SUJET D'INTELLIGENCE ARTIFICIELLE

3ème Année CENTRALE 1990

CORRIGE

JEU DE LA VIE

C JOUSSELIN

Life's too important a matter to be taken seriously.[1]

Introduction

Aujourd'hui les réseaux d'automates et de neurones ont un regain d'intérêt et sont de plus en plus étudiés dans les laboratoires de recherche. Le but de cet exercice est de vous faire comprendre comment fonctionne un réseau d'automates cellulaires reposant sur des règles simples et de voir comment transformer ces règles par une approche neuronale. Le meilleur moyen d'apprendre étant le jeu, cet exercice consiste + réaliser un jeu.

Sujet

La première partie du sujet consiste + réaliser en PROLOG le "Jeu de la vie" dont la surface de jeu est un tore de 1520 cases.
La seconde partie du sujet consiste + re-spécifier les règles du jeu de la vie, sans rien changer au jeu lui-même, en considérant que chaque cellule est un neurone. Ce qui revient + répondre aux questions:

Quelle est topographie du réseau?
Quelles sont les valeurs des coefficients synaptiques?
Quelle est la valeur du seuil?
Quelle est la forme de la fonction de filtrage?

Spécifications du jeu de la vie

Le jeu de la vie n'est pas + proprement parlé un jeu + un joueur,le "joueur" initialise le jeu et ce dernier se continue tout seul. L'intérêt de ce jeu consiste + le voir évoluer au cours du temps.
La version simplifiée du jeu de la vie se joue sur un tore et utilise des cellules bicolores (,#).
Ces cellules peuvent **naître**, **mourir** par étouffement ou par exposition, ou **survivre.**
Les cellules peuvent être dans un des deux états suivants :

VIVANTE
MORTE

1 OSCAR WILDE

Une cellule C_2 est dite voisine d'une cellule C_1 si elle est + une distance inférieure + 2 de C_1 selon la distance euclidienne:

Remarque: une cellule a au maximum 8 voisines.

Une cellule **VIVANTE** + un instant *t* peut + l'instant *t+1* :

mourir par étouffement si et seulement si cette cellule a 4 ou plus voisines et donc devenir **MORTE** à l'instant *t+1*.

mourir par exposition si et seulement si cette cellule a 1 ou 0 voisine et donc devenir **MORTE** à l'instant *t+1*.

survire si et seulement si cette cellule a seulement 2 ou 3 voisines et donc rester **VIVANTE** à l'instant *t+1*.

Une cellule **MORTE** à un instant *t* peut à l'instant *t+1* :

