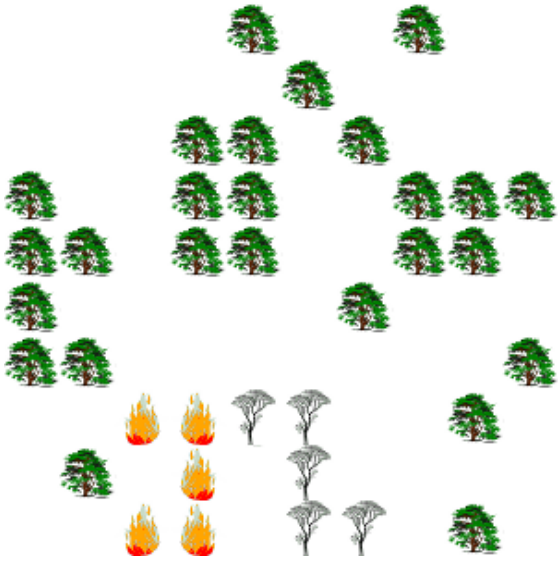
Projet C : Niveau avancé

Émulation d'Incendie en Milieu Forestier



Année universitaire: 2023/2024

Pr. Khadija SLIMANI

1. Présentation

L'objectif du projet consiste à simuler la propagation d'un incendie au sein d'une forêt. Nous envisageons une forêt avec des dimensions de longueur et de largeur, ce qui implique que la forêt peut être représentée sous forme de matrice. Cette matrice de la forêt est composée de cellules avec une taille correspondant à la longueur et à la largeur. Chaque cellule peut revêtir différentes caractéristiques :

- sol
- arbre
- feuille
- roche
- herbe
- eau
- cendres
- cendres éteintes

Chaque type est symbolisé graphiquement à l'aide de symboles spécifiques (dans le cas où le projet est développé en mode console/terminal), conçus exclusivement pour l'affichage visuel.

Type	Symbole	Degré	État
SOL	**+**	**0**	**0**
ARBRE	*****	**4**	**0**
FEUILLE		**2**	**0**
ROCHE	**#**	**5**	**0**
HERBE	**x**	**3**	**0**
EAU	**/**	**0**	**0**
CENDRES	**-**	**1**	**0**
CENDRES ETEINTES	**@**	**0**	**0**

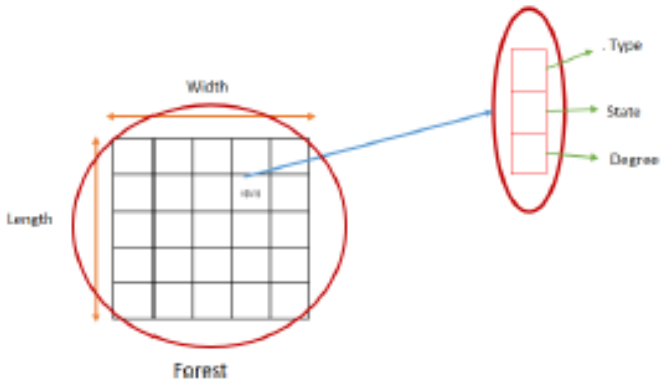


Figure 1 : Représentation de la forêt en mémoire

Chaque cellule de la forêt est caractérisée par

- un type,
- un état entier indiquant si la case est en feu (1) ou non (0),
- et un degré positif indiquant la rapidité avec laquelle le type peut brûler.

Par exemple, une feuille à terre brûle rapidement, donc son degré est de 2, tandis qu'un arbre, dont le degré est de 4, met beaucoup plus de temps à brûler.

Vous devez être en mesure d'initialiser la forêt en plaçant de l'herbe, des roches, des arbres, des feuilles, du sol et de l'eau. Ce remplissage peut être effectué de manière aléatoire ou manuelle.

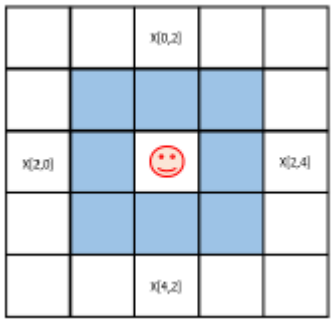


Figure 2 : Voisinage de la cellule (2,2)

Pour simuler la propagation du feu, les règles suivantes sont définies :

- Si une cellule [i, j] est en feu à l'itération de simulation t et que son degré est de 2, elle devient en **cendres** à l'itération suivante t + 1.

- Si une cellule [i, j] est en **cendres** à l'itération de simulation t, elle devient des **cendres éteintes** à l'itération suivante t + 1.
- Si une cellule [i, j] est un **arbre**, une **roche**, de **l'herbe** ou une **feuille** et qu'au moins l'un de ses 8 voisins (voir Figure 2) est en feu, alors la cellule [i, j] sera en feu à t + 1 et son degré diminue de 1.
- Si une cellule [i, j] est en feu et que son degré est supérieur à 2, son degré est diminué de 1.
- Si une cellule [i, j] contient de **l'eau** ou du **sol**, rien ne se produit.

