Rhenan Abachioni Gomes SP3069184

Resumo – MC-CD02: Algoritmos e Modelos de Programação para Big Data

Introdução a Big Data e Apache Spark

Uma execução de sucesso no big data deve ter 3 qualidades: Um código bem construído, uma boa configuração do ambiente de execução e o casamento entre os requisitos da aplicação e a infraestrutura disponível.

A video aula faz uma introdução à big data e ao Apache Spark – Sistema aberto desenvolvido originariamente em UC, Berkeley e AmpLab, sendo uma extensão do modelo MapReduce para algoritmos iterativos (machine learning e grafos) e mineração de dados (R, Python) e possuindo melhor desempenho para casos especiais como processamento in-memory e grafo de tarefas, em comparação à outra solução da Apache para Big Data – O Hadoop. Possui APIs para as linguagens Java, Scala, Python e R.

Conforme foi evoluindo, foi sendo aplicado de diferentes formas, como por exemplo em queries SQL, sendo que para essas diferentes aplicações foi construído um framework.

Seu ambiente de execução consiste em um esquema de mestre e escravo, sendo um driver program junto com um gerenciador de cluster, responsável pelo gerenciamento dos worker nodes.

Suas aplicações são executadas a partir do Driver, o qual executa diversos Jobs, sendo divididos em diversos estágios, que por sua vez são divididos em tarefas. Para a execução, é utilizado um SparkContext (podendo existir apenas um).

Quanto a sua alocação, um job é iniciado pelo nó driver, o qual distribui tarefas pelos workers para que então sejam retornados os resultados das mesmas ao driver. Faz uso de RDD (Resilient Distributed Datasets) – Estrutura de dado armazenada em memória – para maior eficiência, possuindo definições dos tipos de dados estruturados nesses datasets. O RDD está contido no SparkContext.

Para o compartilhamento de dados entre diferentes jobs, torna-se necessário o armazenamento dos dados em disco Para isso é utilizado cache (ou persist). O Spark tentará utilizar esse cache em memória e caso não seja possível, armazenará em disco.

A execução do spark gera, para cada task, uma nova partição (executado por um núcleo). O resultado final pode ser coletado através da operação collect(), coletando as informações das partições e enviado para o SparkContext no driver.