

ZOMBIE CELULAR

Autómata Celular | Sistema Inmunológico vs Virus

INTEGRANTES

Alejandra Díaz Navarro - 2240063

Heydan Ricardo Calderón Villamizar - 2240077

Juan Manuel Niño Piña - 2240040

Daniel Santiago Borrero García - 2221917

Sara Sofía Solano Rocha - 2240058

Paula Andrea Vanges Franco - 2231861

CONTEXTO

El Virus Wildfire es un patógeno ficticio ultra-agresivo presente de forma latente en todos los humanos, concentrado en el cerebro. Permanece dormido sin síntomas hasta activarse por mordida zombi o muerte natural, reactivando el tallo cerebral para crear vectores agresivos (zombis) que solo buscan infectar o consumir vivos.



Virus Wildfire

- **Introducción al Cuerpo por Mordida**
- **Reacción del Cuerpo**
- **Momento de la Fiebre**

IMPLEMENTACIÓN DEL CONTEXTO

Análisis del Vecindario

Cada célula analiza su vecindario para contabilizar:

- Cantidad de virus alrededor.
- Células inmunes presentes y su nivel de anticuerpos.
- Carga viral acumulada y energía disponible.
- Células infectadas, zombis, coágulos y plasma.

ESTADOS CELULARES

💧 Células Sanas



Plasma



Glóbulo Rojo (Eritrocito)



Neutrófilo



Linfocito



Macrófago



Plaqueta

⚡ Sistema Inmune Activado



Neutrófilo Activado

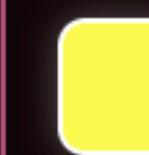


Linfocito Activado



Macrófago Activado

Proceso de Infección



Partícula Viral



Glóbulo Rojo Infectado



Glóbulo Blanco Infectado



Explosión Viral



Transformación Zombie



Glóbulo Rojo Zombificado



Glóbulo Blanco Zombificado



Tejido Necrótico



Coagulación y Barreras



Coágulo de Sangre



Pared de Vaso Sanguíneo



ZOMBIE CELULAR

Reglas del Autómata Celular

⚠ Sistema Inmunológico vs Infección Viral

✳ Simulación Biológica en Python



➡ Desplázate para ver todas las reglas

● GLÓBULOS ROJOS



Infección Viral

Los glóbulos rojos son vulnerables al virus. El riesgo aumenta con la exposición:



```
# Cálculo del riesgo de infección
def aplicarReglasGlobuloRojo(self, a):
    riesgoInfeccion = (a.cantidadVirus * 0.25) + \
        (a.cantidadInfectadas * 0.15) + \
        (self.cargaViral / 200.0)

    if random.random() < riesgoInfeccion:
        return EstadoCelular.CelulaRojaInfectada
```



Zombificación

Rodeado por 6+ zombies, la célula se transforma:



```
# Zombificación directa
if a.cantidadZombies >= 6:
    return EstadoCelular.GlobuloRojoZombie

# Protección por anticuerpos
if self.nivelAnticuerpos > 60 and a.cantidadInmunes >= 3:
    self.cargaViral = max(0, self.cargaViral - 20)
```



Desplázate para ver todas las reglas

⚡ NEUTRÓFILOS - Respuesta Rápida



Activación y Fagocitosis

Primera línea de defensa. Se activan rápidamente y devoran patógenos:  → 

```
# Activación ante amenaza
if not estaActivado and (a.cantidadVirus + a.cantidadInfectadas >= 2):
    self.nivelEnergia = 100
    return EstadoCelular.NeutrofiloActivado

# Fagocitosis - Destruye virus
if a.cantidadVirus > 0 and self.nivelEnergia > 30:
    probabilidadFagocitosis = 0.65 - (self.cargaViral/200.0)
    if random.random() < probabilidadFagocitosis:
        self.cargaViral = 0
        self.nivelEnergia -= 40 # Gasta energía

# Puede morir tras fagocitar
if self.nivelEnergia < 20 or random.random() < 0.3:
    return EstadoCelular.Plasma
```



➡ Desplázate para ver todas las reglas



LINFOCITOS - Memoria Inmune



Aprendizaje Inmunológico

Generan memoria del patógeno para respuestas futuras más rápidas

```
# Activación con memoria
umbralActivacion = 2
if self.memoriaInmune > 60:
    umbralActivacion = 1 # ¡Más rápido!

if a.cantidadInfectadas + a.cantidadZombies >= umbralActivacion:
    self.nivelAnticuerpos += 25
    self.memoriaInmune += 15
return EstadoCelular.LinfocitoActivado
```



