Collège du Sud, Marcel Délèze

- **Liens hypertextes**

Page mère:
http://www.deleze.name/marcel/mathematica/index.html

Logiciel *Tri du couvain* pour *Windows:*
http://www.deleze.name/marcel/logiciels/index.php

Tri du couvain - Programme de simulation

La représentation interne des données

- **Une case du nid**

Chaque case du nid contient les informations suivantes :
e la sorte d'objet qui y est posé; l'état d'une case peut être vide, bleu ou rouge;
f le numéro de la fourmi qui s'y trouve; le numéro 0 indique qu'il n'y a pas de fourmi.

Voici un premier exemple:
e = bleu un objet bleu est posé sur la case;
f = 0 aucune fourmi ne se trouve sur la case.

Voici un deuxième exemple:
e = vide aucun objet n'est posé sur la case;
f = 17 la fourmi n° 17 se trouve sur la case.
La fourmi n° 17 porte un objet ou n'en porte pas; on peut le savoir en se référant à la fourmi n° 17.

- **Le nid**

Le nid est un tableau de cases.
En *Mathematica*, on peut déclarer une telle structure de la manière suivante :

```
(* Nombre de fourmis *)
    nF = 24;
(* Etats d'une case: vide, bleu ou rouge*)
    vide="";
    bleu=0.65;
    rouge=0.95;
(* Dimensions du nid: nombre de colonnes et de lignes *)
    nCol = 40;
    nLn = 30;
(* Constantes *)
    Protect[vide, bleu, rouge];

nidEtat[x, y] = (* état de la case [x, y] *);
nidDossard[x, y]= (* numéro de la fourmi qui se trouve à la case [x, y] *);
```

Une fourmi

Chaque fourmi contient les informations suivantes:

x	l'abscisse de la case où elle se trouve;
y	l'ordonnée de la case où elle se trouve;
e	la sorte d'objet que porte la fourmi; l'état d'une fourmi peut être vide, bleu ou rouge;
mem	la mémoire des états des 15 dernières cases visitées; les états rencontrés y sont écrits cycliquement;
pt	un pointeur indique dans laquelle des 15 cases il faudra écrire l'état de la prochaine case visitée.

Voici un exemple :

x = 25	la fourmi se trouve dans la colonne 25
y = 9	la fourmi se trouve à la ligne 9
e = vide	la fourmi ne porte aucun objet
mem = (vide, vide, vide, bleu, vide, vide, rouge, vide, rouge, rouge, vide, vide, rouge, vide, rouge)	les états des 15 dernières cases que la fourmi a visitées
pt = 4	la prochaine information sera écrite dans la 4-ème position de la mémoire (à la place de "bleu")

La colonie de fourmis

L'ensemble des fourmis est un tableau.
En *Mathematica*, une telle structure peut se déclarer de la manière suivante :

```
(* Nombre de mémoires d'une fourmi *)}
    nMem = 15;

fourmiX[n]= (* numéro de la colonne où se situe la fourmi numéro n *);
fourmiY[n]= (* numéro de la ligne où se situe la fourmi numéro n *);
fourmiCh[n]= (* charge portée par la fourmi numéro n;
            les charges possibles sont {vide, bleu, rouge}
            représentées graphiquement par {X, =, ||} respectivement *);
fourmiMem[n, k]= (* pour la fourmi numéro n, contenu de la case mémoire numéro k *);
fourmiPt[n]= (* pour la fourmi numéro n, pointeur vers la prochaine case mémoire *);
```

Autres variables globales

```
nD      = (* nombre de destinations possibles, c'est-à-dire de cellules adjacentes
               existantes et dépourvues de fourmi *);
destX[n]= (* numéro de la colonne où se situe la destination possible numéro n *);
destY[n]= (* numéro de la ligne où se situe la destination numéro n *);
```

```
(* Données *)
nF = 36       (*Nombre de fourmis*);
nCol = 40      (* Dimensions du nid:nombre de colonnes et de lignes *);
nLn = 30         ;
nB = 240      (* nombre d'objets bleus *);
nR = 160      (* nombre d'objets rouges *);
T1 = 4000       (* nombre d'unités de temps entre deux graphiques;
                nombre d'unités de temps T = T1*T2    *);
```

```
(* Constantes *)
vide = "-"       (* Etats d'une case:vide,bleu ou rouge *);
bleu = 0.65       ;
rouge = 0.95      ;
T2 = 5          (* nombre de graphiques moins un;
                  nombre d'unités de temps T = T1*T2    *);
nMem = 45       (* nombre de mémoires d'une fourmi *);
kP = 0.1;
kM = 0.1;
Protect[vide, bleu, rouge];
```

