한국정보전자통신기술학회논문지(jkiiect)'18-10, Vol.11 No.5
논문 18-11-05-593 http://dx.doi.org/10.17661/jkiiect.2018.11.5.593

로그 분석 처리율 향상을 위한 맵리듀스 기반 분할 빅데이터 분석 기법

이협건*, 김영운, 박지용, 이진우

MapReduce-Based Partitioner Big Data Analysis Scheme for Processing Rate of Log Analysis

Hyeopgeon Lee, Young-Woon Kim, Jiyong Park, Jin-Woo Lee

요 약 인터넷과 스마트기기의 발달로 인해 소셜미디어 등 다양한 미디어의 접근의 용이해짐에 따라 많은 양의 빅데이터들이 생성되고 있다. 특히 다양한 인터넷 서비스를 제공하는 기업들은 고객 성향 및 패턴, 보안성 강화를 위해 맵리듀스 기반 빅데이터 분석 기법들을 활용하여 빅데이터 분석하고 있다. 그러나 맵리듀스는 리듀스 단계에서 생성되는 리듀서 객체의 수를 한 개로 정의하고 있어, 빅데이터 분석할 때 처리될 많은 데이터들이 하나의 리듀서 객체에 집중된다. 이로 인해 리듀서 객체는 병목현상이 발생으로 빅데이터 분석 처리율이 감소한다. 이에 본 논문에서는 로그 분석 처리율 향상을 위한 맵리듀스 기반 분할 빅데이터 분석 기법을 제안한다. 제안한 기법은 리듀서 분할 단계와 분석 결과 병합 단계로 구분하며 리듀서 객체의 수를 유동적으로 생성하여 병목현상을 감소시켜 빅데이터 처리율을 향상시킨다.

Abstract Owing to the advancement of Internet and smart devices, access to various media such as social media became easy; thus, a large amount of big data is being produced. Particularly, the companies that provide various Internet services are analyzing the big data by using the MapReduce-based big data analysis techniques to investigate the customer preferences and patterns and strengthen the security. However, with MapReduce, when the big data is analyzed by defining the number of reducer objects generated in the reduce stage as one, the processing rate of big data analysis decreases. Therefore, in this paper, a MapReduce-based split big data analysis method is proposed to improve the log analysis processing rate. The proposed method separates the reducer partitioning stage and the analysis result combining stage and improves the big data processing rate by decreasing the bottleneck phenomenon by generating the number of reducer objects dynamically.

Key Words : Big Data, Hadoop, Hadoop Distributed File System, MapReduce, Log Analysis

1. 서론

빅데이터 분석에 사용되는 데이터들의 양은 인터넷과 스마트기기의 발달로 인해 소셜미디어 등 다양한 미디어의 접근의 용이해짐에 따라 매년 기하급수적으로 증가하고 있다. 특히 다양한 서비스를 제공하는 기업들의 시스템에 저장되는 방대한 양의 로그들은 고객 성향 및 패턴, 보안성 강화 등 중요한 데이터로 사용되

This research was supported by the Industrial Strategic Technology Development Program (10069080, Music contents production and Individual initiative education service development through Artificial Intelligence) funded By the Ministry of Trade, Industry & Energy(MOTIE, Korea)
*Department of Data Analysis, Seoul Gangseo Campus of Korea Polytech
Department of Applied Music, Hanyang University
Contents Development Office, Sysone
**Corresponding Author : Department of Data Analysis, Seoul Gangseo Campus of Korea Polytech(hglee67@kopo.ac.kr)
Received September 07, 2018 Revised September 12, 2018 Accepted October 01, 2018

고 있다[1, 2].

이러한 로그 분석을 위해 사용되는 빅데이터 분석 기법인 하둡 에코시스템과 다양한 빅데이터 플랫폼들은 대부분 맵리듀스를 기반으로 빅데이터 분석을 수행하고 있다[3].

