PFLOCK Report

Andres Calderon

University of California, Riverside

January 10, 2020

- ▶ Apache Spark divides the work into a number of Stages, each one is also divide into a number of Tasks.
- ► Each Task evaluates the data from a particular Partition and it is sent to an Executor for evaluation.
- ► Tracking the Tasks will give us a notion of the Partitions and Executors performance...

- SparkListener class allows us to monitor Taks metrics. We can capture info about:
 - executor: The id of the executor where the Task was evaluated.
 - duration: Execution time of the task.
 - recordsRead/Written: Total number of records read or written.
 - bytesRead/Written: Total number of bytes read or written.
 - ▶ shuffleRead/Written: Total number of records read or written from the shuffle by this task.

- ▶ Running experiments with Epsilon=30, Mu=3 and Delta=3.
- ▶ Collecting Task metrics. Focus on Time-by-time implementation...

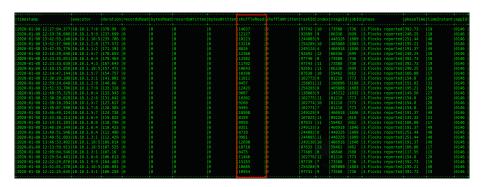


Figure: Top 30 longest tasks

- ▶ Running experiments with Epsilon=30, Mu=3 and Delta=3.
- ▶ Collecting Task metrics. Focus on Time by time implementation...

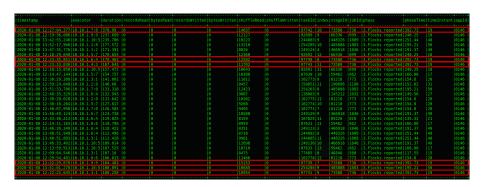


Figure: Top 30 longest tasks

Stage analysis...

- SparkListener also provide info about each Stage. Some interesting metrics are:
 - ▶ numTasks: The number of tasks in which the Stage is divided.
 - name: The Spark function which invoke the Stage and its line of code.
 - details: The stack trace when the Stage was called.



Figure: Top 20 longest stages

Stage analysis...

- ▶ FF.scala:191 makes a call to the redundant flocks pruning routine.
- ▶ Recently I modify this routine to be call during the join between time instants. I will double-check the code.
- ➤ Currently I have extracted some data for testing (time instant 19) and making some improvements.

What is next...

- Fixing the shuffling overheard during the prunning routine.
- ▶ Performing similar analysis for the Time-window implementation.