Set::wrsym : Symbol vide is Protected. ≫

Set::wrsym : Symbol bleu is Protected. ≫

Set::wrsym : Symbol rouge is Protected. ≫

```
<< "Combinatorica`"
```

```
initialiseNid[] := Module[{perm, x, y, z, j},
  If[nB + nR + nF > nCol * nLn, Print["Le nid est trop rempli d'objets et de fourmis."];
   Exit[]];
  Do[nidEtat[x, y] = vide;
   nidDossard[x, y] = 0, {x, 1, nCol}, {y, 1, nLn}];
  perm = RandomPermutation[nCol * nLn];
  (* Relation:    (z-1) == (x-1) nCol + (y-1)    *)
  Do[
       x = Mod[perm[[z]] - 1, nCol] + 1;
       y = Quotient[perm[[z]] - 1, nCol] + 1;
       nidDossard[x, y] = z;
       fourmiX[z] = x;
       fourmiY[z] = y;
       fourmiCh[z] = vide;
       Do[fourmiMem[z, j] = vide, {j, 0, nMem - 1}];
       fourmiPt[z] = 0,
   {z, 1, nF}];
  Do[
       x = Mod[perm[[z]] - 1, nCol] + 1;
       y = Quotient[perm[[z]] - 1, nCol] + 1;
       nidEtat[x, y] = bleu,
   {z, nF + 1, nF + nB}];
  Do[
       x = Mod[perm[[z]] - 1, nCol] + 1;
       y = Quotient[perm[[z]] - 1, nCol] + 1;
       nidEtat[x, y] = rouge,
   {z, nF + nB + 1, nF + nB + nR}]
 ]
```

```
afficheCouvain[j_Integer] := Module[{x, y, i, hu, hv}, hu = 512/nCol; hv = 512/nLn;
  Show[Graphics[{Line[{{0, 0}, {0, 512}, {512, 512}, {512, 0}, {0, 0}}],
     DeleteCases[Table[Which[nidEtat[x, y] == bleu,
        {Hue[bleu], Rectangle[{(x - 1) hu, (y - 1) hv}, {x hu, y hv}]}, nidEtat[x, y] == rouge,
        {Hue[rouge], Rectangle[{(x - 1) hu, (y - 1) hv}, {x hu, y hv}]}],
       {x, 1, nCol}, {y, 1, nLn}], Null, ∞], Table[Which[fourmiCh[i] == vide,
       {Line[{{(fourmiX[i] - 1) hu, fourmiY[i] hv}, {fourmiX[i] hu, (fourmiY[i] - 1) hv}}],
        Line[{{(fourmiX[i] - 1) hu, (fourmiY[i] - 1) hv}, {fourmiX[i] hu, fourmiY[i] hv}}]},
       fourmiCh[i] == bleu, {Hue[bleu, 0.3`, 1], Thickness[0.005`], Line[
         {{(fourmiX[i] - 1) hu, (fourmiY[i] - 2/3) hv}, {fourmiX[i] hu, (fourmiY[i] - 2/3) hv}}],
        Line[{{(fourmiX[i] - 1) hu, (fourmiY[i] - 1/3) hv},
          {fourmiX[i] hu, (fourmiY[i] - 1/3) hv}}]}, fourmiCh[i] == rouge,
       {Hue[rouge, 0.3`, 1], Thickness[0.005`], Line[{{(fourmiX[i] - 2/3) hu, fourmiY[i] hv},
