UNIVERSITÉ DE GENÈVE
Sections d'Informatique et de Physique

FACULTÉ DES SCIENCES
Profs. B. Chopard et M. Droz

Outils, modélisation et simulation en calcul numérique – Corrigé série 1

15 mars 2005

Exercice 1.
La règle de parité.

i) De la définition de la règle de parité, il est clair qu'elle peut être exprimée sous la forme

$$\psi(i,j;t+1) = \psi(i+1,j;t) \oplus \psi(i-1,j;t) \oplus \psi(i,j+1;t) \oplus \psi(i,j-1;t). \qquad (1)$$

ii) Programme Matlab :

```
m=128; %size of the system
X=zeros(m,m); %matrix with zeros everywhere except ...
X(62:66,62:66)=1; %... the initial condition: square
n=[m 1:m-1]; %array: m,1,2,...,m-1
e=[2:m 1]; %array: 2,3,4,...,m,1
s=[2:m 1]; %array: 2,3,4,...,m,1
w=[m 1:m-1]; %array: m,1,2,...,m-1
update=X;
while any(any(update)) %loop as long as structures do not go beyond system size
  A= X(n,:) + X(s,:) + X(:,e) + X(:,w); %the parity rule without modulo 2
  update=X ~= mod(A,2); %to check if there is an overlap of the structures
  X=mod(A,2); %the modulo 2 condition
  pcolor(X)
  axis off
  axis square
  shading flat %removes the lattice
  drawnow %draws the picture
end
```

iii) En appliquant la règle deux fois on obtient $\psi(\cdot,\cdot;t+1)$ en fonction de $\psi(\cdot,\cdot;t-1)$:

$$\begin{aligned}\psi(i,j;t+1) = \quad & \psi(i+2,j;t-1) \oplus \psi(i,j;t-1) \oplus \psi(i+1,j+1;t-1) \oplus \psi(i+1,j-1;t-1) \\ & \oplus\psi(i,j;t-1) \oplus \psi(i-2,j;t-1) \oplus \psi(i-1,j+1;t-1) \oplus \psi(i-1,j-1;t-1) \\ & \oplus\psi(i+1,j+1;t-1) \oplus \psi(i-1,j+1;t-1) \oplus \psi(i,j+2;t-1) \oplus \psi(i,j;t-1) \\ & \oplus\psi(i+1,j-1;t-1) \oplus \psi(i-1,j-1;t-1) \oplus \psi(i,j;t-1) \oplus \psi(i,j-2;t-1).\end{aligned}$$

Utilisant $X \oplus X = 0$ et $X \oplus 0 = X$ on obtient le résultat cherché.

iv) Supposons la relation (1) vraie pour $T \in \mathbb{N}$ itérations, i.e.,

$$\psi(i,j;t+T) = \bigoplus_{(k,l)\in\Omega} \psi(k,l;t), \qquad \Omega = \{(i+T,j),(i-T,j),(i,j+T),(i,j-T)\}. \qquad (2)$$

En appliquant la règle sur le membre de droite de (2) on obtient

$$\begin{aligned}\psi(i,j;t+T) = \quad & \psi(i+2T,j;t-T)\oplus\psi(i,j;t-T)\oplus\psi(i+T,j+T;t-T)\\ & \oplus\psi(i+T,j-T;t-T)\oplus\psi(i,j;t-T)\oplus\psi(i-2T,j;t-T)\\ & \oplus\psi(i-T,j+T;t-T)\oplus\psi(i-T,j-T;t-T)\oplus\psi(i+T,j+T;t-T)\\ & \oplus\psi(i-T,j+T;t-T)\oplus\psi(i,j+2T;t-T)\oplus\psi(i,j;t-T)\\ & \oplus\psi(i+T,j-T;t-T)\oplus\psi(i-T,j-T;t-T)\oplus\psi(i,j;t-T)\\ & \oplus\psi(i,j-2T;t-T).\end{aligned}$$

A nouveau, en utilisant $X \oplus X = 0$ et $X \oplus 0 = X$ on obtient

$$\psi(i,j;t+2T) = \bigoplus_{(k,l)\in\Omega''} \psi(k,l;t), \qquad \Omega'' = \{(i+2T,j),(i-2T,j),(i,j+2T),(i,j-2T)\}. \quad (3)$$

Nous savons que la relation (2) est vraie pour $T = 1$ et $T = 2$. Ce résultat montre donc que si la propriété (2) est vraie, alors (3) est satisfait. L'Eq. (3) est donc vérifiée pour tout nombre d'itérations qui est une puissance de 2. La condition initiale est donc répliquée pour $T = 2^n$ itérations, $n \in \mathbb{N}^*$.

v) L'action de la règle est de translater la configuration initiale dans les quatre directions d'une distance $2T$ et de les superposer. Comme les conditions aux bords sont périodiques, si $T = N/2$ l'Eq. (3) implique que les copies de la condition initiale seront exactement superposées sur elles-même. Par définition de la règle de parité, le système sera alors uniformément dans l'état 0.

