

# Simuler les feux de forêt

Victor Sarrazin

22 mars 2025

## Introduction

Avec le changement climatique, les feux de forêt sont de plus en plus fréquents et dévastateurs, à l'image de ceux en Californie en janvier 2025.

Dans ce cadre, les modélisations informatiques des feux de forêt permettent de simuler l'évolution des feux, afin de prévoir les zones à risques. Mais les simulations informatiques permettent aussi de tester l'impact de certaines transformations sur ces feux, afin de trouver des manières de réduire l'impact des catastrophes, sans pour autant dénaturer les forêts.

## 1 Automates cellulaires

### 1.1 Généralités

Pour réaliser nos modélisations, nous allons utiliser des automates cellulaires. Les automates cellulaires permettent de représenter des systèmes avec des interactions locales entre les éléments qui constituent ces derniers. Les automates cellulaires ont notamment été popularisés avec *Le Jeu de la Vie* de *Conway*.

**Définition 1.** Un automate cellulaire est la donnée d'un triplet  $(Q, M, f)$  avec :

- $Q$  un ensemble d'états
- $M$  une matrice de taille  $n \times m$ , où chaque  $m_{i,j}$  représente une case de la grille
- $f : Q^k \longrightarrow Q$  une *fonction de transition* qui à l'état de la case et de  $k - 1$  voisins renvoie l'état suivant

// TODO : Améliorer la définition des états de transition

La définition de la fonction de transition dépend donc du type de voisinage utilisé. Il existe deux principaux types de voisinages, celui de *von Neumann* et celui de *Moore*. Nous avons

choisi d'utiliser le voisinage de *Moore* pour nos modélisations pour prendre en compte le plus d'interactions possibles, celui de *von Neumann* étant limitant notamment sur les diagonales.

**Définition 2.** Le voisinage de *Moore* est composé du noeud et de ses 8 voisins.  
 Dans le cas du voisinage de *Moore*, on a donc  $f : Q^9 \longrightarrow Q$

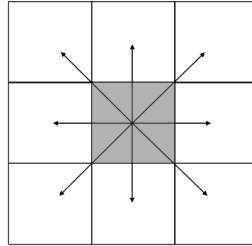


FIGURE 1 – Voisinage de Moore

Il serait possible dans une optique d'avoir des modèles encore plus précis d'utiliser des voisinages de *Moore* étendus, avec plusieurs rayons de voisins, mais dans un souci de simplicité nous nous sommes restreints à un rayon.

## 1.2 Représentation d'un forêt

Afin de représenter une forêt avec un automate cellulaire nous avons défini une liste d'état que les cases pouvaient prendre.

**Définition 3.** Les états possibles sont :

- Arbres
- Arbres denses
- Champs
- Chemins/Tranchée
- Feu
- Case brûlée \*
- Eau \*

A noter que les cases (\*) ne peuvent brûler.

Nous avons choisi de travailler avec une matrice de taille  $256 \times 256$  pour des raisons techniques : au delà le temps de calcul devenait trop long et l'ajout de lignes/colonnes ne représentait pas un intérêt suffisant pour le faire par rapport au temps de calcul que cela impliquait.

## 2 Modèle d’Alexandridis

Dans un premier temps il est possible de faire une modélisation naïve des feux de forêt en associant à chaque case une probabilité fixe de brûler si un de ses voisins est en feu. Une telle tentative a été réalisée avec des résultats quelque peu satisfaisants, c’est pourquoi l’objectif de cette deuxième partie est de voir un modèle plus poussé de modélisation des feux de forêts, dit modèle d’*Alexandridis*.

### 2.1 Présentation du modèle

Le modèle d’*Alexandridis* prend en compte divers phénomènes pour modéliser cette propagation. Dans la suite nous avons choisi de nous restreindre à la densité de végétation, au type de végétation ainsi qu’au vent dans la zone. Le modèle original quant à lui prend aussi en compte l’élévation du terrain.

**Proposition 1.** On utilise les règles de transition suivantes, pour  $(i, j) \in \mathbb{N}^2$ ,  $t \in \mathbb{N}$  :

- Si  $m_{i,j}(t) = \text{feu}$  alors  $m_{i,j}(t+1) = \text{brulé}$
- Si  $m_{i,j}(t) = \text{feu}$  alors  $m_{i\pm 1, j\pm 1}(t+1) = \text{feu}$  avec une probabilité  $p_b$
- Si  $m_{i,j}(t) = \text{brulé}$  alors  $m_{i,j}(t+1) = \text{brulé}$

**Proposition 2.** La probabilité  $p_b$  qu’une case brûle est définie par  $p_b = p_h(1+p_{veg})(1+p_{den})p_{vent}$  avec  $p_h = 0.27$  une constante,  $p_{veg}$  et  $p_{den}$  relatives au type de case.