          {(fourmiX[i] - 2/3) hu, (fourmiY[i] - 1) hv}}], Line[{{(fourmiX[i] - 1/3) hu,
          fourmiY[i] hv}, {(fourmiX[i] - 1/3) hu, (fourmiY[i] - 1) hv}}]}], {i, 1, nF}],
     Text["Image " <> ToString[j] <> "/" <> ToString[T2], {0, -10}, {-1, 0}]},
    ImageSize → {512, 524}, AspectRatio → Automatic]]]

exploreCase[i_Integer, x_Integer, y_Integer] :=
 (* place l'état de la case nid[x,y] dans la mémoire de la fourmi numéro i;
 si la case est dépourvue de fourmi,
 mets (x,y) dans la liste des destinations possibles *)
 Module[{pt},
  pt = fourmiPt[i];
  fourmiMem[i, pt] = nidEtat[x, y];
  fourmiPt[i] = Mod[pt + 1, nMem];
  If[ nidDossard[x, y] == 0,
        nD = nD + 1;
        destX[nD] = x;
        destY[nD] = y]
 ]

frequence[i_Integer, sorte_] := Module[{k},
 Count[Table[fourmiMem[i, k], {k, 0, nMem - 1}], sorte]/nMem]
```

```
pasFourmi[i_Integer] := Module[{x0, x, y0, y, f, p, q}, x0 = fourmiX[i]; y0 = fourmiY[i];
   nD = 0; If[x0 < nCol, exploreCase[i, x0 + 1, y0]]; If[y0 < nLn, exploreCase[i, x0, y0 + 1]];
   If[1 < x0, exploreCase[i, x0 - 1, y0]]; If[1 < y0, exploreCase[i, x0, y0 - 1]];
   If[x0 < nCol && y0 < nLn, exploreCase[i, x0 + 1, y0 + 1]];
   If[1 < x0 && y0 < nLn, exploreCase[i, x0 - 1, y0 + 1]]; If[1 < x0 && 1 < y0,
    exploreCase[i, x0 - 1, y0 - 1]]; If[x0 < nCol && 1 < y0, exploreCase[i, x0 + 1, y0 - 1]];
   If[nD > 0, q = RandomInteger[{1, nD}]; x = destX[q]; y = destY[q]; fourmiX[i] = x;
    fourmiY[i] = y; nidDossard[x0, y0] = 0; nidDossard[x, y] = i;
    If[nidEtat[x, y] ≠ vide && fourmiCh[i] == vide, f = frequence[i, nidEtat[x, y]];
     p = (kP/(kP + f))^2; If[RandomReal[] < p, fourmiCh[i] = nidEtat[x, y]; nidEtat[x, y] = vide],
     If[nidEtat[x, y] == vide && fourmiCh[i] ≠ vide, f = frequence[i, fourmiCh[i]]; p =
       (f/(kM + f))^2; If[RandomReal[] < p, nidEtat[x, y] = fourmiCh[i]; fourmiCh[i] = vide]]]];

pasColonie[] := Module[{perm, i},
  perm = RandomPermutation[nF];
  Do[pasFourmi[perm[[i]]], {i, 1, nF}]]
```

```
(* TRI DU COUVAIN *)
initialiseNid[];
afficheCouvain[0]
```

