Vie Artificielle
Automates Cellulaires

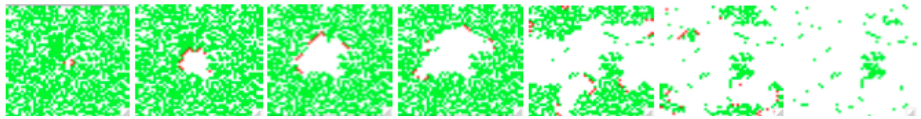
Mise à jour : Janvier 2026

Mots clés: *automates cellulaires, embouteillages, feux de forêt*

Les ressources nécessaires pour cette UE (lien vers les sources, les sujets, etc.) sont disponibles sur Moodle. En particulier vous trouverez un lien vers les codes sources donnés en annexe.

Automates cellulaires 2D : Les feux de forêts [sur ordinateur]

On s'intéresse à la simulation du développement d'un feu de forêt. Pour mémoire, il s'agit d'un automate cellulaire 2D ou une cellule ne devient active que si un seul de ces voisins est actif (voisinage de von Neumann). Pour réaliser cet automate, on vous demande d'étudier le code source en annexe *ForestFire.java* et de le modifier.



Les règles de mise à jour sont les suivantes (et leur interprétation):
- une cellule blanche reste blanche. C'est-à-dire un emplacement vide reste vide
- une cellule verte reste verte sauf si au moins une des cellules voisines est rouge. C'est-à-dire: un arbre prend feu si au moins un de ces voisins est en feu
- une cellule rouge devient blanche. C'est-à-dire: un feu meurt en une itération

QUESTION 1. Programmer les règles de cet automate cellulaire. On considère que les voisines hors champs des cellules au bord sont des cellules vides (ie. état blanc). On vous demande ensuite de découvrir expérimentalement le seuil de percolation au dessus duquel la majorité de la forêt brûle.

QUESTION 2. Même question, mais on considère maintenant un environnement périodique, c'est à dire la cellule sur le bord gauche a pour voisine *à sa gauche* la cellule sur la même ligne qui se trouve sur le bord droit. De même pour les cellules sur les bords droit, haut et bas.

QUESTION 3. Etendez l'automate cellulaire avec un état "cendres" (cellule de couleur noire) et modifiez la règle transformant une cellule rouge en cellule noire (ie. lorsqu'un feu meurt, on trouve des cendres). Une cellule noire disparaît après une itération. Indice: regardez la classe CAImageBuffer, fonction updateForest

QUESTION 4. *V*ous devez maintenant tracer l'évolution du nombre d'arbres sains au cours du temps, avec les itérations (axe X) et le nb_arbres_vivants_actuellement / nb_arbres_vivants_au_depart (axe Y).

Vous pouvez le script Python plotCSV fourni, qui fonctionne avec Matplotlib. Autres options: LibrOffice Calc, Google Sheet, etc. Vous devez donc générer à partir de votre programme des données interprétables pour pouvoir être affiché (typiquement: format CSV) (cf. fichier *aide.txt* et *exemple.csv* dans *plotCSV.zip*). Vous devez tracer 2 graphes pour chacune des densités initiales 0.5 et 0.6 (ie. 4 graphes au total).

Astuce: *Il va falloir mesurer l'évolution de la densité d'arbres au cours du temps et laisser tourner votre simulation jusqu'à stabilisation (ie. plus d'arbres en feu).*

QUESTION 5. Introduisez de nouvelles règles associées à une probabilité de réalisation.

- un arbre peut spontanément prendre feu (avec une probabilité *p1*)
- un emplacement vide peut spontanément voir apparaître un nouvel arbre (avec une probabilité *p2*)

Explorer l'influence de plusieurs couples de valeurs *p1* et *p2*. Pour trois valeurs de *p1* et *p2*, tracez le nombre d'arbres vivants au cours du temps pendant 1000+ itérations.

Modélisation d'une file d'automobiles [sur ordinateur]

On souhaite maintenant simuler les déplacements d'automobiles sur une seule file. On considère un tableau à une dimension (la route), sur laquelle on met des voitures (cases actives) ou non (cases inactives). Voici un exemple d'initialisation:

On souhaite simuler le déplacement selon la règle suivante:

une voiture avance avec une probabilité *p* si et seulement si la case à sa droite est libre.

On fait l'hypothèse que les déplacements sont synchronisés (ie. toutes les voitures essayent de se déplacer au même moment). Dans la suite, vous choisirez au hasard l'état initial sur un automate de taille 50.

QUESTION 1. Simuler cet automate en modifiant le fichier donné - On fait l'hypothèse que *p=1.0*, c'est à dire qu'une voiture se déplace toujours si elle le peut, et qu'une voiture disparaissant à droite réapparait à gauche (ie. les voitures se déplacent sur un circuit). Qu'observez-vous?

QUESTION 2. *p=1.0* est une hypothèse irréaliste (il est peut probable que les automobilistes se déplacent de manière synchronisée). Relancer votre simulation avec *p=0.5*, qu'observez vous?

QUESTION 3. en partant dans la question 1, construisez un automate à 6 états comme vu dans le cours. L'état 0 signifie qu'il n'y a pas de voiture, les états de 1 à 5 signifient qu'il y a une voiture et le nombre permet de calculer la probabilité de déplacement, qui est de (x-1)/5 avec x la valeur de la case (i.e. "6" se déplace à chaque pas de temps).

PREMIER PAS VERS LE PROJET, A PARTIR DES FEUX DE FORÊT

Il n'est pas nécessaire de faire cette partie pendant le TME. Il s'agit d'une suggestion pour commencer le projet.

QUESTION 6.

Etendez votre feu de forêt pour intégrer la dispersion des cendres (une case noire redevient blanche après N=10 itérations).

Implémentez les règles permettant de modéliser l'étalement d'une quantité d'eau dans l'environnement. Vous pourrez par exemple modéliser l'eau présente en prenant en compte la quantité d'eau.

Construisez un monde avec des arbres (+feu) et des étendues d'eau, et où les arbres (non brûlés) agissent comme barrage pour l'eau. Vous pourrez par exemple utiliser deux automates cellulaires différents et composer un affichage combinant les deux.

Etendez votre environnement avec une carte d'altitude (ie. une matrice qui pour chaque case de l'environnement donne l'altitude) et modifier les règles d'écoulement d'eau pour prendre en compte l'altitude dans la direction de l'écoulement. On souhaite pouvoir observer à la fois des écoulements et le remplissage d'un bassin (ie. une quantité d'eau initialement sur une cellule est répartie sur les cellules voisines, mais limitée par la géographie, ie. les cases dont l'altitude bloquent l'écoulement d'eau).

Suggestion pour le projet: l'intérêt d'une méthode de mise à jour asynchrone est que vous pourrez facilement moduler le nombre de mises à jour faites à chaque pas. Par exemple, vous pourrez décider qu'à chaque déplacement de vos agents dans l'environnement, seul 1/10e de votre environnement sera mis à jour. C'est très utile lorsque la mise à jour de votre environnement est très couteuse en temps, afin d'éviter qu'elle ralentisse l'ensemble de la simulation.