그러나 맵리듀스는 리듀스 단계에서 생성되는 리듀서 객체를 한 개로 정의하고 있다. 이로 인해 맵리듀스를 통한 빅데이터 분석은 Shuffle and Sort 단계를 통해 전달되는 많은 데이터들이 하나의 리듀서 객체에 집중적으로 모이게 되며, 병목 현상이 발생하게 된다.

이에 본 논문에서는 로그 분석 처리율 향상을 위한 맵리듀스 기반 분할 빅데이터 분석 기법을 제안한다. 제안한 기법은 리듀서 분할 단계와 분석 결과 병합 단계로 구분하여 구현한다. 리듀서 분할 단계는 로그 분석 유형에 맞춰 리듀서 객체 수를 유동적으로 생성하여 병목 현상을 감소시킨다. 분석 결과 병합 단계는 리듀서 객체들로부터 분할 저장되는 분석 결과들을 하나의 분석 결과로 병합한다. 이로 인해 제안한 기법은 로그 분석 처리율을 향상시킨다.

본 논문의 구성은 다음과 같다. 2장에서는 맵리듀스 기반 빅데이터 분석 기법을 살펴보고, 요구사항을 분석한다. 3장에서는 본 논문에서 제안한 로그 분석 처리율 향상을 위한 맵리듀스 기반 분할 빅데이터 분석 기법을 제시한다. 4장에서는 제안한 기법의 성능을 분석하고, 마지막 5장에서는 결론 및 향후 연구 과제를 제시한다.

2. 관련 연구

본 장에서는 맵리듀스 기반 빅데이터 분석 기법을 살펴보고, 제안한 빅데이터 분석 기법을 구현하기 위해 요구사항을 분석한다.

2.1 맵리듀스 기반 빅데이터 분석 기법

맵리듀스[4]는 하둡의 필수 핵심 기술 중 하나로 방대한 양의 데이터들을 분산되어 설치된 노드들을 활용하여 처리 및 분석을 수행한다. 이 맵리듀스는 빅데이터 처리 및 분석 기술로 가장 보편화된 기술로 하둡 분산 파일 시스템(Hadoop Distributed File System, HDFS)을 기반으로 동작된다. 따라서 맵리듀스 기반 빅데이터 분석 기법은 기본적으로 하둡 분산 파일 시스템과 그 상위 스택인 맵리듀스를 활용하여 빅데이터를 분석한다. 〈그림 1〉은 맵리듀스와 하둡 분산 파일 시스템과의 구조를 나타낸다.

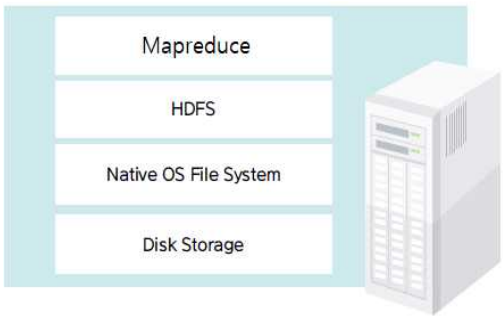

그림 1. 맵리듀스와 하둡 분산 파일 시스템의 구조
Fig. 1. The structure of MapReduce and HDFS

맵리듀스 기반 빅데이터 분석 기법[5, 6]은 하둡 분산 파일 시스템에 기본적으로 제공하는 리소스 매니저(Resource Manager), 애플리케이션 마스터(Application Master), 노드 매니저(Node Manager)와 잡 히스토리(Job History) 데몬을 활용하여 빅데이터를 분석한다.

리소스 매니저는 하둡 분산 파일 시스템의 마스터 노드에 설치되며, 하나의 하둡 클러스터마다 존재한다. 리소스 매니저의 역할은 슬레이브 노드에 존재하는 애플리케이션 마스터를 실행 및 슬레이브 노드의 자원을 할당한다.

애플리케이션 마스터는 슬레이브 노드에 설치되며, 하나의 맵리듀스 잡마다 실행된다. 애플리케이션 마스터의 역할은 자원을 요청하고, 맵과 리듀스 작업을 개별적으로 관리한다.

