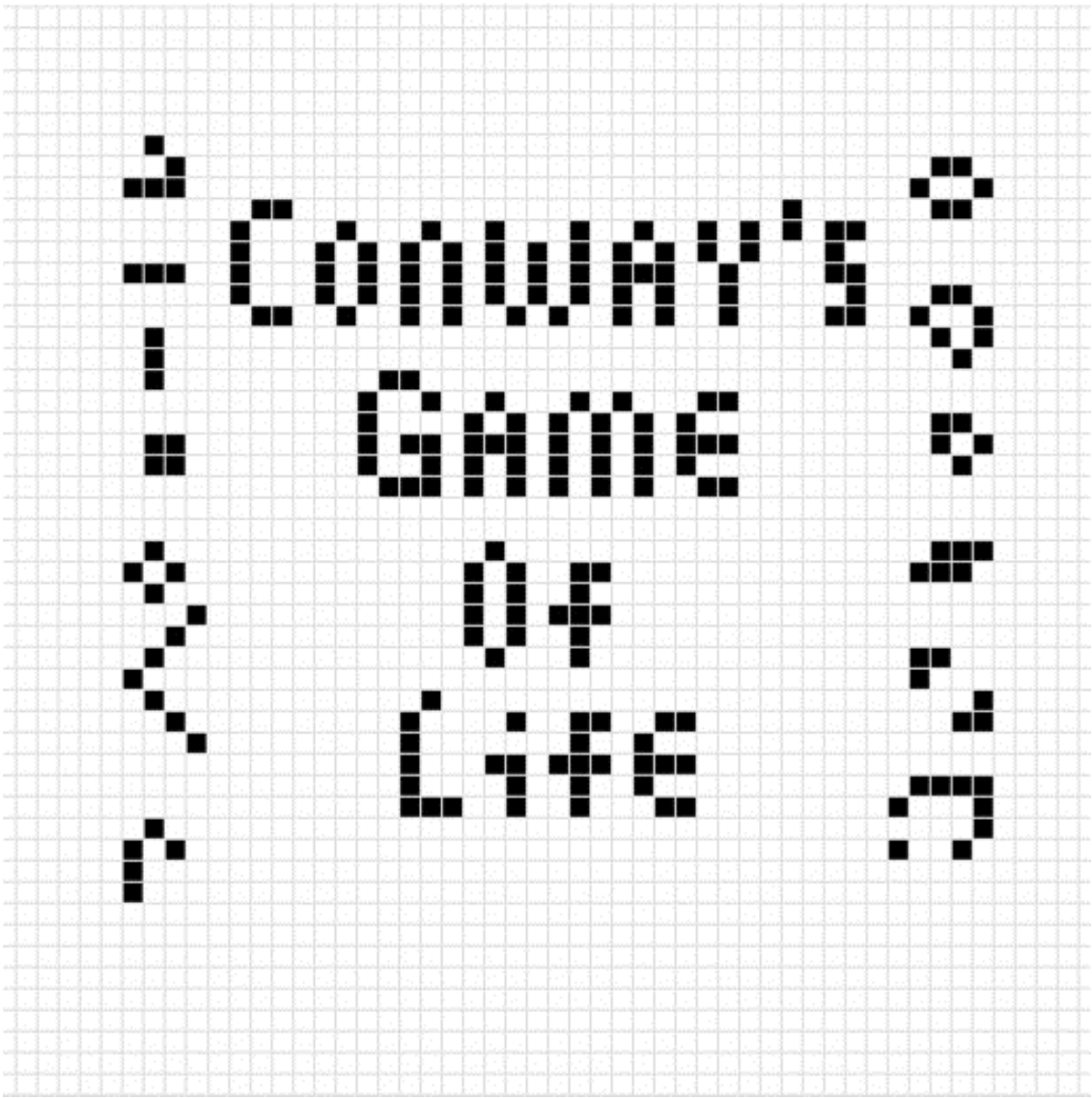
Soufiane Chaieb

L2 informatique

TP4 : le jeu de la vie par John Conway



Introduction :

Le jeu de la vie ne s'agit pas d'un simple jeu dans le sens conventionnel du terme, parce que c'est un jeu où on trouve aucun joueur, et donc ni gagnant ni perdant; une fois les cellules ont été initialisées, ce sont les règles qui déterminent ce qui va passer après. Cependant, ce jeu est plein de surprises ! car il est impossible de regarder la forme de départ et dire à quoi ressemblera la forme à la fin, le seul moyen c'est de suivre les règles.

