Documento de análisis reto 4

Daniel Arango Cruz – 202110646

Oscar Iván García – 201630048

Para realizar el análisis de este cuarto reto una empezaremos analizando el nivel de complejidad de cada requerimiento en orden a la complejidad de sus funciones. Adicionalmente, para poder hacer una comparativa efectiva de tiempos de ejecución tomaremos y compararemos los tiempos que toma cada requerimiento con un tamaño de datos distintos, siendo en primer lugar los archivos small, luego el de tipo “50pct” y finalmente los archivos large.

Requerimiento 1:

|  |  |  |
| --- | --- | --- |
| Funciones | Complejidad | Justificación |
| maxoutdigree(graph) |  | El algoritmo toma O(N) debido a que por cada vértice en el grafo toma su indegree y outdegree. |
| printReq1() |  | Tiene esta complejidad debido a que retira 5 elementos de la cola de prioridad para ser imprimidos. |

**Requerimiento 2:**

|  |  |  |
| --- | --- | --- |
| Funciones | Complejidad | Justificación |
| SCC() |  | Este algoritmo tiene esta complejidad puesto que el algoritmo de kosaraju realiza 3 recorridos de vértices y arcos distintos, primero invierte la lista, luego dfo, luego con el stak final dfo de nuevo en el grafo original. |
|  |  |  |

Requerimiento 3

|  |  |  |
| --- | --- | --- |
| Funciones | Complejidad | Justificación |
| BuscarHomonimos() |  | Puesto que usa funciones como get que llevan o(k) obtener un elemento |
| AereopuertosCercanos() |  | Este algoritmo tiene esta complejidad debido a que analiza todos los aeropuertos cercanos a la ciudad 1 y luego todos los cercanos a la ciudad 2 y calculando su distancia con Harvesine. |
| DijkstraReqcorrido() |  | Este algoritmo toma este tiempo debido a la naturaleza recursiva del algoritmo. |

Requerimiento 4

|  |  |  |
| --- | --- | --- |
| Funciones | Complejidad | Justificación |
| ArbolExpanciónMinima() |  | Este algoritmo toma esta complejidad debido a que por cada vertice dentro del resultado del mst se calcula el camino que representaría con respecto al origen, para de esta forma calcular cual seria el mas largo. |

Requerimiento 5

|  |  |  |
| --- | --- | --- |
| Funciones | Complejidad | Justificación |
| AfectedVertex() |  | Este algoritmo toma esta complejidad debido a que recorre todos los vértices para saber si alguno de estos apunta al vertice que se desea eliminar, a su vez recorre los que salen del vertice a eliminar y finalmente en el no dirigido los adyacentes al vertice a eliminar |