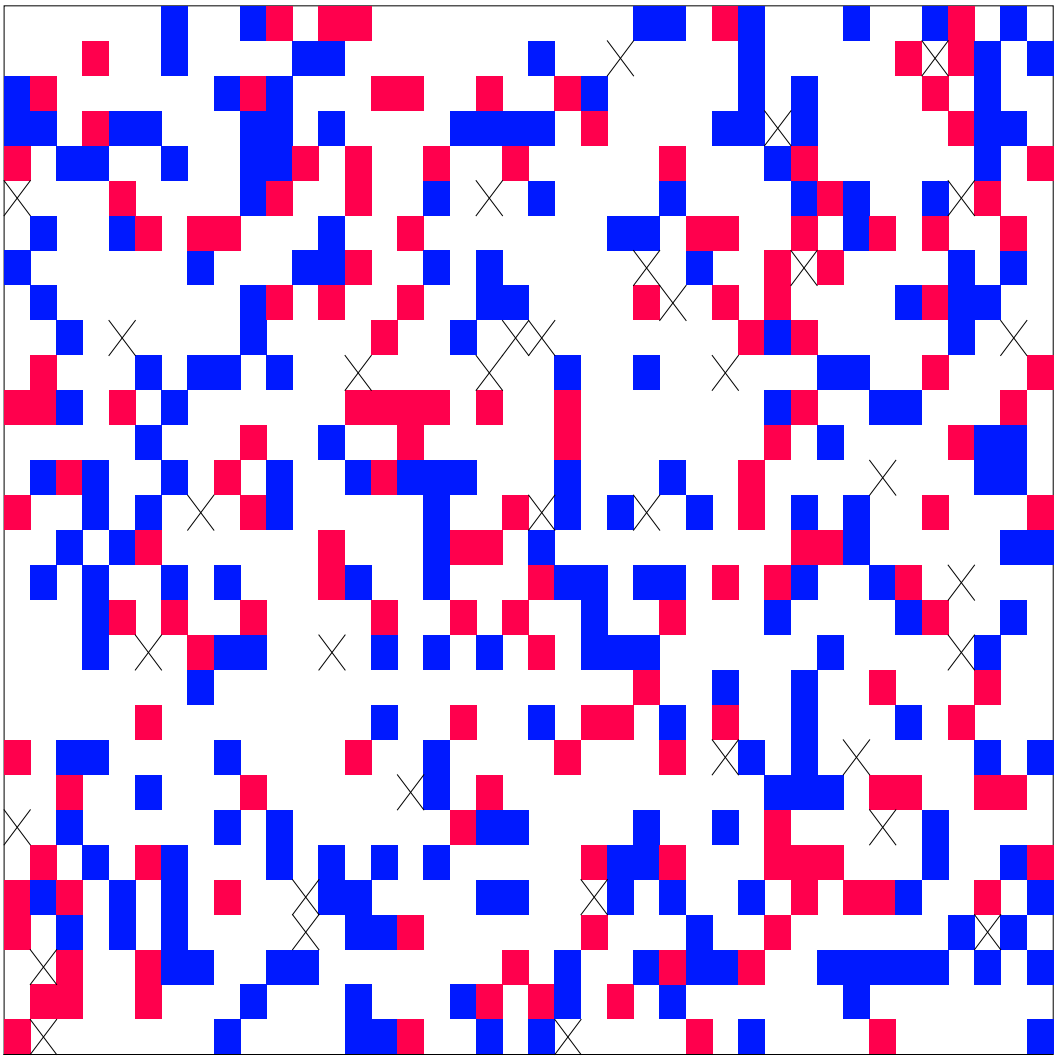

Image 0/5

```
Do[pasColonie[], {i, 1, T1}]; afficheCouvain[1]
```