Exercice 2.
Le nombre de règles possibles pour un automate cellulaire.
Le nombre possible d'états différents pour le système de r cellules est :

$$\underbrace{k \times k \times \ldots \times k \times k}_{r \text{ cellules } \Rightarrow k^r \text{ états}}.$$

La cellule qui est mise à jour peut prendre k valeurs différentes, et ceci pour chacun des k^r états différents du système de r cellules :

$$\underbrace{k \times k \times \ldots \times k \times k}_{k^r \text{ fois}}.$$

Il y a donc k^{k^r} règles possibles.

Exercice 3.
La classification de Wolfram.

i) La décomposition binaire donne :
- Règle 40 : $\{\alpha_0,\alpha_1,\ldots,\alpha_7\} = \{0,0,0,1,0,1,0,0\}$,
- Règle 56 : $\{\alpha_0,\alpha_1,\ldots,\alpha_7\} = \{0,0,0,1,1,1,0,0\}$,
- Règle 18 : $\{\alpha_0,\alpha_1,\ldots,\alpha_7\} = \{0,1,0,0,1,0,0,0\}$,

– Règle 110 : $\{\alpha_0, \alpha_1, \ldots, \alpha_7\} = \{0, 1, 1, 1, 0, 1, 1, 0\}$.

Programme Matlab :

```
clear;
RULE = 18; %rule \in [1,256]
N = 200; %number of cells
ITMAX = 200; %number of time steps
%decimal -> binary
rb = zeros(1,8); %array for binary rule
d1 = RULE;
for (n=[0:7])
    nn=7-n;
    d2 = 2^nn;
    if (d1-d2 >= 0)
        d1 = d1-d2;
        rb(nn+1) = 1;
    end
end
%black-white-map
black = [0 0 0];
white = [1 1 1];
bcmap(1,:) = black;
bcmap(2,:) = white;
colormap(bcmap);
%initial conditions
ca = zeros(ITMAX,N);
ca(1,:)=round(rand(1,N));
%time loop
for i=2:ITMAX
    j=i-1;
    tmp=zeros(1,N);
    for (n=[2:N-1])
        x=4*ca(j,n-1)+2*ca(j,n)+ca(j,n+1);
        tmp(1,n)=rb(1,x+1);
    end
    %periodic boundary conditions:
    x=4*ca(j,N)+2*ca(j,1)+ca(j,2);
    tmp(1,1)=rb(1,x+1);
    x=4*ca(j,N-1)+2*ca(j,N)+ca(j,1);
    tmp(1,N)=rb(1,x+1);
    %update line i
    ca(i,1:N)=tmp(1,1:N);
    %plot
    imagesc(ca);
    axis off equal;
    shading flat;
    drawnow;
end
```

ii) Les règles peuvent être classées dans 4 classes :

- Règle 40 : élément de la classe 1 : l'automate cellulaire évolue après un nombre fini de pas de temps, pour presque toute condition initiale, vers un état homogène unique.

FIG. 1 – Règle 40.

- Règle 56 : élément de la classe 2 : des structures stables ou périodiques de petite période sont formées pour presque toutes les conditions initiales.

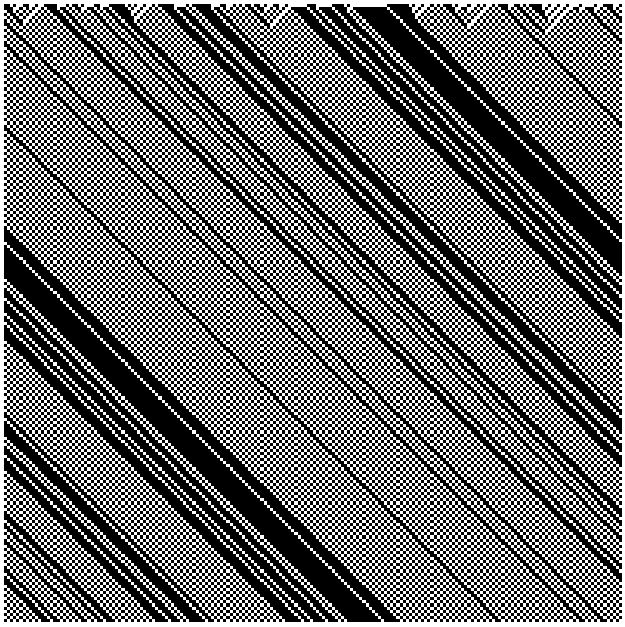


FIG. 2 – Règle 56.