2. Principales fonctionnalités attendues : ce que vous devez implémenter

Créez un programme qui commence par demander à l'utilisateur les dimensions souhaitées pour la forêt, la longueur et la largeur, puis initialise la forêt ; il est nécessaire de définir un type structuré adapté pour représenter une forêt.

- **Possède un menu avec les fonctionnalités suivantes :**
  - Remplissage aléatoire de la forêt
  - Remplissage manuel (L'utilisateur fournira pour chaque cellule de la forêt son type.)
- **simule un incendie :**
  - L'utilisateur doit fournir le nombre d'itérations de simulation.
  - Le programme lance la simulation de propagation du feu et affiche l'état de la forêt à chaque tour de simulation. L'utilisateur pourra déclencher le feu (case à choisir) dans la fonction "Simulation".
- **Revenir en arrière dans une simulation pour un nombre d'étapes choisi.**
- **interruptions et changements :**
  - L'utilisateur peut choisir une cellule, puis la modifier pour un autre type (changer le type de la case, l'état ou le degré).
  - Le programme peut reprendre la simulation. Cette fonctionnalité peut être lancée soit après que l'utilisateur ait effectué une modification sur une cellule, soit après une autre simulation.
- **Vérifier si un incendie peut se propager d'un point A à un point B** ? Si oui, pour le chemin (le plus court), calculer sa longueur et combien d'étapes de simulation sont nécessaires.
- **Quitter la simulation.**

3. Contraintes

Il est impératif de porter une attention particulière à la qualité du code, en respectant les éléments suivants :

1. Assurez-vous que le code soit modulaire, en utilisant différents fichiers .c et .h, et en ayant un maximum de fonctions pour une meilleure organisation.
2. Implémentez une allocation dynamique de mémoire pour une utilisation efficace des ressources.
3. Intégrez des fonctions récursives pour des approches appropriées et adaptées aux problèmes nécessitant une telle approche.

4. Consignes et critères d'évaluation

Vous devrez réaliser ce projet en trinômes (au maximum), sur une période d'environ deux mois. À la fin, vous présenterez votre travail à votre enseignante lors d'une défense, accompagnée d'un rapport de quelques pages.

Votre projet sera évalué en fonction de plusieurs critères détaillés ci-dessous (avec une hiérarchie approximative, les éléments en tête de liste étant plus importants que ceux en bas) :

- **Aspect fonctionnel :** Votre programme doit être opérationnel et permettre de jouer au jeu. Toutes les fonctionnalités anticipées décrites dans la section précédente doivent être disponibles. **C'est le point le plus crucial.**
- **Qualité technique de votre projet :** Cela concerne la clarté et la lisibilité de votre code source (soyez organisé et cohérent), ainsi que la qualité de vos algorithmes.
- **Votre performance lors de la défense** : Efforcez-vous d'être clair et captivant pour votre auditoire ! Préparez votre présentation et évitez de commencer directement par expliquer les détails du code.
- **L'intérêt, la pertinence et la qualité rédactionnelle de votre rapport** : Sa longueur doit être de quelques pages et il doit fournir un détail approfondi de votre approche et de votre réalisation. Particulièrement, **des commentaires sont attendus** concernant les fonctionnalités que vous avez implémentées, les algorithmes importants que vous avez conçus, les défis que vous avez rencontrés et les améliorations éventuelles.

- **L'interface utilisateur et l'expérience utilisateur de votre jeu** : Une interface de jeu agréable et ludique sera valorisée.
- **Les améliorations que vous avez apportées :** Plusieurs idées sont proposées dans la section suivante.

5. Pour aller plus loin… (BONUS)

Après avoir terminé les fonctionnalités attendues, vous avez la possibilité - mais ce n'est pas obligatoire - d'ajouter davantage de fonctionnalités et d'améliorations pour enrichir encore davantage votre projet. Si vous le faites, ces fonctionnalités seront considérées comme des bonus.

Voici quelques idées (Cette liste n'est ni exhaustive ni triée, n'hésitez pas à faire preuve d'imagination) :

- Ajouter un menu attrayant pour lancer une simulation et accéder à diverses fonctionnalités.
- Proposer une fonction pour calculer le nombre de simulations nécessaires pour brûler complètement une forêt à partir d'un point choisi (en excluant les zones d'eau et de sol).
- Permettre à l'utilisateur de sauvegarder une simulation pour y revenir ultérieurement.
- Offrir la possibilité à l'utilisateur de configurer la simulation d'incendie en ajoutant une variante de la simulation dans laquelle il est possible d'ajouter de nouveaux types de cellules ou de règles.
- Ajouter une interface utilisateur graphique (GUI) en plus de l'interface en ligne de commande (CLI).
- Pour les utilisateurs les plus avancés : Mettre en place une intelligence artificielle (IA) pour déterminer le moyen le plus efficace de provoquer un incendie en calculant le point de départ optimal.

Remarque : Les suggestions ci-dessus ne sont pas exhaustives, et vous êtes encouragés à proposer vos propres idées créatives également.