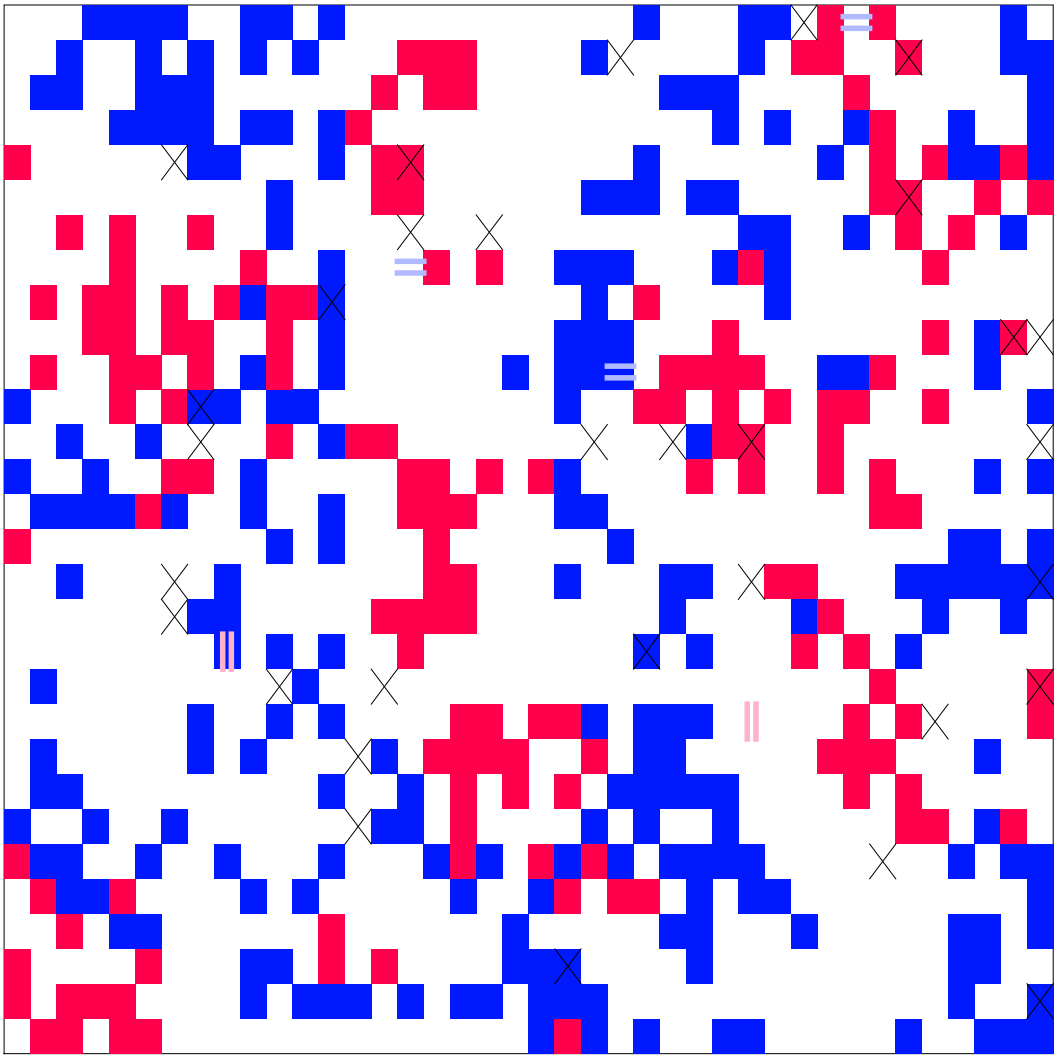

Image 1/5

```
Do[pasColonie[], {i, 1, T1}]; afficheCouvain[2]
```