- Règle 18 : élément de la classe 3 : l'automate cellulaire évolue pour presque toute condition

initiale vers une structure apériodique, ou chaotique. Une petite modification des conditions initiales sera amplifiée par la dynamique.

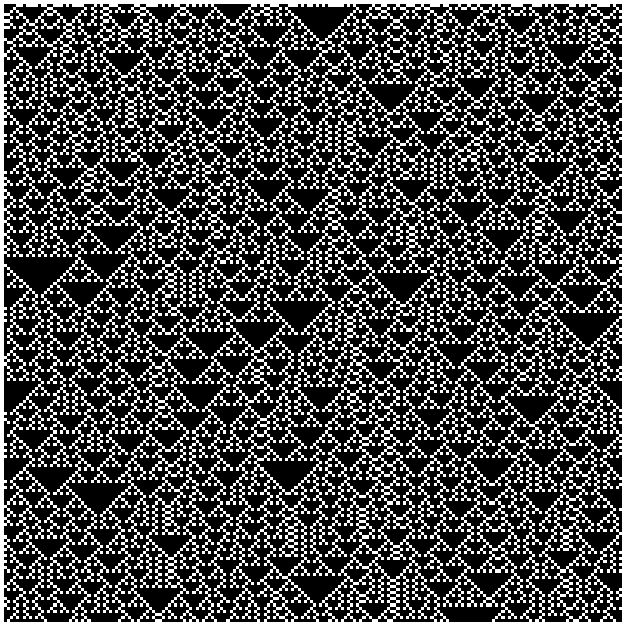


FIG. 3 – Règle 18.

– Règle 110 : élément de la classe 4 : des structures complexes *persistantes* sont formées par la dynamique pour la majeure partie des conditions initiales.

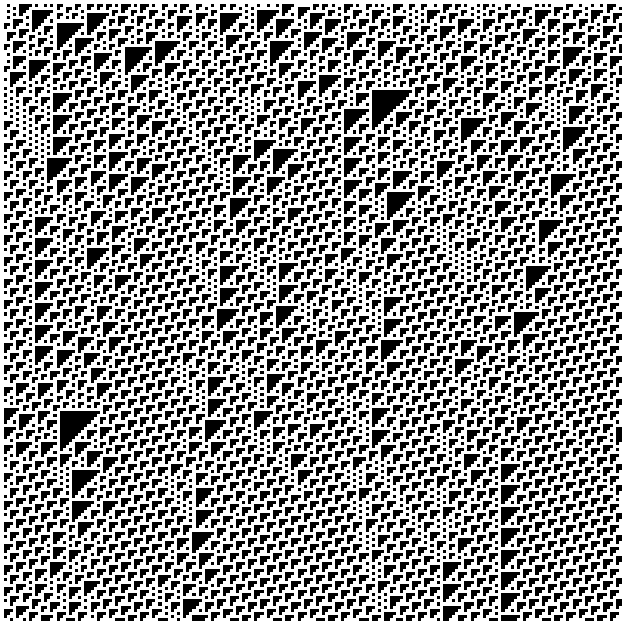


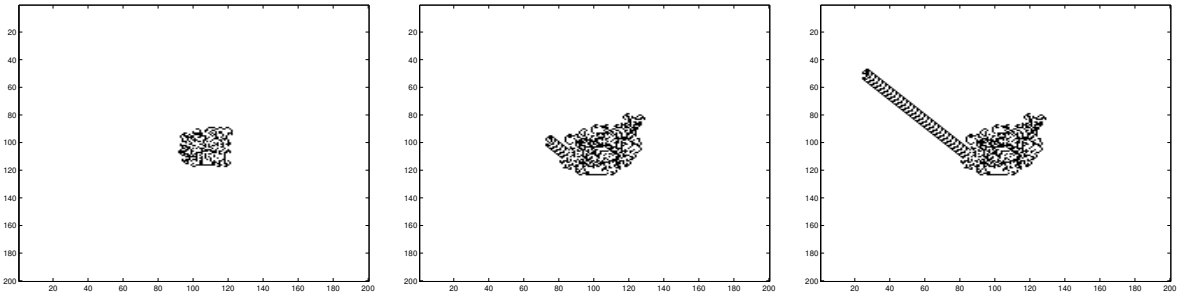
FIG. 4 – Règle 110.

