**Andrés Manrique Ardila 201713672**

**Juan Diego Barrios 20163576**



* *1) Para este requerimiento tocaría recorrer todos los arcos pues estos tienen la información de llegada y salida de los servicios. Al recorrer guardamos esa información en una SeparateChainingHashST (según la modificación que le hemos hecho). Luego, Recorremos la tabla de hash que vendrían a ser los vértices para mirar la cantidad de servicios que hay en cada sublista. Por lo tanto, la complejidad vendría a ser A+V.*
* *2) Para realizar este requerimiento tenemos que recorrer todos los vértices y sus adyacentes. Recorrer todos los vértices nos cuesta V, pero como no sabemos la cantidad de vértices suponemos que en el peor caso es que están conectados con todos los demás vértices sin incluirse. De este peor caso obtenemos que la complejidad es de V\*(V-1).*
* *3) Usando el punto 2) solo nos quedaría recorrer toda la lista de servicios para saber el volumen en cada vértice. Dado esto la complejidad sería de V\*(V-1) + V.*
* *4) En este método tocaría aplicar Dijkstra lo cual nos costaría A Log(V).*
* *5) Para poder realizar este requerimiento tenemos que aplicar Dijkstra* para obtener el de costo mínimo y toca invertir la implementación de *Dijkstra de tal forma que nos devuelva el de costo mayor 2A Log(V).*
* *6) En este requerimiento tenemos que realizar Dijkstra de costo mínimo sobre peajes además realizar una modificación a este algoritmo de tal forma que los caminos de costos iguales los guarde en una lista es decir que deje de ser estable. Luego toca ordenar esa lista lo cual nos implicaría usar un algoritmo de ordenamiento. Debido a todas estas cosas que toca hacer y a que desconocemos la cantidad de caminos que puede retornar la complejidad en el peor caso de este algoritmo es de Log(V) \* (A +V).*



**Requerimientos**

**Complejidad**

**Proyecto 3-201810**

**Taxis de Chicago**

**Parte C:**