Levi Davis, ljd3frf

Big Data Systems

Homework 1 Report

For this homework we explore Hadoop and Apache Spark. First, we went through the setup process, and this was by far the hardest task. At each step errors were encountered, and I was pushed to the limits of my troubleshooting abilities. Once everything was configured correctly, we wrote PySpark programs to explore using the Hadoop/Spark ecosystem. We implemented the PageRank algorithm and made adjustments in the code to examine how they effected the program execution, specifically runtime.

In part 3 task 1, we ran the PageRank program with the two ec2 instances and the runtime is showed below.

Completed Applications (1)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20230222074007-0000	PySpark PageRank program	4	4.0 GiB		2023/02/22 07:40:07	ubuntu	FINISHED	9.3 min

In part3 task 2, we implemented partitions to achieve faster runtime.

Completed Applications (5)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20230222083500-0004	PySpark PageRank program with partitions	4	4.0 GiB		2023/02/22 08:35:00	ubuntu	FINISHED	6.2 min

Even with an initial task that took 9.3 minutes, by introducing partitions the runtime was reduced to 6 minutes, about 33% less time.

For task 3 I killed the vm2 worker at 410/700 steps completed. With partitions each worker's read and write shuffle rates range from about 300-700 MiB. Without partitions these are 0. As seen below the runtime was greatly affected and took much longer.

Completed Applications (6)

Application ID	Name	Cores	Memory per Executor	Resources Per Executor	Submitted Time	User	State	Duration
app-20230222084443-0005	PySpark PageRank program with partitions and worker killed halfway	2	4.0 GiB		2023/02/22 08:44:43	ubuntu	FINISHED	11 min

Each worker has a set number of cores, which in this case is two each. Partitions can be introduced to split the process into smaller chunks for more efficient parallel processing.

From these experiments we can infer the impacts of number of workers and partitions. In task 1, without partitions, there are two active tasks. In task 2, with partitions, there are up to 6 active tasks at one time, which explains the greatly increase performance compared to task 1. For task 3, which initially had at least 4 active tasks, killing a worker decreased it to two active tasks, explaining the great increase in runtime post-kill.