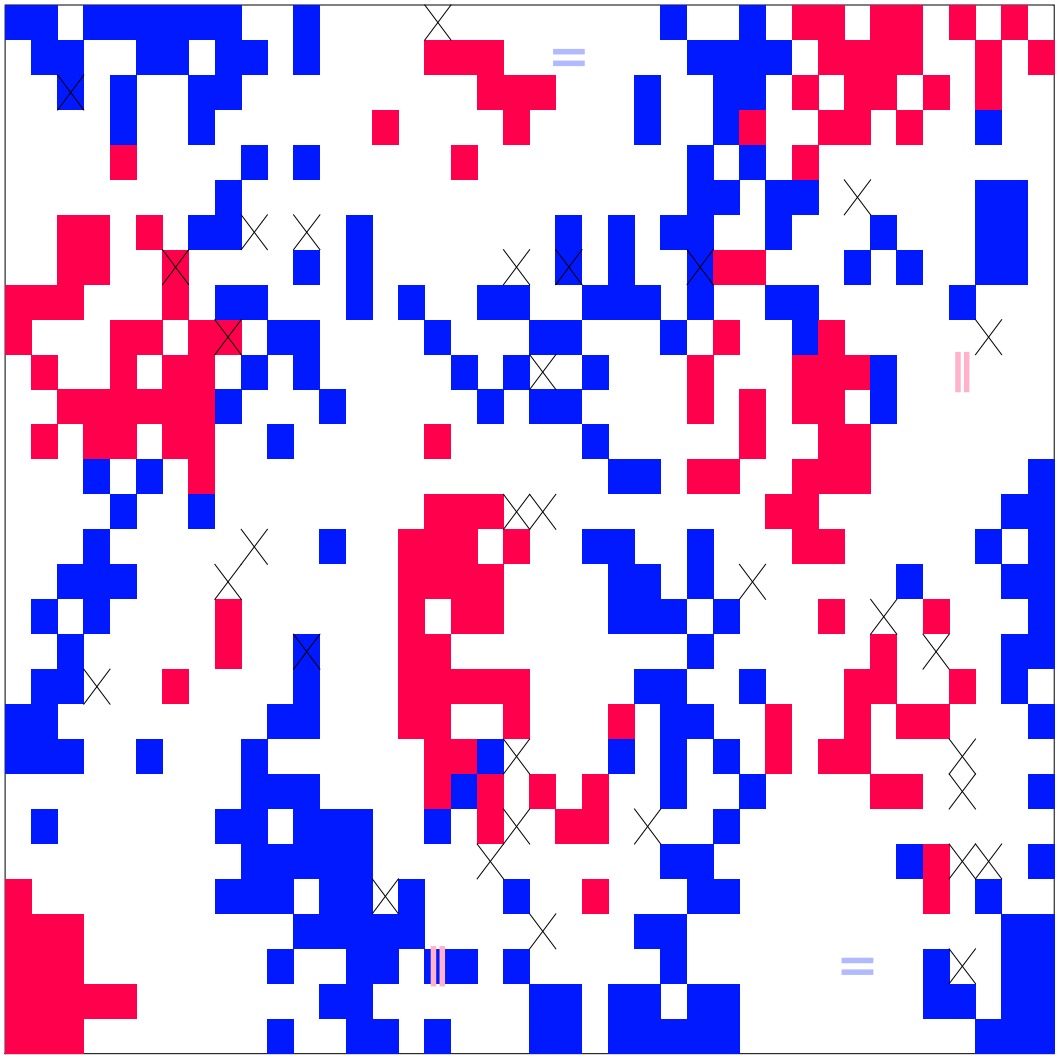

Image 2/5

```
Do[pasColonie[], {i, 1, T1}]; afficheCouvain[3]
```

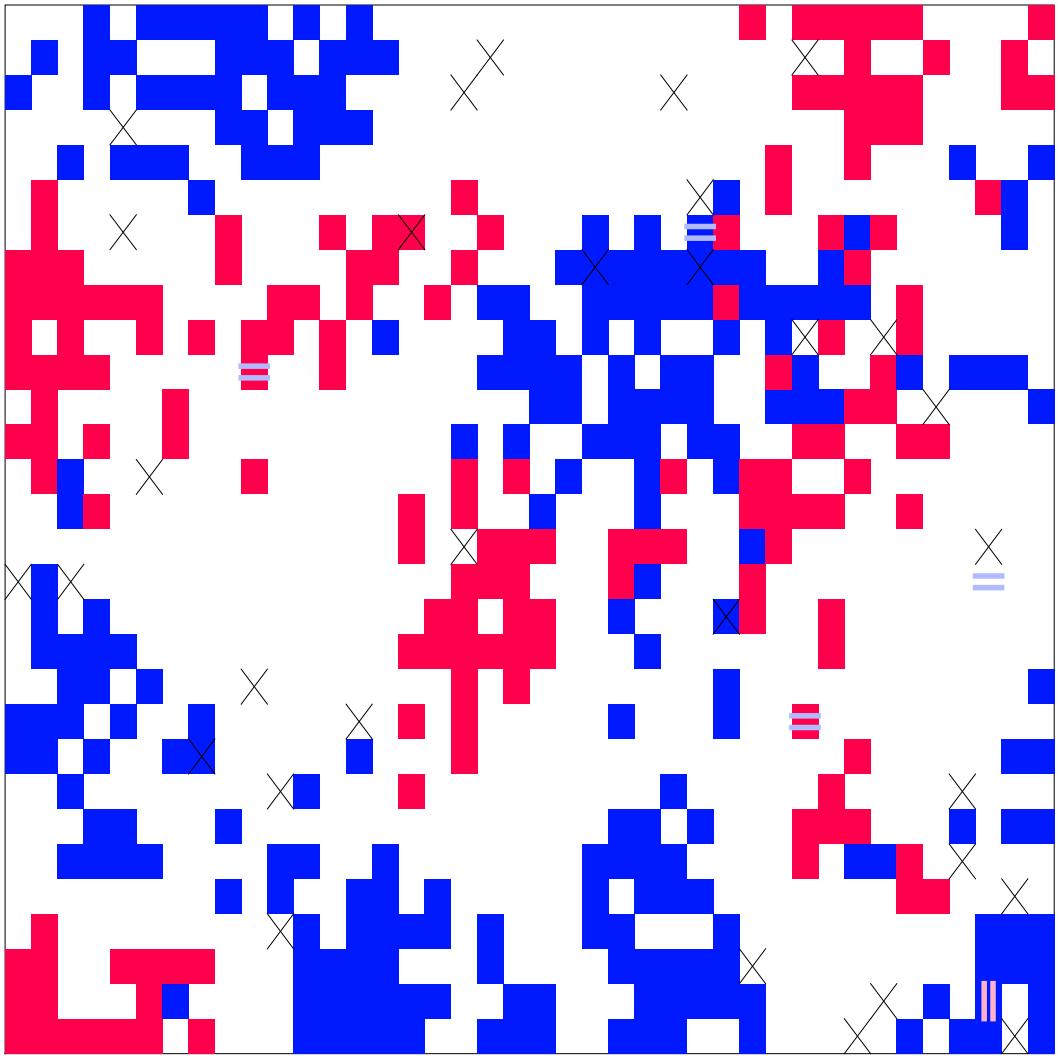

