UNIVERSITAT POLITÈCNICA DE CATALUNYA
BARCELONATECH
Facultat d'Informàtica de Barcelona

FIB

Universitat Politècnica de Catalunya

Facultat d'Informàtica de Barcelona

Práctica 3 - Treballant amb ca i heurístiques

OPTIMITZACIÓ

Grau en Intel·ligència Artificial
Daniel Álvarez 23857151X
Albert Roca 48106974J

02/05/2025

Contents

1 1a Part

1.1 Introducció

En aquesta primera part de la pràctica, hem desenvolupat un model d'autòmat cel·lular unidimensional basat en les regles elementals definides per Wolfram, les quals descriuen com evoluciona l'estat d'una cèl·lula a partir del seu propi estat i dels dels seus dos veïns immediats. Aquestes regles permeten simular comportaments complexos a partir de normes senzilles aplicades iterativament.

Inicialment, hem implementat un autòmat que segueix una única regla de Wolfram i mostra l'evolució temporal del sistema en forma de gràfic. Posteriorment, el model s'ha generalitzat per permetre la combinació de múltiples regles dins d'un mateix procés evolutiu, construint així una estructura multicapa inspirada en les presentades a teoria. Aquesta versió multiregla permet observar com interactuen diferents comportaments locals quan es combinen de manera seqüencial, generant patrons més rics i diversos.

Aquesta extensió ha requerit dissenyar una arquitectura flexible que gestioni l'execució separada de cada regla, la divisió de l'evolució en segments (terços), i la construcció final d'una evolució global combinada. Finalment, s'han implementat eines gràfiques per visualitzar tant les evolucions individuals de cada regla com la seva combinació, facilitant l'anàlisi visual dels efectes d'aquesta fusió de dinàmiques.

1.2 Plantejament de la 1a Part

Tal i com hem definit a la introducció, l'objectiu d'aquesta primera part és implementar un autòmat cel·lular unidimensional. Aquest tipus d'autòmats es defineixen per una línia de cel·les binàries (actives o inactives) que evolucionen en el temps segons una regla local que depèn del seu propi estat i del dels seus veïns immediats. Cada regla es pot codificar com un nombre enter entre 0 i 255, que representa un conjunt de transicions binàries per a cada possible combinació del veïnat.

El plantejament es divideix en dues fases principals:

1. **Implementació d'un autòmat simple amb una sola regla:** inicialment, construim un sistema capaç d'aplicar una única regla de Wolfram sobre una línia de cel·les i representar-ne l'evolució temporal. Aquest pas serveix com a base per validar el funcionament bàsic de l'autòmat i entendre el comportament característic de cada regla.
2. **Generalització a un autòmat multiregla:** a partir del sistema simple, ampliem la funcionalitat per permetre l'aplicació combinada de diverses regles. En lloc d'aplicar una única regla durant tota l'evolució, es genera una seqüència de passos on s'utilitza un terç de l'evolució generada per cada regla. Això permet construir una evolució global on es combinen dinàmiques diferents, simulant així un entorn heterogeni.

Per tal de construir aquesta evolució combinada de forma estructurada, definim els passos següents:

- Per a cada regla, s'executa la seva evolució completa a partir del mateix estat inicial.
- Es divideix l'evolució de cada regla en tres parts iguals (terços).
- Se selecciona un terç concret de cada regla (per exemple, el primer de la primera regla, el segon de la segona, etc.).
- S'uneixen aquests fragments per formar una evolució global contínua.

Aquest plantejament permet no només observar el comportament individual de cada regla, sinó també analitzar com poden interactuar entre si quan s'apliquen en una mateixa simulació, proporcionant un punt de partida per a futurs models híbrids o adaptatius.

1.2.1 Diferents imatges mostrades

Durant el desenvolupament del nostre autòmat cel·lular multiregla, vam considerar essencial implementar una sèrie de visualitzacions progressives que ens permetessin verificar el funcionament correcte del model, tant a nivell individual com combinat. L'objectiu era assegurar-nos que els terços de cada regla utilitzats per formar la imatge final realment coincidissin amb els fragments generats per l'evolució completa de cada regla.

El Pau ens va proposar una manera de comprovar l'exit del nostre autòmat, vam adoptar una estratègia per comprovar de manera visual estructurada en tres nivells:

1. **Visualització completa de cada regla:** es mostra l'evolució temporal de cadascuna de les regles aplicades individualment, amb línies horitzontals que separen els tres terços (primer, segon i tercer). Posarem l'exemple de les regles: [94, 10, 30]

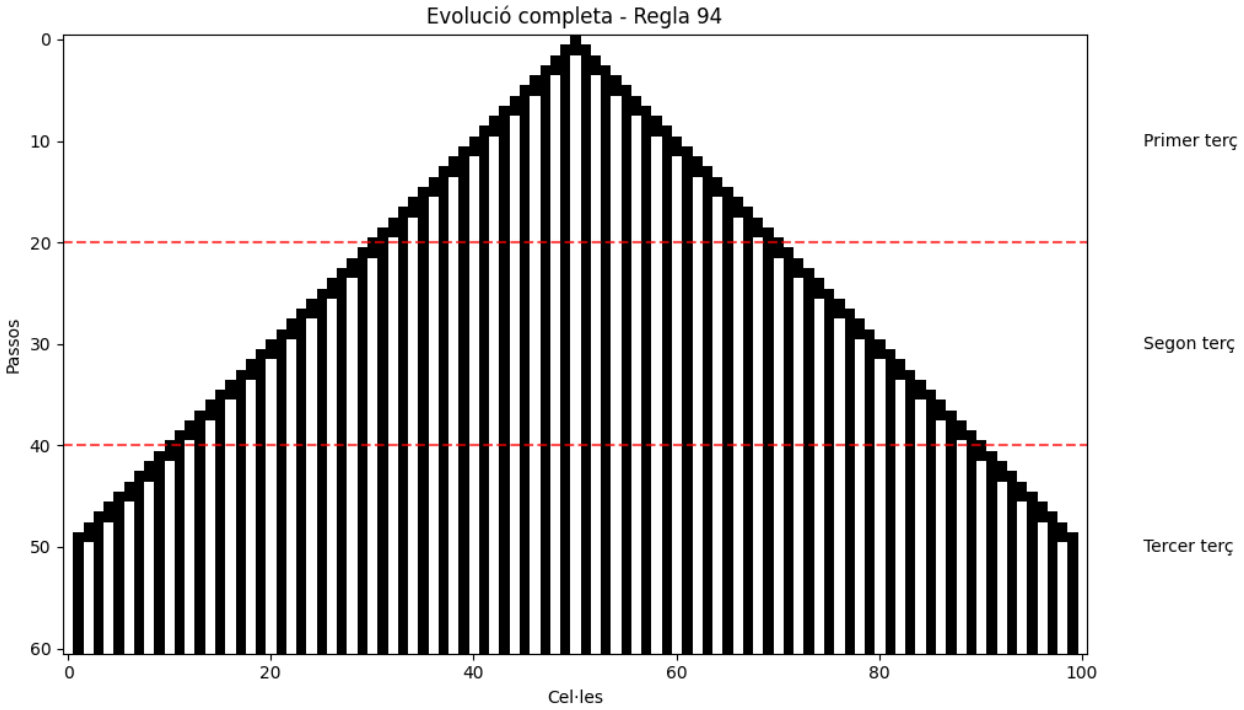

Figure 1: Visualització completa de la regla 94.

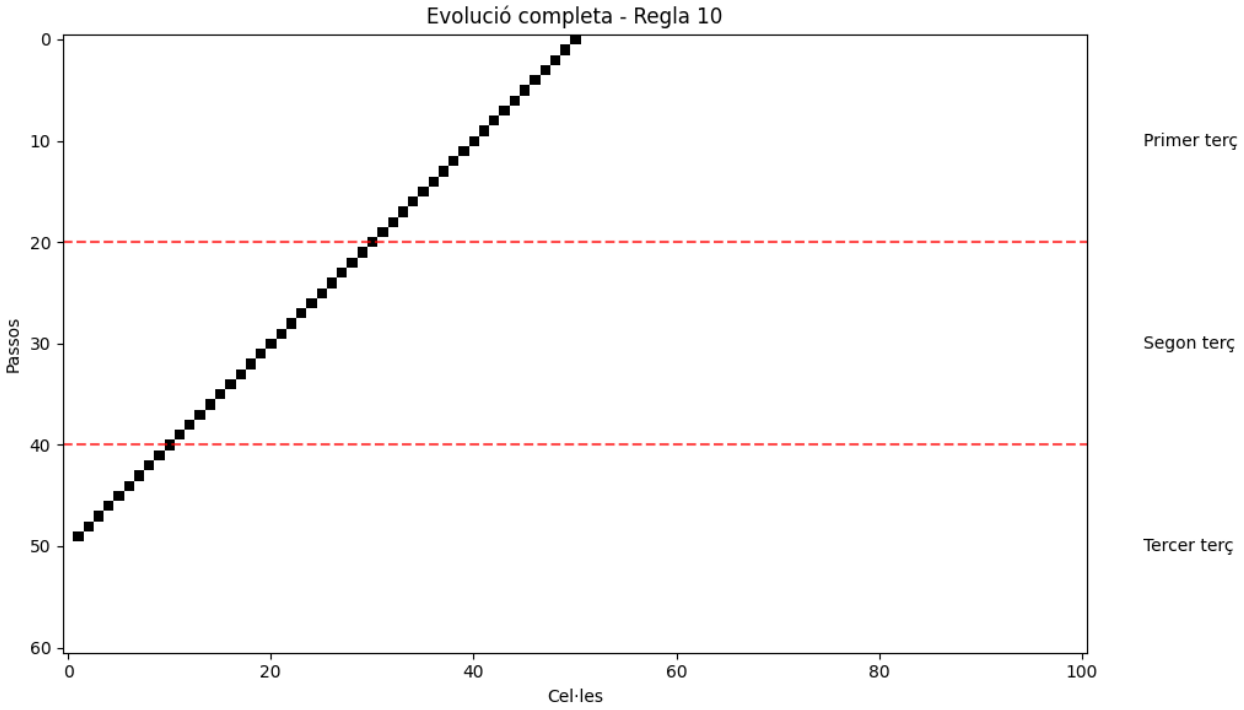

Figure 2: Visualització completa de la regla 10.

