UNIVERSIDAD POLITÉCNICA DE VICTORIA

Ingeniería en Tecnologías de la Información

Proyecto Final

Algoritmo KOSARAJU

Integrantes:

Kenia Elizondo Maravilla - 2130110

Yandi Melina Solis Berrones - 2330047

Alberto Azahel Villanueva Bocanegra - 2330068

Docente: Lidia Ivaanery Garcia Juarez

Ciudad Victoria, Tamaulipas a 05 de noviembre 2025

01. INTRODUCCIÓN

EXPLICACIÓN DEL PROBLEMA A RESOLVER

En una red social como tiktok, los usuarios pueden seguir a otros, formando un grafo dirigido donde los nodos representan usuarios y las aristas indican una relación de seguimiento.

Un SCC (Strongly Connected Component) es un grupo de nodos donde cada usuario puede llegar a cualquier otro dentro del mismo grupo mediante los enlaces dirigidos.

Ejemplo: Si A sigue a B y B sigue a C, pero C no sigue a A, entonces no es un SCC. Si A sigue a B, B sigue a C y C sigue a A, entonces forman un SCC. El algoritmo de Kosaraju detecta estos SCCs en O(V + E), lo que lo hace eficiente incluso en grafos grandes.

01. INTRODUCCIÓN

JUSTIFICACIÓN DEL USO DEL ALGORITMO

El código representa una red social como un grafo dirigido, donde los nodos son usuarios y las aristas simbolizan relaciones de seguimiento. Para descubrir comunidades dentro de esta red, se utilizan dos algoritmos: Kosaraju y Tarjan. Ambos tienen una complejidad de O(V + E), lo que los hace rápidos y eficientes para analizar redes grandes

Kosaraju funciona realizando dos recorridos en profundidad (DFS) y transponiendo el grafo, lo que permite detectar grupos de usuarios fuertemente conectados de manera clara y estructurada.
Tarjan, por otro lado, utiliza una sola pasada de DFS junto con valores de baja accesibilidad (low-link values), lo que agiliza la identificación de estas comunidades.

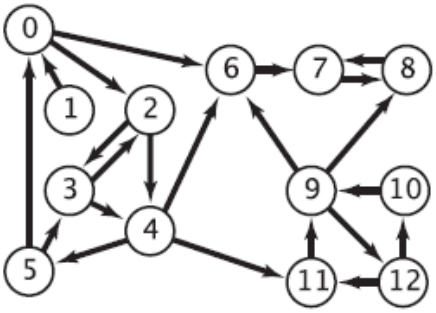
Ambos algoritmos son útiles para encontrar grupos de usuarios que están estrechamente interconectados, lo que ayuda a entender cómo se influyen entre sí o cómo se propaga la información en la red. Además, comparar sus tiempos de ejecución permite elegir el más adecuado según el tamaño y la dinámica de la red.

02. MARCO TEÓRICO

DEFINICIÓN

El algoritmo de Kosaraju-Sharir, o simplemente algoritmo de Kosaraju, es un método eficiente para encontrar los componentes fuertemente conectados (SCC) en un grafo dirigido. Una SCC es un subconjunto de vértices donde cada uno es accesible desde cualquier otro dentro del mismo grupo.

El algoritmo de Kosaraju-Sharir encuentra los componentes fuertemente conectados (SCC) en un grafo dirigido, donde cada vértice es accesible desde cualquier otro dentro del mismo grupo.



Características.

- Es un algoritmo de tiempo lineal.
- Se basa en el hecho de que el grafo transpuesto (el mismo grafo con la dirección de cada borde invertida) tiene los mismos componentes fuertemente conexos que el grafo original.

02. MARCO TEÓRICO

FUNCIONAMIENTO DE ALGORITMO KOSARAJU

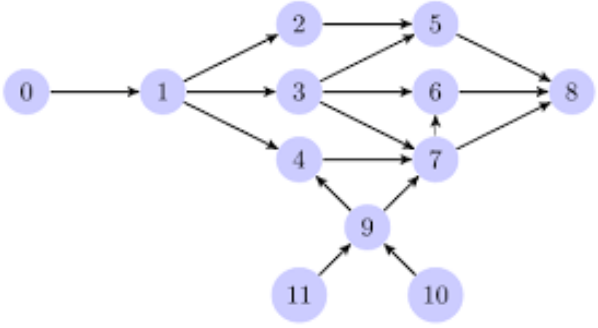
El funcionamiento de el algoritmo de Kosaraju es un algoritmo que realiza dos recorridos de un grafo para identificar sus componentes fuertemente conexas, es decir, identifica los componentes fuertemente conectados (SCC) en un grafo dirigido mediante dos recorridos en profundidad (DFS), utilizando la transposición del grafo.

Paso 1

El algoritmo de Kosaraju se ejecuta en un grafo descrito por una lista de adyacencia, donde cada nodo almacena los nodos a los que está conectado.

Paso 2

El algoritmo de Kosaraju realiza dos DFS: uno en el grafo original para registrar el orden de finalización y otro en el grafo transpuesto, tras invertir las aristas, para identificar las componentes fuertemente conectadas (SCC).