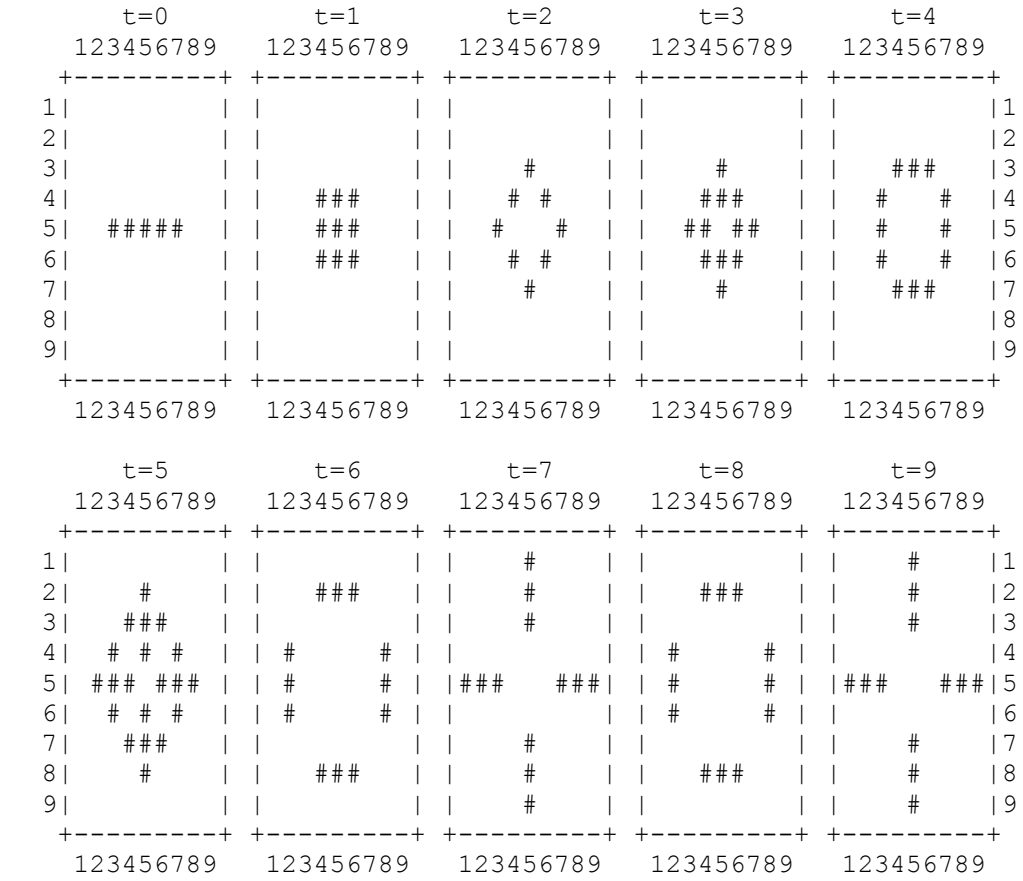
naître si et seulement si exactement 3 de ces voisines sont **VIVANTE**s et donc devenir **VIVANTE** à l'instant *t+1*.

Ces 4 règles simples peuvent se résumer par cette proposition :

Il faut juste 3 voisines pour naître et 2 ou 3 pour survivre!

Exemples :

Initialisation avec une ligne de 5 cellules.



L'intérêt du jeu se termine au temps t=8, car cette configuration se trouve être un bistable (états : t=6, t=7). Le plus simple des bistables est une ligne de 3 cellules (###), cette configuration, appelée un **blinker**, est donc une image du temps et peut servir d'horloge.

Il existe des configurations stables, par exemples :

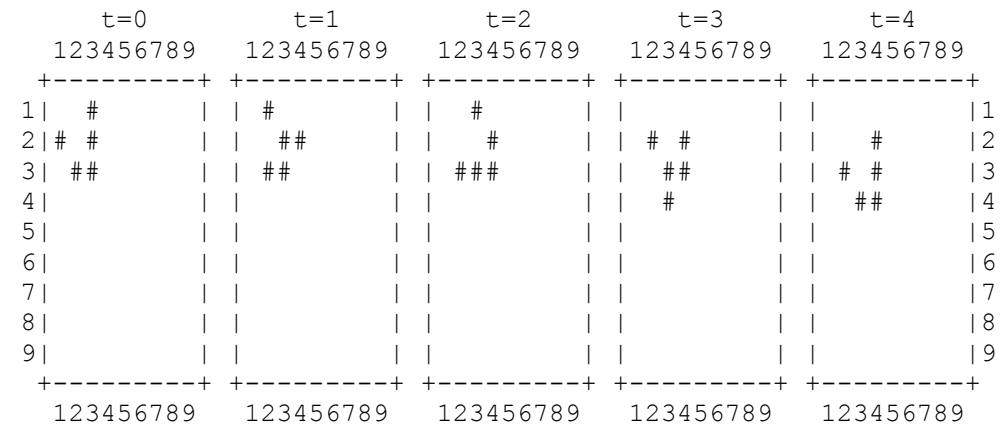
```
##    Block  #    Tub
##          # #
             #

 ##   Pond   #    Beehive
#  #        # #
#  #        # #
 ##          #

 #    Boat   ##   Loaf
# #         #  #
 ##          # #
              #
```

Les configurations stables peuvent servir à mémoriser des informations.

