Felipe Paladino

Reflexión de fin de unidad (UT7)

Al finalizar esta unidad temática, puedo libremente decir que es hasta ahora la que más me ha gustado. O al menos, a la que más tiempo le he dedicado.
Sin duda los grafos dirigidos son una herramienta sumamente útil para representar relaciones entre dos objetos. Particularmente, ya tenía algunos conocimientos del tópico, pues fue uno de los temas abarcados en Matemática Discreta I, el semestre pasado.

¿Qué cosas aprendí?
Bueno, si nos vamos a los temas internos de la unidad temática como tal, diría que una de las cosas que aprendimos fue el cómo representar grafos dirigidos, existiendo dos métodos principales: las listas de adyacencias y matrices de adyacencias. Encuentro particularmente útiles (sobre todo por su característica visual) a las matrices sobre las listas, pues es más sencillo, a mi parecer, comprender las relaciones entre vértices, mediante esa representación.

Es indispensable saber cómo recorrer estos dichosos grafos. Existen dos métodos, el dfs (depth-first search) y el bfs (breadth-first search). En el UT7, la principal herramienta fue el dfs, que esencialmente, es como un recorrido en preorden en un árbol, tal y como vimos en el UT4. El bfs por otro lado, vendría a ser como un recorrido por niveles en un árbol.

Los algoritmos principales trabajados en esta unidad fueron los de Dijkstra y Floyd. Dijkstra plantea un algoritmo para encontrar los caminos más cortos desde un vértice inicial a todos los demás nodos del grafo. Su complejidad temporal (Big O) es paupérrima, pues es de O(n**2), pero eso no niega su buena utilidad para resolver problemas.
Por otro lado Floyd, plantea un algoritmo para encontrar el camino de más corto entre todos los vértices del grafo. Vendría a ser como un Dijkstra para los n vértices del grafo. Esto implica que su Big O es absolutamente desastroso, misérrimo y/o pernicioso, pues es de O(n**3). Empero, al igual que Dijkstra, tiene mucha utilidad para resolver problemas.

Otros algoritmos que vimos son el de Warshall, que esencialmente determina si existe conexión entre cualquier par de vértices. Su implementación utiliza floyd(), por lo que su Big O es de O(n**3) también.

Felipe Paladino

Aunque no lo vimos de forma práctica en la unidad temática, se que existen formas de mejorar el Big O de Dijkstra utilizando colas de prioridad. Esto es algo que investigaré por fuera de la unidad para complementar lo ya visto.

Otros mecanismos vistos en la unidad fueron: Cerradura transitiva, cálculo de excentricidad y centro del grafo, y recorrido en orden topológico.
A recalcar, el recorrido en orden topológico solo puede realizarse si el grafo en cuestión es acíclico!
Que el grafo sea acíclico implica que no tiene ciclos, es decir, no existe un camino simple, de al menos longitud 3, tal que el mismo comience y termine en el mismo vértice.
La forma práctica de identificar si un grafo tiene consigo algún ciclo, es haciendo un recorrido en profundidad (dfs) y si se encuentra una arista de retroceso (va a un vértice ya visitado), entonces acabas de encontrar un ciclo. (Dios te bendiga).

En resumen, aunque haya sido una unidad con mucho trabajo que hacer, considero que ha sido en la que más he aprendido hasta la fecha. Un punto positivo, es que en la sección de grafos dirigidos, muchas de las cosas vistas en esta unidad se repiten.