TPE 2: ANÁLISIS DE DATOS DE VUELOS

Programación de objetos distribuidos

Grupo 6

Diseño:

- Query 1: Movimientos por aeropuerto
  Se utilizó un trabajo *map/reduce* con los siguientes componentes:
  - Un *mapper* que emite un 1 por cada movimiento que tiene un *oaci* específico como origen de un despegue o como destino de un aterrizaje.
  - Un *reducer* que suma dichos valores para cada *oaci*.
  - Un *collator* para agregarle la denominación y pasarlo a formato de lista ordenada.

- Query 2: Top *N* aerolíneas según el porcentaje de movimientos de cabotaje
  Se utilizó un trabajo *map/reduce* con los siguientes componentes:
  - Un *mapper* que emite un 1 por cada vuelo de cabotaje.
  - Un *combiner* que emite la suma de dichos valores para cada aerolínea.
  - Un *reducer* que acumula los valores recibidos del *combiner*.
  - Un *collator* que calcula los porcentajes por cada aerolínea y genera el ranking.

- Query 3: Pares de aeropuertos que registran la misma cantidad de miles de movimientos
  Se utilizaron dos trabajos *map/reduce* donde el primero es igual al de la *query* 1 (pero sin el *collator*) y el segundo contiene los siguientes componentes:
  - Un *mapper* que emite la cantidad de miles de movimientos por cada *oaci* y descarta los que tienen menos de mil.
  - Un *reducer* que genera la lista de pares de aeropuertos que comparten la misma cantidad de miles de movimientos.
  - Un *collator* que descarta los aeropuertos que no comparten miles de movimientos con ningún otro aeropuerto y genera la lista ordenada.

- Query 4: Los *N* aeropuertos destino con mayor cantidad de movimientos de despegue que tienen como origen a un aeropuerto *oaci*
  Se utilizó un trabajo *map/reduce* con los siguientes componentes:
  - Un *mapper* que emite un 1 por cada movimiento que tiene como origen al *oaci* indicado.
  - Un *combiner* que emite la suma de dichos valores para cada aeropuerto.
  - Un *reducer* que acumula los valores recibidos del *combiner*.
  - Un *collator* que genera el ranking.

Análisis de tiempos de resolución:

Se realizaron distintas corridas de cada *query* para cada configuración de nodos (de 1 a 4 nodos inclusive). A continuación, se presenta el promedio de los resultados obtenidos:

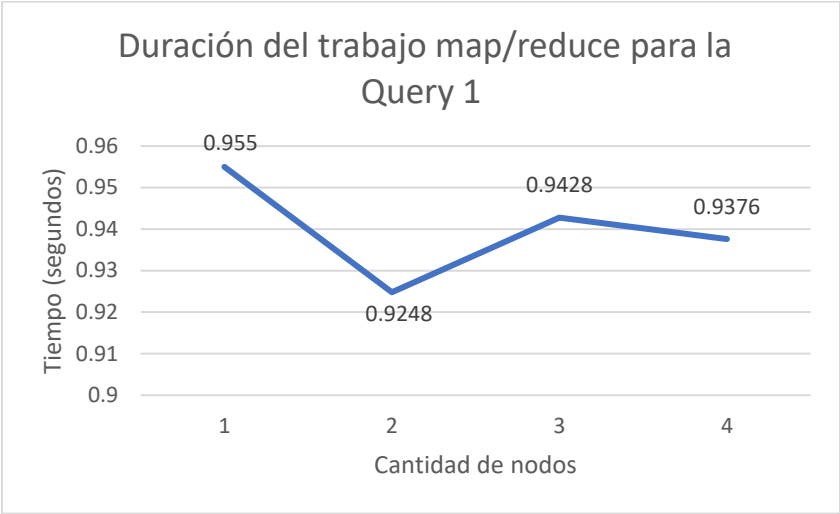

Figura 1: Duración del trabajo map/reduce según la cantidad de nodos para la Query 1