노드 매니저는 슬레이브 노드마다 설치된다. 노드 매니저의 역할은 각 슬레이브 노드들의 자원을 관리한다.

잡 히스토리는 마스터 노드와 슬레이브 노드가 아닌 별도의 노드에 설치되며, 하나의 하둡 클러스터마다 존재한다. 잡 히스토리의 역할은 잡 실행 내역을 메타데이터에 기록 및 관리한다.

〈그림 2〉는 하둡 분산 파일 시스템의 데몬을 활용한 맵리듀스 기반 빅데이터 분석 과정을 나타낸다.

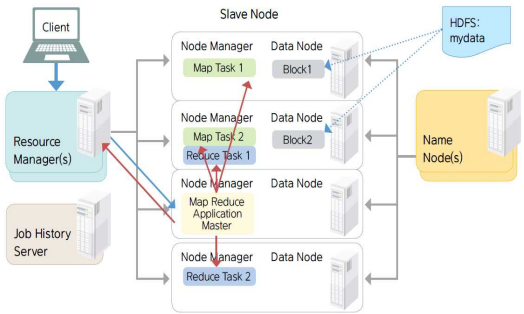

그림 2. 하둡 분산 파일 시스템의 데몬을 활용한 맵리듀스 기반 빅데이터 분석 과정
Fig. 2. The process of MapReduce-based big data analysis using HDFS daemon

빅데이터 분석을 위한 맵리듀스의 주요 동작 단계는 드라이버, 맵, Shuffle and Sort와 리듀스 단계로 구성되며, 맵리듀스의 실행은 이 순서를 따른다. 〈그림 3〉은 맵리듀스이 주요 동작 단계를 나타낸다.

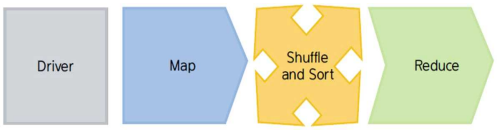

그림 3. 맵리듀스의 주요 동작 단계
Fig. 3. The process of MapReduce

드라이버 단계는 맵리듀스의 실행에 관련된 환경 설정을 정의와 실행하며, 하둡 분산 파일 시스템에 저장된 데이터를 로드하는 역할을 수행한다. 맵 단계는 드라이버에 단계로부터 로드된 빅데이터들에 대해 1차 처리를 수행한다. Shuffle and Sort 단계는 맵 단계로부터 처리된 데이터들을 병합 및 정렬을 수행한다. 리듀스 단계는 2차 가공 및 분석된 결과 데이터를 생성한다.

2.2 요구사항 분석

앞서 설명한 맵리듀스 기반 빅데이터 분석 기법은 하둡의 핵심 기술 중에 하나인 맵리듀스를 활용하여 빅데이터 분석을 수행한다. 맵리듀스의 기본 설정은 리듀스 단계에서 생성되는 리듀서 객체를 한 개로 정의한다. 이로 인해 맵리듀스를 통한 빅데이터 분석은 Shuffle and Sort 단계를 통해 전달되는 많은 데이터들이 하나의 리듀서 객체에 집중적으로 모이게 되며, 병목 현상이 발생하게 된다.

따라서 맵리듀스 기반 빅데이터 분석 기법은 리듀스 단계에서 발생되는 병목 현상을 최소화하여 보다 빠른 빅데이터 분석이 가능해야 한다.

3. 제안한 분할 빅데이터 분석 기법

제안한 분할 빅데이터 분석 기법은 앞서 설명한 요구사항 분석 결과에 적합하도록 병목 현상을 최소화하도록 설계한다. 제안한 기법의 리듀스 단계에서 생성되는 리듀서 객체를 유동적으로 생성한 뒤, 분석 결과의 유형에 따라 하나의 결과를 생성한다. 제안한 빅데이터 분석 기법의 구성은 리듀서 분할 단계와 분석 결과 병합 단계이다.

