Cloud Computing에서의 데이터 복제 성능 개선

이준규, 이봉환

대전대학교 정보통신공학과

Performance Improvement of Data Replication in Cloud Computing

Joon-Kyu Lee and Bong-Hwan Lee

Dept. of Information & Communications Engineering, Daejeon University

E-mail : {izzyjk@naver.com, blee@dju.ac.kr}

요 약

최근 분산시스템은 사용자들에게 데이터센터의 컴퓨팅 자원 및 서비스를 온라인 상에서 효율적으로 제공하는 개념의 클라우드 컴퓨팅으로 진화하고 있다. 클라우드 컴퓨팅은 그리드 컴퓨팅에서 지역적으로 분산되어 있는 컴퓨팅 시스템 자원의 공유로 인하여 발생하는 잠재적 위험성을 중앙집중화된 형태의 데이터센터를 구축하여 줄이고, 거대한 데이터 세트를 분산 처리할 수 있게 하는 서비스 플랫폼이다. 본 논문에서는 대표적인 클라우드 컴퓨팅 미들웨어인 하둡에서 데이터 복제 시 사용하는 1:1 전송 방식을 1:N 방식으로 수정하여 성능을 개선하는 메커니즘을 제안하였다. 제안한 1:N 데이터 복제 방식은 기존의 방식에 비하여 데이터 전송 시간을 현저히 개선하였다.

ABSTRACT

Recently, the distributed system is being evolved into a new paradigm, named cloud computing, which provides users with efficient computing resources and services from data centers. Cloud computing would reduce the potential danger of Grid computing which utilizes resource sharing by constructing centralized data center. In this paper, a new data replication scheme is proposed for Hadoop distributed file system by changing 1:1 data transmission to 1:N. The proposed scheme considerably reduced the data transmission delay comparing to the current mechanism.

키워드

클라우드 컴퓨팅(Cloud Computing), 하둡(Hadoop), 데이터 복제(Data Replication)

Ⅰ. 서 론

클라우드 컴퓨팅(Cloud Computing)은 IT 인프라를 기반으로 사용자의 요구사항에 적합한 온라인 서비스를 제공하는 새로운 서비스 패러다임이다. 사용자는 자신이 원하는 만큼의 필요한 자원만을 사용하게 되어 지금과 같은 컴퓨팅 자원의 낭비뿐만 아니라 자원의 효용성 및 공간 이동성 제공 등 여러 분야에 미치는 파급효과가 크다. 물론 이러한 결과는 과거 분산병렬처리 및 최근에 개발된 그리드 컴퓨팅(Grid Computing)[1]의 과정을 거치면서 발전하게 되었다.

클라우드 컴퓨팅 이전의 그리드 컴퓨팅은 지리적으로 분산되어 있는 컴퓨팅 자원들을 하나로 묶어 사용하는 가상화 그룹 개념을 도입했다. 가상화 그룹이라는 개념의 도입은 훌륭하였으나 이에 반하여 잠재적 보안 위험성과 컴퓨팅 자원의 안정성 유지에 문제점이 발생하였다.

클라우드 컴퓨팅은 과거 메인프레임에 접속되어 있는 여러 터미널들을 사용자가 각자의 계정으로 사용하는 것과 비슷한 환경구조를 가지며, 데이터센터의 컴퓨팅 자원을 효율적으로 사용하자는 취지로 등장하게 되었다. 과거 그리드 컴퓨팅과의 큰 차이점은 지리적으로 산재되어 있는 그룹들을 중앙 집중관리방식으로 전환하는 것이다.

중앙 집중형식으로의 전환은 보다 효율적인 관리의 이점을 갖게 된다. 따라서 상용적인 측면이 강하게 대두되었고, 현재 IT 대기업들을 중심으로 활발히 개발되고 있는 실정이다. 클라우드 컴퓨팅은 사용상의 실수와 손실에 대비하기 위해 노드간 상호 보완 능력을 제공하며, 풍부하고 안전한 컴퓨팅자원을 사용자에게 제공하기 위해 분산 데이터센터 구조를 갖는다. 사용자는 장소에 구애받지 않고 필요한 일을 클라우드에 요청하고 결과

를 확인하는 환경이 제공된다.

