비-전용 분산 컴퓨팅 환경에서 맵-리듀스 처리 성능 최적화를 위한효율적인 데이터 재배치 알고리즘

An Efficient Data Replacement Algorithm for Performance Optimization of MapReduce in Non-Dedicated Distributed Computing Environments

류 은 경, 손 인 국, 박 준 호, 복 경 수, 유 재 수
충북대학교 정보통신공학부

Eunkyung Ryu, Ingook Son, Junho Park,
Kyoungsoo Bok, Jaesoo Yoo
School of Information and Communication Engineering

요약

최근 소셜 미디어의 성정과 모바일 장치와 같은 디지털 기기의 활용이 증가함에 따라 데이터가 기하급수적으로 증가하였다. 이러한 대용량의 데이터를 처리하기 위한 대표적인 프레임워크로 맵-리듀스가 등장하였다. 하지만 전용 분산 컴퓨팅 환경에서의 균등한 데이터 배치를 기반으로 수행되는 기존 맵-리듀스는 가용성이 다른 비-전용 분산 컴퓨팅 환경에서는 적합하지 않다. 이를 고려한 비-전용 분산 컴퓨팅 환경에 최적화된 데이터 재배치 알고리즘이 제안되었지만, 데이터 재배치 알고리즘을 수행함으로써 재배치에 많은 시간을 필요로 하고, 불필요한 데이터 전송에 의한 네트워크 부하가 발생한다. 본 논문에서는 비-전용 분산 컴퓨팅 환경에서 맵-리듀스의 성능 최적화를 위한 효율적인 데이터 재배치 알고리즘을 제안한다. 제안하는 기법에서는 노드의 가용성 분석 모델을 기반으로 노드의 데이터 블록 비율을 연산하고, 기존의 데이터 배치를 고려하여 전송함으로써 네트워크 부하를 감소시킨다. 성능평가 결과 기존 기법에 비해 데이터 재배치 블록 비율이 약 75% 감소하였다.

Ⅰ. 서론

최근 소셜 미디어의 성정과 모바일 장치와 같은 디지털 기기의 활용에 따라 디지털 정보량이 기하급수적으로 증가하여 기존 데이터 저장 및 분석 시스템의 처리 한계를 넘어서게 되었다. 이러한 대용량 데이터를 처리하기 위한 대표적인 프레임워크로써, 하둡(Hadoop)[1]은 대규모 자료의 저장 및 처리를 위한 분산 응용 프로그램을 지원하는 대표적인 오픈소스 소프트웨어이다. 하둡은 페티바이트 이상의 대규모 데이터를 클러스터 환경에서 저장하기 위한 하둡 분산 파일 시스템(HDFS)과 이를 기반으로 병렬 처리를 지원하기 위한 맵-리듀스(MapReduce) 프레임워크로 구성된다.[1)]

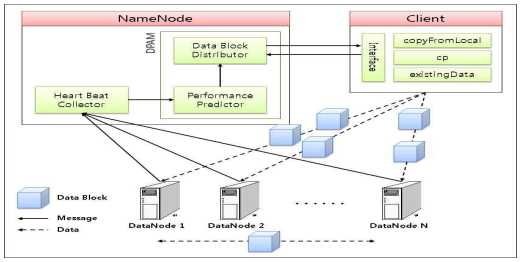
기본적으로 맵-리듀스 프레임워크는 전용 분산 컴퓨팅 환경(Dedicated Distributed Computing)을 기반으로 제안되었으므로 작업의 효율성을 고려하여 저장된 데이터의 편차가 발생하지 않도록 각 노드에 데이터 블록을 균등하게 배치한다. 하지만, 비-전용 분산 컴퓨팅에서는 클러스터를 구성하는 노드의 다른 작업의 활용 및 결함 발생으로 인한 가용성이 모두 다르므로 기존 맵-리듀스 프레임워크와 같은 균등한 데이터 배치는 적합하지 않다. 이러한 문제점을 고려하여, [2]에서는 노드의 가용성 분석을 통해 비-전용 분산 컴퓨팅 환경에 최적화된 데이터 재배치 알고리즘을 제안함으로써 맵-리듀스의 처리 시간을 감소시켰다. 그러나 [2]은 초기 데이터 배치 상태를 전혀 고려하지 않고 전체 데이터를 재배치함으로써 데이터 재배치 과정에서 불필요한 데이터 전송에 의한 네트워크 부하가 발생하고, 재배치에 많은 시간을 필요로 한다.