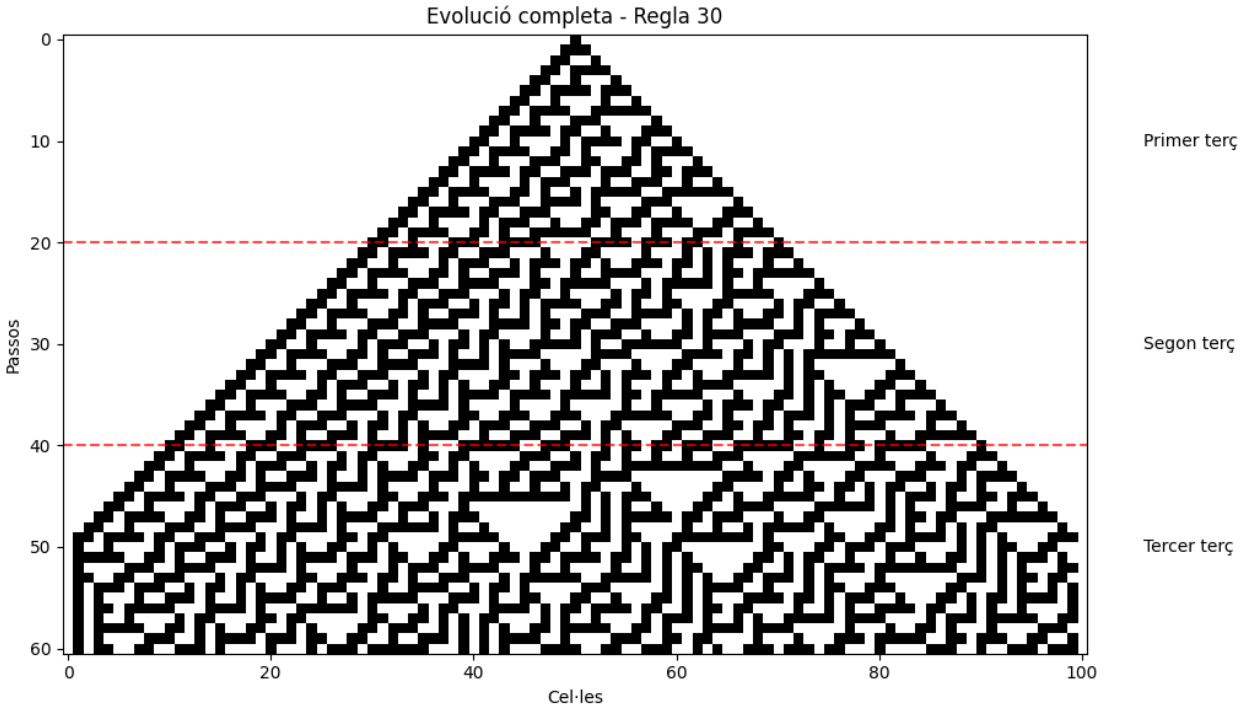

Figure 3: Visualització completa de la regla 30.

2. **Visualització del terç seleccionat:** per cada regla, es mostra només el terç que es farà servir a la imatge final combinada (seguint el patró: primer terç de la primera regla, segon de la segona, etc.). Aquesta imatge permet comparar fàcilment si el fragment s'ha extret correctament de la seva evolució completa.

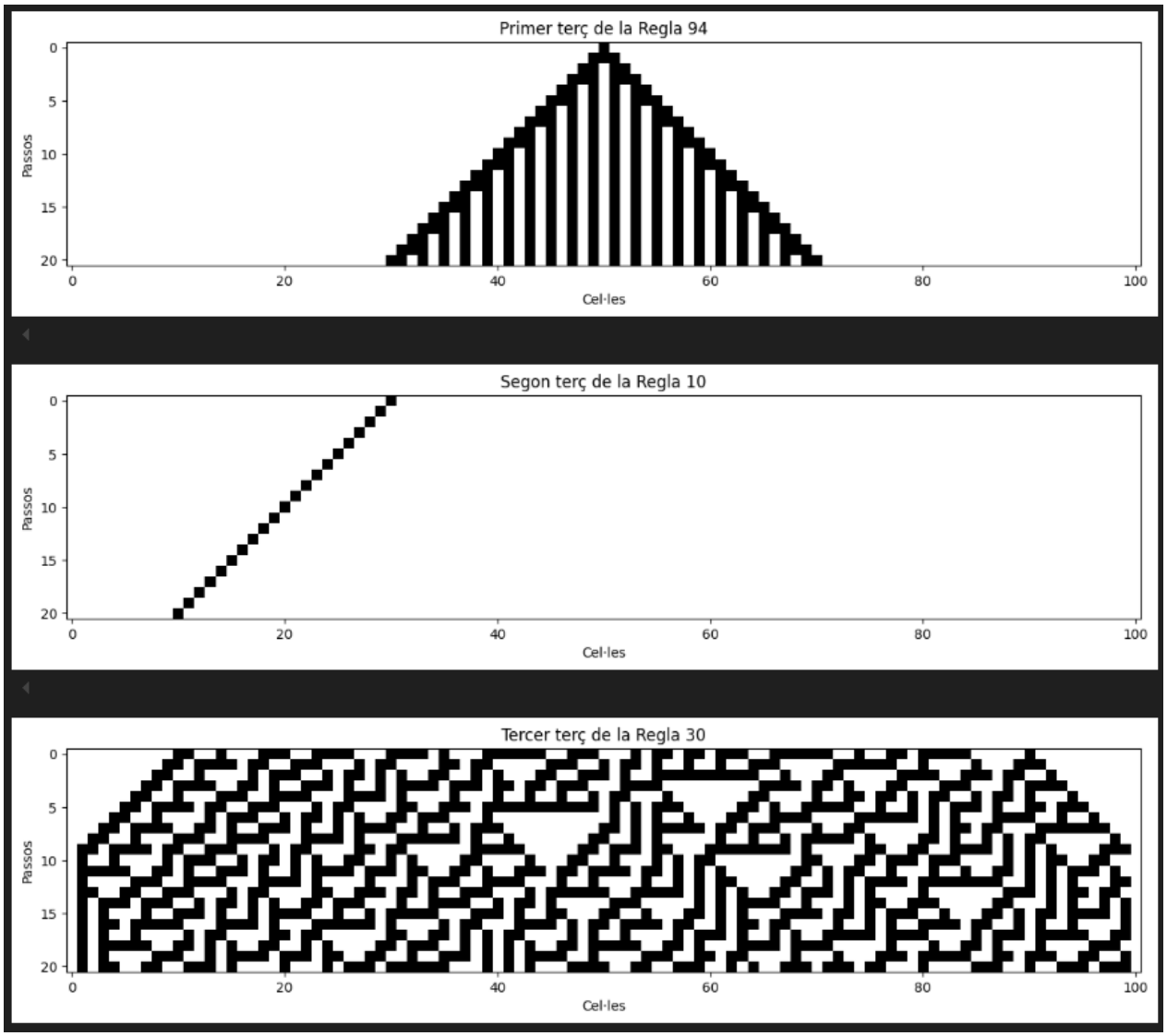

Figure 4: Visualització del terç seleccionat

3. **Visualització combinada:** s'afegeixen seqüencialment els terços seleccionats de cada regla per generar una única evolució que representa l'autòmat multiregla. Aquesta imatge inclou etiquetes que indiquen quin terç de quina regla està representat a cada secció.

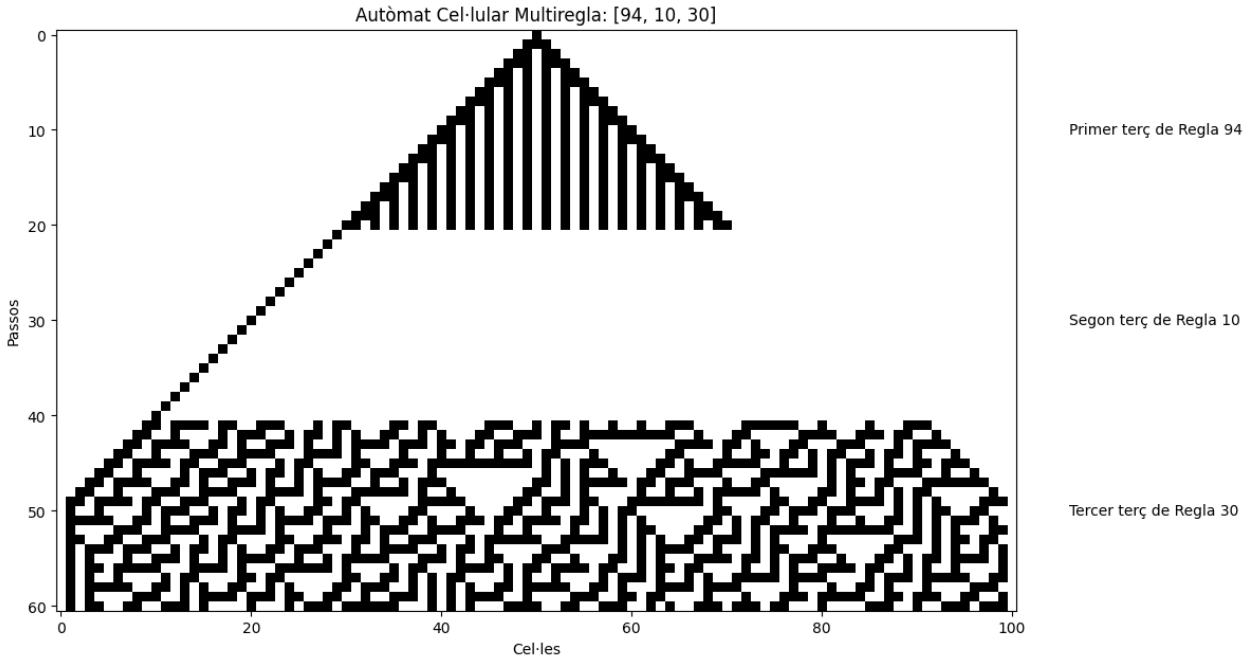

Figure 5: Visualització combinada

La seqüència de gràfiques facilita una validació clara i intuïtiva del model, tot garantint que la imatge final és una composició fidel dels fragments generats per cada regla.

El nostre autòmat és capaç de combinar de manera coherent tres regles de Wolfram i representar-ne la seva evolució conjunta en un sol gràfic. Aquest plantejament modular permetria en el futur generalitzar-lo a un nombre arbitrari de regles, tot i que l'actual implementació està optimitzada per al cas de tres.

1.3 Assumpcions del codi

Durant la implementació de l'autòmat cel·lular multiregla hem adoptat diverses assumpcions implícites que cal tenir en compte per entendre el funcionament correcte del model i la seva interpretació:

- **Nombre de regles fix:** El model està dissenyat per combinar exactament tres regles. Encara que tècnicament es podria modificar per admetre més, la lògica de combinació de terços es basa en el patró: primer terç de la primera regla, segon terç de la segona i tercer terç de la tercera.
- **Divisió en terços iguals:** L'evolució de cada regla es divideix en tres parts iguals. Es pressuposa que el nombre total de passos és divisible per tres o que les petites diferències no afecten significativament la qualitat de la simulació.
- **Estat inicial idèntic:** Totes les regles parteixen del mateix estat inicial, amb una única cel·la central activada. Això garanteix una base comuna per comparar els comportaments de cada regla de forma coherent.
- **Condicions de frontera fixes:** Les cèl·lules dels extrems no s'actualitzen durant l'evolució. Això implica una condició de frontera nul·la (zero fix), que evita que la informació "surti" dels límits de l'autòmat.

- **No hi ha transició suau entre terços:** Quan es construeix la imatge final combinada, cada terç s'uneix al següent sense cap procés d'ajustament. Es pressuposa que la transició entre fragments d'evolució generats per diferents regles no necessita suavització i es pot considerar vàlida directament.

