Simuler les feux de forêt

Victor Sarrazin

24 mai 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 leur évolution et ainsi de prévoir les zones à risque. Elles offrent également la possibilité de tester l'impact de certaines transformations sur ces incendies, afin d'identifier des moyens de réduire les conséquences des catastrophes sans dénaturer les forêts.

1 Automates cellulaires

Pour réaliser nos modélisations, nous allons utiliser des automates cellulaires. Ces outils permettent de représenter des systèmes avec des interactions locales entre les éléments qui les constituent. 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 cellule de la grille

— $f : \mathcal{M}_{n,m}(Q) \longrightarrow \mathcal{M}_{n,m}(Q)$ une *fonction de transition* qui à une grille renvoie la grille suivante

La définition de la fonction de transition dépend donc du type de voisinage considéré. 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'intéractions possibles, celui de *von Neumann* étant plus limitatif, notamment en ce qui concerne les interactions diagonales.

Définition 2. Le voisinage de *Moore* est composé de la cellule centrale et de ses 8 voisins adjacents (horizontalement, verticalement et diagonalement).

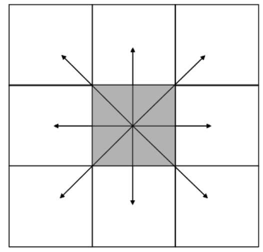

FIGURE 1 – Voisinage de Moore

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. Cependant par souci de simplicité nous nous sommes restreints à un rayon.

2 Modèle d'Alexandridis

Dans un premier temps, il est possible de réaliser une modélisation simple des feux de forêt en associant à chaque cellule une probabilité fixe de s'enflammer si l'un de ses voisins est en feu. Une telle tentative a donné des résultats mitigés. C'est pourquoi l'objectif de cette deuxième partie est de présenter un modèle plus avancé de modélisation des feux de forêts, appelé modèle d'*Alexandridis*.

2.1 Présentation du modèle

Le modèle d'*Alexandridis* prend en compte divers phénomènes pour simuler la propagation des incendies. Par la suite, nous avons choisi de nous concentrer sur la densité de végétation, le type de végétation ainsi que le vent dans la zone. Le modèle original prend également en considération 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 $m_{i,j}(t) = \texttt{feu}$ alors $m_{i,j}(t+1) = \texttt{brulé}$

— Si $m_{i,j}(t) = \texttt{feu}$ alors $m_{i\pm1,j\pm1}(t+1) = \texttt{feu}$ avec une probabilité p_b

— Si $m_{i,j}(t) = \texttt{brulé}$ alors $m_{i,j}(t+1) = \texttt{brulé}$

Proposition 2. La probabilité p_b qu'une cellule brûle est définie par $p_b = p_h(1+p_{veg})(1+p_{den})p_{vent}$ avec $p_h = 0.27$ une constante, et p_{veg} et p_{den} des coefficients relatifs au type de cellule.

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 θ l'angle entre la propagation du feu et la direction du vent, et v la vitesse du vent (en m/s)

	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

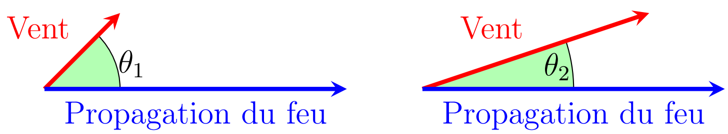

Figure 3 – Configurations de vent

On a $\theta_1 > \theta_2$ et une vitesse plus élevée dans la seconde situation, ainsi la probabilité p_{vent} sera plus élevée dans la seconde situation.

2.2 Résultats

Voici un résultat de modélisation de feu, avec un vent de $15m/s$ orienté vers l'est :

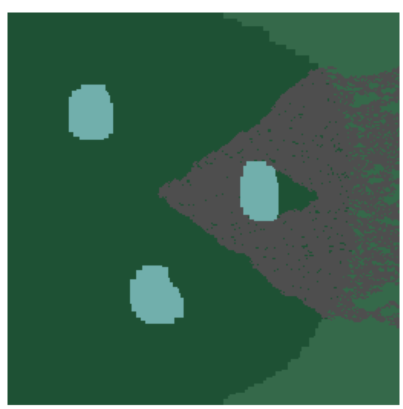

Figure 4 – Simulation Alexandridis

3 Hashlife

Afin de réaliser des modélisations plus conséquentes (notamment en augmentant la taille de la grille) il faut trouver une manière de réduire le nombre de calculs. C'est l'objectif de l'algorithme de mémoïsation `Hashlife` notamment utilisé dans le logiciel `Golly` [1] pour les simulations du *Le Jeu de la Vie* de *Conway*.

