빅데이터 처리 효율 향상을 위한 데이터 압축 방식에 따른 맵리듀스 성능 비교 분석

*김원집[1], 김하윤[2], 이협건[3], 김영운[4]
한국폴리텍대학 서울강서캠퍼스 빅데이터과
kbg5174@naver.com[1], let_hykim@naver.com[2], hglee67@kopo.ac.kr[3],
luckkim@kopo.ac.kr[4]

Comparative Analysis of Map Reduction Performance by Data Compression Method for Improving Big Data Processing Efficiency

*Wonjib Kim[1], Hayoon Kim[2], HyeopGeon Lee[3], Young-Woon Kim[4]
[1]Department of Big Data, Seoul Gangseo Campus
of Korea Polytechnics College.

요 약

하둡은 대용량 데이터 처리를 위한 대표적인 오픈소스 프레임워크이다. 특히 하둡의 맵리듀스 프로그래밍 모델은 분산 환경에서 병렬 처리를 가능하게 한다. 데이터를 처리하기 위한 환경에서는 데이터의 양의 방대해짐에 따라 저장 공간의 한계와 데이터의 전송 속도의 병목현상이 발생이 빈번하다. 이를 해결하기 위한 방법 중 하나는 데이터 압축 기술의 활용이다. 대표적으로 Gzip, Bzip2, Zstd 등이 있으며, 각 방식은 압축률, CPU 사용량, 메모리 사용 측면에서 상이한 특성을 보인다. 이에 본 논문은 하둡의 맵리듀스 프로그래밍 모델에서 대표적인 압축 방식을 구분하여 압축 방식에 따른 성능 평가를 수행한다. 수행 결과는 빅데이터 개발자들에게 맵리듀스 환경의 압축의 효율적 선택을 위한 가이드라인을 제시할 수 있다.

1. 서론

인터넷 서비스의 발전은 데이터의 양이 기하급수적으로 증가를 야기한다. 이에 따라 기업과 연구에서는 대용량 데이터를 처리하기 위한 기술과 방법론의 필요성이 요구된다.

하둡(Hadoop)은 대용량 데이터 처리를 위한 대표적인 오픈소스 프레임워크이다. 특히 하둡의 맵리듀스(MapReduce)프로그래밍 모델은 분산 환경에서 병렬 처리를 가능하게 한다. 맵리듀스는 Map 단계에서 분할하고 처리한 후, Reduce 단계에서 결과를 집계하여 최종 결과를 도출한다[1]. 이러한 처리 방식은 대규모 데이터 세트를 효율적으로 처리할 수 있도록 해주며, 하둡의 HDFS(Hadoop Distributed File System)과 결합하여 높은 확장성과 내결함성을 제공한다. 이러한 장점 덕분에 맵리듀스는 빅데이터 처리 표준 도구로 기업과 연구에서 널리 활용한다.

데이터를 처리하기 위한 환경에서는 데이터의 양의 방대해짐에 따라 저장 공간의 한계와 데이터의 전송 속도의 병목현상이 발생이 빈번하다. 이를 해결하기 위한 방법 중 하나는 데이터 압축 기술의 활용이다. 대표적으로 압축방식은 Gzip, Bzip2, Zstd 등이 있으며, 압축률, CPU 사용량 등 다양한 측면에서 상이한 특성을 보인다. 따라서 압축 방식을 선택할 때는 데이터의 특성과 시스템 자원을 고려해야 한다.

이에 본 논문은 하둡의 맵리듀스 프로그래밍 모델에서 대표적인 압축 방식인 Gzip, Bzip2, Zstd 구분하여 압축 방식에 따른 성능 평가를 수행한다.

2. 관련 연구

본 장은 대표적인 압축 방식인 Gzip, Bzip2, Zstd를 설명한다.

2.1. GZIP(GNU ZIP)

Gzip은 초기 UNIX 시스템에서 사용되던 압축 프로그램인 Compress를 대체하기 위해 개발된 압축방식이다. Gzip은 DEFLATE 알고리즘을 사용하여 단일 파일과 데이터 스트림의 무손실 압축을 지원한다[2]. Gzip은 ZIP 파일과 유사한 알고리즘을 사용하면서, 단일 파일을 압축하는 알고리즘이다. 따라서 Gzip은 여러 파일이나 디렉터리를 단일 파일로 압축하기 위해서는 보통 tar 명령어와 결합하여 사용한다. 'tar.gz'형식 파일은 ZIP 파일보다 더 작은 용량으로 압축한다. tar 방식은 서로 다른 파일끼리의 중

복되는 부분을 압축시킬 수 있기 때문이다.

