

# Le jeu de la vie (automate de Conway)

11 février

Le jeu de la vie est un automate cellulaire, proposé par Conway. Il ne s'agit pas à proprement parler d'un jeu, mais d'une modélisation simplissime du comportement d'une population de cellules sur une grille évoluant au fil du temps via des règles de disparition et de reproduction.

Le jeu consiste en une grille bi-dimensionnelle dont chaque case, peut être soit occupée par une cellule soit inoccupée. À chaque pas de temps, l'état d'une case de la grille est déterminé de façon unique par l'état des huit cases voisines de la façon suivante :

**reproduction** : une case inoccupée ayant exactement trois voisines occupées par des cellules devient vivante ;

**isolement** : une cellule vivante n'ayant pas plus qu'une case voisine occupée disparaît ;

**étouffement** : une cellule entourée par au moins quatre cellules meurt ;

**maintient** : une cellule entourée de deux ou trois case occupées se maintient en vie.

Le but de ce TP est de mettre en place une implémentation de cet automate cellulaire permettant de créer une configuration initiale de cellule sur une grille, puis de suivre l'évolution au cours du temps de cette colonie obéissant aux règles d'évolution ci-dessus.

## Exercice 1. Création du plateau

---

- a. Écrire une class `Grille` qui représentera la grille où les cellules évolueront. Elle devra contenir
  - un nombre de lignes et de colonnes en champs privés de type `int`.
  - un champ privé `grille` de type `boolean[][]` de la dimensions indiquée par le nombre de lignes et de colonnes
  - des méthodes `AjouteCellule` et `SupprimeCellule`
  - une méthode `Affiche` affichant la grille sur la sortie standard, en marquant les cellules par un caractère `'*'` et l'absence de cellule par un caractère `' '`.
  - une méthode `estOccupee(int i, int j)` indiquant si la case  $(i, j)$  est occupée.
  - une méthode `NbVoisins(int i, int j)` retournant le nombre de cellules voisines de la cellule courante.

## Exercice 2. Création du Jeu

---

On souhaite maintenant gérer le déroulement du jeu au fil du temps. Il faut pour cela écrire une classe `Jeu` qui maintient l'état courant d'une grille et dispose d'une méthode `Avance` qui effectue l'évolution d'un pas de temps.

a. Réfléchissez à comment effectuer cette mise à jour de la grille en limitant au maximum le nombre d'allocation en mémoire. Écrire cette classe `Jeu`. Elle devra disposer en outre, de méthodes permettant d'ajouter et de supprimer une cellule à un endroit donné, et une methode affichant la grille courante.

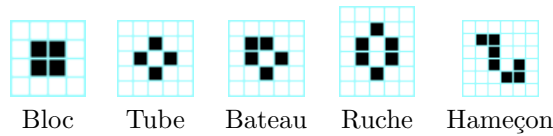
b. Tester votre jeu en créant dans un `main` une boucle infinie effectuant successivement un appel à `Avance`, `Affiche`, et une attente d'un temps fixé par un champ de votre classe `Jeu`.

c. Modifier vos classes pour utiliser deux threads : l'un en charge du calcul de l'évolution du jeu, et l'autre en charge de l'affichage, tout en faisant attention à leur synchronisation.

## A Quelques exemples d'automates cellulaires

### Structures stables

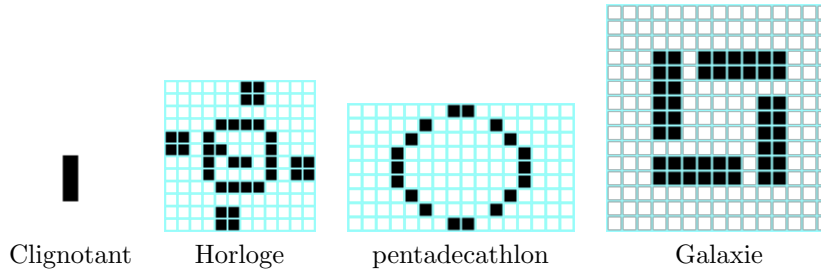
Les configuration suivantes sont stables : elles n'évoluent pas au fil du temps :



## Structures périodiques

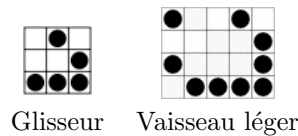
La première question qui s'est posée est de savoir s'il existe des configurations qui restent indéfiniment instables.

La réponse est oui. Voici une liste quelques un des plus célèbres qui sont périodiques, c'est à dire qui évoluent et reviennent à leur situation initiale :



## Structure vaisseau

Une structure est en vaisseau lorsqu'elle *se déplace* dans la grille. Il s'agit en général d'une structure cyclique mais dont la position après un cycle est translatée de la position précédente. Le plus petit vaisseau est le planeur (ou glisseur).



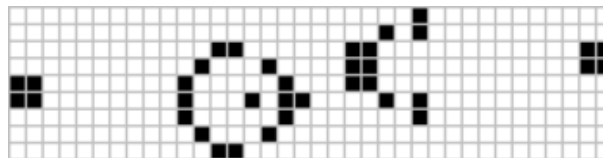
## Autres structures

Le Pentamino R est un automate cellulaire de type Matusalem, c'est à dire qu'il évolue pendant une longue période (ici 1103 générations) avant de se stabiliser vers des structures périodiques ou vaisseau.



## Structures infinies

Enfin, la question suivante est de savoir s'il existe des configurations qui vont engendrer un nombre croissant de cellules. La réponse est oui, et une preuve est apportée par l'existence d'une configuration, le canon à glisseur de Gosper, dont une partie est périodique et qui à chaque cycle va générer un glisseur qui partira glisser vers l'infini.



## Référence et Licence

Les images de cet énoncé proviennent des pages concernant les automates cellulaires et du jeu de la vie de wikipedia.fr et sont protégées par les licences CC-by-SA. On trouvera plus d'information sur le sujet sur ces pages : [https://fr.wikipedia.org/wiki/Jeu\\_de\\_la\\_vie](https://fr.wikipedia.org/wiki/Jeu_de_la_vie)