Apache Spark 2 for Beginners

1장. 스파크 기초

데이터는 기업의 가장 중요한 자산 중 하나이다. 기업에서 수집되고 사용되는 데이터의 규모는 상상 이상으로 커지고 있다. 데이터가 소비되는 속도, 사용되는 데이터 유형의 다양성, 그리고 처리되고 저장되는 데이터의 양은 매순간 그 기록이 갱신되고 있다. 데이터의 양이 기가 바이트에서 테라 바이트로, 더 나아가 페타 바이트로 증가되는 이런 현상은 요즘 작은 규모의 조직에서 조차 매우 일상적인 것이 되었다. 이와 같은 이유로 정적인 데이터 뿐만 아니라 동적인 데이터까지 처리할 수 있는 가용성에 대한 요구도 또한 점점 커지고 있다.

한 기업을 예로 들어보자. 기업의 성공은 리더에 의해 내려지는 의사결정에 좌우되며, 올바른 의사결정을 내리기 위해서는 좋은 데이터와 데이터 처리를 통해 생성된 정보가 뒷받침되어야 한다. 이 사실은 올바른 의사결정을 위해 어떻게 시기 적절하게 그리고 비용 효과적인 방법으로 데이터를 처리할 것인가 하는 문제를 제기한다. 데이터 처리 기법은 컴퓨터 초창기 때부터 발전을 거듭해 왔다. 셀 수 없이 많은 데이터 처리용 제품들과 프레임워크들이 시장에 나타났다가 사라졌다. 이러한 대부분의 제품 및 프레임워크들은 사실 일반적인 목적으로 만들어진 것이 아니었다. 대부분의 조직들은 자신의 데이터 처리 요구사항을 충족시키기 위한 자기만의 맞춤형 애플리케이션에 의존했다.

대규모 인터넷 애플리케이션—일반적으로 **사물인터넷(Internet of Things, IOT)** 애플리케이션으로 알려진—들은 다양한 유형으로 빠르게 소비되는 엄청난 양의 데이터를 처리하기 위한 오픈 프레임워크가 있어야 한다는 공통적인 필요성을 예고했다. 대규모 웹사이트, 미디어 스트리밍 애플리케이션, 그리고 대량의 배치 처리가 필요한 기업들이 이 필요성에 의미를 더했다. 또한 인터넷의 성장과 함께 오픈 소스 커뮤니티들이 무섭게 성장했고, 명망있는 소프트웨어 회사들의 지원을 받아 높은 품질의 소프트웨어를 공급하고 있다. 엄청나게 많은 회사들이 오픈 소스 소프트웨어를 사용하거나 이들 제품군에 자신의 제품을 배포하면서 창업을 했다.

기술적인 관점에서, 데이터 처리에 대한 요구는 커다란 도전을 받았었다. 데이터의 양은 1대의 머신을 넘어서 수많은 머신들로 이루어진 클러스터로 흘러 들어갔다. 정체된 1개 CPU의 처리 능력 때문에 요즘 컴퓨터들은 멀티 코어라고 불리듯이 여러 CPU를 묶어서 처리하고 있다. 멀티 코어 컴퓨터의 모든 프로세서를 사용하도록 설계되거나 개발되지 않은 애플리케이션들은 일반적인 현대의 컴퓨터에서 CPU의 처리능력을 낭비하고 있는 것이다.

[정보]

이 책에서 언급되는 노드(node), 호스트(host), 머신(machine) 등의 용어는 단독으로 또는 클러스터에서 운용되고 있는 하나의 컴퓨터를 가리킨다.

이런 상황에서 이상적인 데이터 처리 프레임워크가 가져야 할 조건은 무엇일까?

* 컴퓨터의 클러스터에 분산되어 있는 데이터 블럭들을 처리할 수 있어야 한다.
* 대량의 데이터 처리 업무 하나를 여러 개의 작업으로 분할하여 병렬로 처리함으로써 처리 시간을 크게 절감할 수 있어야 한다.
* 컴퓨터 내의 모든 CPU 또는 프로세서의 처리 능력을 사용할 수 있어야 한다.
* 클러스터 내의 가용한 모든 컴퓨터를 사용할 수 있어야 한다.
* 평범하고 일반적인 하드웨어에서도 잘 운용되어야 한다.

이런 모든 요구조건을 만족시키는 오픈 소스 데이터 처리 프레임워크는 두 가지가 있다. 그 첫번째는 아파치 하둡(Apache Hadoop)이고, 두번째는 아파치 스파크(Apache Spark)이다.

이번 장에서는 다음과 같은 항목을 다룰 것이다.

* Apache Hadoop
* Apache Spark
* Spark 2.0 설치

Apache Hadoop 살펴보기

Apache Hadoop은 컴퓨터의 클러스터에 분산 데이터를 저장하고, 여러 컴퓨터의 클러스터들에 퍼져 있는 분산 데이터를 처리하기 위해 설계된 오픈 소스 프레임워크이다. 이 프레임워크는 데이터 저장을 위한 분산 파일시스템인 **하둡 분산 파일 시스템(Hadoop Distributed File System, HDFS)**와 데이터 처리 프레임워크인 맵리듀스(MapReduce)로 이루어져 있다. HDFS는 구글의 연구 논문인 *“구글 파일 시스템(The Google File System)”*에 영감을 받아 만들어졌고, 맵리듀스도 역시 구글 논문인 *“맵리듀스: 대규모 클러스터에서의 단순화된 데이터 처리(MapReduce: Simplified Data Processing on Large Clusters)”*를 기반으로 한다.

하둡은 실제로 데이터 처리를 위한 대규모의 하둡 클러스터를 구축하는 방식으로 조직들에 수용되었다. 하둡 맵리듀스 버전 1(MRv1)에서 하둡 맵리듀스 버전 2(MRv2)로 바뀌면서 엄청난 발전이 있었다. 순수한 데이터 처리 관점에서 봤을 때, MRv1은 핵심 구성요소인 HDFS와 맵리듀스로 이루어져 있다. 이 맵리듀스 프레임워크 위에 일반적으로 SQL-on-Hadoop 애플리케이션이라고 불리는 하이브(Hive)나 피그(Pig)같은 많은 애플리케이션들이 올려졌다. 이런 유형의 애플리케이션들은 비록 아파치 프로젝트에서 분리되었지만, 나름대로의 큰 가치를 제공하고 있다.

하둡 생태계에서 맵리듀스와는 다른 타입으로 운용되는 프레임워크인 Yet Another Resource Negotiator (YARN) 프로젝트가 등장했다. 간략하게 말하면 YARN은 HDFS 위에서, 그리고 맵리듀스 밑에서 돌아가는 분산처리환경이다. 사용자들은 YARN과 HDFS에서 분산 데이터 저장소와 하둡 생태계의 데이터 처리 능력을 사용하며 실행되는 자신만의 애플리케이션을 작성할 수 있다. 다른 말로 하면, 새롭게 정비된 맵리듀스 버전 2(MRv2)는 HDFS와 YARN 상위에 있는 애플리케이션 프레임워크 중에 하나가 된 것이다.

p.12