1.4 Implementació del codi

```
import matplotlib.pyplot as plt
import numpy as np

class AutomatCellularMultiregla:
    def __init__(self, regles, longitud, passos):
        self.regles = regles
        self.longitud = longitud
        self.passos = passos

        # Calculamos cuántos pasos para cada regla (división equitativa)
        self.passos_per_regla = []
        passos_restants = passos
        for i in range(len(regles) - 1):
            passos_regla = passos // len(regles)
            self.passos_per_regla.append(passos_regla)
            passos_restants -= passos_regla
        self.passos_per_regla.append(passos_restants)  # El último recibe los pasos restantes

        # Estado inicial: solo la celda central activada
        self.estat_actual = np.zeros(longitud, dtype=int)
        self.estat_actual[longitud // 2] = 1

        # Para almacenar la evolución completa
        self.historial = []
        # Para almacenar la evolución por cada regla
        self.historials_per_regla = [[] for _ in regles]

        # Para almacenar los tercios de cada regla
        self.tercios_por_regla = []

    def aplica_regla(self, regla_bin):
        estat_nou = np.zeros_like(self.estat_actual)
        for i in range(1, self.longitud - 1):
            veinat = self.estat_actual[i - 1:i + 2]
            index = 7 - int(''.join(veinat.astype(str)), 2)
            estat_nou[i] = regla_bin[index]
        self.estat_actual = estat_nou
```

```
def executa(self):
    # Guardamos el estado inicial en el historial completo
    self.historial.append(self.estat_actual.copy())
    estat_inicial = self.estat_actual.copy()

    # Ejecutamos cada regla completa para obtener sus tercios
    for idx, regla in enumerate(self.regles):
        # Reiniciamos al estado inicial para cada regla
        self.estat_actual = estat_inicial.copy()

        # Convertimos la regla a binario (8 bits)
        regla_bin = np.array([int(bit) for bit in np.binary_repr(regla, width=8)])

        # Ejecutamos la regla para todos los pasos
        historial_regla = [self.estat_actual.copy()]
        for _ in range(self.passos):
            self.aplica_regla(regla_bin)
            historial_regla.append(self.estat_actual.copy())

        # Guardamos el historial completo de esta regla
        self.historials_per_regla[idx] = historial_regla

        # Dividimos el historial en tercios
        tercio_size = self.passos // 3
        tercios = [
            historial_regla[:tercio_size + 1],  # Primer tercio (incluye estado inicial)
            historial_regla[tercio_size:2*tercio_size + 1],  # Segundo tercio
            historial_regla[2*tercio_size:]  # Tercer tercio
        ]
        self.tercios_por_regla.append(tercios)

    # Ahora construimos el historial combinado usando diferentes tercios
    self.historial = [estat_inicial.copy()]  # Reiniciamos el historial

    for idx, regla in enumerate(self.regles):
        # Seleccionamos el tercio correspondiente según el índice de la regla
        tercio_a_usar = idx % 3  # 0: primer tercio, 1: segundo tercio, 2: tercer tercio

        # Añadimos los estados de este tercio (sin el estado inicial si no es el primero)
        if idx == 0:
            self.historial.extend(self.tercios_por_regla[idx][tercio_a_usar][1:])
        else:
```

```
            # Saltamos el primer estado ya que continúa del anterior
            self.historial.extend(self.tercios_por_regla[idx][tercio_a_usar][1:])

    def mostra_regla_tercio(self, idx_regla, idx_tercio):
        plt.figure(figsize=(12, 6))
        plt.imshow(self.tercios_por_regla[idx_regla][idx_tercio], cmap='binary')
        plt.xlabel('Cel·les')
        plt.ylabel('Passos')
        tercio_nombres = ["Primer", "Segon", "Tercer"]
        plt.title(f'{tercio_nombres[idx_tercio]} terç de la Regla {self.regles[idx_regla]}')
        plt.tight_layout()
        plt.show()

    def mostra_regla_completa(self, idx_regla):
        plt.figure(figsize=(12, 6))
        plt.imshow(self.historials_per_regla[idx_regla], cmap='binary')
        plt.xlabel('Cel·les')
        plt.ylabel('Passos')
        plt.title(f'Evolució completa - Regla {self.regles[idx_regla]}')

        # Añadimos líneas para separar visualmente los tercios
        tercio_size = self.passos // 3
        plt.axhline(y=tercio_size, color='red', linestyle='--', alpha=0.7)
        plt.axhline(y=2*tercio_size, color='red', linestyle='--', alpha=0.7)

        # Añadimos etiquetas para cada tercio
        plt.text(self.longitud + 5, tercio_size // 2, "Primer terç",
                 verticalalignment='center', color='black')
        plt.text(self.longitud + 5, tercio_size + tercio_size // 2, "Segon terç",
                 verticalalignment='center', color='black')
        plt.text(self.longitud + 5, 2*tercio_size + (self.passos - 2*tercio_size) // 2, "Tercer terç",
                 verticalalignment='center', color='black')

        plt.tight_layout()
        plt.show()

    def mostra_tot(self):
        plt.figure(figsize=(12, 8))
        plt.imshow(self.historial, cmap='binary')
        plt.xlabel('Cel·les')
        plt.ylabel('Passos')

        # Añadimos etiquetas para cada sección (sin líneas divisorias)
```

```
        tercio_size = len(self.historial) // len(self.regles)
        tercio_nombres = ["Primer", "Segon", "Tercer"]
        for i in range(len(self.regles)):
            pos_y = i*tercio_size + tercio_size // 2
            plt.text(self.longitud + 5, pos_y,
                     f"{tercio_nombres[i % 3]} terç de Regla {self.regles[i]}",
                     verticalalignment='center', color='black')

        plt.title(f'Autòmat Cel·lular Multiregla: {self.regles}')
        plt.tight_layout()
        plt.show()

    def mostra_tot_amb_separacio(self):
        # Primero mostramos cada regla completa con sus tercios marcados
        for i in range(len(self.regles)):
            self.mostra_regla_completa(i)

        # Luego mostramos los tercios específicos que estamos usando
        for i in range(len(self.regles)):
            tercio_a_usar = i % 3  # 0: primer tercio, 1: segundo tercio, 2: tercer tercio
            self.mostra_regla_tercio(i, tercio_a_usar)

        # Finalmente mostramos la evolución combinada (sin líneas divisorias)
        self.mostra_tot()

# Ejemplo de uso con tres reglas
aut_multiregla = AutomatCellularMultiregla(regles=[94, 10, 30], longitud=101, passos=60)
aut_multiregla.executa()
aut_multiregla.mostra_tot_amb_separacio()
```

1.5 Descripció del codi

Aquest codi implementa una classe en Python anomenada `AutomatCellularMultiregla`, que simula l'evolució d'un autòmat cel·lular unidimensional, aplicant diverses regles de forma combinada. A diferència d'un autòmat clàssic que utilitza una sola regla (com la 30 o la 110), aquest en combina diverses, i construeix una evolució formada per un terç de cada regla.

1.5.1 Objectiu General

El propòsit d'aquest autòmat multiregla és visualitzar com interaccionen les diferents regles cel·lulars quan s'apliquen de manera seqüencial. Per a cada regla:

- Es simula tota l'evolució des de l'estat inicial.
- Es divideix aquesta evolució en tres parts (terços).

- Es construeix una evolució final combinada agafant un terç de cada regla (el primer de la primera, el segon de la segona, etc.).

1.5.2 Constructor: `__init__`

Inicialitza l'objecte amb els paràmetres següents:

- `regles`: Llista de regles cel·lulars (ex. [94, 10, 30]).
- `longitud`: Nombre de cel·les (amplada de la línia).
- `passos`: Nombre total de passos d'evolució.

També divideix els passos equitativament entre les regles, inicialitza l'estat amb la cel·la central activada, i prepara estructures per guardar l'evolució.

1.5.3 `aplica_regla`

Aplica una regla binària (de 8 bits) a l'estat actual. Per cada cel·la (excloent les dues extrems), s'examina el seu veïnat (3 cel·les) i s'utilitza aquest patró per determinar el nou estat segons la regla.

1.5.4 `executa`

Executa el procés complet:

1. Guarda l'estat inicial.
2. Per cada regla:
    - Reestableix l'estat inicial.
    - Aplica la regla durant tots els passos.
    - Guarda l'evolució completa d'aquesta regla.
    - Divideix aquesta evolució en tres terços.
3. Construeix un historial final combinat amb un terç de cada regla:
    - Primer terç de la primera regla,
    - Segon terç de la segona,
    - Tercer terç de la tercera, etc.

1.5.5 `mostra_regla_tercio`

Mostra un terç concret de l'evolució d'una regla específica utilitzant una imatge amb tons binaris. Per exemple, el primer terç de la regla 94.

1.5.6 `mostra_regla_completa`

Mostra tota l'evolució d'una regla concreta, amb línies i etiquetes que separen els tres terços (visualment útil per analitzar patrons locals).

1.5.7 `mostra_tot`

Mostra l'evolució final combinada que s'ha construït seleccionant un terç de cada regla. Afegeix etiquetes a la dreta per indicar quin terç i quina regla s'estan mostrant a cada segment de l'evolució.

1.5.8 `mostra_tot_amb_separacio`

Funció que genera una visualització completa:

- Mostra l'evolució completa de cada regla (amb terços separats).
- Mostra només el terç utilitzat de cada regla.
- Mostra l'evolució final combinada.

1.5.9 Exemple d'ús

```
aut_multiregla = AutomatCellularMultiregla(regles=[94, 10, 30], longitud=101, passos=60)
aut_multiregla.executa()
aut_multiregla.mostra_tot_amb_separacio()
```

Aquest exemple crea un autòmat amb tres regles, una longitud de 101 cel·les, i 60 passos d'evolució. Després d'executar-lo, es mostra una visualització completa que permet entendre les dinàmiques de cada regla i el comportament emergent quan es combinen.

2 2a Part

2.1 Introducció

En aquesta pràctica hem desenvolupat un model de simulació de la propagació d'un incendi forestal en un entorn bidimensional, utilitzant com a base un autòmat cel·lular i dades reals emmagatzemades en format IDRISI32. L'objectiu principal ha estat analitzar com diferents factors ambientals afecten el comportament del foc, com ara la humitat del terreny, la densitat de vegetació, la presència d'obstacles naturals (com els llacs), el nombre de focus inicials de foc i la direcció del vent.

El model es basa en una funció d'evolució que gestiona l'estat de cada cel·la del mapa al llarg del temps, considerant les interaccions amb les cel·les veïnes i l'actualització progressiva de les capes de vegetació, humitat i estat del foc. Per a cada pas de la simulació, s'actualitza la propagació de l'incendi tenint en compte si les cel·les poden començar a cremar, continuar cremant o extingir-se, segons les condicions locals.