Ce jeu a été inventé par le mathématicien John Conway en 1970. Il a choisi les règles de ce fameux jeu soigneusement, de façon que les cellules ne meurent pas rapidement ou qu'elles naissent avec des quantités énormes, le jeu veille que ces deux tendances soient bien équilibrées, et par conséquence, il est très difficile de prédire si une population prédéfinie, va à un moment donné disparaitre complétement ou continue à se reproduire infiniment.

Le jeu de la vie est un exemple du *cellular automaton,* qui est un système dans lequel plusieurs règles se sont appliquées à des cellules et leurs voisines dans une grille, c'est un domaine de recherche en mathématiques, où plusieurs cellular automata ont été inventé, pour faire des simulations de systèmes complexes.

1. Les règles du jeu de la vie :

Le jeu de la vie est joué sur une grille comme celle de jeu des échecs sauf qu'elle est plus grande(théoriquement de taille infinie), une cellule(case dans la grille) peut être soit vivante soit morte. Chaque cellule possède un voisinage de huit cellules qui elle entoure.

Pour passer d'une génération à un instant t à une autre à l'instant t+1, on compte le nombre de voisins de chaque cellule, l'évolution de la grille dépend de ce nombre de voisins.

- **La Naissance :** Une cellule morte et ayant exactement trois voisins vivants à un instant t devient une cellule vivante à l'instant t+1.

Naissance de la cellule entourée en jaune(instant t+1)

- **La Mort :** une cellule vivante peut mourir pour deux raisons :
    - **La surpopulation :** si la cellule à un instant t possède 4 voisins ou plus, elle meurt à l'instant t+1.
    - **La solitude :** si la cellule à l'instant t ne possède pas de voisins ou elle qu'un seul, elle meurt de solitude(ou sous-population).

Morte de solitude

morte de surpopulation

- **La survie :** une cellule vivante à un instant t, et ayant deux ou trois voisins vivants seulement, continue d'exister à l'instant t+1.

Cellules survivantes à l'instant t+1

2. Implémentation du jeu de la vie en langage C :

Pour implémenter ce jeu, on a utilisé des tableaux car plus efficaces et rapides (permettent d'accéder à nos cases dans un temps constant) et principalement deux fonctions :

- Première fonction : permet de compter le nombre des voisins de chaque case donnée en argument, avec deux boucles for qui parcourt le tableau de deux dimensions et un compteur qui serait incrémenter à chaque fois qu'une cellule voisine est vivante. Cette fonction a été écrite avec deux variant, suivant le sujet du TP.
    - *Count_nghbrs_simple_grid* : fonction qui compte le nombre de voisins d'une cellule sans prendre en compte le voisinage de torus c-à-d dans un tableau normal de taille appelé size, où les cases situées aux quatre coins du tableau ont seulement 3 voisins, celles situées sur les lignes/colonnes 0 et size-1(sauf celles des coins) possèdent 5 voisins, les autres ont 8 voisins.
    - *Count_nghbrs_torus_univers* : cette fonction fait la même chose mais cette fois-ci en prenant on compte l'univers de torus c-à-d dans un tableau circulaire où chaque case possède exactement huit voisins.

- Deuxième fonction : appelé *game_of_life*, c'est dans cette fonction où on a implémenté les règles du jeu de la vie, à l'aide de structures conditionnelles if (conditions) then, elle parcourt un tableau dit de référence (instant t), puis pour chaque case récupère le nombre de voisins, et rempli la nouvelle grille (instant t+1) avec des valeurs appropriées suivant les règles appliquées.

