2. 배경지식 & 선행연구

2.1 Alluxio 구조

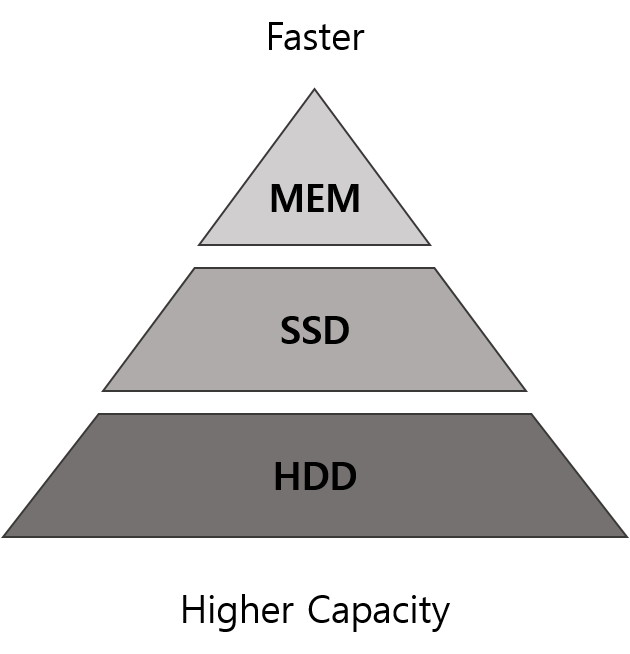
Tachyon으로도 잘 알려진 Alluxio는 메모리 속도의 가상 분산 스토리지 시스템이다. Spark와 마찬가지로 Alluxio는 메모리 IO가 디스크 IO보다 훨씬 빠르다는 사실을 이용해 큰 데이터 환경에서 매우 중요한 데이터 접근 시간을 향상시켜준다. Alluxio는 저장될 수 있는 모든 데이터에 쉽게 접근할 수 있도록 가상 이름 지정 기능을 제공한다. 파일 시스템에서 파일이 어떤 디스크(혹은 호스트)에 있는지 몰라도 파일에 대한 경로만 안다면 모든 파일에 쉽게 접근할 수 있도록 한다. Alluxio는 모든 애플리케이션이 메모리 속도로 모든 스토리지 시스템의 모든 데이터와 상호 작용이 가능하도록 해준다.

Alluxio는 Spark, Hadoop등의 데이터간의 계산을 수행하는 빅데이터 프레임워크와 HDFS, S3, GlusterFS 등의 분산 파일 시스템간의 중간 역할을 한다. Alluxio의 구조는 마스터 노드와 워커 노드들로 이루어진다. 마스터 노드는 통합 이름 공간(파일 시스템 트리)를 유지하고 관리하며 사용 가능한 워커를 추적하는 역할을 한다. 워커는 로컬 자원(RAM, SSD, 디스크)를 관리하고 데이터 청크를 저장하며 물리적 스토리지의 데이터를 가져오는 역할을 한다. 또한 클라이언트의 요청에 응답하고 마스터 노드에게 heartbeat를 보고한다. 이런 구조를 통해 시스템에 워커를 추가함으로써 쉽게 스토리지 용량을 늘릴 수 있다는 장점이 있다. 또한 Alluxio는 애플리케이션과 동일한 클러스터에 위치하여 데이터 지역성이 가능하다는 장점이 있다.

Alluxio는 어플리케이션과 물리적인 스토리지 사이에 위치하므로 물리적인 스토리지에서 데이터를 가져올 때 데이터를 자체 메모리에 저장하여 빠른 속도를 낼 수 있도록 한다.

2.2 Alluxio Tiered Storage

Alluxio는 계층적인 스토리지를 제공하여 메모리뿐만이 아닌 다른 스토리지 타입도 커버할 수 있도록 한다. 현재 Alluxio는 그림과 같은 Mem, SSD, HDD의 3가지 유형의 스토리지를 제공한다. Alluxio의 높은 계층일수록 데이터를 읽어오는 속도가 빠르나, 용량이 적다. 계층화된 스토리지에 저장된 데이터는 Alluxio에서 관리하며 직접 접근은 불가능하다.



2.3. C. Chen, T. Hsia, Y. Huang, and S. Kuo, “Scheduling-Aware Data Prefetching for Data Processing Services in Cloud”, 2017 IEEE International Conference on Advanced Information Networking and Applications, 2017

Scheduling-Aware Data Prefetching (SADP) 기법을 제안해 데이터 prefetching과 데이터 eviction을 포함하여 대용량 클라우드 데이터 센터의 데이터 prefetching문제점을 해결하였다.

2.4. O. Yildiz, A. Zhuo, and S. Ibrahim, “Eley: On the Effectiveness of Burst Buffers for Big Data Processing in HPC Systems”, 2017 IEEE International Conference on Cluster Computing, 2017

HPC(High Performance Computing) 어플리케이션의 효율을 보장하면서 빅데이터 어플리케이션의 효율을 높이기 위해서 Eley라는 burst buffer solution을 제시하였다. Eley는 두 어플리케이션에서 input 데이터를 읽어오는 시간을 줄이기 위해 데이터를 prefetch하여 computing node의 근처에 저장하도록 하였다.

2.5. Y. Chen, H. Zhu, H. Jin, and X.-H. Sun, “Algorithm-level Feedback-controlled Adaptive data prefetcher: Accelerating data access for high-performance processors”, Parallel Comput., vol. 38, no. 10-11, pp. 553-551, Oct. 2012.

Algorithm-level Feedback-controlled Adaptive (AFA) data prefetcher를 제안하여 기존의 prefetching이 어플리케이션의 접근 패턴이 변할 때 효율적이지 못한 점을 해결하였다.

2.6. Y. Chen, S. Byna, and X. H. Sun, “Data access history cache and associated data prefetching mechanisms”, in Proceedings of the2007 ACM/IEEE Conference on Supercomputing, 2007. SC ’07, 2007, pp. 1-12

새로운 cache 구조인 Data Access History Cache(DAHC)를 제안하고 관련된 prefetching 기술을 연구했다. DAHC는 data prefetching의 이점을 얻는 현실적인 접근을 제시하고 관련된 prefetching 기술은 기존의 접근보다 더 효율적임을 증명했다.