A més de la implementació del model, s'ha dut a terme un conjunt d'experiments per analitzar com varien la durada i l'extensió de l'incendi segons diferents paràmetres, com ara els intervals de vegetació i humitat, el nombre de focus inicials i la direcció del vent. Finalment, s'han generat gràfiques comparatives per facilitar la interpretació dels resultats obtinguts i extreure conclusions sobre la dinàmica del foc en entorns naturals.

2.2 Implementació/Descripció del codi

2.2.1 Estructura

Per tal de garantir un desenvolupament modular, entenedor i reutilitzable, el codi del projecte s'ha estructurat en diversos fitxers Python, cadascun encarregat d'una funcionalitat concreta dins la simulació. Aquesta divisió permet separar clarament les tasques de generació de dades, lectura i escriptura de fitxers, gestió de la simulació i visualització dels resultats.

En concret, el fitxer `generador_dades.py` s'encarrega de crear les capes de vegetació i humitat, mentre que `lectura_idrisi.py` gestiona el format IDRISI per a la lectura i l'escriptura dels mapes. El cor de la dinàmica del foc es troba en `simulacio.py`, que defineix la lògica de propagació i actualització de l'estat del foc pas a pas. Finalment, `visualitzacio.py` conté les funcions per representar gràficament les diferents capes, i `main.py` actua com a script principal per executar la simulació, definint els paràmetres inicials i controlant el bucle temporal.

2.2.2 Implementació general del model

La implementació del model de simulació es basa en un autòmat cel·lular, en què cada cel·la del mapa evoluciona al llarg del temps en funció del seu propi estat i dels estats de les cel·les veïnes. Per a cada iteració temporal (considerada com una hora simulada), el sistema actualitza tres capes principals: la **capa de vegetació**, que indica el nombre d'hores que pot cremar-se una cel·la; la **capa de humitat**, que representa el retard en l'inici del foc a causa de la presència d'humitat; i la **capa d'estat de l'incendi**, que defineix si una cel·la està pendent de cremar, cremant-se o ja ha estat cremada.

Cada cel·la pot trobar-se en un d'aquests estats: pendent (`0`), cremant-se (`1`) o cremada (`2`). El foc només es pot propagar a cel·les veïnes que tinguin humitat igual a zero i vegetació positiva. A més, si s'especifica una direcció del vent, la propagació es veu afavorida en aquella direcció concreta.

La simulació continua fins que ja no queden cel·les en estat `1` (cremant-se), moment en què es considera que l'incendi s'ha extingit completament. Aquesta arquitectura modular i basada en capes facilita tant la visualització del procés com l'anàlisi dels resultats en funció de diferents configuracions ambientals.

Així doncs, passem a descriure cada arxiu del simulador d'incendis.

2.2.3 main.py

Aquest fitxer actua com a programa principal del projecte i conté el punt d'entrada de la simulació. S'hi defineixen els paràmetres inicials de l'escenari, es genera el mapa ambiental (humitat i vegetació), s'inicialitzen els focus d'incendi i s'executa el bucle de simulació temporal fins que el foc s'extingeix completament.

Primerament, es fixa una llavor aleatòria (`SEED`) per garantir la reproduïbilitat dels resultats. A continuació, es defineixen els paràmetres del model: la mida del mapa (`dimensio_mapa`), el nombre de llacs, el nombre de focs inicials i la direcció del vent (si n'hi ha). Amb aquestes dades, es generen les capes de humitat i vegetació mitjançant les funcions del mòdul `generador_dades.py`, i es desen en format IDRISI amb `guardar_fitxer_idrisi()`.

Tot seguit, es genera la capa d'estat de l'incendi inicial (amb els focus actius) i es guarda també en format IDRISI. A partir d'aquí, s'inicia el bucle de simulació gràfica, que s'executa fins que totes les cel·les han deixat d'estar en flames. En cada iteració del bucle, es visualitza l'estat actual i s'actualitzen les tres capes principals (humitat, vegetació i incendi) mitjançant la funció `simular_incendi()` del mòdul `simulacio.py`.

Quan ja no hi ha cap cel·la cremant-se, es mostra per pantalla el nombre total d'hores simulades (equivalent al nombre de passos del bucle) i finalitza la visualització.

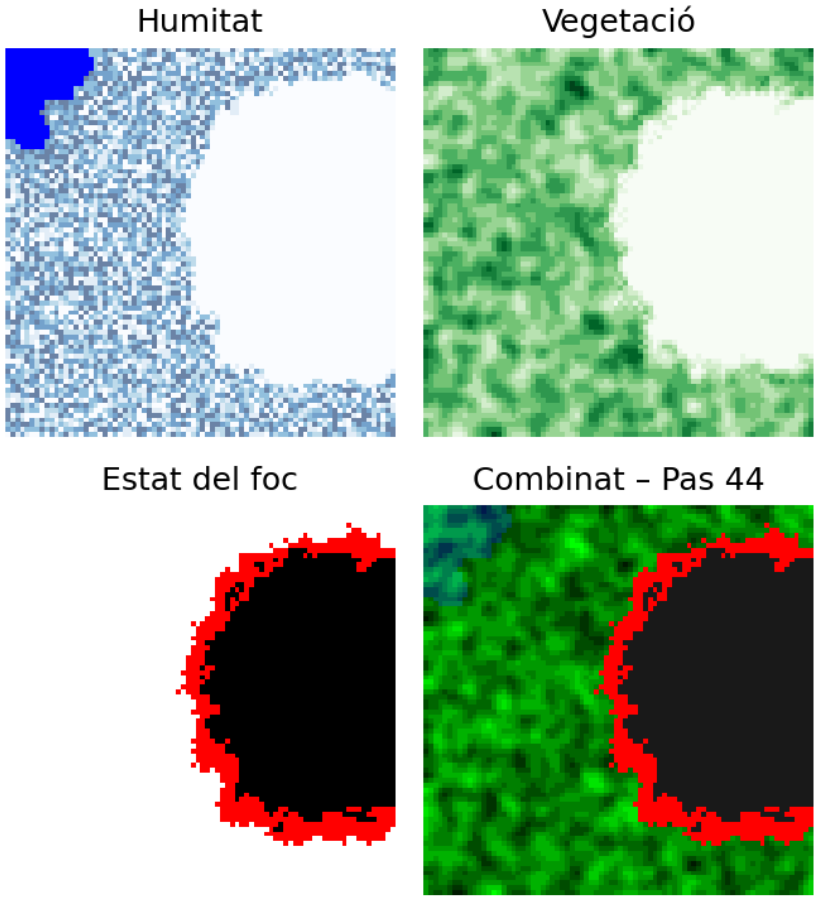

Figure 6: Execució de main.py amb les respectives capes

2.2.4 generador_dades.py

Aquest fitxer conté les funcions encarregades de generar les dues capes ambientals principals del sistema: la capa de **humitat** i la capa de **vegetació**. Aquestes capes es generen de manera aleatòria, però amb una estructura realista que simula característiques pròpies d'un entorn natural.

La funció `generar_humitat()` crea una matriu on cada cel·la conté un valor enter que representa el nombre d'hores que una cel·la ha d'esperar abans de poder començar a cremar-se, si una veïna està en flames. A més, s'hi pot generar un nombre determinat de llacs, assignant humitat infinita (`np.inf`) a les cel·les corre-

sponents, que es consideren incombustibles. Les cel·les properes a un llac tenen més probabilitats de tenir una humitat elevada, mentre que la resta del mapa té valors aleatoris dins d'un interval configurable (per exemple, entre 0 i 5, o entre 0 i 20 segons l'experiment).

La funció `generar_vegetacio()` crea la capa de vegetació, que indica el nombre d'hores que triga una cel·la a cremar-se completament un cop ha començat el foc. Per tal de simular una distribució més natural, es genera una matriu aleatòria i s'hi aplica un filtre gaussià per suavitzar les diferències brusques. A continuació, els valors s'escales i s'arrodoneixen per obtenir enters dins un rang desitjat (per exemple, entre 1 i 10, o entre 5 i 20 en altres experiments).

2.2.5 lectura_idrisi.py

El fitxer `lectura_idrisi.py` conté les funcions responsables de la lectura i escriptura de fitxers en format IDRISI32, que permeten emmagatzemar i recuperar les diferents capes de dades utilitzades en la simulació. Aquest format es compon de dos fitxers per capa: un fitxer `.img` que conté els valors numèrics (una línia per cel·la) i un fitxer `.doc` que conté les metadades com el nombre de files i columnes, el títol, la resolució, etc.

La funció `guardar_fitxer_idrisi()` permet escriure una matriu en aquests dos formats, creant automàticament la carpeta de destinació si no existeix. Aquesta funció és útil tant per guardar les capes generades com per exportar l'estat de l'incendi en diferents moments de la simulació.

D'altra banda, la funció `llegir_fitxer_idrisi()` permet carregar una capa a partir dels fitxers `.doc` i `.img`, reconstruint la matriu original amb la seva mida corresponent. Finalment, `llegir_fitxers_idrisi()` facilita la lectura conjunta de dues capes (per exemple, humitat i vegetació), retornant-les com a tupla per a una integració fàcil amb la resta del sistema.

2.2.6 simulacio.py

Aquest mòdul conté la lògica principal de la propagació del foc i la seva evolució al llarg del temps. Està format per dues funcions: `inicialitzar_incendi()` i `simular_incendi()`.

La funció `inicialitzar_incendi()` s'encarrega de generar una matriu amb l'estat inicial de l'incendi. Aquesta matriu té el mateix tamany que el mapa i assigna un valor d'estat a cada cel·la. Les cel·les comencen per defecte en estat `0` (pendent de cremar), i se seleccionen aleatòriament un nombre determinat de cel·les (determinat pel paràmetre `num_focs`) per encendre-les inicialment (estat `1`). Aquesta selecció s'assegura que cap d'aquestes cel·les inicials sigui un llac (és a dir, amb humitat infinita).

La funció `simular_incendi()` actualitza l'estat de totes les cel·les del mapa en un pas temporal de la simulació. En cada iteració, si una cel·la està cremant-se (`1`), es redueix la seva vegetació en una unitat. Quan la vegetació arriba a zero, la cel·la es marca com a cremada (`2`). A més, la funció redueix la humitat de les cel·les veïnes i, si la humitat d'una cel·la veïna arriba a zero i té una cel·la veïna que crema, aquesta cel·la comença a cremar-se.

La propagació també pot veure's influïda per la direcció del vent, que afavoreix la transmissió del foc en una direcció concreta (nord, sud, est o oest). Aquesta propagació addicional s'aplica a una cel·la situada

dues unitats en la direcció del vent, si compleix les condicions per començar a cremar-se.

La funció retorna l'estat actualitzat de les tres capes (humitat, vegetació i incendi), així com un indicador booleà que determina si encara hi ha foc actiu o no.

2.2.7 visualitzacio.py

El fitxer `visualitzacio.py` conté la funció `visualitzar_estat()`, encarregada de mostrar gràficament l'evolució de la simulació pas a pas. Aquesta funció utilitza la llibreria `matplotlib` per representar, en una figura subdividida en quatre subgràfics, les diferents capes que intervenen en la dinàmica de l'incendi.