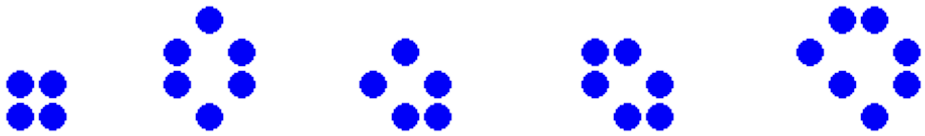
D'autres fonctions ont été utilisées, notamment *print_2d_grid* qui permet d'afficher la grille remplie(affiche un point si la cellule est morte, une croix si la cellule est vivante), i*nit_grid_manually* permet l'utilisateur d'initialiser la grille avec des cellules vivantes avant de commencer le jeu, *init_grid_randomly* s'appuie sur le générateur de Mersenne Twister (vu en TP 2) pour remplir la grille aléatoirement avec les valeurs 0 et 1.

L'utilisateur a le droit de préciser la taille de la grille(Attention : affichage encombrant dans le terminal si la taille est très grande), il a aussi le choix d'initialiser la grille à la main ou bien aléatoirement, le choix de préciser l'univers qui le souhaite(univers torus vs grille simple), et enfin il peut entrer le nombre de génération qu'il veut voir(le nombre de steps).

3. Jouons le jeu ! :

Pour reconnaitre à quel point ce jeu de vie est magique, il faut essayer l'initialiser avec des figures différents même aléatoires et voir que ces figures donnent d'autres figures identifiables, parmi ses figures qu'on peut identifier facilement :

- **Figures toujours vivantes (still life) :** ces figures restent les mêmes, stable d'une génération à l'autre. Les cellules ne meurent pas, et pas de nouvelle cellule vivante qui apparait. Pour qu'une figure soit toujours vivante, il faut chacune de ses cellules vivantes possède 2 ou 3 voisins vivants, et chaque cellule morte doit avoir un nombre de voisins diffèrent de 3. Parmi ces figures on trouve :



(block) (beehive) (boat) (ship) (loaf)

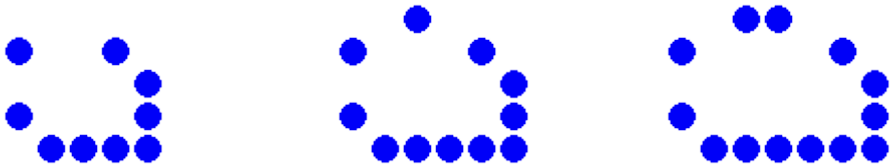
- **Figures périodiques/oscillatoires :** se sont des figures qui change d'une génération à l'autre mais éventuellement revient à leur forme. Parmi elles on trouve(reviennent à leurs formes après seulement deux génération) :

(blinker) (Toad)

- **Gliders :** un glider est une figure de 5 cellules vivantes qui lui aussi revient à ça forme originale(glider) après 4 générations, mais il se déplace dans la grille diagonalement par une cellule quand il revient à sa forme.

(**glider**)

- **Vaisseau spatial (spaceships) :** se sont des figures qui se déplacent à gauche, à droite, en haut et vers le bas, contrairement au glider qui se déplace sur la diagonale.



(light weight spaceship) (middle weight) (heavy weight)

il n’est pas évident de prédire les figures qui surgissent à partir d’une figure donnée, ni de dire si elle va se stabiliser à instant t ou va continuer d’évoluer infiniment. Par exemple, en initialisant le jeu avec une ligne de N cellules vivantes, on aura pour :

- N = 1 ou 2, ces figures disparaissent complètement.
- N = 3 : un Blinker(voir figure ci-dessus).
- N = 4 : devient un Beehive à t = 2.
- N = 5 : devient feu de circulation à t = 6.
- N = 6 : disparaissent à t = 12.
- N = 8 : donne 4 blocks et 4 beehives.
- N = 9 : donne deux feu de circulation.
- N = 11 : devient deux blikers.

Conclusion :

Le jeu de la vie est un exemple simple de système auto-organisateur(self-organizing systems), qui est un domaine de recherche qui étudie les comportements et l’évolution de d’un objet à partir de quelques règles qui y sont appliquées. Par exemple : comprendre comment les pétales d’une rose ou les rayures d’un zèbre peuvent surgissent d’un tissu des mêmes cellules. Ce domaine pourrait nous aider, à comprendre la diversité de la vie sur notre planète.