Exercice 4.
La fourmi de Langton.

i) Programme Matlab :

```
clear;
N=200; %system size
tmax=13000; %max time iteration
tskip=100; %display every tskip iterations => speed up
%black-white-map
black=[0 0 0];
white=[1 1 1];
bcmap(1,:)=black;
bcmap(2,:)=white;
colormap(bcmap);
%initial conditions
ca = ones(N,N);
pos=[N/2;N/2];
direction=[1;0];
%rotation matrix = [cos(x) -sin(x); sin(x) cos(x)]
rotateleft=[0 -1;1 0];
rotateright=[0 1;-1 0];
%time loop
for t=1:tmax
    pos=pos+direction;
    if (ca(pos(1,1),pos(2,1))==1)
        direction=rotateleft*direction;
        ca(pos(1,1),pos(2,1))=0;
    else
        direction=rotateright*direction;
        ca(pos(1,1),pos(2,1))=1;
    end
    if (mod(t,tskip)==0)
       imagesc(ca);
       drawnow;
    end
end
```

ii) On constate qu'au-delà d'un certain temps la fourmi "s'échappe" et emprunte une trajectoire orientée à 45 degrés (cf. figure).



$t = 5000$ $t = 10500$ $t = 13000$

Cette trajectoire est formée d'une structure constituée de 104 itérations, qui est répétée indéfiniment. La règle élémentaire de Langton génère donc un comportement (macroscopique) très complexe qui ne peut pas être prédit sur la base des règles microscopiques uniquement.

iii) Supposons que le domaine visité par la fourmi soit borné. Le nombre de cellules est donc fini, et comme le nombre d'itérations est infini, certaines cellules seront visitées un nombre infini de fois. A cause de la dynamique, on peut distinguer deux types de cellules. Une cellule H est une cellule qui est visitée par la fourmi entrant horizontalement, tandis qu'une cellule V est une cellule qui est visitée par la fourmi entrant verticalement. Etant donné que la fourmi tourne ensuite de 90 degrés, une cellule H sera entourée de 4 cellules V, et réciproquement. Les cellules H et V sous-tendent donc une structure alternée statique sur l'automate cellulaire (cf. Fig. 5).

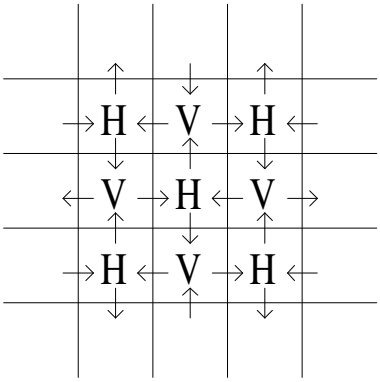
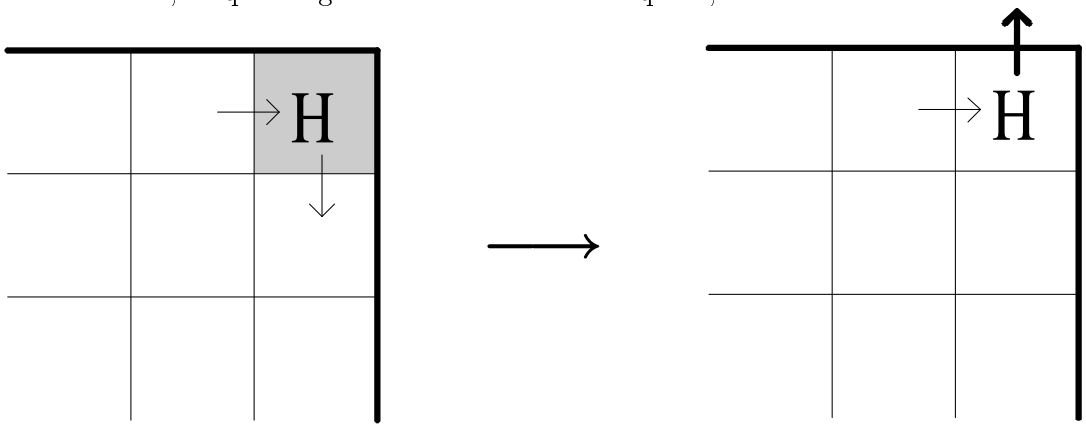


FIG. 5 – Structure des mouvements sur l'automate cellulaire.

Considérons à présent la cellule dans le coin supérieur droit de l'automate cellulaire. Cette cellule est définie si comme on l'a supposé la trajectoire est bornée. Supposons que cette cellule soit de type H. Bien entendu cette cellule reste du même type pour tout temps. Cette cellule doit donc être visitée en provenance de la gauche, et est quittée verticalement vers le bas. Un tel mouvement n'est possible que si la cellule était noire. Après que la fourmi quitte cette cellule, elle la peint en blanc. Par conséquent, lors de sa prochaine visite, le seul mouvement possible sera de quitter la cellule vers le haut, ce qui élargit le domaine. Par conséquent, le domaine est non borné.



mouvement possible
lors de la première visite

mouvement possible
lors de la seconde visite