본 연구에서는 클라우드 컴퓨팅에서 데이터 무결성과 연계된 데이터 분산방법을 개선하는 메커니즘을 제안하고 그 성능을 분석하고자 한다.

II. 클라우드 컴퓨팅

클라우드 컴퓨팅은 현대의 가스, 전기, 전화, 수도공급과 같은 공공서비스적인 의미를 갖는 IT 인프라를 구현하고자 하는 관점에서 시작되었고, 이전의 병렬처리, 그리드 컴퓨팅, P2P컴퓨팅의 성격을 모두 담고 있다[2].

최근 웹2.0은 지식의 집중화와 다양한 웹 응용형식을 시도하고 있다. 이에 맞춰 클라우드 컴퓨팅도 휴대 무선단말기와 같은 간단한 형태의 단말기에서도 서비스 제공자(Service Provider)에 접속하여 서비스를 받는 형태로 발전하고 있다.

사용자는 간단한 유·무선 단말기를 사용하여 시간의 구애를 받지 않고 안정성과 기밀성이 보장되는 클라우드 컴퓨팅을 사용하여 자유롭게 정보를 활용할 수 있다. 그림 1은 IT 인프라 상에 구축된 데이터 센터에서 클라우드 컴퓨팅을 서비스하는 개념도로써 사용자는 서비스 카테고리에 등록 되어 있는 필요한 서비스를 요청하고 이를 제공받는 형식으로 이루어져 있다.

이러한 클라우드 컴퓨팅의 대표적인 미들웨어로 하둡(Hadoop)이라는 Apache 산하의 오픈 개발그룹이 있다. 자바 클래스를 기본으로 필요한 기능을 모아 오픈소스 형태로 제공하여 여러 개발자에게 참여를 유도하고 있다.

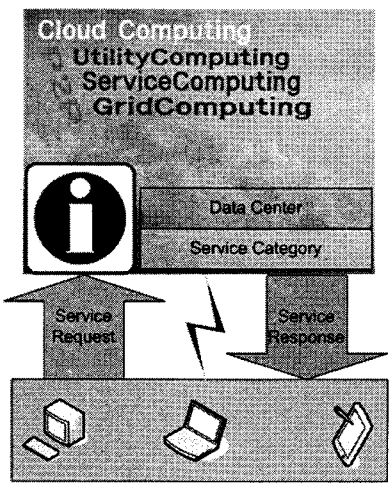

그림 1. 클라우드 컴퓨팅 개념

하둡을 기본으로 두고 있는 대표적인 IT 기업으로는 IBM, Yahoo, Google 등이 있다[3].

III. 하둡 분산 파일 시스템

하둡의 분산 파일시스템(Hadoop Distributed File System, HDFS)[4]의 기술 문서에 따르면 HDFS는 크게 6가지 기능에 중점을 두고 있다.

(1) 하드웨어상의 오류보정

이는 수백에서 수천 대에 이르는 데이터센터 내 노드들 중 일부 노드에서 오류가 발생 시 신속하게 자동으로 복구하는 기능

(2) 스트리밍 자료 접근

범용 파일시스템과 달리 반응속도 보다는 시간당 처리량에 최적화되어 있다.

(3) 대용량 데이터 집합

한 파일이 기가 바이트나 테라바이트 정도의 크기를 갖는 것을 목적으로 설계되었다. 자료의 대역폭 총량이 높고 하나의 클러스터에 수 백개의 노드를 둘 수 있으며, 하나의 인스턴스에서 수천만여 개의 파일을 지원한다.

(4) 간단명료한 데이터 관리모델

하둡의 응용은 파일 모델상 한 번의 기록하고 여러 번 읽는 모델에 적합한 구조이다. 이렇게 함으로써 처리량을 극대화할 수 있다. 파일 또는 자원에 접근한 기록들을 간결하고 정확하게 유지해야 된다.