Au lieu de calculer chaque cellule indépendamment, l'algorithme considère des groupements de cellule dits *macro-cellules* de taille $2^n \times 2^n$, que l'on peut redécouper en 4 quadrants de taille $2^{n-1} \times 2^{n-1}$ (partie en bleu sur la figure).

On appelle *résultat* la macro-cellule centrale de taille $2^{n-1} \times 2^{n-1}$ (partie en rouge sur la figure).

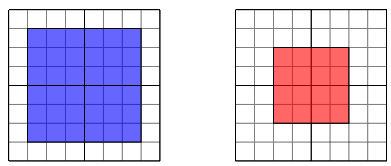

FIGURE 5 – Une macro-cellule de taille $2^3 \times 2^3$

Ce découpage en macro-cellule permet de calculer le résultat sans considérer aucune information extérieure à la macro-cellule pendant 2^{n-2} unités de temps. Ce calcul est fait récursivement selon deux cas :

Soit $n = 2$, le résultat est composé d'uniquement 4 cellules, dans ce cas on le calcule directement en appliquant les règles de notre automate cellulaire.

Sinon $n > 2$, on a donc 4 quadrants de taille $2^{n-1} \times 2^{n-1}$ pour lesquels on calcule récursivement leur résultat (ici en bleu). Ensuite il faut calculer les 5 macro-cellules de taille $2^{n-2} \times 2^{n-2}$ (en rouge) en considérant les macro-cellules de taille $2^{n-1} \times 2^{n-1}$ associées (exemple en vert).

Enfin après ces précalculs effectués, on peut calculer les quatre macro-cellules de taille 2^{n-2} (en jaune) afin d'obtenir le résultat que l'on recherche.

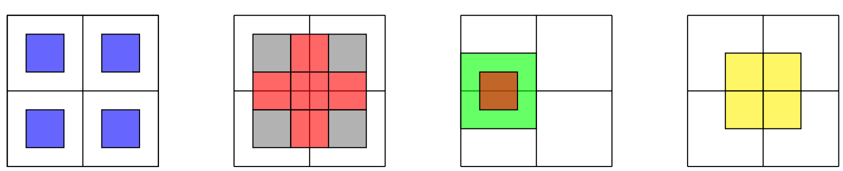

FIGURE 6 – Calcul dans le cas $n > 2$

Si dans le cas $n = 2$ on fait le même nombre d'opérations que dans la simulation naïve, le cas $n > 2$ réalise une réelle économie de calcul.

Mais `Hashlife` n'est pas uniquement une réduction du nombre de calculs dans les itérations. En effet l'algorithme exploite le fait que les automates cellulaires présentent des répétitions de motifs durant l'exécution, notamment avec les *gliders* dans *Le Jeu de la Vie* volant jusqu'aux bordures (infinies) de la grille :

1. https ://golly.sourceforge.io/

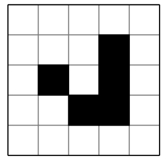

FIGURE 7 – Un *glider*

C'est pourquoi `Hashlife` réalise aussi une mémoïsation des résultats de chaque quadrant à l'aide d'une table de hashage, d'où le nom de l'algorithme. Celà permet de réutiliser au maximum les calculs et d'éviter de recalculer les situations récurrentes.

Si `Hashlife` permet de réduire drastiquement le temps de calcul pour des grandes grilles et de s'affranchir de bornes de ces dernières en exploitant les motifs qui se répètent, il existe toujours des cas dégénérés où l'algorithme a une complexité similaire à celui naïf si les motifs ne se répètent jamais [2].

4 Conclusion

Les automates cellulaires permettent de simuler de nombreux systèmes avec des interactions tels que les feux de forêt. Cependant pour réussir à avoir des simulations d'une taille convaincante (et dans un temps raisonnable) il faut implémenter des algorithmes plus poussés tels que `Hashlife`.

2. L'algorithme `Quicklife` permet de résoudre ce problème