CS322:Big Data

Final Class Project Report

Project (FPL Analytics / YACS coding): YACS CODING Date: 06/12/2020

SNo	Name	SRN	Class/Section
1	MAKNOOR SHALINI	PES1201800253	A
2	SHREE AKSHAYA A T	PES1201801466	D
3	SINDHU CH	PES1201801526	D
4	SHRUTHI SHANKARAN	PES1201801677	D

Introduction

As we know that Big Data consists of multiple jobs from different applications. These tasks are too large to even consider running on a single machine. In order to run these tasks we run on clusters of interconnected machines. These machines get the tasks allocated based on the scheduling algorithm. Running the task on different machines will reduce the overall execution time of the job ,since the job contains many tasks that are distributed among different machines which is scheduled based on scheduling algorithms.

The Aim of the project : Simulate the working of a centralized scheduling framework, Implementation of 3 different scheduling algorithms and calculation of the metrics and analyzing the result.

Related work

Class lectures and notes, websites like Stack Overflow, Geeks for Geeks

Design

Implementation of master.py

Initially 2 threads are created, one for getting the request for a job and the other for getting the update from the worker. The request for a job is sent using socket connection to the port number 5000. Details of job arrival time is stored in a file. Each map task and reduce task in a job is launched and assigned to a worker (whose slot is reduced by 1) according to the scheduling algorithm chosen by the user. The second thread is for receiving updates from workers through port 5001. Once an update of task completion is received, the slot is increased by 1.

Implementation of worker.py

A thread to receive task from Master through its respective port is implemented. The arrival time is noted and the task is executed (simulated) through a thread. Once execution is done, the completion time of the task is noted and an update is sent to the Master.

Implementation of analysis.py

Files containing time details for jobs and tasks are read. Mean and median are calculated accordingly and graphs are plotted.

Results

Considering 15 job requests given to the master, based on the 3 scheduling algorithm the mean time of the task and job completion and also the median time of the task and job completion is calculated for each scheduling algorithms:

For **Random Scheduling Algorithm**:

The mean time of task completion is 2.57 s and job completion is 9.27 s

The median time of task completion is 3.002 s and job completion is 9.86 s

For **Least Loaded Algorithm**:

The mean time of task completion is 2.40 s and job completion is 9.85 s

The median time of task completion is 2.006 s and job completion is 10.03 s

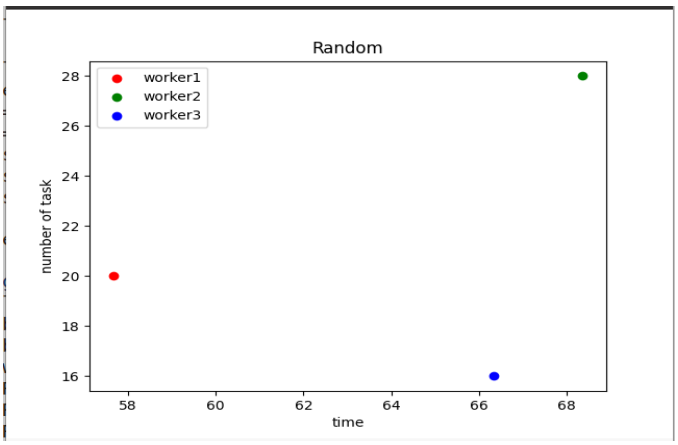
For **Round Robin Algorithm**:

The mean time of task completion is 2.24 s and job completion is 7.03 s

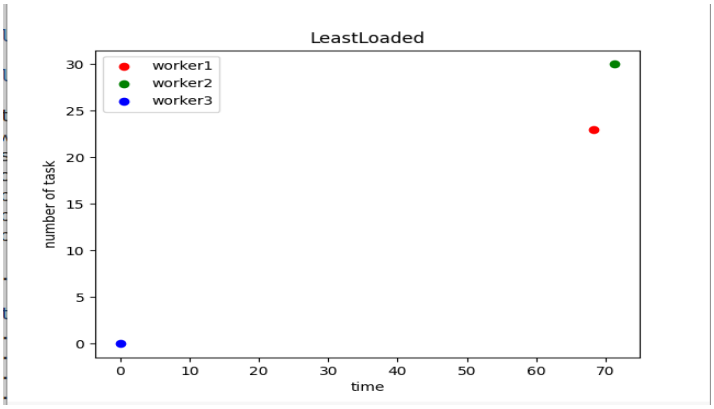
The median time of task completion is 2.004 s and job completion is 7.99s

Plotting of graph between number of task assigned in each worker vs time taken, for each scheduling algorithm

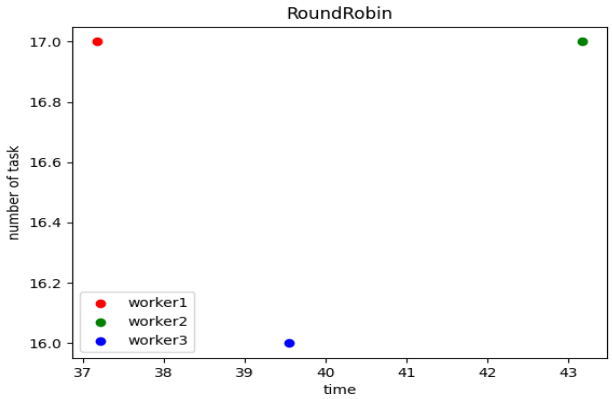
1) Random Scheduling Algorithm



2) Least Loaded Scheduling Algorithm



3) Round Robin Scheduling Algorithm



Problems

1) In socket connection between the master and worker to update the completion of task from the worker through the port number 5001, which gave connection refused error when included threading.
2) Getting accurate results using analysis.py, for which we have changed the format of data written to files for more efficient access.
3) Map and Reduce Dependency

Conclusion

We gained insight into how to implement the process of distributing big data workload among clusters.

From the Results of each scheduling algorithm we inferred that the Round Robin is the most efficient algorithm for scheduling the tasks on each machine. From the Result section we can see that the average time taken for a task to complete and time taken for

the job to complete is less compared to the other scheduling algorithms . And also from the plot between number tasks in each worker vs the time taken we can infer that the load on each worker is balanced unlike that of other scheduling algorithms.

EVALUATIONS:

SNo	Name	SRN	Contribution (Individual)
1	MAKNOOR SHALINI	PES1201800253	Master.py, Random scheduling,Analysis.py
2	SHREE AKSHAYA A T	PES1201801466	Master.py(connection initialization), LeastLoaded scheduling,Worker.py,Analysis .py
3	CH SINDHU	PES1201801526	master.py worker.py
4	SHRUTHI SHANKARAN	PES1201801677	RoundRobin Scheduling, Worker.py, Analysis.py

(Leave this for the faculty)

Date	Evaluator	Comments	Score

CHECKLIST:

SNo	Item	Status
1.	Source code documented	Done

2.	Source code uploaded to GitHub – (access link for the same, to be added in status)	Done https://github.com/shruthi-sh/bigdataproject.git
3.	Instructions for building and running the code. Your code must be usable out of the box. PTO	***In separate terminals:*** python3 master.py path to config/config.json <algo_name> python3 worker.py 4000 1 python3 worker.py 4001 2 python3 worker.py 4002 3 python3 requests.py <num> ***For analysis.py file:*** python3 analysis.py