(5) 데이터 접근의 효율성

데이터를 많이 옮기면 대역폭이 많이 들기 때문에 네트워크 혼잡으로 인하여 전체 처리율이 감소한다. 따라서 가까운 곳에 있는 자료를 처리하도록 계산 작업 자체를 옮기면 전체적인 처리율이 더 높아진다.

(6) 이 기종 하드웨어와 소프트웨어 플랫폼과의 호환성

서로 다른 하드웨어와 소프트웨어 플랫폼들을 연동해도 잘 동작한다.

그림 2는 HDFS의 데이터 복제방법과 순서를 나타낸 것이다. 그림 2에서 보는 바와 같이 파일이나 데이터 스트림의 Read 또는 Write 시 클라이언트와 데이터 노드 사이에 1:1의 연관 관계가 성립된다. 원본 데이터의 복제는 전송이 끝난 후 네임노드의 통제 하에 다시 이루어지는 형식이다. Hadoop은 스트림 처리에 Out of Memory 현상이 발생하고 데이터 맵핑이 Pipelining 방식으로 처리됨으로 인하여 순차적 처리에 의한 시간지연이 발생하는 문제점을 가지고 있다.

본 논문에서는 네임 노드와 데이터 노드 간의 상관관계와 클라이언트에서의 데이터스트림 분할 방식을 이용하여 데이터 복제 성능 개선 방법을 제안한다.

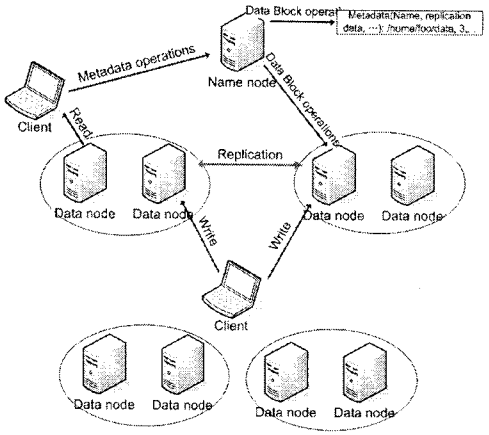

그림 2. HDFS의 데이터스트림 이동과 복제서비스 과정

IV. 분산 데이터 복제 모델 및 성능 측정

본 논문에서는 기존의 1:1 파일전송 방식을 1:N 전송 방식으로 변경하여 데이터 전송 효율을 개선하는 방법을 제안하였다. 기존의 하둡 네임노드가 서비스 카테고리를 제공하는 것에 추가하여 클라이언트 자신이 통신해야 하는 노드 리스트를 제공하는 형식이다. 통신을 해야 하는 노드를 파악한 뒤 파일 전송을 기존의 1:1이 아닌 통신 대상 데이터노드의 숫자만큼 파일 전송을 N개의 단편으로 보내는 방법이다.

이 과정에서 정의한 프로토콜에 따라 소켓통신이 이루어지지 않는 데이터노드가 발생 시 통신 오류처리로 네임노드의 DB에 전송실패 노드와 발생 시간을 기록하게 된다. 기록한 뒤에는 실패된 데이터스트림의 단편을 다음 대상 데이터노드로 전송함으로써 무결성을 보장한다.

이 로그를 바탕으로 네임노드는 데이터노드들의 상태를 참조하여 복구에 활용한다. 표 1은 실험에 사용된 PC들의 사양을 나타낸 것이며, Node 1은 데스크탑이고 Node 2는 네임노드와 데이터노드로 사용한 노트북이다.

표 1. 실험에 사용된 PC 사양

	Node1	Node2
CPU	IntelP4-2.00GHz	IntelC2-2.0
RAM	754MB	2GB
HDD	60GB	20GB
OS	Fedora 8	Fedora 8
대역폭	10Mbps	
역할	데이터 노드	데이터노드/네임노드

그림 3은 실험 시스템에서의 데이터 복제 과정을 나타낸다. 클라이언트가 데이터 전송 요청 메시지를 네임노드로 전송(1)하면 네임노드는 데이터노드의 상태를 체크(2)하고 난 다음 클라이언트에게 대상 노드 리스트 정보를 통보함과 동시에 사용 승인 메시지를 보낸다(3). 클라이언트는 전송 대상 데이터 노드들의 정보를 이용하여 파일을 분할하여 각각 전송하고(4) 전송 완료 통보 메시지를 네임노드에 전송한다(5). 마지막으로 네임노드는 데이터 노드들에 분할되어 수신된 파일들을 병합하여 데이터 복제를 완료한다(6).