Producción de Anticuerpos

Fabrican anticuerpos que neutralizan el virus y curan células infectadas

```
# Producción acelerada
incrementoAnticuerpos = 5
if self.memoriaInmune > 70:
    incrementoAnticuerpos = 8

# Curación de infectadas
if a.cantidadInfectadas > 0:
    bonusMemoria = self.memoriaInmune / 200.0
    probabilidadCuracion = (0.35 + bonusMemoria) * \
        (self.nivelAnticuerpos / 100.0)
```



Memoria Inmune: 0-40



Novato | 41-70



Entrenado | 71-100



Experto



Desplázate para ver todas las reglas



MACRÓFAGOS - Los Gigantes



Respuesta Lenta pero Poderosa

Grandes y lentos, pero devoran células infectadas enteras

```
# Requiere amenaza alta
if a.cantidadVirus + a.cantidadInfectadas >= 3:
    self.nivelEnergia = 100
    return EstadoCelular.MacrophagoActivado

# Fagocitosis de células completas
if a.cantidadInfectadas > 0:
    probabilidadFagocitosis = 0.55
    if random.random() < probabilidadFagocitosis:
        self.cargaViral = max(0, self.cargaViral - 25)
        self.nivelEnergia -= 30
```

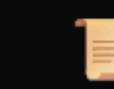


Presentación de Antígenos

"Enseñan" a los linfocitos, aumentando la memoria inmune del sistema

```
# Entrena otras células
if a.cantidadInmunes > 0 and self.edadEstado % 3 == 0:
    self.memoriaInmune += 10

# Muy resistente pero no inmune
if a.cantidadZombies >= 6 and self.cargaViral > 90:
    probabilidadInfeccion = 0.15 + (self.cargaViral/300.0)
    if random.random() < probabilidadInfeccion:
        return EstadoCelular.CelulaBlancaInfectada
```



Desplázate para ver todas las reglas



PARTÍCULAS VIRALES



Estrategia de Infección

Prefieren glóbulos rojos (más abundantes) pero pueden infectar células inmunes

```
# Preferencia por glóbulos rojos
if a.cantidadGlobulosRojos > 0 and random.random() < 0.75:
    return EstadoCelular.CelulaRojaInfectada

# Si no hay rojas, ataca inmunes
if a.cantidadGlobulosRojos == 0 and a.cantidadInmunes > 0:
    if random.random() < 0.4:
        return EstadoCelular.CelulaBlancaInfectada
```

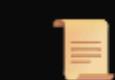


Replicación y Explosión

Después de 4 turnos, puede explotar liberando más virus

```
# Ciclo de vida del virus
if self.edadEstado > 4 and a.cantidadVirus < 3:
    if random.random() < 0.25:
        return EstadoCelular.ExplosionViral

# Eliminación por sistema inmune
if a.cantidadInmunes >= 2 or self.nivelAnticuerpos > 60:
    probabilidadEliminacion = 0.6 + (self.nivelAnticuerpos / 200)
    if random.random() < probabilidadEliminacion:
        return EstadoCelular.Plasma
```



Desplázate para ver todas las reglas



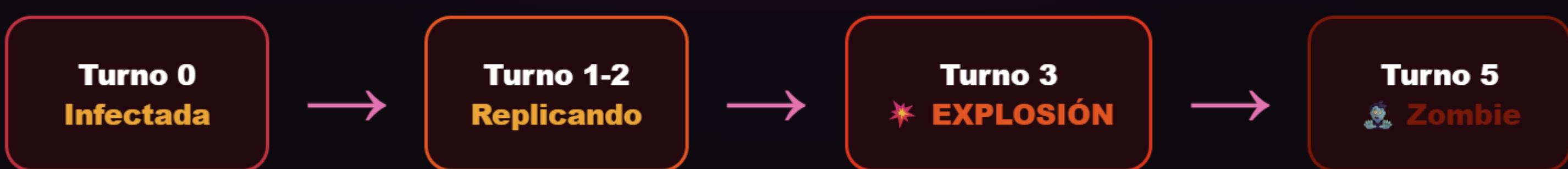
CÉLULAS INFECTADAS



Período de Incubación

El virus se replica dentro de la célula. ¡El tiempo corre!  →  → 

```
# Período de incubación diferente  
esRojaInfectada = (self.estadoActual == EstadoCelular.CelulaRojaInfectada)  
periodoIncubacion = 3 if esRojaInfectada else 4  
  
# Explosión viral al completar replicación  
if self.edadEstado >= periodoIncubacion:  
    return EstadoCelular.ExplosionViral  
  
# Rescate por sistema inmune (más difícil con el tiempo)  
probabilidadRescate = 0.25 - (self.edadEstado * 0.05)  
  
if esRojaInfectada:  
    if a.cantidadInmunes >= 3 and self.nivelAnticuerpos > 70:  
        if random.random() < probabilidadRescate:  
            return EstadoCelular.GlobuloRojo # ¡Salvada!
```