Il existe aussi des configurations qui peuvent se déplacer, ces dernières sont appelées **gliders**. Le plus simple des **gliders** est la configuration :

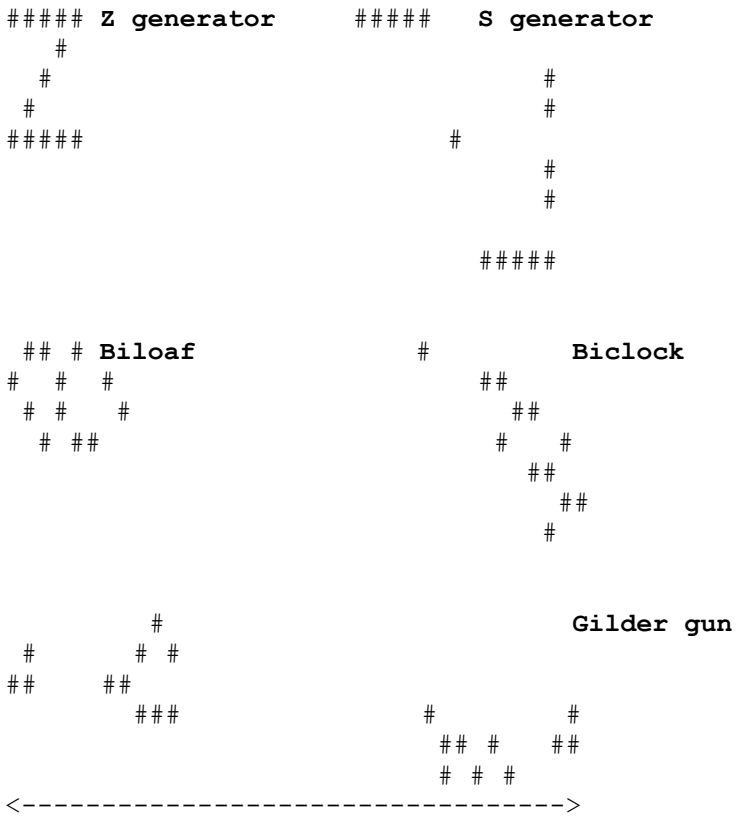
```
  #   Glider
# #
 ##
```

Ce **Glider** se déplace en diagonal d'une case en 4 unité de temps:



Les **gliders** peuvent servir de support à la transmission de l'information.

Comment générer des **gliders**? Voici des exemples de générateur de **gliders** :



LEGENDE Générateurs de **Gliders**

Conception du Jeu de la vie

La surface de jeu est un tore de 1520 cases qui développé sur l'écran se présente comme un rectangle de 20x76 cases; prenons un repère cartésien d'origine O en haut à gauche de l'écran d'axe OL verticale descendant et OC horizontal vers la droite de l'écran.

Une cellule C à un instant *t* est caractérisée par deux informations:

sa position [L,C] dans le repère (O,OL,OC)
sa valeur **VIVANTE** ou **MORTE**

La valeur à l'instant *t+1* d'une cellule ne dépend que de sa valeur à l'instant *t* et de la valeur à l'instant *t* de ses voisines.

Pour distinguer les différentes configurations (configuration à l'instant *t*, configuration à l'instant *t+1*, configuration à l'instant 0) dans une seule base de données une caractéristique est ajoutée à une cellule: la valeur du temps modulo 2. La configuration initiale qui doit être conservée aura pour caractéristique la valeur 2.