본 논문에서는 비-전용 분산 컴퓨팅 환경에서 맵-리듀스의 성능 최적화를 위한 효율적인 데이터 재배치 알고리즘을 제안한다. 제안하는 기법에서는 노드의 가용성 분석 모델을 기반으로 데이터 블록 비율을 연산하고, 기존의 데이터 배치를 고려하여 데이터를 재배치함으로써 네트워크의 부하를 감소시키고 처리 성능을 최적화 하는 것이 가능하다.

Ⅱ. 제안하는 데이터 재배치 알고리즘

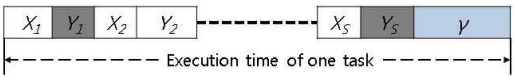
본 논문에서는 비-전용 분산 컴퓨팅 환경에서 맵-리듀스의 성능 최적화를 위한 효율적인 데이터 재배치 알고리즘을 제안한다. 제안하는 기법에서는 노드의 가용성 분석 모델을 기반으로 데이터 블록 비율을 연산하고, 기존의 데이터 배치를 고려하여 데이터를 전송함으로써 네트워크 부하를 감소시킨다. 그림 1은 제안하는 데이터 재배치 알고리즘을 위한 시스템 아키텍쳐를 나타낸다. 제안하는 기법의 시스템 아키텍쳐는 기존 기법의 시스템 아키텍쳐와 유사하게 하둡 프레임워크의 NameNode와 Client를 수정하였다. NameNode는 Heart-Beat Collector를 통해 노드의 장애시간을 기록하고, Performance Predictor를 추가하여 노드의 평균 장애시간을 분석한다. 분석 모델을 기반으로 Data Block Distributor는 데이터 블록을 배치한다. Client는 데이터 재배치를 효율적으로 처리하기 위해, 기존의 데이터 위치를 파악할 수 있도록 하둡 명령어로 existingData를 추가한다.

1) 본 연구는 KISTI의 국가슈퍼컴퓨팅서비스 개발 및 기술 연구과제(K-13-L01-C02)의 지원 및 2012년도 정부(교육과학기술부)의 재원으로 한국연구재단의 기초연구사업 지원을 받아 수행된 것임(2012R1A1A2041898)



▶▶ 그림 1. 제안하는 아키텍쳐

본 논문에서는 기존의 비-전용 분산 컴퓨팅 환경에서의 노드 가용성 분석 모델[2] 기반의 새로운 데이터 재배치 알고리즘을 제안한다. 비-전용 분산 컴퓨팅 환경에서의 가용성 분석 모델은 노드에 발생되는 평균 장애시간을 포함한 태스크의 실행시간을 기반으로 노드의 평균 맵-리듀스의 처리시간을 예측하여 노드의 가용성을 판단한다. 장애시간을 포함한 단일 태스크 실행시간은 그림 2와 같이 단일 태스크가 수행되는 동안 장애가 발생하지 않고 완벽하게 태스크가 수행되는 시간이다. 예를 들어 태스크를 실행하는 동안 X동안 장애가 발생하면 처음부터 다시 태스크를 수행하며 단일 태스크가 완전히 완료될 때까지 반복한다. 이와 같은 과정을 통해 단일 태스크가 수행되는 시간은 장애 발생 시간 X, 장애로 인해 재시작하는 시간 Y, 단일 태스크를 완전히 수행한 시간 γ의 합과 같다.



▶▶ 그림 2. 단일 태스크 실행 시간

단일 태스크의 평균 수행시간은 고장확률 밀도함수와 푸아송 분포를 통하여 평균 장애 시간(E(X))과 평균 재시작 시간(E(Y)), 그리고 장애가 일어날 평균 횟수(E(S))를 예측한다. 그러므로 단일 태스크의 평균 수행 시간은 식 (1)과 같이 나타낸다.

