Université Côte d'Azur
Automates & Langages

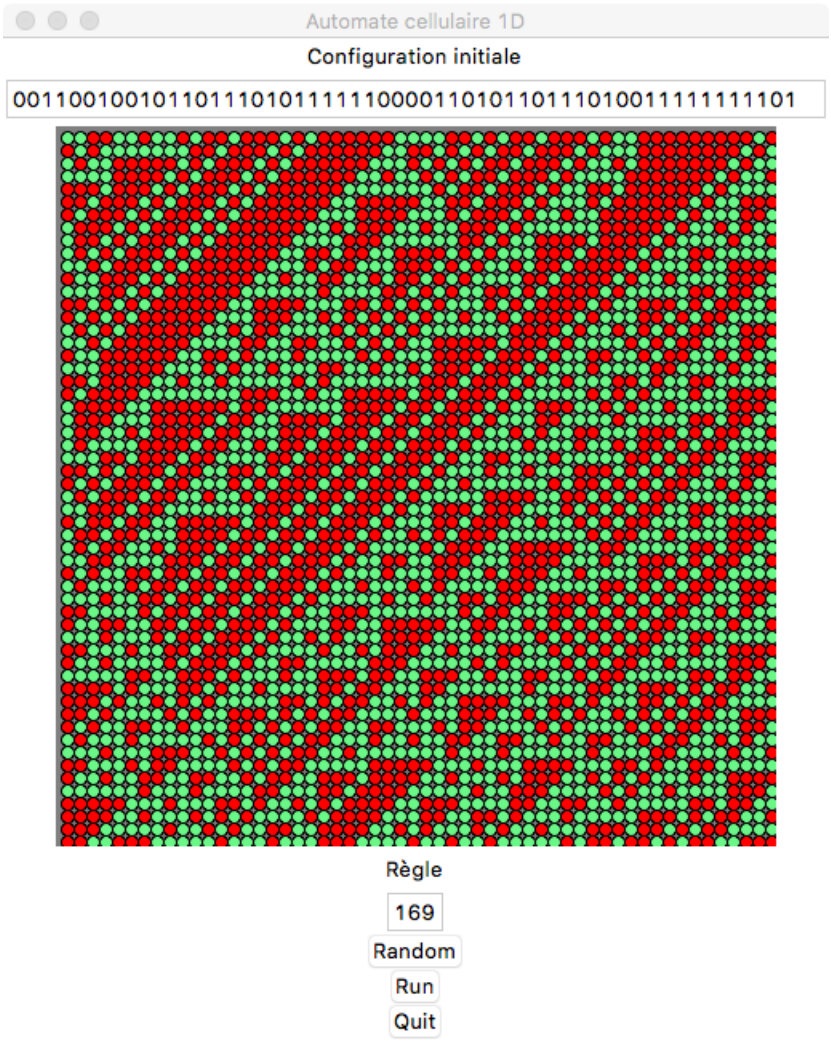
L3 Info, L3 MI, L3 ST
2023–24

TP n°6

Automates cellulaires

Présentation

Le but de ce TP est de visualiser le diagramme espace-temps d'un automate cellulaire élémentaire. L'utilisateur entre le numéro d'un automate cellulaire (AC) et une configuration initiale qui peut aussi être choisie aléatoirement. Il lance l'affichage du diagramme espace-temps en appuyant sur le bouton Run d'une fenêtre graphique.
Une version minimale de l'interface graphique programmée en `tkinter` est fournie dans la classe `CA-GUI`. Il reste donc à programmer la classe `CA` qui va calculer l'évolution en fonction de la règle de l'AC choisie et aussi de la configuration initiale.



La classe `CA`

La classe `CA` est composée du constructeur et de deux méthodes, c'est la seconde qui contient le cœur de l'évolution de l'AC. Récupérez les fichier `CA-GUI.py` et `CA.py`. Le premier contient le code de l'interface graphique, le second, à compléter, contient la classe `CA`.

Exercice 1.

On commence par écrire le constructeur `_init_(self,cfg,n)` qui à partir d'une configuration `cfg` et du numéro n de la règle de l'AC choisi (entre 0 et 255) instancie les deux attributs `config` et `regle`.

1. On reçoit la configuration `cfg` choisie par l'utilisateur sous la forme d'une chaîne de 56 caractères. Il faut la transformer en une liste d'entiers 0 ou 1. Pour pouvoir appliquer la fonction de transition aux cellules des bords, deux possibilités :
   - soit on ajoute deux 0 aux extrémités qui seront immuables et l'AC évoluera entre ces deux *murs* qui ne seront pas affichés ;
   - soit, bien plus élégamment, on recopie la valeur la plus à gauche de la configuration à sa droite et la valeur la plus à droite à sa gauche de sorte à ce que l'AC évolue dans un *cylindre*. Là encore, on n'affichera pas les cellules extrémales dupliquées.

   L'attribut `config` reçoit alors cette adaptation de la configutation initiale entrée par l'utilisateur.
2. L'attribut `regle` est initialisé à partir de l'entier n qui doit donc être compris entre 0 et 255. Il contient dans l'ordre la liste des images des 8 triplets `(1,1,1)`, `(1,1,0)`, `(1,0,1)` ... jusqu'à `(0,0,1)` et `(0,0,0)`.

Exercice 2. *(facultatif)*

Ecrivez une méthode `_str_(self)` qui retourne la chaîne correspondante à la liste `config` privée de ses deux extrémités. Elle pourra éventuellement vous aider à mettre au point votre programme dans l'Exercice 3. On peut s'en passer mais n'hésitez pas à y revenir au besoin.

Exercice 3.

La méthode `next(self)` est chargée de calculer l'évolution d'une configuration donnée au temps suivant. Pour chaque cellule non-extrémale i de `config`, elle considère son état et les états de ses cellules voisines pour connaître le triplet auquel appliquer la fonction de transition.

Par exemple, si une cellule est dans l'état 1, sa voisine de gauche dans l'état 1 et celle de droite dans l'état 0, le triplet 110 donne lieu à l'entier $k = 6$. L'image du triplet k se situe dans `regle[7-k]` du fait que les triplets sont ordonnés par ordre décroissant de leurs équivalents en binaire comme vu à la fin de l'Exercice 1.

Ecrivez la méthode `next(self)` qui met à jour l'attribut `config` : cela revient à appliquer de façon synchrone la fonction de transition à chacune des cellules, sauf à celles rajoutées aux bords.

La classe `CA-GUI`

Exercice 4.

1. Prenez connaissance de la classe `CA-GUI` fournie et exécutez-la. Si tout est correctement programmé auparavant, vous pouvez entrer une configuration initiale et visualiser l'exécution d'un automate cellulaire de votre choix dessus.
2. Il manque tout de même un bouton Random afin de remplir automatiquement et de façon aléatoire la configuration initiale et le numéro de l'AC. Ajoutez le code permettant l'activation d'un tel bouton.