3.1 리듀서 분할 단계

리듀서 분할 단계는 맵리듀스의 기본 설정을 변경하여 리듀서 객체를 여러 개를 생성하여 Shuffle and Sort 단계를 통해 전달받는 데이터를 분산 처리할 수 있도록 한다. 리듀서 분할 단계의 역할은 유동적인 리듀서 객체 생성과 리듀서 객체와 분석할 빅데이터를 매칭하고, 분석된 결과들을 생성한다.

유동적인 리듀서 객체 생성은 우선 맵리듀스의 기본 설정을 변경을 통해 진행된다. 맵리듀스의 기본 설정 변경은 드라이버에서 변경이 가능하며, 로그 분석 유형에 맞게 리듀서 객체를 생성한다. 로그 분석 유형은 일반적인 로그에 반드시 기입되는 날짜·시간 정보를 활용하여 날짜별, 요일별, 시간대별 3가지로 정의한다. 〈표 1〉은 로그 분석 유형별 정의한 리듀서 객체의 수를 나타낸다.

표 1. 로그 분석 유형별 정의한 리듀서 객체의 수
Table 1. The count of reducer object for log analysis type

로그 분석 유형	리듀서 객체 수	비고
날짜	1부터 최대 31개	검색기간에 따라 유동적 변경
요일	7개	
시간대	24개	24시간 기준

로그 분석 유형 중 날짜별 유형은 로그 분석을 위한 검색기간에 유동적으로 하루마다 1개의 리듀서 객체를 생성하고, 요일별 유형은 요일별 총 7개의 리듀서 객체를 생성하고, 시간대별은 24시간으로 기준으로 총 24개의 리듀서 객체를 생성한다. 〈그림 4〉는 로그 분석 유형별 리듀서 생성을 위한 의사 코드를 나타낸다.

```
1  //맵리듀스 잡 생성
2  Job job = new Job();
3
4  //날짜별: 1, 요일별: 2. 시간대별: 3
5  logType = getConf(args[2]);
6
7  //리듀서 객체 생성 수
8  if (logType==1)
9    job.setNumeReduce(getDays());
10 else if (logType==2)
11   job.setNumeReduce(7);
12 else if (logType==3)
13   job.setNumeReduce(24);
14
15 //리듀서 객체 생성 수
16 setConf("logType", logType)
17
```

그림 4. 로그 분석 유형별 리듀서 생성을 위한 의사 코드
Fig. 4. The pseudo code of create reduce object for log analysis types

리듀서 생성을 위한 기본 환경 변경은 맵리듀스의 주요 단계 중 드라이버 단계에 정의한다. 드라이버 단계에 사용되는 드라이버 객체는 정의된 로그 분석 유형(logType)에 따라 맵리듀스 잡에 생성되는 리듀서 객체의 수를 정의한다.

분석할 빅데이터 매칭은 로그 분석 유형에 따라 생성된 다 수의 리듀서 객체마다 처리해야할 로드 데이터를 매칭한다. 데이터 매칭은 맵리듀스 프레임워크에서 제공하는 파티셔너(Partitioner) 객체를 상속받아 오버라이드하여 사용한다. 〈그림 5〉는 분석할 빅데이터 매칭을 위한 의사 코드를 나타낸다.

```
1  //맵리듀스의 정의된 파티셔너 객체 상속
2  extends Partitioner
3
4  //유형별 데이터 매칭
5  int getPartition(getTimeStamp()){
6    //지정될 리듀서 객체 번호
7    int res = 0;
8    if (getConf(logType)==1)
9      setDateReducer(getTimeStamp());
10   else if (getConf(logType)==2)
11     setWeekReducer(getTimeStamp());
12   else if (getConf(logType)==3)
13     setTimeReducer(getTimeStamp());
14 }
15
16 //로그의 날짜·시간 정보 가져오기
17 int getTimeStamp(logType)
18
19 //날짜별 리듀서 객체 매칭
20 int setDateReducer(timeStamp){
21   for each (int i = reducerCnt){
22     reducer(i) = timeStamp;
23   }
24 }
25
26 //요일별 리듀서 객체 매칭
27 int setDateReducer(timeStamp){
28
29 //시간대별 리듀서 객체 매칭
30 int setDateReducer(timeStamp){
31
```

