Diseño y Análisis de Algoritmos

Práctica 2

Árboles abarcadores mínimos:
Multigrafos. Puntos de articulación. Algoritmo de Kruskal.

Andrián Navas Ajenjo
Gloria del Valle Cano
adrian.navas@estudiante.uam.es & gloria.valle@estudiante.uam.es

Grado en Ingeniería Informática
Escuela Politécnica Superior
Universidad Autónoma de Madrid
17-11-2019

Índice general

Índice de figuras

1. Introducción

En la segunda práctica hemos implementado algoritmos básicos para multigrafos, detección de puntos de articulación, TAD conjunto disjunto y Kruskal.
Asimismo, abordaremos los problemas propuestos con la representación de gráficas que nos ayuden a realizar un análisis de los tiempos de los algoritmos con el fin de abstraer la complejidad y la estrategia de los algoritmos.

2. Cuestiones

Cuestiones sobre puntos de articulación

Cuestión 1

Tomando como base el código de las funciones `o_a_tables` y `p_o_a_driver`, dar razonadamente una estimación teórica del coste de detectar si un grafo conexo tiene o no puntos de articulación. Contrastar este análisis con las gráficas a elaborar mediante la función `time_pda` considerando únicamente grafos con `prob` 0.7 y 0.9.

Como podemos observar, `p_o_a_driver` tiene un coste de O(|V|) ya que tiene que inicializar los diccionarios para cada nodo el grafo. `o_a_tables` realiza una función básica 2 veces para el número de ramas de cada nodo, por lo cual al final tendremos O(2*|E|)=O(|E|) más la recursividad mediante la cual aplica `o_a_tables` a todas las ramas del grafo también por lo que su coste final es de O(|E|*|E|). Sumando ambos costes, obtenemos un coste total de O(E|*|E| + |V|).

En las siguientes gráficas podemos observar el tiempo de ejecución de time_pda para grafos con 0.7 y 0.9 de probabilidad en sus ramas.

```
times = time_pda(10,10,100,10,0.7)
plt.plot(list(times.keys()),list(times.values()))
plt.show()
```

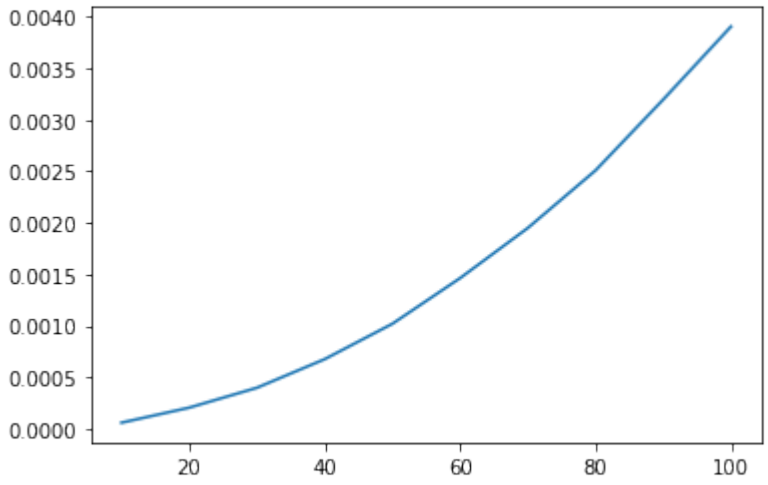

Figura 2.1: Comparación de tiempos de PDA con prob 0.7

```
times = time_pda(10,10,100,10,0.9)
plt.plot(list(times.keys()),list(times.values()))
plt.show()
```

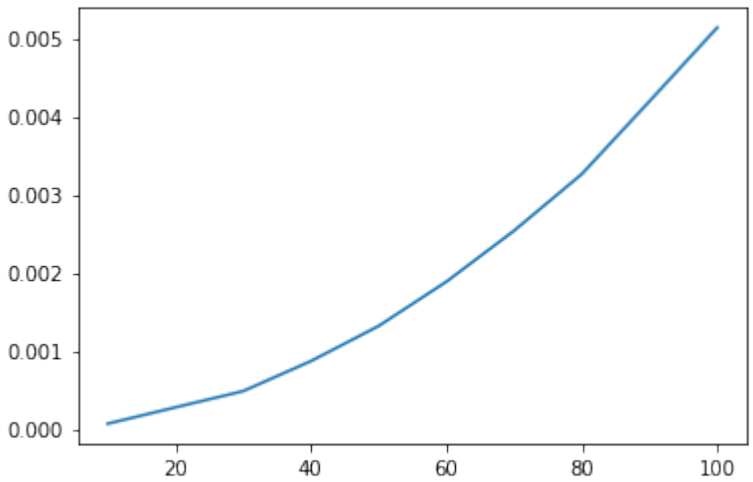

Figura 2.2: Comparación de tiempos de PDA con prob 0.9

Cómo podemos observar, las gráficas tienen una tendencia creciente y exponencial cómo hemos predicho previamente en el estudio teórico.