Els quatre subgràfics generats són els següents:

- **Humitat:** mostra el mapa de humitat en tons blaus, destacant amb color blau fosc les zones amb humitat infinita (llacs).
- **Vegetació:** representa la quantitat de vegetació restant a cada cel·la amb una escala de verds.
- **Estat de l'incendi:** visualitza l'estat de cada cel·la segons el seu valor: blanc per a cel·les pendents de cremar, vermell per a cel·les cremant-se, i negre per a cel·les ja cremades.
- **Vista combinada:** integra totes les capes anteriors en una sola imatge per mostrar de manera simultània la vegetació, la humitat i la propagació del foc. El fons verd indica la vegetació, el blau suau representa la humitat, i el vermell i negre es fan servir per superposar el foc actiu i les zones cremades respectivament.

Aquesta visualització dinàmica permet seguir la simulació de manera intuïtiva i detectar fàcilment com es propaga l'incendi en funció de les condicions ambientals. Cada pas es mostra amb una petita pausa temporal per crear l'efecte d'animació.

2.3 Diagrames

2.3.1 Diagrama d'estats

El diagrama d'estats següent descriu de manera simplificada el comportament dinàmic d'una cel·la dins del nostre model de simulació. Hem considerat tres estats possibles: *Not Burning*, *Burning* i *Burned*, així com les condicions que provoquen les transicions entre ells.

Inicialment, una cel·la es troba en estat *Not Burning*. Aquest estat pot canviar a *Burning* si alguna cel·la veïna està cremant-se i la humitat de la cel·la actual és zero. Aquesta condició reflecteix el mecanisme de propagació del foc quan el terreny és prou sec per encendre's.

Un cop la cel·la entra en estat *Burning*, es mantenen diverses accions internes. Per una banda, el sistema *actualitza* els valors ambientals de la cel·la (com la vegetació i la humitat). Per altra banda, si es compleixen les condicions adequades, el foc pot *propagar-se* a cel·les veïnes. Aquestes accions internes no canvien l'estat de la cel·la, però afecten el seu entorn i determinen l'evolució de la simulació.

Finalment, quan la vegetació de la cel·la arriba a zero, la cel·la canvia a l'estat *Burned*. En aquest estat, la cel·la ja no pot propagar més el foc ni tornar-se a encendre, i per tant representa un punt inert dins la graella.

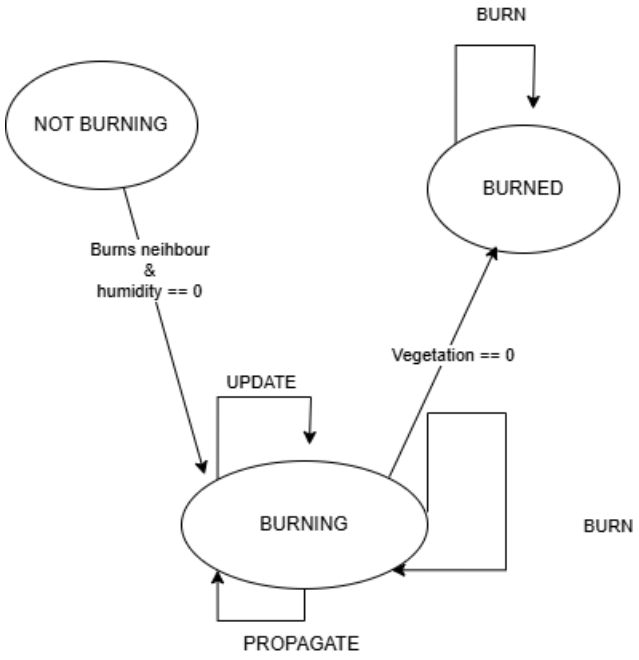

Figure 7: Diagrama d'estats

2.3.2 Diagrama de procés

Per tal de representar visualment la lògica que segueix el nostre model de propagació d'incendis, s'ha elaborat un diagrama de flux que descriu pas a pas l'evolució de l'estat de cada cel·la del mapa. Aquest diagrama mostra de manera clara les diferents condicions que es tenen en compte en cada iteració, com ara la presència d'humitat infinita (que identifica llacs), la quantitat de vegetació disponible, la humitat restant, la presència de foc en cel·les adjacents, i la influència del vent. Cada cel·la és avaluada individualment segons aquests criteris per decidir si comença a cremar, si continua cremant-se o si ja ha estat completament cremada. Aquesta estructura ens ha permès implementar una simulació modular i comprensible, facilitant tant el desenvolupament del codi com l'anàlisi del comportament del sistema.

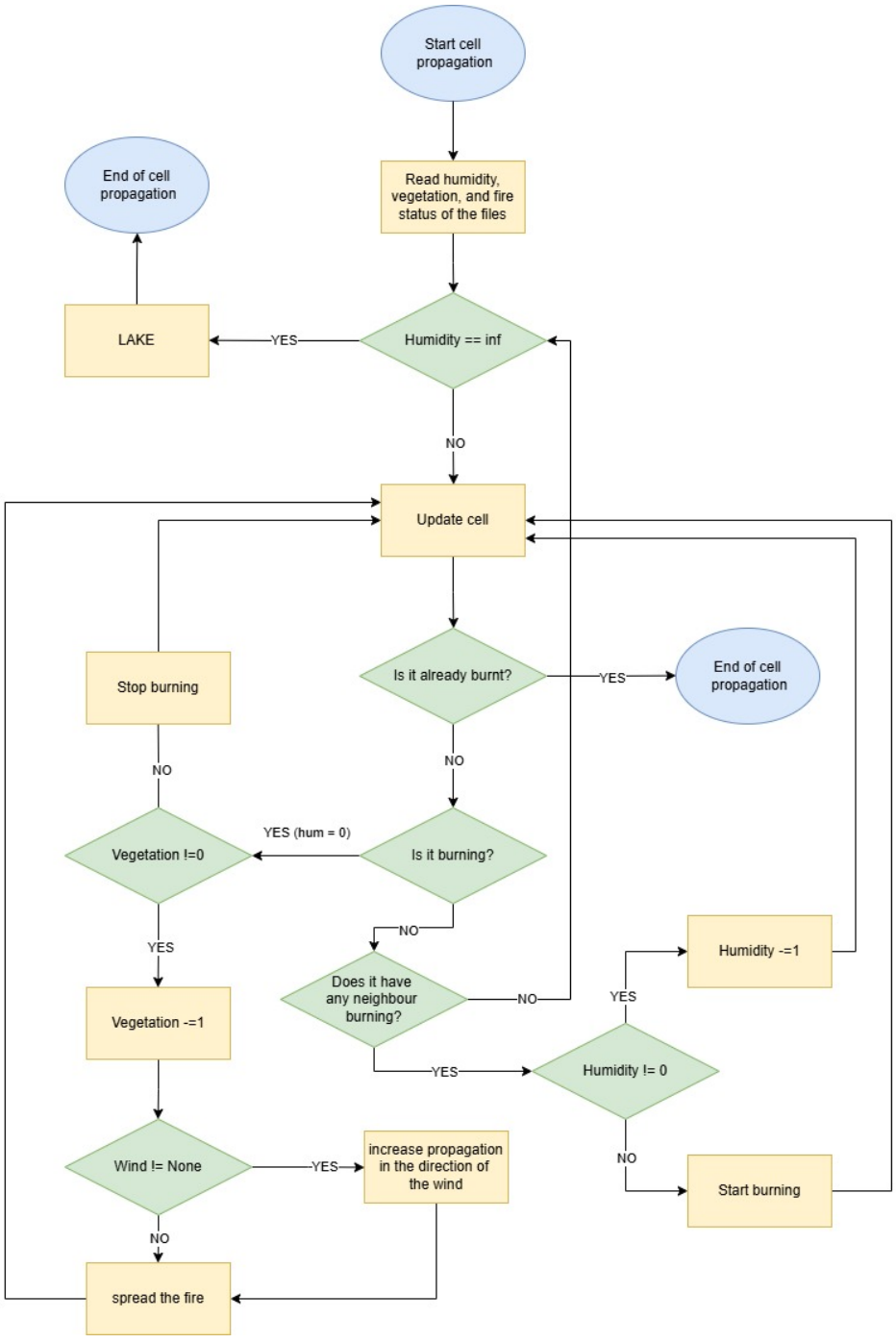

Figure 8: Diagrama del Procés

En primer lloc, es llegeixen els valors d'humitat, vegetació i estat del foc a partir dels fitxers d'entrada corresponents. Aquestes dades representen l'estat inicial de cada cel·la. Tot seguit, es comprova si la cel·la té una humitat infinita. Aquesta condició implica que la cel·la forma part d'un llac i, per tant, és incombustible. En aquest cas, la propagació del foc no hi tindrà lloc.

Si la cel·la és susceptible de cremar-se, s'actualitza el seu estat i es comprova si ja està totalment cremada.

En aquest cas, ja no pot continuar propagant el foc, i es marca com a finalitzada. Si encara no s'ha cremat, s'analitza si s'està cremant en el moment actual. Si és així, això significa que la seva humitat és zero, ja que aquesta és una condició necessària per a la ignició segons el nostre model.

Un cop iniciada la combustió, es comprova si la cel·la encara té vegetació. Si en té, es redueix una unitat el nivell de vegetació, representant una hora de combustió. A continuació, si s'ha definit una direcció de vent, aquest factor afavoreix la propagació cap a la direcció especificada, accelerant el procés. En cas contrari, el foc es propaga de manera uniforme cap a les cel·les veïnes, incloent les diagonals. Si la cel·la s'ha quedat sense vegetació, deixa de cremar i s'actualitza el seu estat com a cel·la cremada.

En cas que la cel·la no estigui cremant, es comprova si alguna de les cel·les veïnes està en flames. Si es troba una cel·la veïna encesa, la propagació pot començar. En aquest punt, es verifica si la cel·la té humitat. Si en té, es redueix una unitat; si no en té, l'estat de la cel·la es modifica per passar a cremant, i l'incendi comença a expandir-se des d'aquesta nova font.

Aquest procés es repeteix per a totes les cel·les i per a cada pas temporal fins que l'incendi s'ha extingit completament, proporcionant una simulació detallada i dinàmica del comportament del foc en funció de les condicions ambientals.

2.4 Experimentació

Aquesta secció del treball es dedica a l'anàlisi experimental del model de l'incendi implementat. L'objectiu principal és estudiar com diferents paràmetres afecten la durada total de l'incendi, entesa com el nombre total d'iteracions (hores simulades) que han de passar fins que totes les cel·les cremables han estat completament consumides pel foc. Mitjançant aquest conjunt d'experiments, es pretén comprendre millor la influència de diversos factors ambientals i estructurals del terreny, com ara la humitat, la densitat de vegetació, el nombre de nuclis de foc inicials, la presència d'obstacles naturals (llacs), o la direcció del vent.

Per tal de garantir que les conclusions extretes siguin consistents i comparables entre si, s'ha optat per fixar una sèrie de paràmetres globals que es mantindran constants en tots els experiments, excepte en aquells casos en què es vulgui estudiar específicament l'efecte d'un paràmetre concret. Aquesta metodologia permet aïllar l'impacte de cada variable i observar-ne el comportament de manera controlada.