그림 5. 분석할 빅데이터 매칭을 위한 의사 코드
Fig. 5. The pseudo code for big data matching

분석할 빅데이터 매칭은 분할 생성한 리듀서 객체에 데이터를 매칭하는 것으로 파티셔너 객체를 상속하며 시작한다. 분석할 빅데이터 매칭에 사용된 함수는 크게 5가지로 etPartition, getTimeStamp,

setDateReducer, setDateReducer와 setDateReducer 함수이다. getPartition 함수는 파티셔너 객체를 상속받아 오버라이드하고, 앞서 생성한 맵리듀스와 데이터간 매칭을 로그 분석 유형별로 수행한다. getTimeStamp 함수는 로그에 기입된 날짜·시간 정보를 로그 분석 유형에 맞게 가져온다. setDateReducer, setDateReducer와 setDateReducer 함수는 로그 유형별 매칭을 수행하며, 앞서 설명한 getPartition 함수에서 호출된다. 이 함수들은 getTimeStamp 함수를 통해 전달받은 날짜·시간 정보의 순서대로 리듀서 객체 0번째 주소부터 매칭된다. 예를 들어, 날짜별 로그 분석유형은 1일은 주소 값이 0번인 리듀서 객체와,시간대별 로그 분석 유형은 0시는 주소 값이 0인 리듀서, 23시는 주소값이 23인 리듀서와 매칭된다.

3.2 분석 결과 병합 단계

분석 결과 병합 단계는 맵리듀스의 리듀서 객체마다 생성되는 분석결과를 하나의 파일로 병합한다. 분석 결과 병합 단계에서의 결과 병합은 앞서 설명한 리듀서 분할 단계를 통해 실행이 완료된 맵리듀스의 결과 파일들을 맵리듀스의 맵 단계를 실행하여 병합한다. 따라서 분석 결과 병합 단계의 동작은 총 1번의 맵리듀스 동작과 추가적인 맵 단계 동작을 수행한다. 〈그림 6〉은 제안한 기법의 주요 동작 단계를 나타낸다.

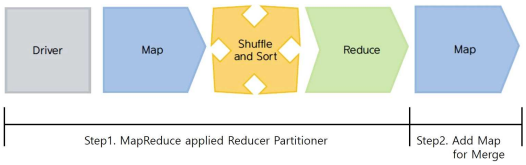

그림 6. 제안한 기법의 주요 동작 단계
Fig. 6. The step of propose scheme

분석 결과 병합 단계는 먼저 실행된 맵리듀스의 결과 데이터를 병합하는 역할만 수행하기 때문에 맵리듀스 전체 동작을 다 수행하지 않고, 맵리듀스의 맵 단계만 실행한다. 〈그림 7〉은 맵리듀스의 맵 단계만 실행하기 위한 의사 코드를 나타낸다.

```
1  //새로운 맵리듀스 잡 생성
2  Job job2 = new Job();
3
4  //맵리듀스 분석 결과 로드
5  FileFormat.setInputPaths();
6
7  //분석 결과 병합될 파일 경로 설정
8  FileFormat.setOutputPaths();
9
10 //리듀서 객체 생성되지 않도록 설정
11 job.setNumReduceTask(0);
12
13 //맵리듀스 실행
14 job.waitForCompletion();
15
```

그림 7. 맵 단계만 실행하기 위한 의사 코드
Fig. 7. The pseudo code of execution map step

분석 결과 병합 단계는 새로운 맵리듀스를 실행하기 때문에 앞서 생성한 맵리듀스 잡이 아닌 새로운 잡을 생성하여 실행한다. 새롭게 생성된 맵리듀스 잡 객체는 앞서 설명한 리듀서 분할 단계에서 생성된 분석 결과들을 병합하기 위해 디렉터리 단위로 로드한다.

