CUCEI

CENTRO UNIVERSITARIO DE
CIENCIAS EXACTAS E INGENIERÍAS

UNIVERSIDAD DE GUADALAJARA
Red Universitaria de Jalisco

División de Tecnologías para la Integración Ciber-Humana

Departamento de Ciencias Computacionales

Carrera: Ingeniería en Computación

Análisis de Algoritmos
Actividad Voraz

Profesor: JORGE ERNESTO LOPEZ ARCE DELGADO
Gutierrez Vazquez Axel 220575328
Quintero Arreola Laura Vanessa 220577347
Sección: D06

Fecha de entrega: 20 de noviembre de 2025

Introducción

En la siguiente actividad se pone en práctica los conocimientos adquiridos durantes las clases donde se explicaron los algoritmos voraces, así como sus diferencias, así mismo para poner a prueba lo aprendido se desarrollara un código en python que implementa los algoritmos de Prim y Kruskal a un mismo grafo, para que de esta manera se puedan comparar ambos algoritmos y tener un mayor entendimiento de cómo funciona.

Objetivos

- Crear un grafo con 6 nodos y 9 aristas
- Implementar el algoritmo de Prim
- Implementar el algoritmo de Kruskal
- Analizar cuales son las diferencias en la implementación
- Comprender cómo usar un algoritmo voraz

Desarrollo

Para desarrollar la actividad tuvimos que realizar un grafo el cual representamos por medio de un diccionario en python, después implementamos el algoritmo de prim usando una cola de prioridad, una lista con nodos visitados, y el arbol de expansion minima que guardaría el destino, origen y peso. Por otro lado en el algoritmo de Kruskal usamos una lista de tuplas para representar las aristas, las cuales ordenamos de menor a mayor peso, y una estructura auxiliar llamada UnionFind para gestionar las conexiones y evitar la formación de ciclos en el resultado final.

Comparación

Comparación	Kruskal	Prim
Diferencias en el proceso	Tiene un enfoque global, es decir ordena todas las aristas de menor a mayor peso y va uniendo componentes desconectados verificando que no se creen ciclos.	Trabaja con un enfoque local: Inicia en un nodo arbitrario y hace crecer el árbol paso a paso, agregando siempre la arista más barata conectada a los nodos ya visitados.

Diferencias en resultados	Peso 15. Seleccionó las aristas basándose estrictamente en el orden numérico de sus pesos (de menor a mayor globalmente).	Peso 15. Seleccionó las mismas aristas, pero descubriéndolas secuencialmente siguiendo la ruta de conexión desde el nodo A.
Complejidad aproximada	O (E log E)	O (E log V)

Conclusion del equipo

Gutierrez Vazquez Axel

Se comprobó que tanto el algoritmo de Prim como el de Kruskal garantizan el mismo resultado óptimo, convergiendo ambos en un peso total de 15. Respecto a la implementación, se considera que el algoritmo de Kruskal presenta una mayor sencillez conceptual, ya que su lógica basada en el ordenamiento lineal de aristas resulta más intuitiva y directa de codificar en comparación con la gestión dinámica de la cola de prioridad y actualización de vecinos requerida en Prim.

Quintero Arreola Laura Vanessa

Ambos algoritmos son buenos para hacer un árbol de expansión mínima, sin embargo sus formas de abordarlo son considerablemente distintos, mientras que kruskal busca las aristas solo basándose en el peso, prim puede empezar desde cualquier nodo, en este caso nuestros resultados fueron de esta manera porque decidimos que prim empezará desde el nodo A y precisamente este tenía su arista con B de peso 1.