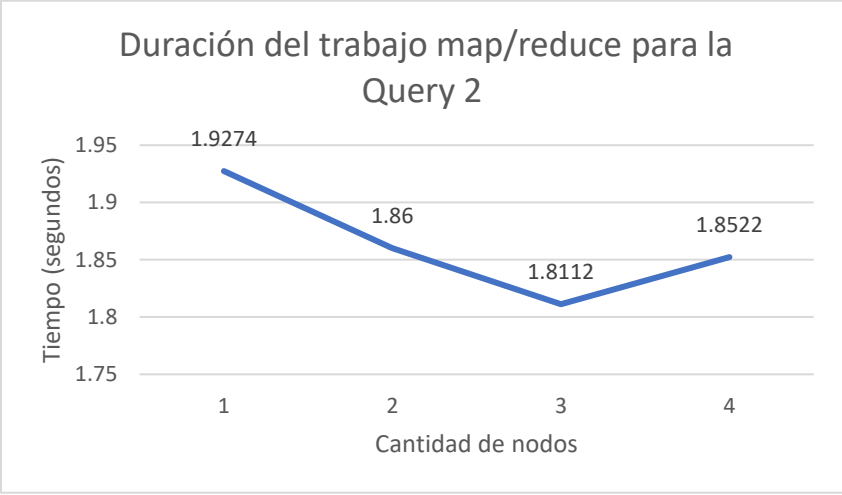

Figura 2: Duración del trabajo map/reduce según la cantidad de nodos para la Query 2

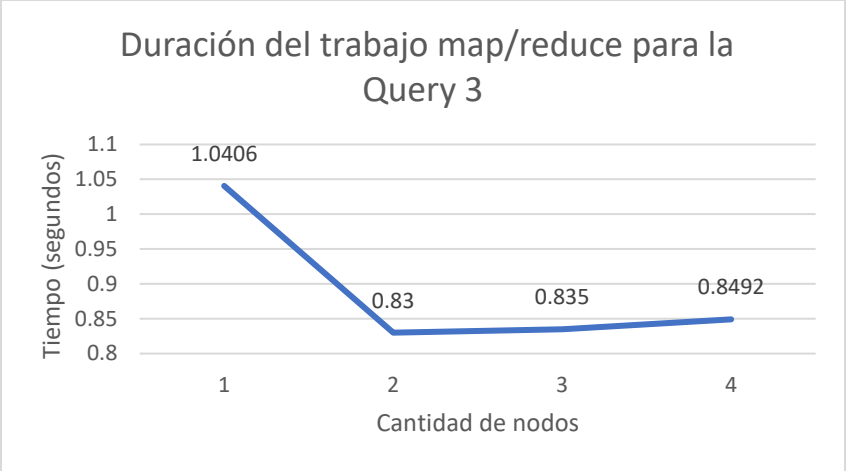

Figura 3: Duración del trabajo map/reduce según la cantidad de nodos para la Query 3

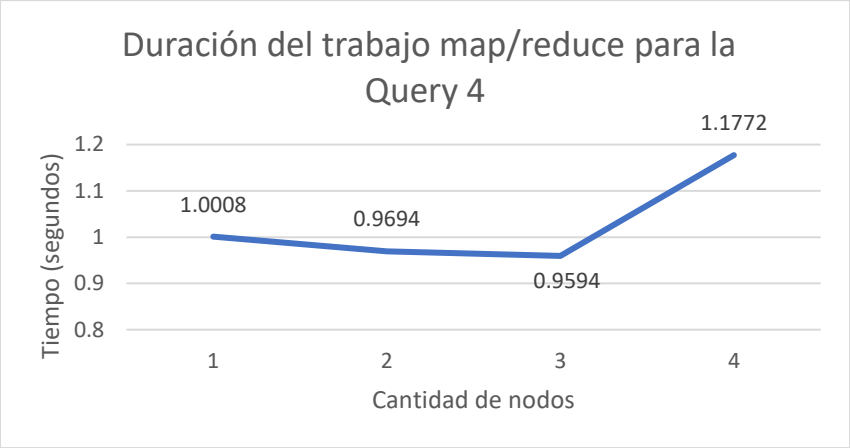

Figura 4: Duración del trabajo map/reduce según la cantidad de nodos para la Query 4

Como se puede ver, no observamos grandes diferencias cambiando la cantidad de nodos. Creemos que esto puede ser porque la escala del problema no ameritaba el uso de múltiples nodos. Otra razón puede ser que las tareas no estén distribuidas de forma eficiente entre los nodos.