Flink BenchMark

1. **Introduction**

* Apache Flink cluster benchmark set for server operators which is able to measure the performance of an Apache Flink cluster. This enables spotlighting irregularities in software or hardware behaviour.
* When operating an Apache Flink cluster, many aspects on different levels of hardware and software define the overall performance.
* The microbenchmark suite is built from an operators perspective, opposed to related work, which measures the performance by real world use cases mainly from a developers view.

1. **Microbenchmark** **Suite**

In order to define our microbenchmark suite, we first define the requirements of this suite. Afterwards, we describe the workloads we defined in order to benchmark the performance of Flink clusters and how we ensure statistic rigor of the measurements.

1. **Requirements**

* The microbenchmark suite should be usable by operators in order to check whether a clusters performance fits its hardwares performance and whether the performance remains constant over time. We want to detect unusual or varying computing, RAM performance. Besides a one-time usage, the microbenchmarks should be executable as short benchmarks on a regular basis in order to detect anomalies occurring due to temporary reasons (e.g: non-functional cooling or network load interference)
* Furthermore, the microbenchmark suite should assure statistic rigor, i.e. that repeating the same measurements produces values we consider to stem from the same distribution.

1. **Workloads**

* In order to check whether computing performance, RAM performance or HDFS access performance differs, we create a workload for each performance measure
* Each workload is a Flink job which is executed on a cluster. For every job the execution time separately. Furthermore, each workload has a variable size and parallelism. The parallelism defines on how many workers the tasks of a job are distributed. The following benchmarks are implemented:
  + In order to benchmark **computing performance**, *size ∗ factor* additions are executed. These are grouped into *N* map jobs. Those jobs contain additions. Therefore, the network traffic is not increased with increasing benchmark size
  + In order to benchmark **RAM performance**, the benchmark reserves *size ∗ factor* bytes of RAM. In every job, a random count of indices of the array is initialized with a random value. Afterwards, the same count of random indices are added. The sum of the random indices is the result of the job.

1. **Executions**

*Source available:* [*https://git.sc.uni-leipzig.de/flink-benchmark/flink-benchmark*](https://git.sc.uni-leipzig.de/flink-benchmark/flink-benchmark)

* **Steps**
* Clone source available
* In build.gradle file:
  + Update version to your flink version
  + Add dependency flink-clients\_${scala.binary.version} for DataSet API
  + Modify 'Main-Class': 'de.flinkbench.StartBenchmark'
* This projects contains benchmarks:
  + The **Addition-Benchmark**, started by de.flinkbench.StartBenchmark -workload add, which adds numbers 10^9 \* size times.
  + The **RAM-Benchmark**, started by

de.flinkbench.StartBenchmark -workload ram, which reserves size ram, fills parts of it with numbers and adds them.