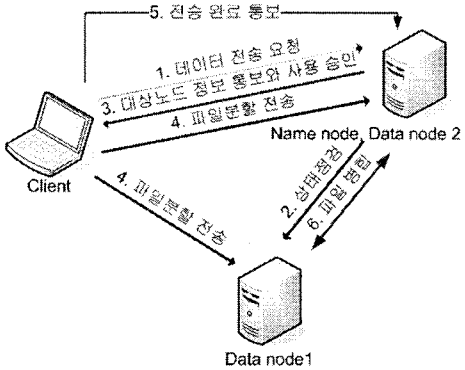

그림 3. 데이터 복제 과정

그림 4는 3.5GB 크기의 파일 전송 시 1:1 전송 방식과 제안한 1:2 전송 방식을 이용하여 파일 스트림을 단편화하여 보내었을 때의 전송 시간을 비교한 것이다.

1:1로 각각의 데이터노드로 데이터 전송할 때보다 1:2로 다중 전송 시 총 전송이 현저히 감소하여 성능이 개선되었음을 알 수 있다.

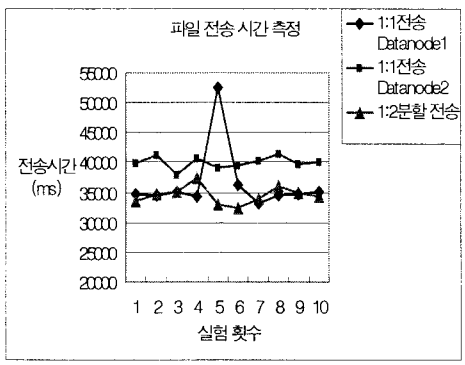

그림 4. 파일전송 시간 측정 결과

V. 결 론

네임노드를 중심으로 구성되어 있는 클라우드 컴퓨팅 분산 시스템은 안전하고 빠른 서비스 능력을 지속적으로 유지해야 한다. 본 논문에서는 파일을 분할하여 전송하는 1:N 다중 전송 메커니즘을 제안하고 그 성능을 1:1 전송 방법과 비교하였다. 두 대의 데이터 노드로 구성된 분산 파일 전송 시스템에서 실험한 결과 제안한 1:N 전송 방법이 기존의 1:1 스트림 데이터 전송 방식에 비해 전송 지연 시간을 감소시킴을 알 수 있었다. 이 방식은 또한 데이터 센터로의 평균부하를 분산함으로써 사용요구 처리지연 시간을 감소시키는 효과도 기대할 수 있을 것으로 사료된다.

향후 연구로서 더 많은 데이터 노드의 수와 다양한 데이터 셋을 이용한 실험 결과 및 분석이 요구된다 하겠다.

Acknowdgement

본 연구는 한국산업기술재단의 지역혁신 인력 양성 사업 및 정보통신연구진흥원의 대학 IT 연구센터 지원사업 (IITA-2008-C1090-0801-0014)의 연구 결과로 수행되었음.

참고문헌

[1] I. Foster, C. Kesselman (eds). *The Anatomy of the Grid: Enabling Scalable Virtual Organizations,* International J. Supercomputer Applications, 15(3), 2001.

[2] Rajkumar Buyya and Chee Shin Yeo (eds),"*Market-Oriented Cloud Computing: Vision, Hype, and Reality for Delivering IT Services as Computing Utilities,*" *In Proc. of* 10th IEEE International Conference on HPCC_08, Sept. 25-27, 2008.

[3] *Data-Intensive Supercomputing: The case for DISC,* Technical Report CMU-CS-07-128, School of Computer Science, Carnegie Mellon University.

[4] *The Hadoop Distributed File System: Architecture and Design, http://hadoop.apache.org/core*