스파크 클러스터의 SLA 기반 오토스케일링

오유리 ○, 최지은, 김윤희

숙명여자대학교 컴퓨터과학과

{yoori0203, jechoi1205, yulan}@sookmyung.ac.kr

요 약

인터넷 및 모바일 기기의 발달로 인해 전세계적인 사용자들이 생성하는 데이터 양이 기하급수적으로 증가하고 있다. 이러한 빅데이터 시대를 반영하여 대량의 데이터를 분석하는 일은 필수적이다. 스파크는 분산처리 프레임워크로 실시간 스트리밍 데이터를 처리가능하며, 디스크 I/O 가 아닌 인메모리 컴퓨팅을 지원하여 기존에 많이 사용되었던 하둡보다 빠른 실행이 가능해졌다. 또한 클라우드 기술은 언제 어디서나 원하는 시점에 원하는 만큼의 자원을 사용할 수 있는 환경이 제공한다. 스파크를 이용한 빅데이터 분석 처리 시, 자원 요구량을 예측하기 어렵기 때문에 유연한 자원사용이 가능한 클라우드 기술을 접목한다면 한층 더 효율적인 작업 실행이 가능하다. 이때, SLA 기반의 적절한 자원 오토스케일링은 자원을 낭비하지 않고 능률적인 실행을 가능하도록 하며 가장 잘 구성되어야 하는 부분이다. 본 연구에서는 스파크 클러스터의 SLA 기반 오토스케일링을 제안한다. 이는 어플리케이션의 SLA 를 만족하는 실행을 보장하기 위하여 실시간 모니터링을 통해 오토스케일링을 실행한다. 특정 어플리케이션이 SLA 를 만족하지 못하는 상황이 예측되었을 때, 다른 어플리케이션의 실행상황을 분석하여 SLA 를 만족하는 선 내에서 자원을 협상(negotiate)한다. 이로써 모든 어플리케이션이 SLA 를 충족하는 효율적인 자원 활용이 가능하도록 한다.

1. 서론

인터넷과 모바일 기기의 발달로 인터넷에서 생성되는 데이터의 양이 기하급수적으로 증가하고 있는 추세이다. 이에 따라 초기에 등장한 빅데이터 분산처리 프레임워크인 하둡[1]은 맵리듀스 알고리즘을 이용하여 단일 머신으로 처리가 불가능한 작업을 수 십대의 머신에서 나누어 처리하는 기능을 지원하였다. 그러나 하둡[1]은 데이터를 디스크에 읽고 쓰기 때문에 I/O 병목현상, 실시간 처리가 불가능하다는 단점이 드러났다. 이러한 상황을 반영하여 최근에 등장한 분산프레임워크인 스파크[2]는 인메모리 컴퓨팅을 이용하며, 실시간 스트리밍 데이터 처리를 지원한다. 빅데이터 분석은 일괄 처리로 끝나기 어렵고, 연속적인 여러 단계를 거쳐 장시간 동안 처리되는 경우가 많다. 이는 다양한 작업이 실시간으로 실행되는 시스템에서는 동적인 자원 관리가 필수적이다.

필요로 하는 자원이 런타임에 동적으로 달라짐에 따라, 대용량의 자원을 원하는 시점에 필요한 만큼 빌려 쓸 수 있는 클라우드 기술은 이러한 문제점에 해결책으로 사용될 수 있다. 클라우드 기술은 가상화를 통해 자원의 물리적인 제약을 받지 않고 유연한 자원의 사용이 가능하므로 스파크의 유동적인 자원 요구량을 충족시킬 수 있는 환경을 제공한다. 이 때, 다양한 SLA(Service Level Agreement; e.g. 비용, 데드라인) 및 자원 요구사항과 예측할 수 없는 작업 제출 시점 등 예측하기 어려운 상황에 대해 적절한 자원 오토스케일링은 필수적이다.

본 논문에서는 클라우드 컴퓨팅 환경에서 스파크 클러스터의 자원을 효율적으로 활용하기 위한 오토스케일링을 제안한다. 제안하는 오토스케일링은 데드라인 SLA 를 만족시키는 것을 목표로 한다. 따라서 데드라인을 만족하는 범위에서 효율적인 오토스케일링이 가능하다.

본 논문의 구성은 다음과 같다. 1 장의 서론에 이어 2 장에서는 관련 연구들을 살펴보고, 3 장에서는 본 논문에서 제안하는 SLA 기반 오토스케일링 기법을 적용한 프레임워크 구조에 대하여 설명한다. 4 장에서는 SLA 기반 오토스케일링 기법에 대해 설명하고 마지막으로 5 장에서 결론을 맺는다.

