

# GraphViz Pro 3D

Sistema Interactivo para la Generación, Análisis y  
Visualización de Grafos

Reporte Técnico

Hector Jesus Vela Acosta

5 de diciembre de 2025

## Resumen

Este reporte documenta el desarrollo de **GraphViz Pro 3D**, una herramienta web para la visualización y análisis de grafos. Se detallan los fundamentos matemáticos de los algoritmos de Conjunto Independiente Máximo (MIS), la geometría utilizada para la generación de grafos de Halin y la implementación técnica mediante WebGL. El sistema integra análisis algorítmico en tiempo real con una representación gráfica avanzada.

# Índice

<b>1. Introducción</b>	<b>3</b>
<b>2. Arquitectura del Sistema</b>	<b>3</b>
<b>3. Análisis Algorítmico (MIS)</b>	<b>3</b>
3.1. Heurística Greedy Aleatorizada . . . . .	3
3.2. Heurística de Prioridad Hojas (From Leaves) . . . . .	4
<b>4. Generación Procedural: Grafos de Halin 3D</b>	<b>4</b>
4.1. Modelo Matemático de Distribución . . . . .	4
<b>5. Detalles de Implementación en Código</b>	<b>5</b>
5.1. Estructuras de Datos Eficientes . . . . .	5
5.2. Sistema de Animación (Timeline) . . . . .	5
<b>6. Interfaz de Usuario (Dashboard)</b>	<b>5</b>

## 1. Introducción

El análisis de grafos complejos requiere herramientas que vayan más allá de las representaciones estáticas en 2D. Este proyecto aborda dos problemas fundamentales:

1. **Visualización:** La representación de topologías complejas (como grafos de Halin) sin cruces visuales ni oclusión.
2. **Optimización Combinatoria:** La resolución y visualización del problema del Conjunto Independiente Máximo (MIS).

## 2. Arquitectura del Sistema

El sistema opera como una aplicación de una sola página (SPA) construida con tecnologías web estándar.

- **Core:** HTML5, CSS3, JavaScript (ES6+).
- **Motor Gráfico:** Three.js (WebGL) para renderizado 3D acelerado por hardware.
- **Post-Procesamiento:** EffectComposer para efectos visuales (Bloom/Neón).

## 3. Análisis Algorítmico (MIS)

El problema del Conjunto Independiente Máximo (MIS) consiste en encontrar un subconjunto de vértices  $S \subseteq V$  tal que para todo par de vértices  $u, v \in S$ , no exista una arista  $(u, v) \in E$ . Dado que es un problema NP-Difícil, implementamos dos heurísticas eficientes.

### 3.1. Heurística Greedy Aleatorizada

Este algoritmo ofrece una solución rápida con una complejidad computacional de  $O(|V| + |E|)$ .

**Definición del Algoritmo:**

1. Sea  $G = (V, E)$  el grafo de entrada.
2. Se genera una permutación aleatoria  $\pi$  de los vértices en  $V$  para evitar sesgos por el orden de entrada.
3. Se inicializa  $S = \emptyset$  (Conjunto Independiente) y  $D = \emptyset$  (Conjunto de Descartados).
4. Se itera sobre cada vértice  $v \in V$  siguiendo el orden  $\pi$ :
  - Si  $v \notin D$ :
  - $S \leftarrow S \cup \{v\}$  (Seleccionar  $v$ ).
  - Para cada vecino  $u \in Vecinos(v)$ :
  - $D \leftarrow D \cup \{u\}$  (Descartar vecinos).
5. **Resultado:** El conjunto  $S$  es un Conjunto Independiente Maximal.

### 3.2. Heurística de Prioridad Hojas (From Leaves)

Esta estrategia explota la topología del grafo para mejorar la cardinalidad de la solución, basándose en la propiedad de que los nodos hoja siempre pueden formar parte de una solución óptima local.

