

T = _(_(2, 1, 3), _(1, 3, 2), _(3, 2, 1), _(3, 1, 2))
Yes (0.00s cpu, solution 5, maybe more) ?

12 solutions en tout
```

Question 4.8

```
getVarListAlt(T, L):-
    dim(T, [NbEquipes, NbConf]),
    (
        for(Conf, 1, NbConf),
        fromto([], AncienneConf, NouvelleConf, L),
        param(T, NbEquipes)
    do

        Moitie #= div(NbEquipes, 2) + 1,
        (
            for(IdGrandeEquipe, 1, Moitie),
            fromto([], AncienneListe, NouvelleListe,
EquipesConf),
            param(T, NbEquipes, Conf)
```

```
            do
                GrandeEquipe is T[IdGrandeEquipe, Conf],
                IdPetiteEquipe #= NbEquipes - IdGrandeEquipe + 1,
                % writeln(test(IdGrandeEquipe, IdPetiteEquipe)),
                ( IdPetiteEquipe #> IdGrandeEquipe ->
                    PetiteEquipe is T[IdPetiteEquipe, Conf],
                    NouvelleListe = [GrandeEquipe, PetiteEquipe |
    AncienneListe]
                ;
                    NouvelleListe = [GrandeEquipe | AncienneListe]
                )

            ),
            append(AncienneConf, EquipesConf, NouvelleConf)
        ).
```

Test:

> T = [](](10, 7, 9, 3, 8, 1, 2), [](11, 9, 8, 7, 2, 10, 1), [](9, 11, 10, 8, 3, 2, 7), [](8, 10, 11, 9, 1, 6, 5), [](7, 8, 6, 10, 1, 2, 3), [](12, 6, 13, 11, 10, 9, 4), [](6, 12, 4, 13, 11, 5, 10), [](5, 6, 7, 1, 9, 13, 12), [](4, 5, 1, 2, 9, 3, 11), [](4, 2, 5, 1, 6, 7, 3), [](3, 1, 2, 6, 7, 8, 4), [](2, 3, 1, 4, 7, 5, 9), [](2, 1, 3, 5, 6, 4, 8), [](1, 3, 2, 5, 4, 7, 6), [](1, 2, 3, 4, 5, 6, 7), [](1, 4, 5, 2, 3, 8, 9), [](1, 5, 4,
> 3, 2, 9, 8), [](2, 4, 6, 1, 5, 3, 10), [](3, 2, 1, 7, 4, 9, 5), [](3, 4, 7, 8, 2, 1, 6), [](5, 1, 4, 2, 10, 12, 13), [](5, 3, 8, 6, 12, 4, 2), [](6, 4, 2, 9, 12, 11, 13), [](6, 5, 3, 10, 13, 12, 1), [](7, 9, 5, 6, 13, 11, 8), [](8, 13, 7, 12, 3, 4, 11), [](9, 13, 12, 2, 4, 11, 1), [](13, 8, 10, 12, 11, 3, 9), [](13, 10, 12, 11, 5, 8, 6))
>
> Yes (152.61s cpu, solution 1, maybe more) ?

II Anexes

Code source

```
:-lib(ic).
:-lib(ic_symbolic).

/* Question 4.4 */
solve(T):-
    getData(TailleEquipes, NbEquipes, CapaBateaux, NbBateaux,
NbConf),
    defineVars(T, NbEquipes, NbConf, NbBateaux),
    pasMemeBateaux(T, NbEquipes, NbConf),
    pasMemePartenaires(T, NbEquipes, NbConf),
    capaBateaux(T, TailleEquipes, NbEquipes, CapaBateaux,
NbBateaux, NbConf),
    getVarListAlt(T, Liste),
    labeling(Liste).

/* Test 4.4 */
/*
solve(T).

T = []([](1, 1, 1), [](1, 1, 1), [](1, 1, 1), [](1, 1, 1))
Yes (0.00s cpu, solution 1, maybe more) ? ;

T = []([](1, 1, 1), [](1, 1, 1), [](1, 1, 1), [](1, 1, 2))
Yes (0.00s cpu, solution 2, maybe more) ? ;

T = []([](1, 1, 1), [](1, 1, 1), [](1, 1, 1), [](1, 1, 3))
Yes (0.00s cpu, solution 3, maybe more) ?
*/

/* Question 4.1 */
% getData(TailleEquipes, NbEquipes, CapaBateaux, NbBateaux,
NbConf):-
%     TailleEquipes = [](5, 5, 2, 1),
%     NbEquipes #= 4,
```

```
%     CapaBateaux = [](7, 6, 5),
%     NbBateaux #= 3,
%     NbConf #= 3.

getData(TailleEquipes, NbEquipes, CapaBateaux, NbBateaux,
NbConf):-
    TailleEquipes = [](7, 6, 5, 5, 5, 4, 4, 4, 4, 4, 4, 4, 4, 3,
3, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2, 2),
    NbEquipes #= 29,
    CapaBateaux = [](10, 10, 9, 8, 8, 8, 8, 8, 8, 7, 6, 4, 4),
    NbBateaux #= 13,
    NbConf #= 7.