Els valors establerts com a configuració base per a totes les simulacions són els següents:

- `dimensió = 80`: mida del mapa bidimensional (80×80).
- `numero_llacs = 0`: no hi ha presència de llacs (cap cel·la amb humitat infinita).
- `direccio_vent = None`: no s'aplica cap efecte de vent en la propagació.
- `numero_focs = 1`: només s'encén una cel·la inicial.
- Rang de valors de la humitat: `0--5`, amb probabilitat de major humitat en zones properes a llacs (si n'hi hagués).
- Rang de valors de la vegetació: `1--10`, indicant les hores que pot cremar una cel·la un cop encesa.

Tenint en compte que el model incorpora múltiples elements aleatoris (com la distribució espacial de la vegetació i la humitat, o la posició inicial del foc), és essencial realitzar diverses repeticions de cada experiment per tal d'obtenir resultats representatius i robustos. Això permet compensar la variabilitat i evitar conclusions basades en casos puntuals atípics.

Amb aquesta finalitat, cada experiment es repeteix set vegades, utilitzant una llavor diferent en cada iteració. Aquestes llavors permeten fixar l'estat inicial del generador aleatori, assegurant la reproductibilitat dels resultats i facilitant tant la verificació com la comparació amb altres configuracions.

Les llavors utilitzades són les següents:

- Iteració 1: `1111`
- Iteració 2: `2222`
- Iteració 3: `3333`
- Iteració 4: `4444`
- Iteració 5: `5555`
- Iteració 6: `6666`
- Iteració 7: `7777`

Aquesta estructura experimental ens permet construir mitjanes i valors comparatius sòlids que seran analitzats en les seccions següents, amb l'objectiu d'identificar tendències i comportaments rellevants dins del model.

2.4.1 Experiment 1: Com afecta el nombre de nuclis d'incendi a la propagació de l'incendi

En aquest primer experiment, es pretén analitzar l'efecte directe que té el nombre de focus d'incendi inicials sobre la durada total de la propagació del foc. En el model implementat, el foc comença amb un cert nombre de cel·les seleccionades aleatòriament com a nuclis inicials, que passen automàticament a l'estat `1` (cremant-se).

Per dur a terme l'experiment, s'han executat simulacions amb un nombre de focus inicials que va de `1` a `10`, mantenint constant la resta de condicions.

A continuació es presenten les dades obtingudes per a cada cas, juntament amb una gràfica que mostra la relació inversa entre el nombre de focs i la durada mitjana de l'incendi.

Table 1: Durada (en hores simulades) segons el nombre de nuclis de foc inicials.

Iteració	**1 foc**	**2 focs**	**3 focs**	**4 focs**	**5 focs**	**6 focs**	**7 focs**	**8 focs**	**9 focs**	**10 focs**
1	98	85	78	78	77	77	77	77	56	56
2	90	90	90	90	68	68	68	68	68	68
3	106	84	84	84	84	84	84	77	77	77
4	109	104	80	80	73	73	73	73	73	73
5	90	83	83	83	83	83	83	83	83	83
6	116	116	116	115	78	66	66	66	66	54
7	101	101	88	81	81	74	66	63	61	61

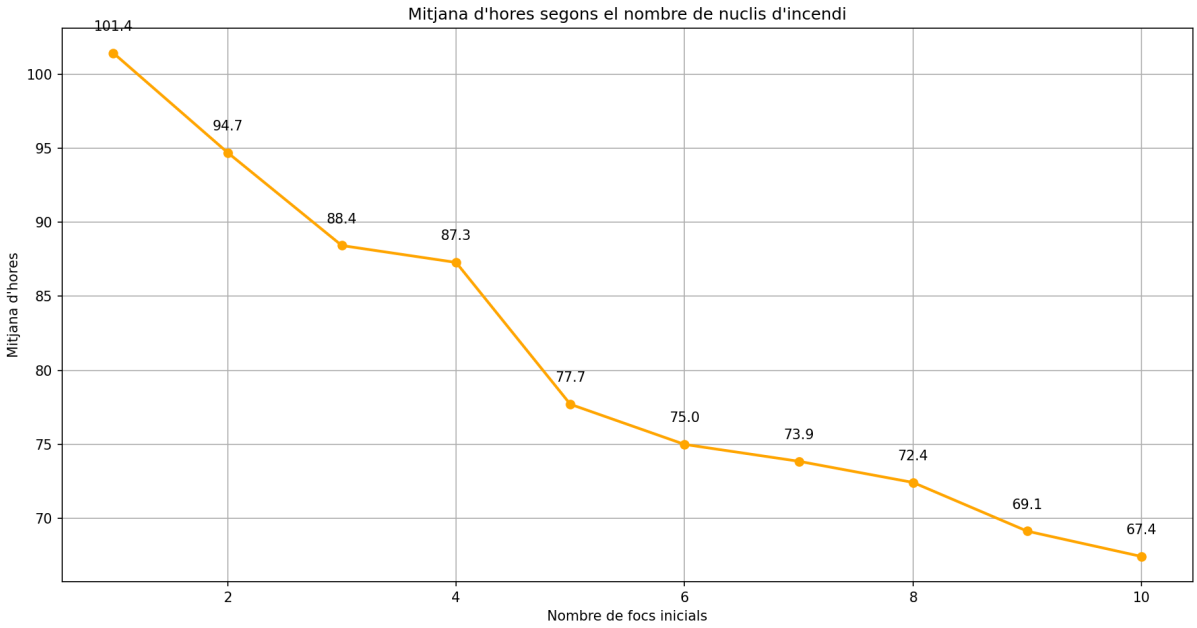

Figure 9: Mitjana d'hores que triga el foc a extingir-se completament en funció del nombre de nuclis d'incendi inicials

Els resultats obtinguts mostren una tendència clara i consistent: a mesura que augmenta el nombre de nuclis d'incendi inicials, el nombre d'hores que triga l'incendi a extingir-se disminueix de forma significativa. La mitjana d'hores per a un únic focus és de 101,4, mentre que amb deu focus la mitjana baixa fins a 67,4 hores. Aquesta reducció no és lineal, però sí notable i progressiva, especialment en els primers increments de nombre de focs, on l'efecte acumulatiu de múltiples fronts actius comença a ser molt evident.

Aquest comportament pot explicar-se fàcilment si s'analitza la naturalesa del model. Quan només hi ha un focus, el foc es propaga radialment a partir d'un únic punt, i necessita moltes iteracions per arribar als extrems de la graella. En canvi, quan s'activen diversos nuclis al mateix temps, s'inicien múltiples cercles de propagació simultània. Això redueix significativament les distàncies que ha de recórrer el foc per abastar tot el territori i accelera la velocitat global de consum de vegetació.

És important destacar que, tot i la variabilitat inherent al model (deguda a la disposició aleatòria dels nuclis, la vegetació i la humitat), la tendència és consistent en totes les repeticions. La disminució en la durada total de l'incendi és especialment marcada fins a arribar als 5 o 6 nuclis, moment en què la millora en el temps comença a presentar una corba d'esmorteïment. Això pot indicar que, a partir d'un cert punt, afegir més focs inicials té un efecte marginal, ja que moltes zones del mapa ja estan cobertes per algun nucli proper.

Aquest experiment demostra que el nombre de punts d'ignició és un factor determinant per a la dinàmica temporal d'un incendi.

2.4.2 Experiment 2: Com afecta el nombre de llacs a la propagació de l'incendi

En aquest segon experiment, l'objectiu és analitzar com la presència d'obstacles naturals, concretament llacs, influeix en la durada total de l'incendi. En el nostre model, els llacs es representen com a cel·les amb humitat infinita, la qual cosa les fa completament incombustibles. Aquestes cel·les actuen com a barreres naturals dins del mapa, impedint que el foc es propagui a través d'elles o per zones immediatament properes, ja que també afecten indirectament la humitat de les cel·les veïnes.

Per estudiar aquest efecte, es realitzen simulacions amb diferents quantitats de llacs, variades de manera incremental. El nombre de llacs considerat en aquest experiment va de 0 a 5, afegint un llac addicional a cada iteració.

Tal com s'ha fet en l'experiment anterior, cada configuració s'ha repetit set vegades utilitzant les mateixes llavors per garantir la comparabilitat entre proves. Aquest disseny experimental permet observar l'impacte mitjà del nombre de llacs sobre el temps necessari per extingir el foc.

L'hipòtesi inicial és que un nombre més elevat de llacs actuarà com a factor limitador de la propagació, obligant el foc a envoltar les zones incombustibles i alentint el procés global. A continuació, es mostren les dades i gràfiques corresponents per analitzar aquest efecte.

Table 2: Durada (en hores simulades) segons el nombre de llacs presents al mapa.

Iteració	**1 llac**	**2 llacs**	**3 llacs**	**4 llacs**	**5 llacs**	**6 llacs**
1	121	130	131	121	141	118
2	146	119	119	178	128	125
3	150	149	117	151	151	139
4	144	182	187	190	142	224
5	143	180	175	175	163	161
6	125	156	153	127	294	200
7	132	130	174	194	141	146

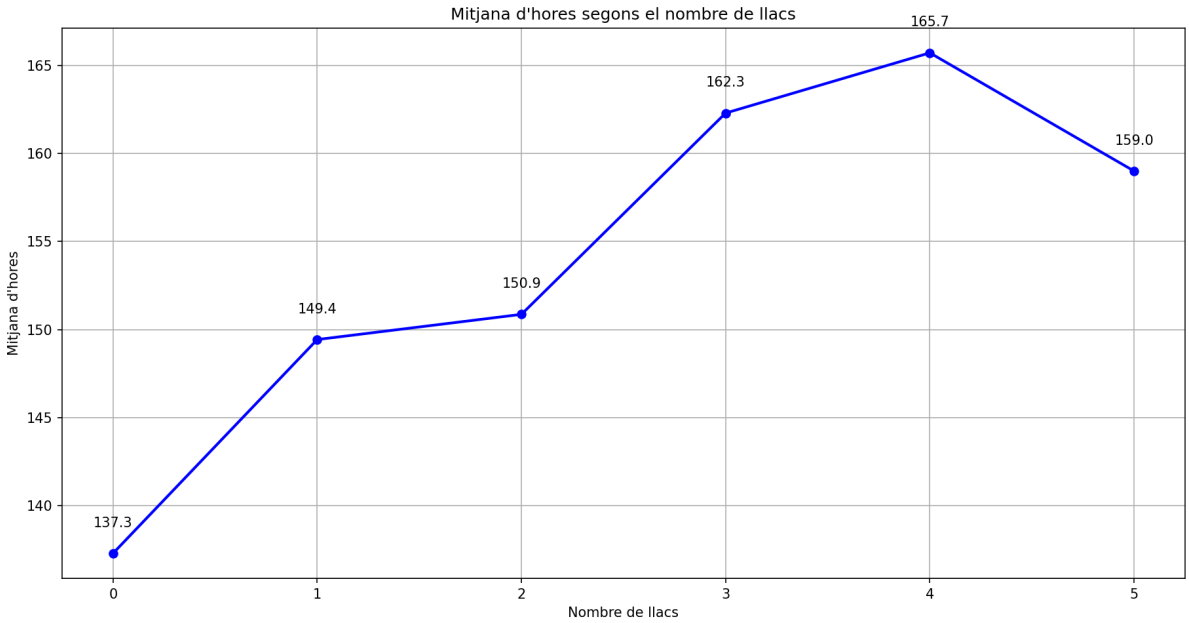

Figure 10: Mitjana d'hores d'incendi a mesura que incrementen els llacs

Els resultats obtinguts en aquest experiment mostren una tendència general segons la qual l'augment del nombre de llacs tendeix a incrementar la durada total de l'incendi. Partint d'un escenari sense llacs, on el foc triga una mitjana de 137,3 hores a consumir tota la vegetació disponible, s'observa un augment progressiu en el nombre d'hores a mesura que es van afegint llacs. Amb cinc llacs, la durada mitjana arriba a 165,7 hores, i fins i tot amb sis llacs, tot i que el temps baixa lleugerament (159,0 hores), es manté molt per sobre dels valors inicials.