2.2. Bzip2

Bzip2는 BWT(Burrows-Wheeler Transform)와 Huffman 인코딩을 사용하여 높은 압축률을 제공하는 압축 알고리즘이다. BWT는 문자열을 재배치해 반복되는 패턴이 더 잘 나타나도록 변환한다. BWT 수행 이후, Huffman 인코딩은 이진 트리를 생성한다. 생성된 이진 트리는 빈번하게 등장하는 패턴을 짧은 코드로 대체하여 추가적인 압축을 수행한다. Bzip2는 블록 기반 압축을 사용하여 데이터를 여러 블록으로 나누어 처리한다. 각 블록은 독립적으로 압축되고 해제한다. 이로 인해 Bzip2는 병렬 처리[3]에서 높은 성능을 발휘한다.

2.3. Zstd (Zstandard)

Zstd는 페이스북에서 개발한 고성능 압축 방식으로, LZ77과 엔트로피 코딩 기법을 사용한다. Zstd는 데이터의 특성에 따라 다양한 압축 수준을 제공하여 맞춤형 압축이 가능하다. 특히 Zstd는 실시간 데이터 처리에 효과적이다[4]. 또한 Zstd는 멀티스레딩을 지원하여 데이터의 압축과 해제가 매우 빠르다.

3. 성능평가

본 장에서는 2장에서 설명한 Gzip, Bzip2, Zstd 압축 방식에 따른 성능 평가를 수행한다. [표 1] 성능 평가를 위한 주요 환경 구성을 나타낸다.

[표 1] 주요 환경 구성

호스트 시스템	
CPU	Intel Xeon W-1250P, 6코어, 4.10GHz
메모리	64 GB
가상화 소프트웨어	VMware Workstation 17
가상머신 구성 (Master, Slave1, Slave2)	
운영체제	Rocky Linux 9.4
할당 코어 수	1
할당 메모리	8 GB
네트워크	NAT
하둡 버전	3 .36 Version
데이터	아파치 웹 로그 2GB
HDFS 블록크기	64 MB

성능평가는 다른 압축 방식으로 압축된 데이터 세트를 사용하여, 아파치 웹 로그에 대한 IP 빈도수 를 분석한다. 또한 성능평가는 Map과 Reduce 단계의 출력에도 동일한 압축 방식을 적용한다.

[표 2]는 각각의 압축방식에 대한 맵리듀스 성능 평가 결과를 보여준다.

[표 2] 압축 방식에 맵리듀스 성능평가 결과

압축방식에 대한 성능평가				
압축방식 / 비교항목	None	Gzip	Bzip2	Zstd
실행된 Map 수	32	1	2	1
Map 처리시간	1276.7 s	70.2 s	308 s	70.3s
Reduce 처리 시간	23.6 s	6.2 s	10.2 s	5.6 s
CPU 사용 시간	201.4 s	87.9 s	304.2 s	85 s
HDFS 읽은 MB	2147 MB	229 MB	135 MB	197 MB
HDFS 쓴 MB	5.3 MB	1.2 MB	1.2 MB	1.4 MB
Spilled 레코드	38106256	57159384	57159384	57159384
가비지 컬렉션 시간	143.34 s	0.63 s	1.2 s	0.62 s
물리적 메모리 사용	0.6 GB	0.62 GB	0.64 GB	0.64 GB
가상 메모리 사용량	2.5 GB	2.5 GB	2.5GB	2.6 GB

CPU 사용시간은 Zstd 압축 방식이 가장 효율적이다. 또한 Zstd와 Gzip는 맵리듀스 실행시간 및 가비지 컬렉션에 소요되는 시간이 적다. 또한 Map 수가 1개로 분산 처리가 불가능하다. Bzip2은 블록 기반 압축을 사용하여 Map 수가 2개로 각 블록에 대한 분산 처리가 가능하다. 또한 HDFS I/O에서 좋은 성능 보인다.

4. 결론

하둡의 맵리듀스 프로그래밍 모델에서는 전반적으로 Zstd 방식이 좋은 성능 결과를 보인다. Bzip2 압축 방식은 다른 압축 방식에 비해 가장 높은 압축률과 분산 처리를 제공한다. 그러나 Bzip2는 CPU 처리 시간이 원본 데이터보다 더 많이 소요되는 비효율적인 결과를 보인다. 그러므로 Bzip2는 데이터를 장기간 보관하거나 네트워크 및 HDFS I/O를 최소화해야 할 때 적합하다. 본 연구에서는 아파치 웹 로그에 대한 빈도수 검사 작업을 통해 압축 방식에 대한 맵리듀스 성능을 분석한다.

향후, 다양한 맵리듀스 프로그래밍 모델에 대한 추가적인 검증이 필요하다.

참고문헌

[1] Apache Hadoop, https://hadoop.apache.org, accessed Sept. 27, 2024

[2] Deutsch, Peter. "GZIP file format specification version 4.3" (1996)

[3] Bzip2, "Bzip2 Homepage," http://www.bzip.org, accessed Sept. 27, 2024

[4] Wassenberg, J., & Fiedorowicz, P. (2020). Zstandard Compression and the 'application/zstd'