La base de données est constituée de 2 types de structures:

```
Etat(Temps,Ch,Tmax)                          en 1 seul exemplaire
            Temps  représente la valeur t,
            Ch     représente le caractère vivant (ici '#'),
            Tmax   représente le temps d'arrêt du jeu en mode
                   automatique (ici 100).

Cell([L,C],Val,TempsMod)                     en plusieurs exemplaires
            [L,C]  représente la position de la cellule dans le
                   repère (O,OL,OC),
            Val    représente la valeur de la cellule:
                   1 VIVANTE ou 0 MORTE,
            TempsMod    représente le temps t modulo 2.
```

L'initialisation du jeu se fait:

en écrivant dans les configurations TempsMod = 2 et TempsMod = 0 les cellules de valeur **VIVANTE**

en initialisant l'état à (0,'#',100) par exemple.

Le calcul de la configuration *t+1* (TempsMod + 1 Modulo 2) se fait:

en supprimant de la base de données toutes les cellules de l'ancienne configuration *t+1* (TempsMod à 1 Modulo 2),

en supprimant de la base de données toutes les cellules de valeur **MORTE** de la configuration *t* (TempsMod),

en ajoutant à la configuration *t* (TempsMod) l'ensemble des voisines **MORTEs** des cellules **VIVANTE** de la configuration, si ces dernières n'appartiennent pas encore à cette configuration,

en calculant la valeur à l'instant *t+1* des cellules de la configuration à traitée en appliquant les règles du jeu.

La reprise du jeu après une sauvegarde se fait:

en rechargeant le fichier **JeuVie.DB** qui contient l'image de la base de données au temps de la sauvegarde.
Comme la base de données contient la configuration au temps_0 (TempsMod = 2), il est possible de redémarrer le jeu.

Réalisation du Jeu de la vie

Voir Annexe A.

Contraintes de réalisation

Les règles du jeu de la vie étant locales, la recherche des nouvelles valeurs des cellules doit être locale; le programme ne doit donc pas être réalisé avec des boucles imbriquées décrivant le tore.

Remarques

L'implémentation du langage PROLOG utilisée est TURBO-PROLOG 1.1 de Borland international.

Initialisation du programme

L'initialisation du programme consiste à présenter la fenêtre de jeu de 20x76 cases représentant le tore, à imposer un état par défaut (Etat(0,'#',100)) et à lancer le menu générale du programme.

Clause concernée: GOAL

Menu général

Le menu général sert à sélectionner les différentes commandes de ce programme:

h : Affichage des possibilités de commande du programme.

i : Initialisation de la configuration de départ,
Clause concernée: Positionner
l : Chargement de la base de données **JeuVie.DB** sauvegardée
Clause concernée: Afficher
s : Sauvegarde de la base de données à cet instant dans **JeuVie.DB**
r : Ré-initialisation de la base de données avec la configuration de départ.
Clause concernée: Afficher
e : Efface la base de données et impose l'état de départ.
Clause concernée: Afficher
t : Calcule et affiche la configuration au temps suivant.
Clause concernée: Tracer
g : Calcule et affiche les configurations suivantes jusqu'au temps max.
Clause concernée: Aller

Clauses concernées: Menu et Menu1

Initialisation de la configuration de départ

L'initialisation de la configuration de départ se fait en sélectionnant les cellules à l'aide des flèches **up**, **down**, **left**, **right** du clavier, et en changeant leur valeur à l'aide de la touche **cr**. L'initialisation se termine par appuis sur la touche **esc**.

Clauses concernées: Positionner, Position et Touche

Calcul de la configuration suivante

Le calcul de la configuration suivante se fait en 5 étapes:

Suppression de l'ancienne configuration *t+1* (Tc1=TempsMod à 1 Modulo 2) par

```
retract(cell(_,_,Tc1)),fail.
```

Suppression de toutes les cellules de valeur **MORTE** de la configuration *t* (Tc=TempsMod Modulo 2) par

```
retract(cell(_0,Tc)),fail.
```

Ajout à la configuration *t* (TempsMod Modulo 2) de l'ensemble des voisines **MORTE** des cellules **VIVANTE** de la configuration, si ces dernières n'appartiennent pas encore dans cette configuration, par

```
cell(Pt,1,Tc),voisine(Pt,Pt1),ajouter(Pt1,Tc),fail.
```