02. MARCO TEÓRICO

COMPLEJIDAD COMPUTACIONAL

La complejidad del algoritmo de Kosaraju es O(n + m). Esto significa que el algoritmo requiere un tiempo lineal para encontrar los componentes fuertemente conectados de un grafo dirigido.

Explicación

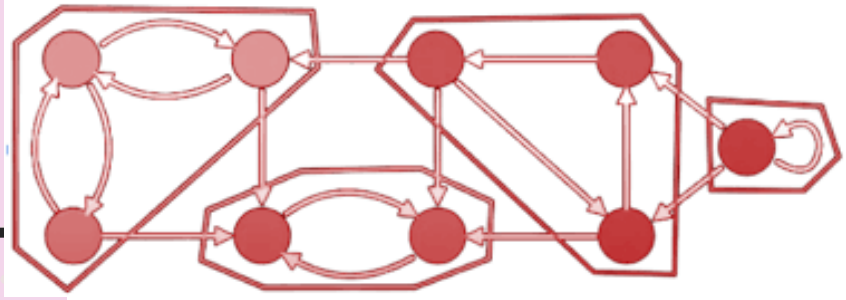
- La complejidad computacional es una medida de la eficiencia de un algoritmo.
- La complejidad computacional asintótica describe cómo el tiempo y el espacio requeridos por un algoritmo aumentan en relación con el tamaño de la entrada.
- La complejidad temporal se expresa utilizando la notación O grande, que describe el límite superior del peor de los casos.
- La complejidad del espacio mide cuánta memoria o almacenamiento se necesita para almacenar una estructura de datos.
- El algoritmo de Kosaraju-Sharir (también conocido como algoritmo de Kosaraju) es un algoritmo de tiempo lineal.

La teoría de la complejidad computacional formaliza la cuantificación de la cantidad de recursos necesarios para resolver problemas.

02. MARCO TEÓRICO

APLICACION REAL DEL ALGORITMO

- Redes sociales: Identifica grupos de usuarios conectados, como en Twitter, donde un retuit puede enlazar a toda una comunidad.
- Motores de búsqueda: Agrupa páginas web interconectadas para mejorar la clasificación y el análisis de comunidades.
- Sistemas de recomendación: Relaciona contenido en plataformas como Netflix o Spotify según interacciones de los usuarios.
- Redes de transporte: Detecta zonas con tráfico interdependiente, útil en carreteras y redes de metro.
- Compiladores: Resuelve dependencias entre módulos y funciones en la programación.
- Circuitos electrónicos: Optimiza diseños al detectar ciclos en señales eléctricas.
- Ciberseguridad: Identifica redes de bots y malware que se comunican entre sí.



03. IMPLEMENTACIÓN TÉCNICA

CÓDIGO Y ESTRUCTURA

El sistema está organizado en dos clases principales que trabajan en conjunto:

- **Grafo**: Lógica de datos y algoritmos SCC
  - dfs_primera: ordenamiento por tiempo de finalización
  - grafo_transpuesto: transposicion del grafo
  - dfs_segunda: segunda DFS en grafo transpuesto

- **AppTiktok**: Interfaz gráfica y controlador principal
  - mostrar_perfil_tiktok: para la conexion usa self.grafo.adyacencia para obtener datos, actualiza la interfaz con datos del grafo
  - seguir_usuario: modifica el grafo subyacente y notifica a la interfaz para actualizar vista

- **Funciones de Visualización:**
  - dibujar_grafo: convierte estructura interna a formato networkx, asigna colores basados en comunidades detectadaas y renderiza usando matplotlib

03. IMPLEMENTACIÓN TÉCNICA

ESTRUCUTRAS DE DATOS

Seleccionamos lista de adyacencia por las siguientes razones:

1. **Optimización para Algoritmos SCC**
   - Los algoritmos Kosaraju y Tarjan requieren:
     - Iteración eficiente sobre vecinos
     - DFS que recorre aristas existentes
2. **Características de Redes Sociales Reales**
   - Propiedades que favorecen lista de adyacencia:
     - Distribución de grado power-law: Pocos usuarios muy conectados, muchos con pocas conexiones
     - Crecimiento incremental: Nuevos usuarios se conectan gradualmente
     - Consultas por vecindario: "¿A quién sigue X?" es más común que "¿X sigue a Y?"
3. **Compatibilidad con Librerías de Visualización:** NetworkX funciona naturalmente con listas
4. **Flexibilidad para Operaciones Dinámicas:** En una red social, las conexiones cambian constantemente