분석 결과 병합 단계는 분석 결과에 대해 하나의 파일로 생성하기 때문에 추가적인 리듀스 단계가 불필요하며, 리듀서 객체가 생성되지 않도록 설정한다.

4. 성능평가

본 장에서는 제안한 맵리듀스 기반 분할 빅데이터 분석 기법에 대한 성능평가를 수행한다. 성능평가 항목은 앞서 관련 연구에서 분석한 요구사항을 기반으로 기존 맵리듀스 기반 빅데이터 분석 기법과 제안한 분할 빅데이터 분석 처리시간 및 분석 결과 병합 단계의 처리시간을 분석한다.

〈표 2〉는 성능평가를 위한 주요 환경 구성을 나타낸다.

표 2. 성능평가를 위한 주요 환경 구성
Table 2. The environment for performance analysis

항목	내용
빅데이터 컴퓨터 환경	실행환경 : 하둡 완전 분산모드 설치버전 : 하둡 2.82
서버(노드) 스팩	Dell PowerEdge R730 – Intel Xeon 3.4Ghz 8Core 64G RAM
노드 구성	네임노드 : 1대 2차네임노드 : 1대 데이터노드 : 3대
분석 대상-웹로그	아파치 발생 로그(하루 약 500MB)
분석 기간	31일
맵리듀스 잡	비정상적인 URL 접속 횟수 분석 잡

성능평가를 위한 환경 구성은 아파치와 같은 웹서버에서 발생되는 웹 로그를 기반으로 빅데이터 분석을 수행하며, 하둡을 기반으로 완전 분산 모드를 구축하여 검증한다. 성능평가에 사용된 하둡의 버전으로 정식 배포된 2.82 버전을 사용한다. 하둡 완전 분산 모드의 구성 노드는 1개의 네임노드와 2차 네임노드, 3개의 데이터 노드로 총 5개의 서버를 활용한다. 웹 로그 분석을 위한 분석 기간은 2018년 3월부터 5월까지 총 3개월로 정의하여 분석한다. 성능평가에 사용되는 맵리듀스 잡은 비정상적인 URL 접속 횟수를 분석 잡을 사용한다.

4.1 제안한 기법의 빅데이터 분석 처리시간

제안한 기법의 빅데이터 분석 처리 시간은 로그 분석 유형별 분할 빅데이터 분석 기법들의 맵리듀스 실행시간을 측정하고, 그 결과를 비교 분석한다. 분석 방법은 월요일이 1일인 날부터 30일까지의 생성된 로그 데이터를 분석하는데 발생한 처리시간을 측정한다. 〈그림 8〉과 〈표3〉은 제안한 기법의 빅데이터 분석 처리 시간을 나타낸다.

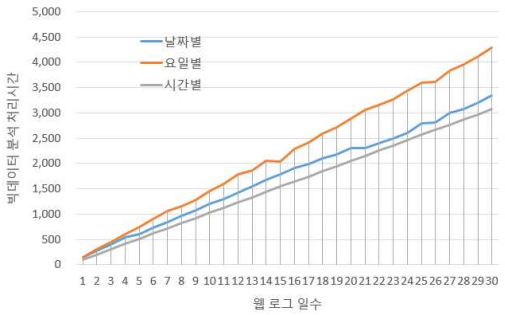

그림 8. 제안한 기법의 빅데이터 분석 처리 시간
Fig. 8. The big data analysis process time of propose d scheme

표 3. 제안한 기법의 구간별 빅데이터 분석 처리 시간
Table 3. The big data analysis process time of propose scheme for each section

	1-5	6-10	11-15	16-20	21-25	26-30
날짜별	394.6	961	1,548	2,097	2,520.2	3,089.8
요일별	452.2	1,169.4	1,872.4	2,582.8	3,300.4	3,960.4
시간대별	307	821	1,335.6	1,847.6	2,360.6	2,869.4

결과에 따르면, 시간대별 분할 빅데이터 분석 기법은 제안한 로그 분석 유형별 분석에서 가장 빠르게 빅데이터를 처리하였다. 빅데이터 처리 시간이 가장 느린 유형은 요일별 분할 빅데이터 분석이며, 30일 기준으로 시간대별 분할 빅데이터 분석 기법에 비해 약 28% 느리게 빅데이터를 분석하였다.