Calcul de la valeur à l'instant *t+1* des cellules de la configuration à traiter en appliquant les règles du jeu, par

```
cell(Pt,V,Tc),findall(X,voisine_1(Pt,Tc,X),L),
somme(L,Nb),règles(Pt,V,Nb,Tc1),fail.
```

Affichage et mise à jour de l'état, par

```
afficher(T1,Tc1,Ch),assertz(état(T1,Ch,Tmax)).
```

Clauses concernées: Tracer, voisine, ajouter, voisine_1, somme et règles

Calcul des configurations suivantes

Le calcul des configurations suivantes consiste à faire une boucle sur le calcul de la configuration suivante tant que le joueur n'appuie pas sur une touche et que le temps n'est pas supérieur au temps maximum.

Clauses concernées: Aller et tracer

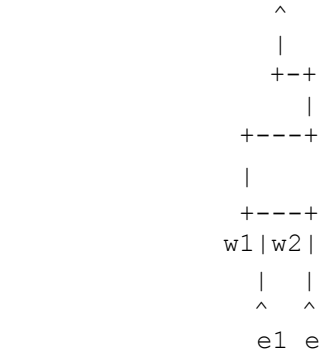
Impression

L'impression sur papier d'une configuration à un instant *t* se fait par appuis sur la touche "hard_copy d'écran".

Approche neuronale du Jeu de la Vie

Rappel

Description d'un neurone formel:



Modélisation d'un neurone

Re-spécification du Jeu de la Vie

Un réseau de neurones formels est une forme particulière d'un réseau d'automates. Nous pouvons donc dire que la version simplifiée du jeu de la vie se joue sur un tore et utilise des neurones bicolores (,#).
Les neurones peuvent **naître**, **mourir** par étouffement ou par exposition, ou **survivre.**
Les neurones peuvent être dans un des deux états suivants :

VIVANT
MORT

Règle: Il faut juste 3 voisins pour naître et 2 ou 3 pour survivre!

Re-conception du Jeu de la Vie

Nous prendrons les mêmes conventions que celles prises pour l'approche réseau d'automates.

La surface de jeu est un tore de 1520 cases qui développé sur l'écran se présente comme un rectangle de 20x76 cases; prenons un repère cartésien d'origine O en haut à gauche de l'écran d'axe OL verticale descendant et OC horizontal vers la droite de l'écran.

Un neurone N_+ un instant t est caractérisée par deux informations:

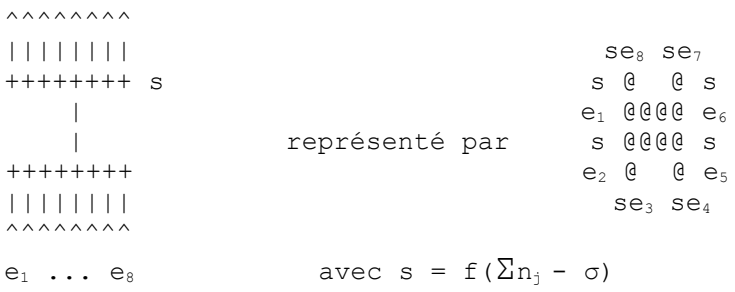
sa position [L,C] dans le repère (O,OL,OC)
sa valeur **VIVANT** ou **MORT**

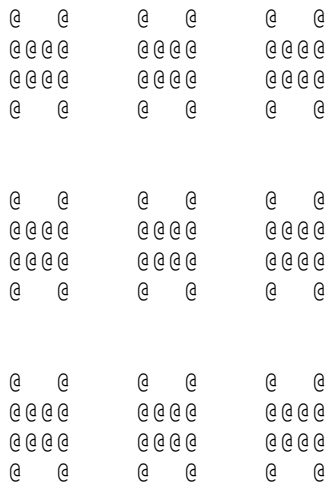
La valeur à l'instant *t+1* d'un neurone ne dépend que de sa valeur à l'instant *t* et de la valeur à l'instant *t* de ses voisins.