	$p_{veg}$	$p_{den}$
Arbres	0.3	0.3
Arbres denses	0.3	0
Champs	−0.1	0

FIGURE 2 – Probabilités  $p_{veg}$  et  $p_{den}$  selon le type de végétation

**Proposition 3.** La probabilité  $p_{vent}$  liée au vent est définie par  $p_{vent} = \exp(0.045v) \times \exp(0.131v \times (\cos(\theta) - 1))$  avec  $\theta$  l’angle entre la propagation du feu et la direction du vent, et  $v$  la vitesse du vent (en  $m/s$ )

### 2.2 Résultats

Dans un premier temps comparons les résultats de la modélisation naïve et celui du modèle d’*Alexandridis* :

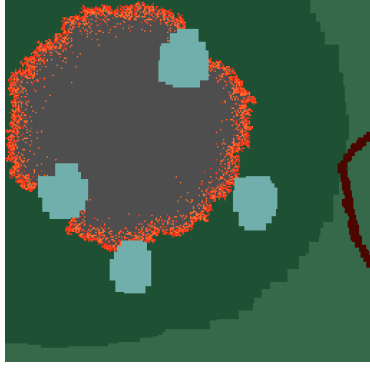


FIGURE 3 – Solution naïve

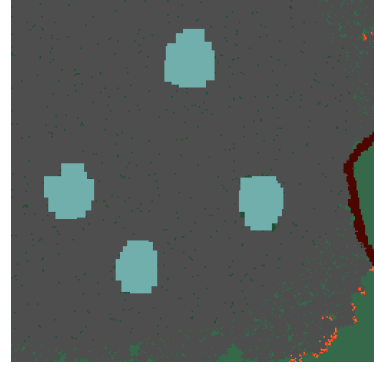


FIGURE 4 – *Alexandridis*

On remarque que la solution naïve a une propagation plus lente et un front de feu plus étendu que celui du modèle d'*Alexandridis*.

Comparons maintenant deux modélisations avec des densités de végétation différentes, et sous  $15m/s$  de vent vers l'est :

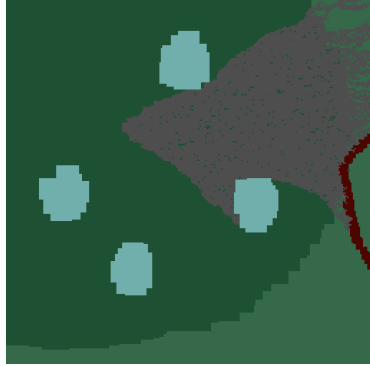


FIGURE 5 – Végétation normale

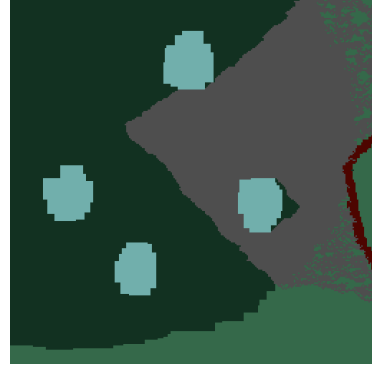


FIGURE 6 – Végétation dense

On remarque principalement deux effets notoires sur les deux modélisations.

D'une part, la progression dans une végétation normale est moins rapide (ce qui fait sens empiriquement), et donc le front est moins étendu. La surface brûlée dans le cas de la densité normale présente aussi des trous là où celle de la densité élevée est lisse, ce phénomène est expliqué par le fait que  $p_b$  est plus élevée pour une densité plus forte.

D'autre part, dans les deux modélisations on voit clairement l'impact du vent : le front de feu s'est déplacé suivant un cône poussé par le vent.

## 2.3 Considérations algorithmiques

Cette sous partie a pour objectif de rendre compte de l'implémentation de la modélisation réalisée et de comprendre le choix de  $256 \times 256$ .

Voici les complexités des différentes fonctions créés :

- Calcul de  $p_b$  :  $O(1)$
- Calcul du passage de  $t$  à  $t + 1$  :  $O(n^2)$  avec  $n$  la longueur de la grille, avec une estimation de 100 opérations élémentaires pour chaque case
- Affichage de l'état  $t$  :  $O(m^2 \times n^2)$  avec  $n$  la longueur de la grille et  $m$  la taille d'une case représentée sur l'écran

Ainsi pour  $N$  itérations, on trouve une complexité totale de  $O(m^2 \times n^2 \times N)$ , numériquement avec  $n = 256$ ,  $m = 2$  et  $N = 200$  on trouve  $\approx 10^5$ , et donc on en déduit grossièrement  $\approx 10^7$  opérations élémentaires, pour un temps de calcul d'environ 5 secondes.

On comprend donc pourquoi augmenter la valeur de  $n$  (toutes choses égales par ailleurs) ou de  $N$  conduirait à des simulations plus longues.

## 3 Transformations

// TODO : Final part

## 4 Annexe

### 4.1 Choix d'implémentation

Pour implémenter les différents modèles nous avons choisi d'utiliser le langage C99. Le choix du langage a été motivé principalement par deux aspects :

- Le C permet par son bas niveau de gagner un certain temps d'exécution par rapport à d'autres langages, utile ici au vu du nombre de calculs à effectuer dans nos grilles.
- Le C étant plus utilisé que le OCaml bénéficie par conséquent de plus de bibliothèques externes.

Nous avons utilisé les bibliothèques non standard suivantes :

- SDL2 : Pour réaliser une interface graphique afin de visualiser les évolutions.
- cJSON : Pour manipuler des fichiers JSON et représenter les grilles afin de pouvoir les importer.

- `png` : Pour créer des fichiers PNG et les manipuler en C afin d'exporter nos simulations.