2. 관련연구

1) 스파크

기존에 분산 프레임워크로 활발하게 연구되었던 하둡[1] 프레임워크에 이어 스파크[2] 프레임워크가 등장하였다. 스파크[2]는 하둡과 유사하게 작업을 여러 노드에 분산처리하며 큰 특징으로 인메모리 컴퓨팅을 지원한다. 데이터 처리를 위해 디스크 I/O 를 실행하였던 하둡[1]과 달리 새로운 개념의 데이터 구조인 RDD(Resilient Distributed Dataset)를 사용한

메모리에서의 컴퓨팅을 지원한다. 이는 반복하는 실행 또는 스트리밍 데이터 처리에 용이하여 결과적으로 더욱 빠른 실행결과를 도출할 수 있다.

이러한 스파크의 효율적인 실행을 위한 연구로는 [3-5]가 있다. [3]은 온라인 병렬 분석 서비스를 제공시, 효율적이고 신뢰성 있는 서비스를 제공하기 위한 엔트로피 개념 기반의 스케쥴링 기법을 제안한다. 엔트로피 개념을 시스템의 무질서도를 측정하는데 사용되며 이는 신뢰성을 판단하는 기준이 된다. 또한 동적인 코어 성능을 측정하여 자원을 스케쥴링하는데 이용한다. [4]는 빅데이터를 처리할 때, 병렬접근의 최적화에 관한 연구이다. 이를 위하여 병렬데이터 요청을 저장 서버와 매칭하는 방법을 제시하며, 대화형 데이터 접근 방식에서 성능 향상을 위한 계획을 제시한다. [5]는 스파크를 이용한 날씨 이벤트 감지 및 추적 시스템을 제안한다. 이를 위하여 새로운 데이터 구조인 sRDD(scientific RDD)를 이용하여 분석을 진행하고, 단계간의 데이터 재사용을 특징으로 언급한다.

2) 오토스케일링 기법

사용자의 요구사항을 충족시키며, 효율적인 자원 사용을 위하여 자원에 대한 오토스케일링에 관한 연구는 [6-8]가 있다. [6]은 하이브리드 클라우드 환경에서의 오토스케일링 기법을 제안한다. 과학응용을 실행하기 위하여 사설 및 공용 클라우드 환경에서 데드라인을 만족하는 비용 효율적인 스케쥴링 방법을 제안한다. [7]은 응용의 패턴을 고려한 하이브리드 클라우드 환경에서 오토스케일링 기법을 제안한다. 응용의 특성 및 작업의 데드라인을 고려하여 효율적인 자원 활용을 증명하였다. [8]은 하둡 클러스터의 오토스케일링 기법을 제안한다. 실행시간에 대한 일반적인 모델을 제시하고, 각 워크로드에 맞게 튜닝 후, 예상 실행시간을 공식화 하였다. 이를 기반으로 오토스케일링 기법을 제안하였다.

스파크 또는 오토스케일링 기법에 관한 여러 관련연구들이 있었으나, 스파크 클러스터에 대한 오토스케일링을 진행한 연구는 없었다. 또한 스파크의 런타임 도중 다양한 자원 및 데드라인의 요구사항과 예측할 수 없는 작업 제출 시점 등의 상황에 대한 클라우드 환경의 오토스케일링은 필수적이다. 따라서 본 연구에서는 스파크 클러스터의 SLA 기반의 오토스케일링을 제안한다.

3. 오토스케일링 프레임워크 구조

본 논문에서 제안하는 오토스케일링 프레임워크 구조는 그림 1 과 같다. 사용자가 SLA(데드라인, 비용)를 제출하고 스파크 어플리케이션을 수행하고자 하는 경우, 스파크 어플리케이션은 오픈스택 자원을 활용하여 작업을 실행한다. 이때, 자원 및 작업에 대한 관리는 'YARN scheduler' 가 담당한다. 'Yarn scheduler'는 동적인 자원관리를 위해 VM 모니터링과 제어를 담당한다. 또한 작업실행을 위해 작업 모니터링과 실행기능을 갖는다.

'Auto-Scaling Service'는 오토스케일링 프레임워크의 핵심적인 서비스로 런타임 동안 모니터링을 통해 오토스케일링을 진행한다. 사용자가 제출한 SLA를 충족시키기 위하여 실시간 작업에 대한 모니터링을 하며, 오토스케일링이 필요한 시점에 자원에 대한 스케쥴링을 진행한다. 'Metadata Management Service'는 작업, 자원, VM 에 대한 요약된 정보를 가지고 있으며 이는 작업 스케쥴링 시, SLA 만족여부를 확인하는 용도로 사용된다.