03. IMPLEMENTACIÓN TÉCNICA

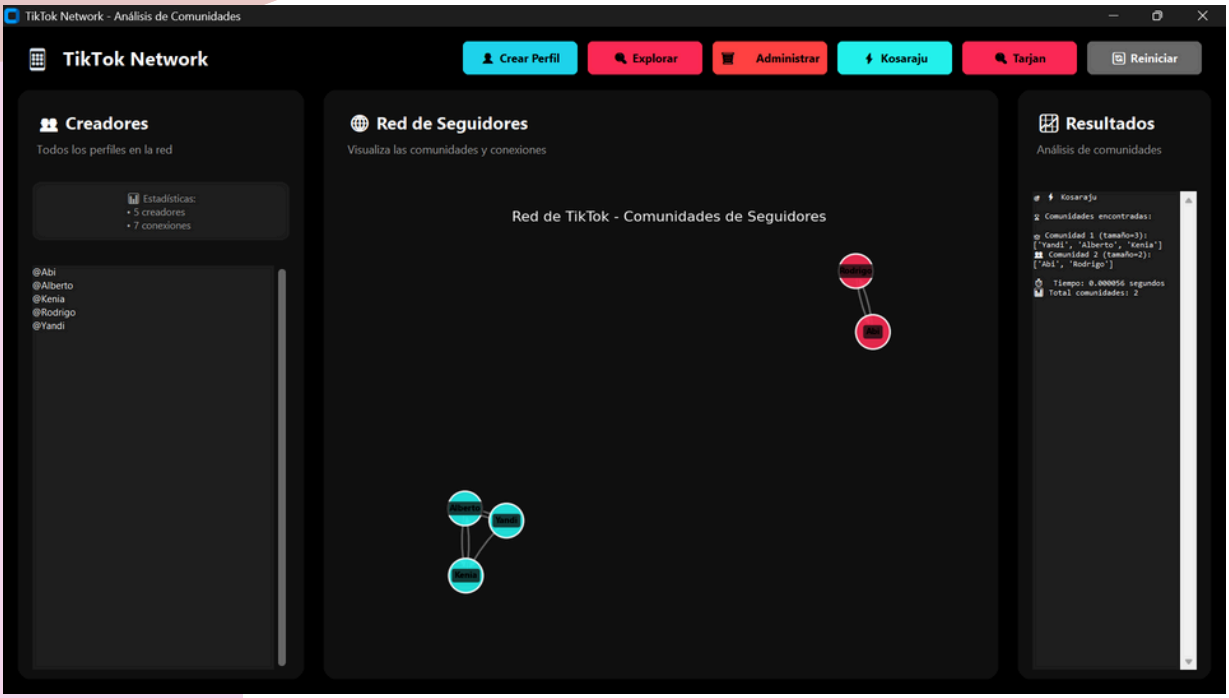
LIBRERÍAS

Librería	Propósito	Justificación
customtkinter	Interfaz gráfica moderna	Reemplaza tkinter estándar con diseño TikTok
networkx	Manipulación de grafos	Algoritmos de layout y visualización
matplotlib	Visualización del grafo	Integración con tkinter y personalización
collections	Estructuras eficientes	defaultdict para lista de adyacencia

04. INTERFAZ

APLICACION REAL DEL ALGORITMO

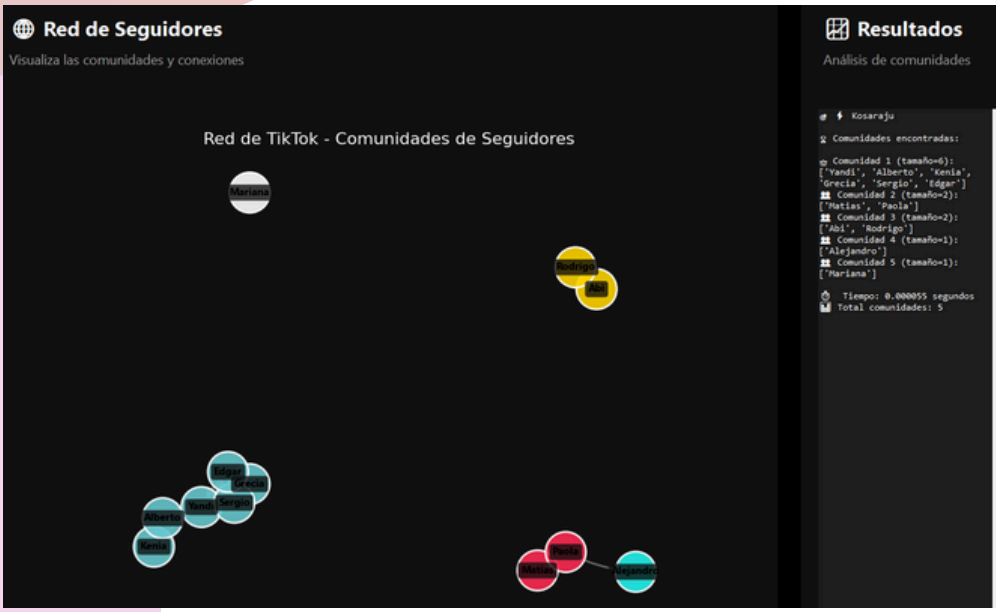
Pagina Principal.- Red de seguidores y comunidades



Aplicación de Kosaraju y comparación con Tarjan

05. PRUEBAS

Representación de un grafo de 12 nodos y 22 conexiones, 5 comunidades y un tiempo de respuesta de 0.000055 segundos



Grafo de 6 nodos y 14 conexiones, 2 comunidades y un tiempo de respuesta de 0.000028 segundos