Group: Group Therapy
Casper Kristiansson
Nicole Wijkman

RQ 5: Data-Intensive Computing

1. Assume we have two types of resources in the system, i.e., CPU and Memory. In total we have 28 CPU and 56GB RAM (e.g., 1 CPU = 2 GB). There are two users in the systems. User 1 needs h2CP U, 2GBi per task, and user 2 needs h1CP U, 4GBi per task. How do you share the resources fairly among these two users, considering (i) the asset fairness, and (ii) DRF.

(i) First, considering the asset fairness, one has to divide the resources so that each user gets a fair share of memory and CPU. Since it is known that 1 CPU = 2 GB of RAM, the total resource units available for memory and CPU can be calculated.

$$Total\ CPU\ Units\ =\ 28\ CPU\ =\ 28\ *\ 1\ =\ 28\ Units$$
$$Total\ Memory\ Units\ =\ 56\ GB\ =\ 56\ *\ 2\ =\ 112\ Units$$

Then, one can calculate the resource units required by each user for memory and CPU and then allocate that number of units to that user. This follows asset fairness since each user gets their fair share of both resources.

User 1:

$$CPU\ Units\ required\ =\ 2\ CPU\ =\ 2\ *\ 1\ =\ 2\ Units$$
$$Memory\ Units\ required\ =\ 2\ GB\ =\ 2\ *\ 2\ =\ 4\ Units$$

User 2:

$$CPU\ Units\ required\ =\ 1\ CPU\ =\ 1\ *\ 1\ =\ 1\ Unit$$
$$Memory\ Units\ required\ =\ 4\ GB\ =\ 4\ *\ 2\ =\ 8\ Units$$

(ii) Next, DRF has to be considered. DRF Considers the dominant resource for each user, then allocates the resources according to this. For User 1, the dominant resource can be seen to be memory, since they need 4 memory units compared to the 2 CPU units. The same goes for User 2. It can be seen that the dominant resource is memory for them as well, since they need 8 memory units compared to 1 CPU Unit. To allocate based on DRF, the dominant resource is allocated first and then the other resources are allocated in proportion.

$$4\ Memory\ Units\ allocated\ to\ User\ 1\ (Their\ dominant\ resource)$$
$$The\ remaining\ 108\ Memory\ Units\ are\ allocated\ to\ User\ 2\ in\ proportion\ to\ CPU$$
$$User\ 2\ thus\ gets\ (1/28)\ *\ 108 \approx 3.86\ Memory\ Units$$

$$2\ CPU\ Units\ are\ allocated\ to\ User\ 1\ (Their\ non\ -\ dominant\ resource)$$
$$The\ remaining\ 26\ CPU\ Units\ are\ allocated\ to\ User\ 2\ in\ proportion\ to\ memory$$
$$User\ 2\ gets\ (8/56)\ *\ 26 \approx 3.71\ CPU\ Units$$

2. What are the similarities and differences among Mesos, YARN, and Borg?

Mesos, YARN, and Borg are all cluster management systems designed for to manage and schedule resources in large-scale distributed computing environments. Their aim is to maximize resource utilization, allowing multiple frameworks to share the same cluster. They all also have multiple framework support, supporting the execution of multiple frameworks or applications on a single cluster. This enables both flexibility and resource sharing. The systems also provide mechanisms for allocations of CPU and memory.

The three systems have multiple differences. They differ in origin, resource abstraction, architecture, and openness. Mesos offers a more fine-grained resource allocation and follows a two-level scheduling model. YARN was initially tailored for Hadoop and attracts resources into containers, while also following a two-level architecture. Borg was initially developed by Google for internal use and thus has more abstract resource model as well as a different architectural approach.

3. What are the differences between Warehouse and Datalake? What is Lakehouse?

A Data Warehouse is a centralized data storage system that originated in the 1980s. It was designed primarily to handle structured data as well as well-defined schemas. It is purpose-built for SQL analytics and Business Intelligence (BI) tasks, with its focus laying on data quality, consistency, and high-performance querying.

A Data Lake emerged later, in the 2010s. It provides a more flexible and cost-effective approach to data storage when compared to a Data Warehouse. It handles a variety of data types, including both unstructured and semi-structured data. It is also very adaptable, since it allows for schema definition after data storage.

A Lakehouse, as suggested by the name, can be described as a modern data storage concept, combining the advantages of Data Warehouses and Data Lakes. It does this by integrating Data Warehouse management and performance features, but with Data Lake's open and flexible data storage approach. This fusion results in “the best of both worlds” by giving high-performance analytics alongside the cost-effectivity of raw data storage.

4. Briefly explain how Delta Lake handle concurrent writing on the same file

Delta Lake is designed so that all transactions are achieved using a concept called “optimistic concurrency control”, using optimistic concurrency protocols against the object store. Delta Lake first identifies what changes have been made by each write operation, then merges these changes together, conflict-free. This process ensures data consistency and integrity, without any conflicts or corruption problems when multiple writes happen concurrently.