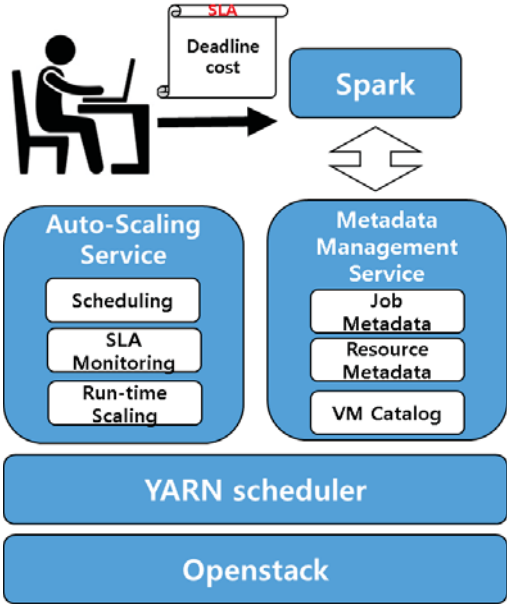

그림 1. 오토스케일링 프레임워크 구조

4. SLA 기반 오토스케일링

본 논문에서는 SLA 기반의 오토스케일링 기법을 제안한다. 스파크 어플리케이션과 같은 빅데이터 분석 응용은 동적으로 작업의 부하가 변화하여 작업 수행에 대한 예측 및 자원 사용에 대한 예측이 어렵다. 따라서 주기적인 모니터링을 통해 실시간으로 변하는 환경을 확인하고 사용자의 요구사항을 충족시키는 것이 중요하다.

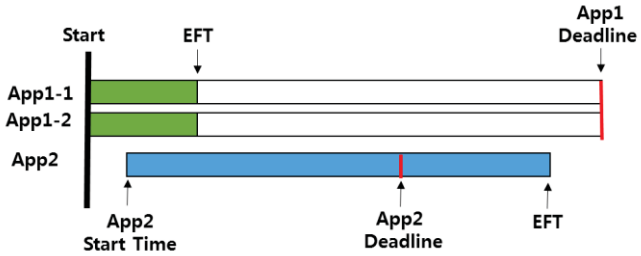

그림 2. 초기 스케쥴링 결과

그림 2 는 초기 스케쥴링 결과이다. 어플리케이션

1 이 제출되고 실행되는 도중 어플리케이션 2 가 제출되어 실행되는 상황이다. 각 어플리케이션의 색깔 막대는 예상 수행시간을 의미하며 EFT(Estimated Finish Time) 은 예상 종료시간을 의미한다. 또한 빨간 선은 사용자가 요구하는 데드라인을 의미한다.

어플리케이션 1 이 제출되었을 때에는 모든 자원을 사용 가능한 상태이다. 따라서, 어플리케이션 1 에 제출된 데드라인보다 상당히 빠른 시간 내에 작업을 끝낼 수 있는 상황이며, 이는 빠른 예상종료시간을 통해 알 수 있다. 어플리케이션 1 이 수행되는 도중에 어플리케이션 2 가 제출되었다. 시스템은 어플리케이션 2 가 제출될 것을 예상하지 못했기 때문에 어플리케이션 2 를 위한 충분한 자원을 할당하지 못했다. 어플리케이션 2 의 데드라인보다 더 많은 시간 작업을 해야 작업이 끝나는 상황이 되어, 이는 시간상으로 데드라인보다 예상종료시간이 뒤에 있음을 통해 확인할 수 있다. 이러한 경우, 자원의 균등한 분배가 일어나지 않아 먼저 수행을 시작한 어플리케이션이 과도하게 빠른 시간 내에 수행되고, 나중에 제출된 작업은 데드라인을 만족시킬 수 없는 상황이 일어난다.