Aquesta evolució és coherent amb el comportament del model. Els llacs es consideren zones d'humitat infinita, per tant, són completament incombustibles i actuen com a barreres que interrompen la propagació natural del foc. Quan el foc arriba a una regió propera a un llac, no pot continuar la propagació en línia recta i ha de rodejar l'obstacle. A més, la presència de llacs també incrementa la probabilitat que les cel·les properes tinguin un valor d'humitat elevat, la qual cosa introdueix un retard addicional abans que aquestes cel·les puguin començar a cremar-se.

En conclusió, aquest experiment confirma que la presència de llacs actua com a element retardador en la propagació del foc. Tot i que la variabilitat introduïda per la posició aleatòria dels obstacles pot alterar lleugerament els resultats, la tendència general mostra una correlació positiva entre el nombre de llacs i la durada de l'incendi.

2.4.3 Experiment 3: Efecte de l'increment de l'interval de vegetació sobre la durada de l'incendi

En aquest experiment s'estudia com afecta l'increment dels valors possibles de vegetació en el temps que triga un incendi a consumir completament la graella. En el model proposat, la vegetació representa la quantitat d'hores que una cel·la pot cremar-se abans de quedar totalment consumida. Així doncs, un valor més alt de

vegetació implica que una cel·la tarda més a ser destruïda, retardant també la seva capacitat de propagar el foc a les cel·les veïnes.

L'interval de vegetació és un paràmetre clau perquè actua com a una forma de "combustible" dins la simulació. A diferència d'altres factors com la humitat o la direcció del vent, que modulen la propagació lateral, la vegetació incideix directament en el temps que una cel·la roman activa dins del procés de cremada. Per aquest motiu, és d'especial interès entendre com aquest interval pot condicionar la durada global de l'incendi.

Per dur a terme aquest estudi, s'han definit cinc intervals diferents de valors possibles de vegetació: `0--10`, `0--20`, `5--10`, `10--20` i `15--20`. A cada cas, la vegetació de la graella s'ha generat respectant exclusivament aquest rang.

Com en els experiments anteriors, cada configuració s'ha repetit set vegades amb les mateixes llavors aleatòries. Això permet calcular una mitjana representativa del comportament del model per a cada cas i detectar com canvia la durada de l'incendi a mesura que el mapa es fa més resistent a la crema a causa d'una vegetació més densa o més lenta de consumir.

Table 3: Durada (en hores simulades) segons l'interval de valors de la vegetació.

Iteració	**0–10**	**0–20**	**5–10**	**10–20**	**15–20**
1	98	104	102	113	118
2	90	98	94	107	112
3	106	112	110	121	126
4	109	118	113	127	132
5	90	97	94	106	111
6	116	122	120	131	136
7	101	106	105	115	120

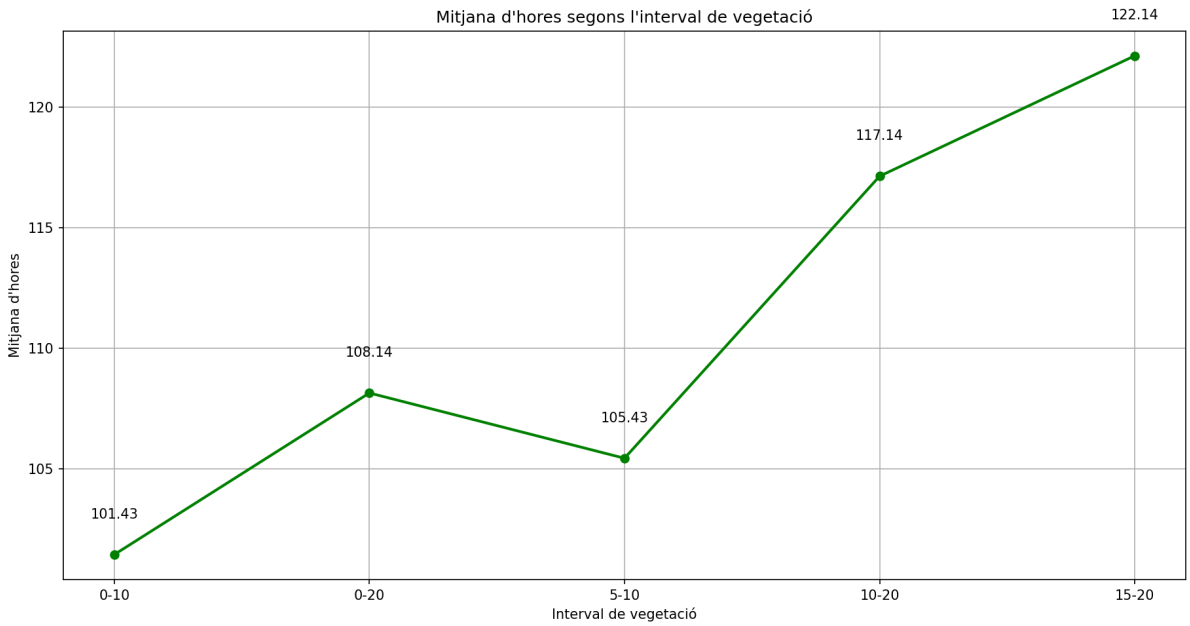

Figure 11: Mitjana de la durada (en hores simulades) segons l'interval de vegetació

Els resultats obtinguts mostren una correlació clara entre l'interval de valors assignat a la vegetació i la durada total de l'incendi. Com s'esperava, quan s'incrementa la quantitat de vegetació disponible (entenent-la com a nombre d'unitats d'hora que una cel·la pot cremar), el foc triga més temps a consumir la totalitat del mapa.

L'escenari amb valors entre `0--10` és el més lleuger, amb una mitjana de 101,4 hores. A mesura que s'amplia l'interval cap a valors superiors, com en el cas de `10--20` o `15--20`, s'observen increments considerables en la durada, arribant a una mitjana de 122,14 hores en aquest últim cas. Això confirma que la vegetació, com a capa que retarda la propagació directa del foc, juga un paper essencial en la dinàmica temporal del model.

També és rellevant observar que els intervals més elevats no només fan que el foc trigui més a consumir cada cel·la individualment, sinó que, en conseqüència, endarrereixen la propagació cap a les cel·les veïnes. Això pot provocar escenaris en què el foc queda temporalment estancat, en espera que la vegetació es consumeixi prou per permetre la propagació.

En resum, aquest experiment valida la hipòtesi que una vegetació més abundant i densa incrementa notablement la resistència a la propagació del foc i, per tant, la durada total de l'incendi.

2.4.4 Experiment 4: Efecte de l'increment de l'interval d'humitat sobre la durada de l'incendi

Aquest experiment té com a objectiu analitzar l'impacte que té l'interval de valors d'humitat sobre la durada total de l'incendi. En el model implementat, la humitat representa el nombre d'unitats de temps durant les quals una cel·la no pot començar a cremar-se, encara que estigui en contacte amb una cel·la veïna en flames. Així, un valor d'humitat elevat retarda significativament l'inici de la combustió i, per tant, la propagació del foc.

S'ha decidit fer servir diferents intervals de generació d'humitat per observar-ne l'efecte: `0--5`, `0--10`, `0--15` i `0--20`. A cada experiment, els valors de la capa d'humitat es generen aleatòriament dins d'aquests intervals, excepte en el cas de les cel·les que formen part dels llacs, les quals mantenen la seva humitat infinita i no es veuen afectades per aquest paràmetre.

A través d'aquesta experimentació, es busca validar la hipòtesi que un augment en l'humitat mitjana del terreny comporta una propagació més lenta del foc i, per tant, una major durada global de l'incendi.

Table 4: Durada (en hores simulades) segons l'interval d'humitat.

Iteració	**0–5**	**0–10**	**0–15**	**0–20**
1	98	137	194	323
2	90	178	212	212
3	106	178	266	206
4	109	166	301	297
5	90	144	196	216
6	116	195	218	304
7	101	132	176	287

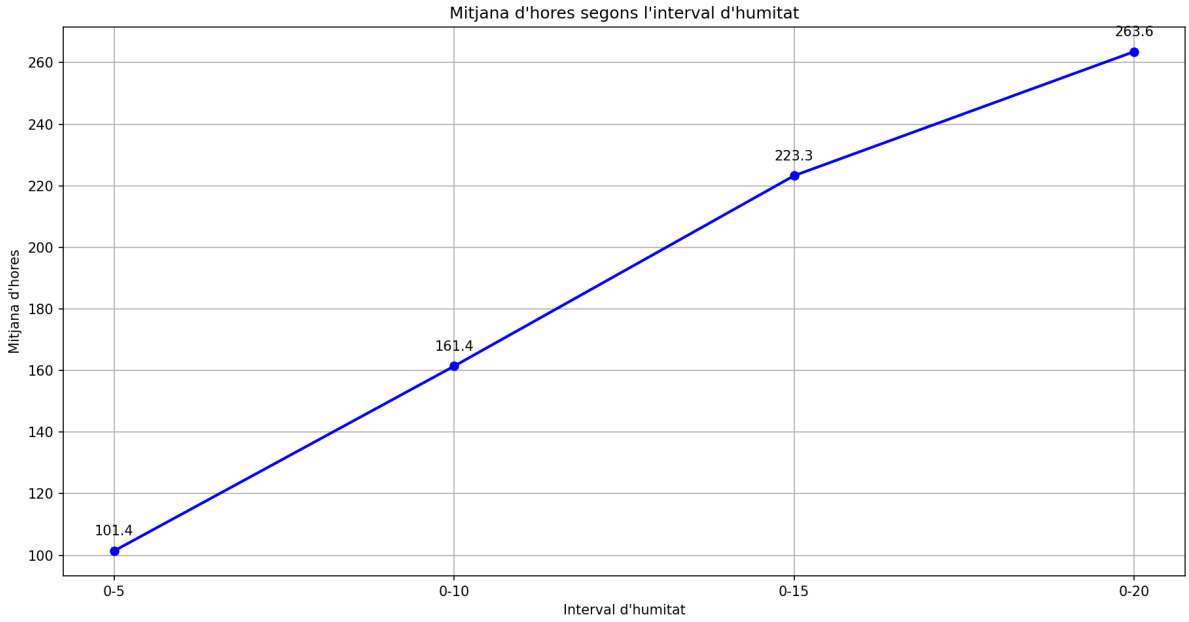

Figure 12: Mitjana de la durada (en hores simulades) segons l'interval d'humitat

Els resultats d'aquest experiment mostren una relació fortament positiva entre l'interval de valors d'humitat i la durada total de l'incendi. A mesura que el rang d'humitat augmenta, el foc triga considerablement més a consumir tota la graella. En concret, la durada mitjana de l'incendi passa de 101,4 hores en l'escenari amb humitats entre 0 i 5, fins a 263,6 hores quan l'interval s'estén de 0 a 20.