Image 3/5

```
Do[pasColonie[], {i, 1, T1}]; afficheCouvain[4]
```

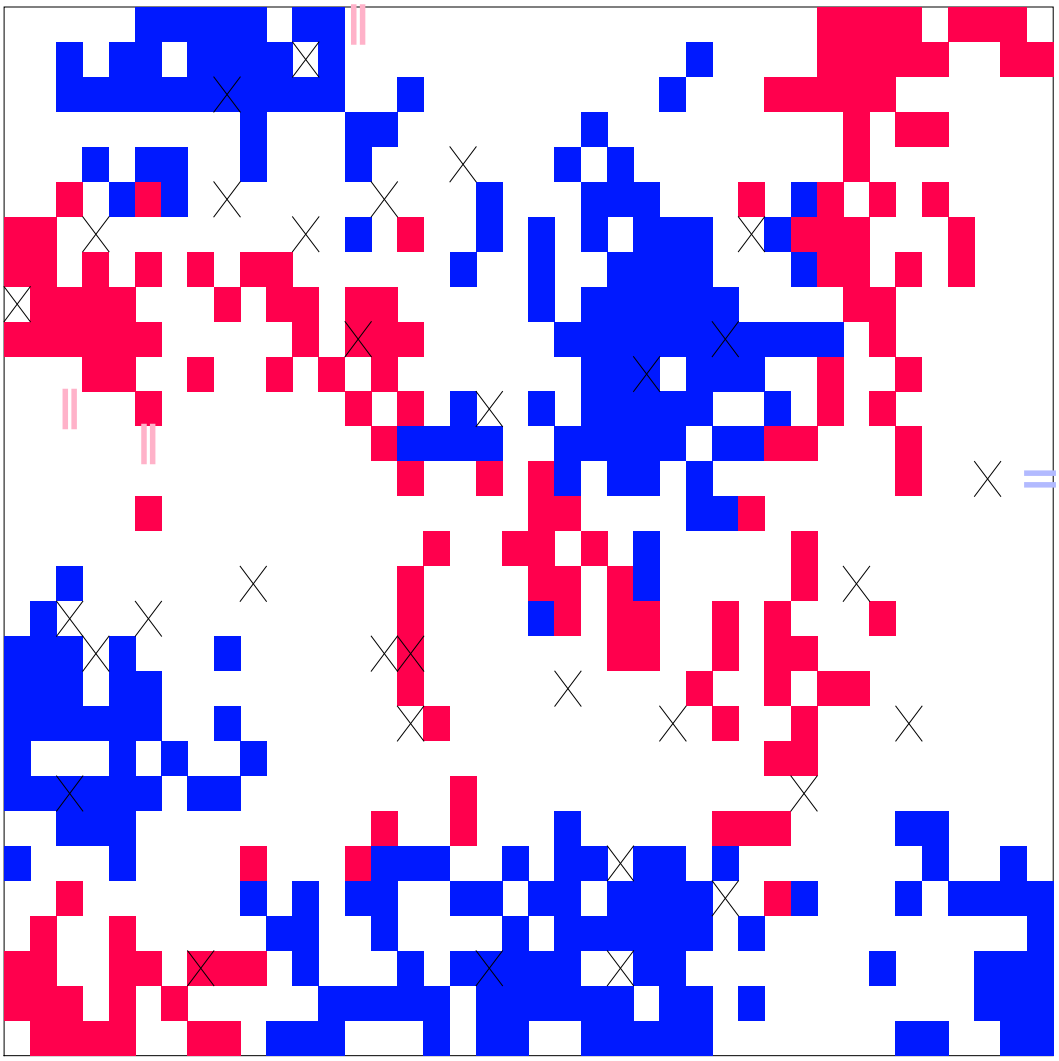