Cuestión 2

¿Tiene sentido el concepto de punto de articulación en multigrafos? Si crees que sí, arguméntalo. ¿Cómo los definirías?¿Y qué habría que cambiar en las funciones anteriores para que funcionen en multigrafos?
Al igual que en los grafos normales, pueden existir puntos de articulación en los multigrafos. La forma de encontrar los puntos de articulación sería la misma pero añadiendo un bucle for más para estudiar todas las ramas entre dos mismos nodos y escoger una de ellas para el estudio de las o_a_tables.

Cuestiones sobre Kruskal

Cuestión 1

Discutir la aportación al coste teórico del algoritmo de Kruskal tanto de la gestión de la cola de prioridad como la del conjunto disjunto. Intentar llegar a la determinación individual de cada aportación.:

- **Cola de prioridad**:
  - Para insertar elementos en la cola de prioridad: O(|V|log|V|).
  - Para extraer elementos en la cola de prioridad: O(log|V|).
  - Finalmente el coste total de la cola de prioridad es de O(|V|log|V|).
- **Conjunto disjunto**:
  - Para inicializar el conjunto disjunto: O(|V|).
  - El coste de `find` acumulado es de O(|E|log*|V|), lo que es en definitiva O(|E|).
  - El coste acumulado de `union` es de O(|V|).
- **Coste total**: Debido a la aportación de cada operación se estima que al menos el coste de Kruskal total debe ser de:

$$O(|E| \log_{|} V|) \tag{2.1}$$

Cuestión 2

Contrastar la discusión anterior con las gráficas a elaborar mediante las funciones desarrolladas en la práctica.

```
times = time_kruskal(50,10,100,5,0.5,False)
plt.plot(list(times.keys()),list(times.values()))
plt.show()
```

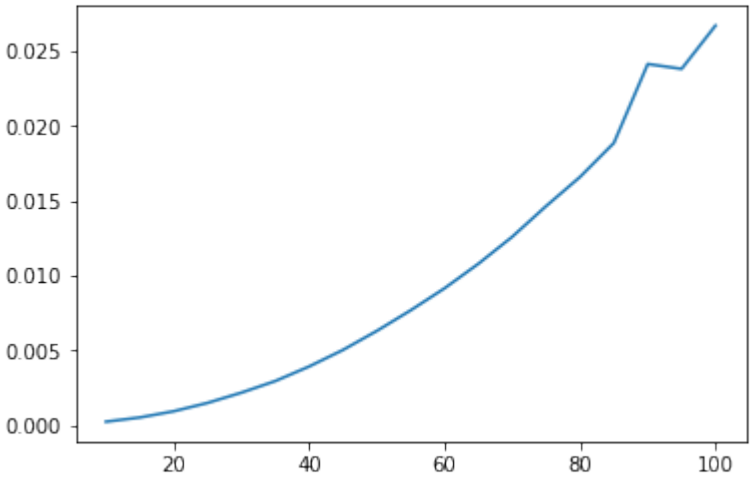

Figura 2.3: Comparación de tiempos de Kruskal

```
times = time_kruskal_2(50,10,100,5,0.5,False)
plt.plot(list(times.keys()),list(times.values()))
plt.show()
```

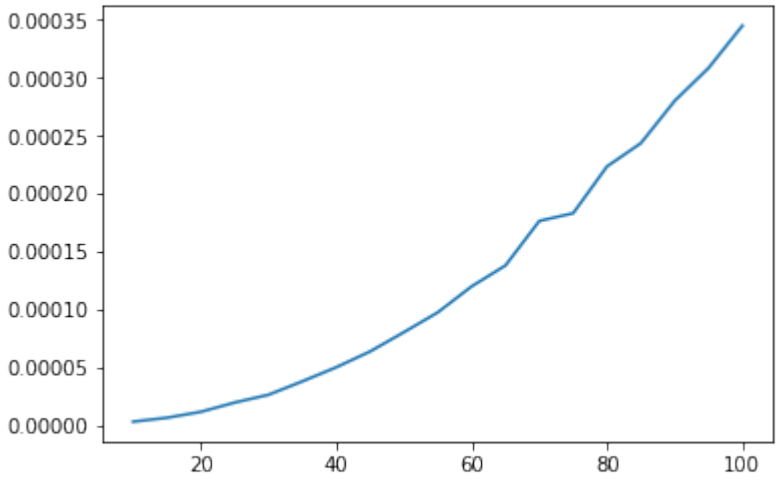

Figura 2.4: Comparación de tiempos de Kruskal2

Observando las gráficas obtenidas por nuestra funcion time-kruskal, podemos asumir nuestras estimaciones teóricas como válidas ya que coinciden los valores teóricos con los experimentales de forma aproximada.

Cuestión 3

¿Tiene sentido el concepto árbol abarcador mínimo en multigrafos? Si crees que sí, ¿cómo los definirías?¿Y qué habría que cambiar en las funciones anteriores para que funcionen en multigrafos?

Sí que lo tiene ya que aún existiendo varias ramas entre dos mismos nodos, el árbol abarcador mínimo continuará elaborando el árbol abarcador mínimo del grafo y descartando las ramas más costosas. Un posible cambio a aplicar para que esto funcionase sería a la hora de insertar las ramas en la cola de prioridad, escoger una única rama entre cada par de nodos (la que tenga menor coste) y después aplicar el mismo algoritmo de Kruskal.