Aquest increment és coherent amb el funcionament del model, on l'humitat actua com a retardador de

la ignició. Cada cel·la amb humitat superior a zero ha de reduir aquest valor progressivament abans de començar a cremar-se. Per tant, si el rang màxim d'humitat és elevat, algunes cel·les poden quedar diverses hores sense propagar foc tot i estar rodejades de cel·les en flames. Això frena el ritme de propagació de manera significativa.

A més, el fet que les cel·les amb humitat infinita (com els llacs) no s'hagin inclòs en aquest experiment permet afirmar amb més claredat que l'efecte observat prové exclusivament de la variació del paràmetre d'humitat i no d'obstacles impassables.

En conjunt, l'experiment confirma que l'humitat és un dels factors més determinants per frenar la propagació d'un incendi en aquest tipus de model, i que la seva variació pot tenir un impacte dràstic en la durada total del fenomen.

2.4.5 Experiment 5: Efecte de la direcció del vent sobre la durada de l'incendi

Aquest experiment té com a objectiu estudiar la influència de la direcció del vent en el comportament temporal de la propagació d'un incendi. A diferència d'altres factors com la humitat o la vegetació, que actuen com a resistències locals a la combustió, el vent modifica el patró espacial de propagació, afavorint el desplaçament del foc cap a una direcció concreta i accelerant la ignició de cel·les situades a major distància.

En el model utilitzat, la direcció del vent s'introdueix com un paràmetre discret amb quatre possibles valors: `nord`, `sud`, `est` i `oest`. En absència de vent, la propagació del foc es produeix només entre cel·les adjacents. En canvi, quan s'especifica una direcció de vent, el foc té capacitat d'afectar directament cel·les situades dues unitats més enllà en la direcció indicada, la qual cosa pot provocar una expansió més ràpida del front de l'incendi.

Això permet valorar com de determinant pot ser el vent a l'hora d'accelerar o frenar la propagació d'un incendi en un entorn determinat.

Table 5: Durada (en hores simulades) segons la direcció del vent.

Iteració	**nord**	**sud**	**est**	**oest**	**None**
1	91	98	90	98	98
2	90	90	85	90	90
3	86	106	106	103	106
4	109	95	106	109	109
5	90	84	90	89	90
6	111	116	116	98	116
7	101	94	101	95	101

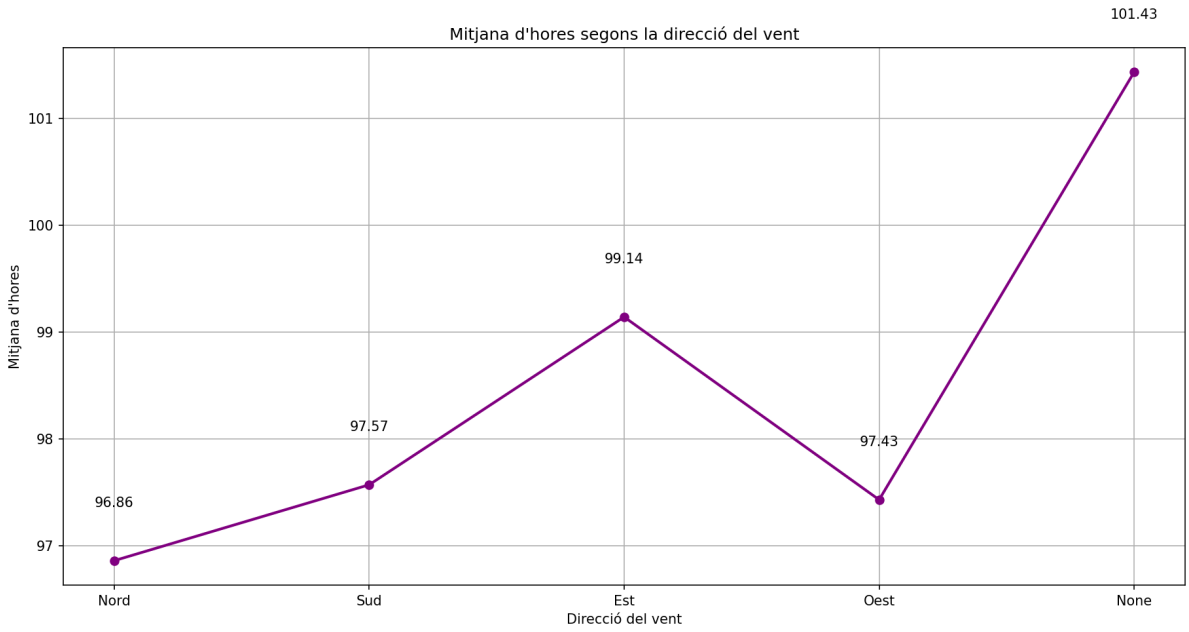

Figure 13: Mitjana de la durada (en hores simulades) segons la direcció del vent

Els resultats obtinguts mostren que la direcció del vent té un efecte lleuger però mesurable en la durada total de l'incendi. Tot i que les diferències no són tan pronunciades com en altres paràmetres com la humitat o la vegetació, sí que s'observa una tendència clara: en presència de vent, el foc tendeix a consumir la graella en menys temps que en absència de vent.

En concret, la durada mitjana més alta es dona quan no hi ha vent (`None`), amb un valor de 101.43 hores. Això indica que, en aquest escenari, el foc es propaga de forma més simètrica i lenta, ja que no rep cap impuls addicional que afavoreixi l'expansió ràpida cap a una direcció concreta. Per contra, les quatre direccions de vent donen lloc a durades lleugerament inferiors, situant-se entre 96.86 (vent del nord) i 99.14 hores (vent de l'est).

Aquestes diferències poden atribuir-se a la capacitat del vent per afavorir la ignició de cel·les més allunyades en la direcció corresponent. Tot i això, com que el vent només afecta una direcció i el mapa és simètric, la seva influència no transforma radicalment la propagació del foc, sinó que simplement introdueix una lleugera acceleració en el procés.

També cal considerar que, com que la ubicació del nucli inicial de foc és aleatòria, el vent pot ser més o menys efectiu depenent de la proximitat d'obstacles o límits del mapa en la direcció afavorida. Per això, les diferències observades no són extremes, però sí consistents.

En resum, podem concloure que la presència de vent redueix lleugerament la durada de l'incendi, però el seu impacte és menys acusat que altres variables com l'humitat o la vegetació.

3 Conclusions

3.1 1a Part

Aquesta primera part de la pràctica ha permès entendre en profunditat el funcionament dels autòmats cel·lulars unidimensionals definits per les regles de Wolfram. A partir d'una implementació bàsica d'un autòmat amb una única regla, s'ha construït un model més general capaç de combinar diferents regles de forma seqüencial i coherent.

El disseny modular de la classe `AutomatCellularMultiregla` ens ha permès:

- Visualitzar de forma clara l'evolució temporal de qualsevol regla de Wolfram.
- Extreure parts específiques (terços) de cada evolució per tal de crear una composició híbrida.
- Validar el comportament del sistema mitjançant gràfiques comparatives i verificacions visuals.

Aquest plantejament ha resultat útil per observar com diferents regles poden generar comportaments contrastats, fins i tot quan parteixen del mateix estat inicial. A més, la combinació de terços ha permès construir una evolució global que barreja dinàmiques diverses, obrint la porta a aplicacions més complexes com models adaptatius o sistemes heterogenis.

Com a possible línia de millora, es podria generalitzar el model per permetre un nombre arbitrari de regles i implementar sistemes on el canvi de regla es produeixi de forma dinàmica durant la simulació (per exemple, cada cert nombre de passos o segons l'estat local). També es podria afegir suport per a condicions de frontera periòdiques o per explorar altres estratègies de combinació entre regles.

En definitiva, aquest exercici ha proporcionat una visió clara i manipulable de les regles cel·lulars i ha posat de manifest la potència d'aquests sistemes per generar comportaments complexos a partir de normes simples.

3.2 2a Part

Al llarg d'aquesta segona part, hem desenvolupat un simulador de propagació d'incendis forestals mitjançant un model basat en cel·les. Aquest enfocament ens ha permès entendre com diferents factors ambientals influeixen en l'evolució del foc, com ara la humitat del terreny, la quantitat i distribució de la vegetació, la presència de llacs i la direcció del vent. Gràcies a una implementació modular, amb la separació entre generació de dades, lectura de fitxers, simulació i visualització, hem aconseguit estructurar el codi d'una manera clara i eficient, afavorint-ne la comprensió i reutilització.

A través dels experiments realitzats, hem pogut observar com petits canvis en els paràmetres poden provocar diferències significatives en la durada total de l'incendi. Per exemple, hem constatat que un increment en la humitat o en la densitat de vegetació pot alentir notablement la propagació del foc, mentre que un augment del nombre de nuclis inicials accelera clarament el procés de combustió. També hem comprovat que la presència de llacs pot actuar com a barrera natural, tot i que el seu efecte depèn molt de la seva ubicació dins la graella. Pel que fa al vent, hem detectat que pot afavorir una direcció de propagació, encara que el seu impacte global és menor en comparació amb altres factors.

En definitiva, aquest projecte ens ha permès aprofundir en la modelització de fenòmens naturals complexos i adquirir una visió més clara dels factors que intervenen en la propagació d'un incendi forestal.

4 Ús de la intel·ligència artificial

Al llarg del desenvolupament d'aquest projecte, hem fet ús d'eines basades en intel·ligència artificial per agilitzar i millorar el procés de programació. Concretament, hem utilitzat **ChatGPT** com a assistent per estructurar i implementar parts del codi, resoldre dubtes conceptuals sobre el funcionament del model, i redactar fragments de documentació i anàlisi. Aquesta eina ens ha permès generar idees clares i modulars, especialment útils durant la definició de la lògica de propagació i la creació d'experiments comparatius.

D'altra banda, també hem comptat amb el suport de **GitHub Copilot**, que ens ha estat útil per suggerir completions automàtiques i detectar possibles errors en temps real durant la codificació. Aquest suport ens ha permès solucionar problemes concrets de manera més àgil i mantenir una estructura coherent al llarg del projecte.