/* Test

getData(T, N, C, Nb, NbConf).

T = [](5, 5, 2, 1)
N = 4
C = [](7, 6, 5)
Nb = 3
NbConf = 3

*/

/* Question 4.2 */
defineVars(T, NbEquipes, NbConf, NbBateaux):-
    dim(T, [NbEquipes, NbConf]),
    T :: 1..NbBateaux.

/*
T = []([](_423{1 .. 3}, _438{1 .. 3}, _453{1 .. 3}), [](_468{1 ..
3}, _483{1 .. 3}, _498{1 .. 3}), [](_513{1 .. 3}, _528{1 .. 3},
_543{1 .. 3}), [](_558{1 .. 3}, _573{1 .. 3}, _588{1 .. 3}))
*/

/* Question 4.3 */
getVarList(T, L):-
    term_variables(T, L).
```

```
/* Q 4.5 */
pasMemeBateaux(T, NbEquipes, NbConf):-
    (
        /* Pour chaque Equipe */
        for(Equipe, 1, NbEquipes),
        param(T, NbConf)
    do
        (
            for(I, 1, NbConf),
            param(T, Equipe, NbConf)
        do
            (
                for(J, I+1, NbConf),
                param(T),
                param(I),
                param(Equipe)
            do
                X is T[Equipe, I],
                Y is T[Equipe, J],
                X #\= Y
            )
        )
    ).

/*
T = []([](1, 2, 3), [](1, 2, 3), [](1, 2, 3), [](1, 2, 3))
Yes (0.00s cpu, solution 1, maybe more) ? ;

T = []([](1, 2, 3), [](1, 2, 3), [](1, 2, 3), [](1, 3, 2))
Yes (0.00s cpu, solution 2, maybe more) ? ;

T = []([](1, 2, 3), [](1, 2, 3), [](1, 2, 3), [](2, 1, 3))
Yes (0.00s cpu, solution 3, maybe more) ? ;
*/


/* Prendre 2 equipes & 2 confrontations
/* Question 4.6 */

pasMemePartenaires(T, NbEquipes, NbConf):-
    (
```

```
        for(Equipe1, 1, NbEquipes),
        param(T, NbEquipes, NbConf)
    do
        (
            for(Equipe2, Equipe1 + 1, NbEquipes),
            param(T, Equipe1, NbConf)
        do
            (
                for(Conf1, 1, NbConf),
                param(T, Equipe1, Equipe2, NbConf)
            do
                (
                    for(Conf2, Conf1+1, NbConf),
                    param(T, Equipe1, Equipe2, Conf1)
                do
                    /* Equipes de la première confrontation */
                    E1_C1 is T[Equipe1, Conf1],
                    E2_C1 is T[Equipe2, Conf1],

                    /* Equipes de la deuxième confrontation */
                    E1_C2 is T[Equipe1, Conf2],
                    E2_C2 is T[Equipe2, Conf2],

                    /* Si deux équies sont ensemble dans une
confrontation alors elles ne le sont pas lors de l'autre
confrontation */
                    (E1_C1 #= E2_C1) => (E1_C2 #\= E2_C2),
                    (E1_C2 #= E2_C2) => (E1_C1 #\= E2_C1)
                )
            )
        )
    ).

/*
solve(T)

T = [][](1, 2, 3), [](1, 3, 2), [](2, 1, 3), [](2, 3, 1))
Yes (0.00s cpu, solution 1, maybe more) ? ;

T = [][](1, 2, 3), [](1, 3, 2), [](2, 1, 3), [](3, 1, 2))
Yes (0.00s cpu, solution 2, maybe more) ? ;
```

```
T = [][[](1, 2, 3), [](1, 3, 2), [](2, 1, 3), [](3, 2, 1))
Yes (0.00s cpu, solution 3, maybe more) ? ;

T = [][[](1, 2, 3), [](1, 3, 2), [](2, 3, 1), [](2, 1, 3))
Yes (0.00s cpu, solution 4, maybe more) ?