Il faut donc que chaque neurone soit relié à ses 8 voisins et à lui-même.

Topographie du réseau

La topographie du réseau est donc très simple:





Topographie du réseau de neurones

Coefficients synaptiques

Les 8 voisins d'un neurone jouent un rôle symétrique, seul le nombre voisins **VIVANT** est important. Nous pouvons donc donner la même valeur aux coefficients synaptiques inter-neurone. Ces coefficients sont pris arbitrairement égaux à :

$$\mathbf{w_{ij} = 1 \text{ pour } i \neq j}$$

La règle générale: Il faut juste 3 voisins pour naître et 2 ou 3 pour survivre! montre que l'influence des voisins d'un neurone est plus importante que sa propre influence (son histoire).
Il faut donc marquer cette différence par le "bon choix" de la valeur du coefficient intra-neurone; deux choix sont possibles:

- une valeur élevée de ce coefficient dépendant du nombre de voisins pris en compte dans le jeu (ici 8),
  $w_{ii} > 7$
- une valeur faible de ce coefficient par rapport au coefficient inter-neurone, indépendant du nombre de voisins pris en compte dans le jeu
  $0 < w_{ii} < 1$

Nous choisissons $\mathbf{w_{ii} = 1/2}$

Fonction de filtrage et seuil

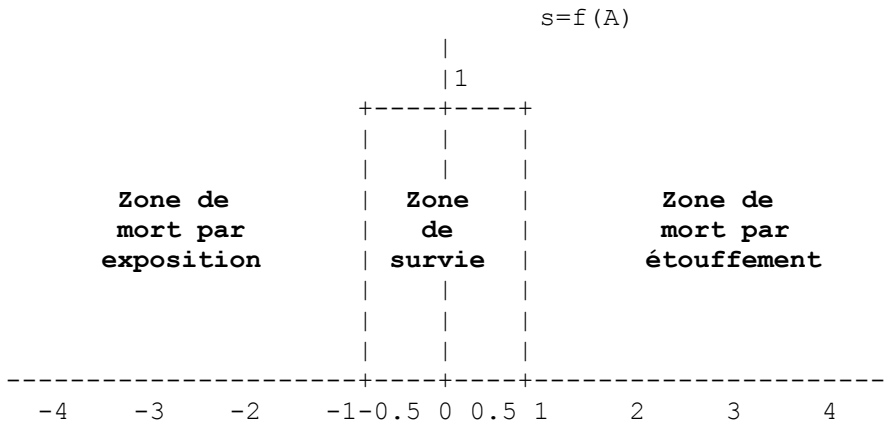
Pour obtenir une valeur de sortie binaire d'un neurone avec la#convention suivante:

1 **VIVANT** ou 0 **MORT,**

il faut bien choisir la forme de la fonction de filtrage. Cette fonction est déterminée par la règle du jeu:

Neurone à t	**Nb vois**	**Val. A**	**Neurone à t+1**	**S=f(A) σ=0**
Mort	0	0	Mort	0
Vivant	0	0.5	Mort	0
Mort	1	1	Mort	0
Vivant	1	1.5	Mort	0
Mort	2	2	Mort	0
Vivant	2	2.5	Vivant	1
Mort	3	3	Vivant	1
Vivant	3	3.5	Vivant	1
Mort	4	4	Mort	0
Vivant	4	4.5	Mort	0
Mort	5	5	Mort	0
Vivant	5	5.5	Mort	0
Mort	6	6	Mort	0
Vivant	6	6.5	Mort	0
Mort	7	7	Mort	0
Vivant	7	7.5	Mort	0
Mort	8	8	Mort	0
Vivant	8	8.5	Mort	0

Pour centrer la fonction f, nous choisissons **σ = 3**.



ANNEXE Sources du Jeu de la vie