Summary of “Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing”

Hou, Jue

[jue.hou@helsinki.fi](mailto:jue.hou@helsinki.fi)

In the article “Resilient Distributed Datasets: A Fault-Tolerant Abstraction for In-Memory Cluster Computing”, authors introduced a memory abstraction for distributed cluster computation which is called resilient distributed datasets (RDD). Unlike other memory abstraction, RDD actually will be kept persistent in the memory after reading and make it read-only. The aim of doing so is, as authors claim, to improve the performance and make data reusable, which also benefits not only the batch stream application but also the applications which have to turn to intermediate result for multiple times such as data mining or machine learning. By using the mechanism of lineage, system can easily know the parents of the selected RDD. This also means that RDD supports fault-tolerance well. Of cause, lineage benefits the RDD in many other aspects. To use the RDD, users have to write a driver program. This program will define RDD and invoke the operations on it. Also, driver will connect to the workers on different server and assign the task to them. Workers will read data from files, transfer it as what is invoked and persist it in memory.

Overall, this article explains the benefit of why using the RDD and how the RDD works on the Spark systems. Along with many examples, I think some basic idea is quite straightforward, though the section of interpreter integration is still a little bit difficult for me.