이러한 결과가 발생한 이유는 맵리듀스 객체의 생성 수와 밀접한 관계가 있다. 날짜별 분할 빅데이터 분석 기법은 총 30개의 맵리듀스 객체를 생성하고, 요일별 분할 빅데이터 분석 기법은 총 7개의 맵리듀스 객체를 생성하고, 마지막으로 시간대별 분할 맵리듀스 빅데이터 분석 기법은 총 24개의 맵리듀스 객체를 생성한다. 시간대별 분할 빅데이터 분석 기법은 날짜별 분할 빅데이터 분석 기법에 비해 생성된 맵리듀스 객체의 수가 적었으며 요일별 분할 빅데이터 분석 기법에 비해 생성된 맵리듀스 객체의 수가 많았다.

따라서 최적의 빅데이터 분석 처리시간에 필요한 리듀서 객체 수는 많을수록 좋은 것이 아니라, 분석할 웹 로그의 양과 로그 데이터의 쌓이는 시점에 따라 적절하게 리듀서 객체를 생성해서 사용해야 한다.

4.2 기존 법과 빅데이터 분석 처리시간 비교

빅데이터 분석 처리 시간은 앞서 성능 분석한 시간 대별 분할 빅데이터 분석 기법과 기존 기법의 실행시간을 측정하고, 그 결과를 비교 분석한다. 분석 방법은 앞서 성능 분석한 방법과 동일하다. 〈그림 9〉와 〈표 4〉는 기존 기법과 빅데이터 분석 처리 시간 비교를 나타낸다.

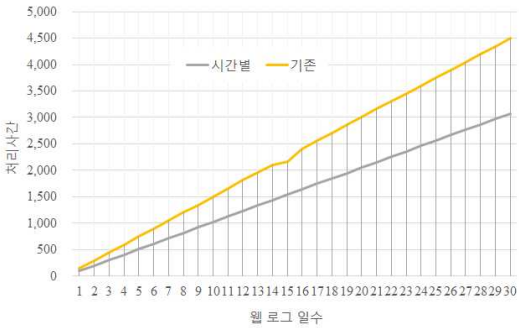

그림 9. 기존 기법과 제안한 기법의 빅데이터 분석 처리 시간 비교
Fig. 9. The comparison of existing scheme and propose scheme for big data analysis process time

표 4. 기존 기법과 제안한 기법의 구간별 빅데이터 분석 처리 시간
Table 4. The existing scheme and propose scheme for each section

	1-5	6-10	11-15	16-20	21-25	26-30
시간대별	307	821	1,335.6	1,847.6	2,360.6	2,869.4
기존	449.8	1,202	1,939.8	2,707	3,455.6	4,198
증 · 감	−142.8	−381	−604.2	−859.4	−1095	−1328.6
처리율	32%	32%	31%	32%	32%	32%

결과에 따르면, 제안한 기법과 기존 기법의 빅데이터 처리 시간은 30일 기준으로 2,869.초와 4,198초로 측정되었으며, 제안한 기법은 기존 기법과 비교하여 1,328.6초 빠르게 빅데이터 분석을 수행하였다.

빅데이터 분석 처리율은 1일부터 30일까지 제안한 기법이 기존 기법보다 평균적으로 약 32% 향상되었다.

5. 결론

로그 분석을 위해 사용되는 빅데이터 분석 기법들은 대부분 맵리듀스를 기반으로 빅데이터 분석을 수행하고 있다. 맵리듀스는 하둡을 기반으로 구현된 빅데이터 플랫폼들에 반드시 사용되는 핵심 기술이다.