#### Definición del Algoritmo:

##### 1. Fase de Poda:

- Identificar el conjunto de hojas  $L = \{v \in V \mid grado(v) = 1\}$ .
- Añadir  $L$  a la solución:  $S \leftarrow S \cup L$ .
- Identificar el conjunto de vecinos de las hojas:  $N(L) = \bigcup_{v \in L} Vecinos(v)$ .
- Eliminar del grafo los nodos procesados:  $V' = V \setminus (L \cup N(L))$ .

##### 2. Fase Greedy:

- Aplicar el algoritmo Greedy Aleatorizado sobre el subgrafo inducido  $G[V']$ .
- Unir el resultado con el conjunto  $S$  obtenido en la fase de poda.

## 4. Generación Procedural: Grafos de Halin 3D

Un grafo de Halin se construye tradicionalmente incrustando un árbol en el plano y conectando sus hojas en un ciclo. Para la visualización 3D, extendemos este concepto a una estructura multinivel cónica.

### 4.1. Modelo Matemático de Distribución

Para posicionar los nodos en el espacio 3D  $(x, y, z)$ , utilizamos coordenadas cilíndricas.

#### 1. Nivel Raíz (Centro):

$$P_{raiz} = (0, altura, 0)$$

**2. Nivel Intermedio (Ramas):** Distribuimos  $k$  nodos equidistantemente en un círculo a media altura. La posición del  $i$ -ésimo nodo es:

$$\theta_i = \frac{2\pi \cdot i}{k}$$

$$x = r_{medio} \cdot \cos(\theta_i), \quad z = r_{medio} \cdot \sin(\theta_i), \quad y = \frac{altura}{2}$$

**3. Nivel Exterior (Hojas):** Las hojas se distribuyen en el plano base ( $y = 0$ ). Para evitar cruces de aristas, cada rama "padre" tiene asignado un sector angular  $\Delta\theta$  exclusivo, dentro del cual se distribuyen sus hijos.

$$x_{hijo} = r_{max} \cdot \cos(\phi), \quad z_{hijo} = r_{max} \cdot \sin(\phi), \quad y = 0$$

Donde  $\phi$  es el ángulo calculado dentro del sector del padre.

## 5. Detalles de Implementación en Código

Esta sección describe cómo los modelos matemáticos anteriores se traducen a JavaScript.

### 5.1. Estructuras de Datos Eficientes

Para soportar la complejidad algorítmica descrita en la Sección 3, el parser utiliza 'Set' y 'Map' para garantizar búsquedas en tiempo constante  $O(1)$ .

```

1 function parseEdgelist(text) {
2   const nodos = new Set(); // Evita duplicados
3   const adj = new Map();   // Lista de adyacencia Hash
4
5   // ... lectura linea por linea ...
6
7   // Insercion eficiente
8   if (!adj.has(u)) adj.set(u, []);
9   adj.get(u).push(v);
10 }

```

### 5.2. Sistema de Animación (Timeline)

Para visualizar la ejecución del algoritmo sin bloquear la interfaz de usuario, implementamos un patrón de "Grabación y Reproducción".

```

1 // En lugar de modificar la vista inmediatamente:
2 timeline.push({
3   node: u,
4   action: 'select' // o 'discard', 'leaf'
5 });
6
7 // Reproduccion controlada
8 setInterval(() => {
9   const paso = timeline[index];
10  aplicarCambioVisual(paso);
11  index++;
12 }, velocidad);

```

## 6. Interfaz de Usuario (Dashboard)

El sistema implementa una interfaz moderna con las siguientes características:

- **Panel Colapsable:** Maximiza el área de trabajo.
- **Feedback Métrico:** Cálculo en tiempo real de densidad ( $D = \frac{2|E|}{|V|(|V|-1)}$ ), grado máximo y tamaño del MIS.
- **Raycasting:** Interacción 3D mediante el lanzamiento de rayos desde la cámara para la detección de nodos bajo el cursor.