*/


/* Question 4.7 */


capaBateaux(T, TailleEquipes, NbEquipes, CapaBateaux, NbBateaux,
NbConf):-
    (
        % Pour chaque confrontation
        for(NumConf, 1, NbConf),
        param(T, TailleEquipes, NbEquipes, CapaBateaux,
NbBateaux)
    do
        (
            for(NumBateau, 1, NbBateaux),
            param(T, TailleEquipes, NbEquipes, CapaBateaux,
NumConf)
        do
            Capa #=< CapaBateaux[NumBateau],
            (
                for(NumEquipe, 1, NbEquipes),
                fromto(0, C, NC, Capa),
                param(T, TailleEquipes, NumBateau, NumConf)
            do
                NB is T[NumEquipe, NumConf],
                NC #= C + TailleEquipes[NumEquipe] * (NB #=
NumBateau)
            )
        )
    ).

/*
```

```
T = [](1, 2, 3), [](2, 3, 1), [](3, 1, 2), [](3, 2, 1))
Yes (0.00s cpu, solution 1, maybe more) ? ;

T = [](1, 2, 3), [](3, 1, 2), [](2, 3, 1), [](1, 3, 2))
Yes (0.00s cpu, solution 2, maybe more) ? ;

T = [](1, 3, 2), [](2, 1, 3), [](3, 2, 1), [](3, 1, 2))
Yes (0.00s cpu, solution 3, maybe more) ? ;

T = [](1, 3, 2), [](3, 2, 1), [](2, 1, 3), [](1, 2, 3))
Yes (0.00s cpu, solution 4, maybe more) ? ;

T = [](2, 1, 3), [](1, 3, 2), [](3, 2, 1), [](3, 1, 2))
Yes (0.00s cpu, solution 5, maybe more) ?

12 solutions en tout
*/


/* Question 4.8 */
getVarListAlt(T, L):-
    dim(T, [NbEquipes, NbConf]),
    (
        for(Conf, 1, NbConf),
        fromto([], AncienneConf, NouvelleConf, L),
        param(T, NbEquipes)
    do

        Moitie #= div(NbEquipes, 2) + 1,
        (
            for(IdGrandeEquipe, 1, Moitie),
            fromto([], AncienneListe, NouvelleListe,
EquipesConf),
            param(T, NbEquipes, Conf)
        do
            GrandeEquipe is T[IdGrandeEquipe, Conf],
            IdPetiteEquipe #= NbEquipes - IdGrandeEquipe + 1,
            % writeln(test(IdGrandeEquipe, IdPetiteEquipe)),
            ( IdPetiteEquipe #> IdGrandeEquipe ->
                PetiteEquipe is T[IdPetiteEquipe, Conf],
```

```
                NouvelleListe = [GrandeEquipe, PetiteEquipe |
AncienneListe]
            ;
                NouvelleListe = [GrandeEquipe | AncienneListe]
            )

        ),
        append(AncienneConf, EquipesConf, NouvelleConf)

        % ((NbEquipes mod 2) #\= 0 ->
        %     M #= Moitie + 1,
        %     Eq is T[M, Conf],
        %     append([Eq], L, L)
        % )
    ).

/*
T = [](10, 7, 9, 3, 8, 1, 2), [](11, 9, 8, 7, 2, 10, 1), [](9,
11, 10, 8, 3, 2, 7), [](8, 10, 11, 9, 1, 6, 5), [](7, 8, 6, 10,
1, 2, 3), [](12, 6, 13, 11, 10, 9, 4), [](6, 12, 4, 13, 11, 5,
10), [](5, 6, 7, 1, 9, 13, 12), [](4, 5, 1, 2, 9, 3, 11), [](4,
2, 5, 1, 6, 7, 3), [](3, 1, 2, 6, 7, 8, 4), [](2, 3, 1, 4, 7, 5,
9), [](2, 1, 3, 5, 6, 4, 8), [](1, 3, 2, 5, 4, 7, 6), [](1, 2, 3,
4, 5, 6, 7), [](1, 4, 5, 2, 3, 8, 9), [](1, 5, 4,
3, 2, 9, 8), [](2, 4, 6, 1, 5, 3, 10), [](3, 2, 1, 7, 4, 9, 5),
[](3, 4, 7, 8, 2, 1, 6), [](5, 1, 4, 2, 10, 12, 13), [](5, 3, 8,
6, 12, 4, 2), [](6, 4, 2, 9, 12, 11, 13), [](6, 5, 3, 10, 13, 12,
1), [](7, 9, 5, 6, 13, 11, 8), [](8, 13, 7, 12, 3, 4, 11), [](9,
13, 12, 2, 4, 11, 1), [](13, 8,
10, 12, 11, 3, 9), [](13, 10, 12, 11, 5, 8, 6))

Yes (152.61s cpu, solution 1, maybe more) ?
*/
```