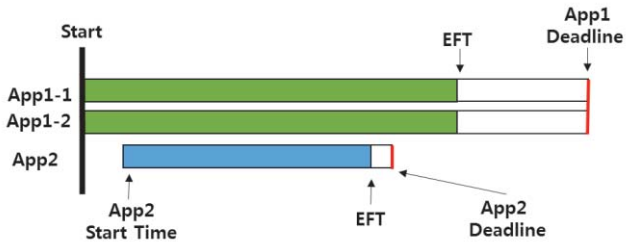

그림 3. 스케쥴링 결과

앞서 초기 스케쥴링을 이용하여 작업을 수행하는 경우, 오토스케일링을 통해 어플리케이션 1 과 어플리케이션 2 모두의 각각 데드라인을 만족하는 작업수행이 가능하다. 주기적인 모니터링을 통해 모든 어플리케이션이 사용자가 요구한 데드라인을 만족하는지 확인하고 데드라인을 만족하지 못하는 어플리케이션이 있는 경우, 해당 어플리케이션을 위하여 다른 어플리케이션의 실행 상황을 분석하여 자원을 여유 있게 사용하는 어플리케이션의 자원을 수거하여 데드라인을 만족하지 못하는 어플리케이션에 배정한다. 위의 예에서 어플리케이션 2 의 예상종료시간과 데드라인을 비교하였을 때, 데드라인을 만족하지 못하는 상황을 인지하고 오토스케일링을 실행한다. 어플리케이션 1 이 데드라인에 비해 예상종료시간이 빨리 끝나므로, 어플리케이션 1 의 자원을 일부 수거하여 어플리케이션 2 에 배정한다. 어플리케이션 1 은 활용하는 자원이 줄어들었기 때문에 이전보다 예상수행시간은 늘어났지만 여전히 예상종료시간은 데드라인을 만족한다. 어플리케이션 2 의 경우는 추가적인 자원을 할당 받았기 때문에 예상종료시간이 이전보다 줄어들어 데드라인 내에 작업을 종료할 수 있게 되었다. 어플리케이션 1 은 일부 수거된 자원으로 인해 예상종료시간이 늘어났지만 여전히 데드라인을 만족하는 실행이 가능하다. 위 오토스케일링 시나리오를 통해 어플리케이션 1 과 어플리케이션 2 은 효율적인 자원 협상으로 인해 모두 데드라인을 만족하는 작업수행이 가능함을 알 수 있다.

5. 결론 및 향후 연구

본 논문에서는 스파크 클러스터에서 효율적인 자원 활용을 위한 SLA 기반 오토스케일링을 제안하였다. 기하급수적으로 늘어난 데이터로 인해 스트림 데이터 등 빅데이터 분석에 적합한 스파크 클러스터를 활용하는 추세이며 제안하는 오토스케일링 프레임워크에서 스파크 클러스터는 클라우드 컴퓨팅의 가상화 기술을 이용하여 동적으로 원하는 시점에 원하는 만큼의 자원활용이 가능하도록 한다. 또한 오토스케일링 기법 적용 시나리오를 제시한다. 시나리오에서는 기존 실행하고 있던 어플리케이션이 SLA 를 만족할 수 있도록 실행되고 있는 상태에서 또 다른 새로운 어플리케이션이 제출되었을 때, 이후에 제출된 어플리케이션이 충분한 자원을 할당받지 못해 SLA(데드라인)을 만족하지 못하는 경우에 대해 언급한다. 이때, 기존에 실행하던 어플리케이션의 자원의 일부를 수거하고 이후에 제출된 어플리케이션을 위해 재할당한다면, 두 어플리케이션 모두 데드라인을 만족하는 실행이 가능함을 보였다. 이는 효율적인 자원활용을 위하여 오토스케일링의 필요성을 나타내었다.

향후에는 SLA 정책에 따른 효율적인 오토스케일링 알고리즘을 제안하고 실험을 통하여 스파크 클러스터에서 오토스케일링 기법의 효과를 증명할 예정이다.

6. 참고문헌

[1] Apache hadoop. [Online]. Available: http://hadoop.apache.org/.

[2] Apache Spark: Lightning-fast cluster computing, “Apache spark,” 2015. [Online]. Available: https://spark.apache.org/

[3] H. Chen and F. Z. Wang, "Spark on entropy: A reliable & efficient scheduler for low-latency parallel jobs in heterogeneous cloud," Local Computer Networks Conference Workshops (LCN Workshops), 2015 IEEE 40th, Clearwater Beach, FL, 2015, pp. 708-713.

[4] J. Yin and J. Wang, "Optimize Parallel Data Access in Big Data Processing," Cluster, Cloud and Grid Computing (CCGrid), 2015 15th IEEE/ACM International Symposium on, Shenzhen, 2015, pp. 721-724.

[5] R. Palamuttam et al., "SciSpark: Applying in-memory distributed computing to weather event detection and tracking," Big Data (Big Data), 2015 IEEE International Conference on, Santa Clara, CA, 2015, pp. 2020-2026.

[6] 강혜정, 고정인, 김윤희, "과학 계산 응용 실행을 위한 하이브리드 클라우드에서의 SLA 기반 VM 오토-스케일링 기법", 정보과학회논문지: 시스템 및 이론 제 40 권 제 6 호, pp. 266-273, 2013 년 12 월
[7] 안윤선, 정솔, 김윤희, "하이브리드 클라우드상의 과학응용을 위한 가상 자원 오토스케일링 기법", 정보과학회논문지: 시스템 및 이론 제 41 권 제 4 호, pp. 266-273, 2014 년 8 월
[8] Gandhi, Anshul, et al. "Autoscaling for Hadoop Clusters."

7. 기타

"이 논문은 2015 년도 정부(미래창조과학부)의 재원으로 한국연구재단-차세대정보・컴퓨팅기술개발사업의 지원을 받아 수행된 연구임(2015M 3C 4A7065646)."