 Desplázate para ver todas las reglas



ZOMBIFICACIÓN



Transformación Zombie

Después de la explosión viral, las células se convierten en zombies

```
# Explosión genera zombies
if self.edadEstado >= 2:
    if random.random() < 0.7:
        return EstadoCelular.GlobuloRojoZombie
    else:
        return EstadoCelular.GlobuloBlancoZombie
```



Necrosis y Eliminación

Los zombies son difíciles de eliminar, pero no inmortales

```
# Eliminación por inmune masivo
if a.cantidadInmunes >= 5 and self.nivelAnticuerpos > 80:
    if self.edadEstado > 20 and random.random() < 0.15:
        return EstadoCelular.TejidoNecrotico

# Necrosis natural (lenta)
if self.edadEstado > 50:
    if random.random() < 0.05:
        return EstadoCelular.TejidoNecrotico
```



Desplázate para ver todas las reglas



PLAQUETAS - Defensa por Barreras



Formación de Coágulos

Las plaquetas crean barreras físicas al detectar amenazas cercanas



```
# Coagulación defensiva
if a.cantidadPlaquetas >= 3 and (a.cantidadInfectadas + a.cantidadZombies) > 1:
    probabilidadCoagulo = 0.3 + (a.cantidadZombies * 0.1)
    if random.random() < probabilidadCoagulo:
        return EstadoCelular.CoaguloSangre

# Vulnerabilidad ante virus
if a.cantidadVirus >= 2 and self.cargaViral > 40:
    return EstadoCelular.ParticulaVirus
```



Desplázate para ver todas las reglas



COÁGULOS SANGUÍNEOS

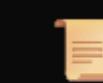


Disolución de Coágulos

Los coágulos se disuelven cuando la amenaza desaparece

```
# Disolución si no hay amenaza
if (self.edadEstado > 25 and a.cantidadZombies == 0 and
    a.cantidadInfectadas == 0):
    if random.random() < 0.2:
        return EstadoCelular.Plasma

# Se mantienen mientras haya infección
return EstadoCelular.CoaguloSangre
```



Desplázate para ver todas las reglas

● TEJIDO NECRÓTICO

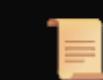


Necrosis y Regeneración

Restos de células zombies destruidas. Pueden regenerarse muy lentamente

```
# Regeneración extremadamente lenta
if self.edadEstado > 80:
    if random.random() < 0.1:
        return EstadoCelular.Plasma

return EstadoCelular.TejidoNecrotico
```



Desplázate para ver todas las reglas



PLASMA SANGUÍNEO



Estado Base del Sistema

El plasma es el estado neutro. Puede regenerar cualquier tipo de célula según las necesidades del sistema

```
# Estado neutro del sistema  
# El plasma es el lienzo  
# para nueva vida celular  
  
return EstadoCelular.Plasma
```



Desplázate para ver todas las reglas



EQUILIBRIO DINÁMICO

El sistema simula una batalla biológica constante entre:

🛡️ **Sistema Inmune** vs 🦠 **Infección Viral**



Virus

Infecta rápido
Se replica
Crea zombies



Neutrófilos

Primera respuesta
Fagocitan virus
Mueren rápido



Linfocitos

Aprenden
Producen anticuerpos
Curan infectadas



Macrófagos

Lentos pero fuertes
Devoran células
Entrenan al sistema



Desplázate para ver todas las reglas



Factores Críticos del Sistema

Carga Viral

Determina riesgo de infección

Nivel de Energía

Capacidad de acción

Nivel de Anticuerpos

Protección y curación

Edad del Estado

Progresión temporal

Memoria Inmune

Respuesta más rápida

Vecindario

Interacciones locales

LA GRILLA DEL AUTÓMATA



Estructura Bidimensional

El torrente sanguíneo se modela como una grilla de 120×80 células, donde cada celda representa un elemento del sistema sanguíneo.

- **Dimensiones: 120 columnas \times 80 filas = 9,600 células totales**
- **Inicialización: Cada célula comienza con un estado según probabilidades fisiológicas**
- **Bordes: Las paredes de los vasos sanguíneos forman límites naturales**
- **Distribución celular: 45% eritrocitos, 53% plasma, 2% leucocitos, 1% plaquetas**

```
class GrillaTorrenteSanguineo: def __init__(self, ancho, alto): self.__ancho = 120 self.__alto = 80  
self.__grilla = [ [CelulaSanguinea() for _ in range(ancho)] for _ in range(alto) ]
```

VECINDARIO DE MOORE

👥 Análisis de 8 Vecinos



- **Cada célula analiza sus 8 vecinos inmediatos**
- **Incluye vecinos horizontales, verticales y diagonales**
- **Los bordes retornan None para posiciones fuera de límites**
- **Permite propagación rápida de virus e infección**

```
def obtenerVecinos(self, fila, col): vecinos = [None] * 8 indice = 0 # Recorrer vecindario 3x3 (excluyendo centro) for df in range(-1, 2): for dc in range(-1, 2): if df == 0 and dc == 0: continue # Saltar célula central if enLimites(fila+df, col+dc): vecinos[indice] = self.__grilla[fila+df][col+dc]
```

ALGORITMO DEL AUTÓMATA

⚙️ Proceso de Actualización en 2 Fases

El autómata celular utiliza un algoritmo de doble buffer para garantizar actualizaciones sincronizadas de todas las células.