Image 4/5

```
Do[pasColonie[], {i, 1, T1}]; afficheCouvain[5]
```

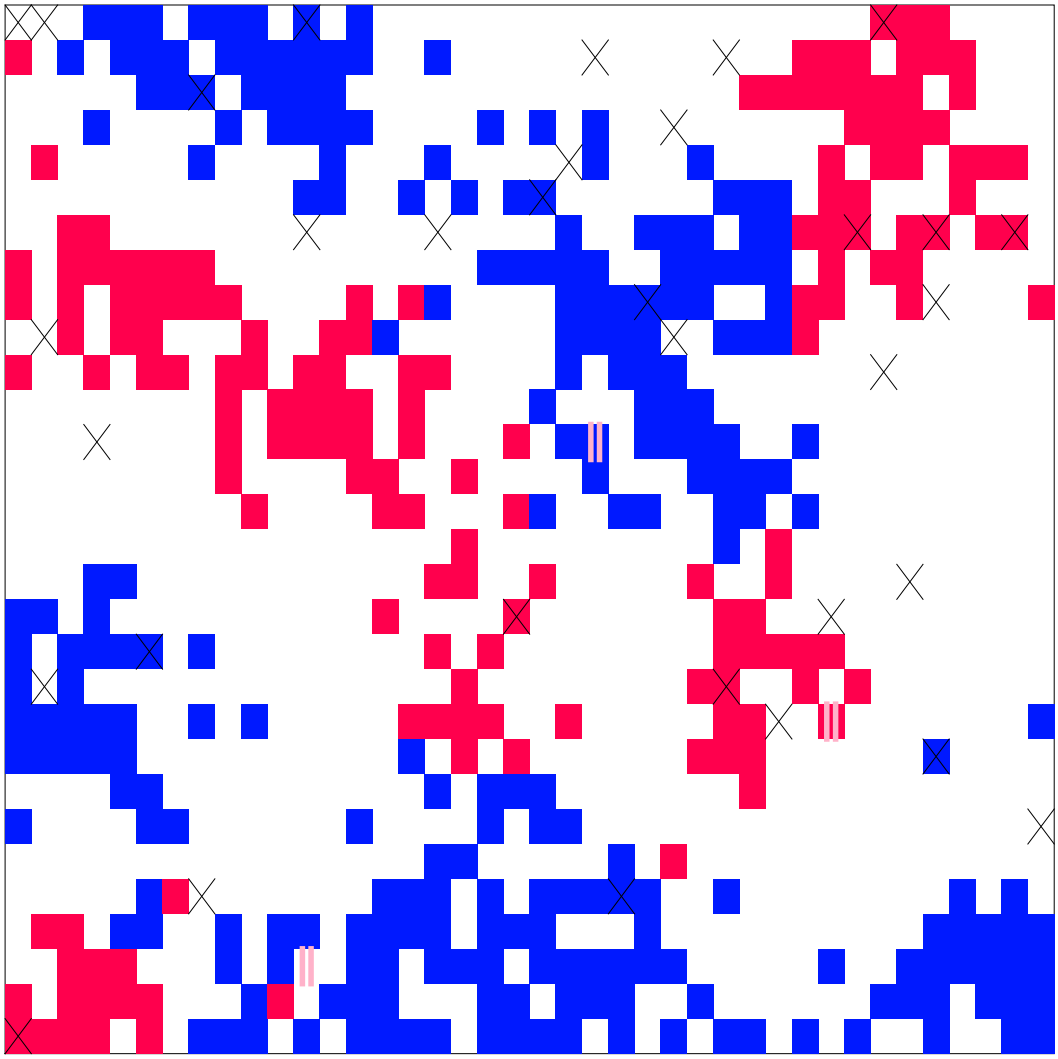

Image 5/5