그러나 맵리듀스는 리듀스 단계에서 생성되는 리듀서 객체를 한 개로 정의하고 있어 많은 문제를 야기하고 있다. 특히 맵리듀스를 통한 빅데이터 분석은 Shuffle and Sort 단계를 통해 전달되는 많은 데이터들이 하나의 리듀서 객체에 집중적으로 모이게 되며, 병목 현상이 발생하게 된다.

이에 본 논문에서는 로그 분석 처리율 향상을 위한 맵리듀스 기반 분할 빅데이터 분석 기법을 제안한다. 제안한 기법은 리듀서 분할 단계와 분석 결과 병합 단계로 구분하여 구현한다. 리듀서 분할 단계는 로그 분석 유형에 맞춰 리듀서 객체 수를 유동적으로 생성하여 병목 현상을 감소시킨다. 분석 결과 병합 단계는 리듀서 객체들로부터 분할 저장되는 분석 결과들을 하나의 분석 결과로 병합한다. 이로 인해 제안한 기법은 기존 기법에 비해 약 32%의 향상된 빅데이터 처리율을 보였다. 향후, 빅데이터 분석 기법에 대한 연구는 다양한 시스템에서 발생시킨 로그들을 기반으로 심층적인 분석이 필요하다.

REFERENCES

[1] H. G. Lee, Y. W. Kim & K. Y. Kim (2017), Implementation of an Efficient Big Data Collection Platform for Smart Manufacturing. Journal of Engineering and Applied Sciences, 12(2Si), 6304-6307.
[2] Y. W. Kim & H. G. Lee (2017). Implementation of Big Data Analysis System to Prevent Illegal Sales in the Cable TV Industry. Journal of Engineering and Applied Sciences, 12(3Si), 6542-6545.
[3] H. G. Lee, Y. W. Kim, K. Y. Kim & J. S. Choi. (2018). Design of GlusterFS Based Big Data Distributed Processing System in Smart Factory, Journal of Korea Institute of Information, Electronics, and Communication Technology, 11 (1), 70-75.
[4] E. H. Jeong & B. K. Lee. (2017). A Design of Hadoop Security Protocol using One Time Ke

y based on Hash-chain, Journal of Korea Institute of Information, Electronics, and Communication Technology, 10(4), 340-349.
[5] Y. S. Lee (2015). Authentication Method for Safe Internet of Things Environments, Journal of Korea Institute of Information, Electronics, and Communication Technology, 8(1), 51-58.
[6] J. T. Seong (2017). Analysis of Signal Recovery for Compressed Sensing using Deep Learning Technique, Journal of Korea Institute of Information, Electronics, and Communication Technology, 10(4), 257-267.

저자약력

이 협 건(Hyeopgeon Lee) [종신회원]

- 2011.03 - 2015.08, 숭실대학교 일반대학원 컴퓨터학과 공학박사
- 2015.12 - 현재, 한국폴리텍대학 서울강서 캠퍼스 데이터분석과 교수

<관심분야> 빅데이터, 실시간 분석, 데이터 분석

김 영 운(Young-Woon Kim) [종신회원]

- 2004.09 - 2018.08, 숭실대학교 일반대학원 컴퓨터학과 공학박사
- 2015.12 - 현재, 한국폴리텍대학 서울강서 캠퍼스 데이터분석과 교수

<관심분야> 빅데이터, 실시간 분산 처리, 데이터분석

박 지 용(Jiyong Park) [정회원]

- 1997.03 - 2004.8 서울대학교 작곡과이론전공 학사
- 2009.01 - 2010.12 New York University 재즈피아노 석사
- 2016.3 - 현재 한양대학교 실용음악학과 겸임교수

<관심분야> 음향미디어, 전자음악, 음악교육

이 진 우(Jin-Woo Lee) [정회원]

- 1987.03-1993.08 숭실대학교 물리학과 학사
- 2000.10-2014.08 코리아닷컴커뮤니케이션즈 본부장
- 2016.06- 현재 (주)시스원 부장

<관심분야> 빅데이터, 무선통신, 사물인터넷