1 Fase de Cálculo

Cada célula analiza su vecindario y determina su próximo estado sin modificar la grilla actual.

2 Fase de Aplicación

Todas las células actualizan simultáneamente a su nuevo estado calculado.

```
def paso(self): # FASE 1: Calcular próximo estado
    for fila in range(self.__alto):
        for col in range(self.__ancho):
            vecinos = self.obtenerVecinos(fila, col)
            siguienteEstado = self.__grilla[fila][col].calcularSiguienteEstado(vecinos)
            self.__grilla[fila][col].establecerSiguienteEstado(siguienteEstado) # FASE 2: Aplicar cambios simultáneos
    for fila in range(self.__alto):
        for col in range(self.__ancho):
            self.__grilla[fila][col].aplicarSiguienteEstado()
```

- Evita efectos de cascada no deseados durante la actualización
- Mantiene la consistencia temporal del sistema
- Permite reversibilidad teórica del autómata

REGLAS DE TRANSICIÓN

5 Lógica de Cambio de Estados

Cada célula determina su próximo estado basándose en conteos de sus vecinos y su estado interno (energía, carga viral, anticuerpos).



Infección Viral

Si una célula sana tiene ≥ 2 virus vecinos, puede infectarse según probabilidades. La carga viral aumenta con cada generación.



Respuesta Inmune

Los leucocitos se activan al detectar ≥ 3 células infectadas. Generan anticuerpos y pueden eliminar virus vecinos.



Explosión Celular

Células con carga viral ≥ 100 explotan, liberando 8 partículas virales hacia sus vecinos de plasma.



Zombificación

Células con energía ≤ 0 y alta carga viral se convierten en zombies, propagando infección agresivamente.

INTERFAZ GRÁFICA - TKINTER



Visualización en Tiempo Real

La interfaz desktop utiliza Tkinter para crear una experiencia visual inmersiva del torrente sanguíneo.

- **Canvas de visualización: 840x560 píxeles que renderizan 9,600 células en tiempo real**
- **Dashboard médico: Panel lateral con métricas vitales actualizadas cada generación**
- **Controles interactivos: Botones para iniciar, pausar, avanzar paso a paso y reiniciar**
- **Inyección de virus: Introducción manual de patógenos en ubicaciones aleatorias**
- **Multithreading: Simulación en hilo separado para mantener la UI responsiva**
- **Scroll vertical: Contenedor scrollable para adaptarse a diferentes resoluciones**

```
class Simulador: def __init__(self): self.root = tk.Tk() self.canvas = tk.Canvas( width=840, height=560,  
bg="#1a1a1a" ) self.crear_dashboard_medico() self.crear_panel_controles()
```

INTERFAZ GOOGLE COLAB

● Versión en la Nube

Para entornos sin capacidad de ejecutar Tkinter, se desarrolló una versión compatible con Google Colab usando bibliotecas web.

Matplotlib

Renderiza la grilla del torrente sanguíneo como una imagen de píxeles coloreados, actualizada en cada frame.

ipywidgets

Proporciona controles interactivos (botones, sliders) que funcionan dentro de notebooks Jupyter.

```
import matplotlib.pyplot as plt from ipywidgets import Button, Output, VBox
def visualizar_grilla():
    fig, ax = plt.subplots(figsize=(12, 8))
    matriz_colores = generar_matriz_rgb(grilla)
    ax.imshow(matriz_colores, interpolation='nearest')
    plt.show()
    # Controles interactivos
    btn_iniciar = Button(description='▶ Iniciar')
    btn_pausar = Button(description='|| Pausar')
    display(VBox([btn_iniciar, btn_pausar, output]))
```

- **Compatibilidad:** Funciona en cualquier navegador sin instalaciones
- **Portabilidad:** Código ejecutable directamente desde GitHub o Drive
- **Limitaciones:** Actualización más lenta que la versión Tkinter nativa

CONCLUSIONES

- **Se puede mejorar la simulación haciendo que las células se muevan.**
- **Sirve para aplicaciones en la vida real como videojuegos o simulaciones de propagación de enfermedad.**
- **Se acerca a una simulación del comportamiento de la propagación de un virus y cómo actúa el cuerpo en respuesta.**