$$E(X) = \int_{t=0}^{\gamma} t f_x(X=t) = \frac{1}{\lambda} + \frac{\gamma}{1-e^{-\lambda t}}$$
$$E(Y) = \frac{\mu}{1-\lambda\mu}$$
$$E(S) = \frac{1-e^{-\gamma\lambda}}{e^{-\gamma\lambda}} = e^{\gamma\lambda} - 1 \quad (1)$$
$$E(T) = E(E(T|S)) = E(\gamma + SE(Y) + SE(Y))$$
$$= (e^{\gamma\lambda} - 1)(\frac{1}{\lambda} + \frac{\mu}{1-\lambda\mu})$$

맵-리듀스는 n개의 노드에서 m개의 입력 데이터 블록을 처리하므로 성능 최적화를 위해 가용성 분석 모델을 기반으로 노드에 데이터 블록 비율을 연산한다. 데이터 블록 비율은 식 (2)와 같이 태스크의 실행시간의 반비례하여 연산하고 식 (3)과 같이 n개의 노드에 m개의 블록들이 비율에 맞게 배치되도록 연산한다.

$$Rate_i = \frac{1/E_i(T)}{\Phi} \quad (2)$$
$$w_i = m \times rate_i \quad (3)$$

데이터 블록의 전송을 최소화하기 위해, 기존의 배치된 데이터를 고려하여 노드의 데이터 비율에 적합하도록 최소한의 데이터 블록을 전송한다. 식 (4)와 같이 기존의 데이터 블록수와 노드에 적합한 데이터 블록수를 비교하여 데이터 블록을 전송해야할 노드와 데이터 블록 수를 연산한다.

$$Deviation_i = pre\_w_i - w_i \quad (4)$$

식(4)에서 계산된 데이터 블록수의 편차가 Deviationi 〉 0 이면, 노드 i에 저장된 데이터 블록수가 재편성한 데이터 블록수보다 많으므로 다른 노드에 전송해야 한다. 이 때, 노드 i에서 전송할 목적지 노드는 식 (5)를 통해 현재 노드에 가까운 노드를 선출한다.

$$Distance = |Chunk\_IP_i - Chunk\_IP_j| \quad (5)$$

선출된 노드를 중심으로 Deviationj 〈 0 인 노드를 탐색하여 데이터 블록을 전송함으로써 네트워크 부하를 감소시키고 맵-리듀스의 처리 성능을 최적화한다.

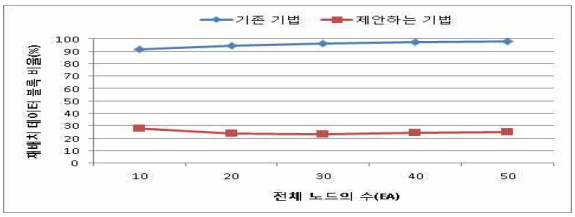
Ⅲ. 성능 평가

본 논문에서는 제안하는 기법의 우수성을 보이기 위해 기존 기법[2]과의 시뮬레이션을 통한 성능 비교 평가를 수행하였다. 노드의 결함 발생은 임의로 발생시키며, 본 시뮬레이션은 Java 시뮬레이터를 바탕으로 표 1과 같은 성능 평가 환경을 구성하여 수행하였다.

표 1. 성능평가 환경

환경변수	설정값
노드의 수 (EA)	10 ~ 50
데이터량 (GB)	6.4~ 32
데이터 블록 사이즈 (MB)	64
노드 당 청크 수 (EA)	10

그림 3은 노드 수에 따른 재배치 데이터 블록 비율을 비교 평가한 결과이다. 기존 재배치 알고리즘[2]의 경우 기존의 데이터 배치를 고려하지 않고 재배치함으로써 데이터 블록의 재전송 할 비율이 높아진다. 제안하는 기법은 기존의 데이터 배치를 고려함으로써 데이터 블록의 재전송 비율을 감소시킨다. 성능 평가 결과, 제안하는 기법의 데이터 재배치 블록의 비율이 평균 약 75% 감소하였다.



▶▶ 그림 3. 노드 수에 따른 재배치 데이터 블록 비율

Ⅳ. 결론

본 논문에서는 비-전용 분산 컴퓨팅 환경에서 맵-리듀스의 성능 최적화를 위한 효율적인 데이터 재배치 알고리즘을 제안하였다. 제안하는 기법에서는 노드의 가용성 분석 모델을 기반으로 데이터 블록 비율을 연산하고, 기존의 데이터 배치를 고려하여 데이터를 전송함으로써 네트워크 부하를 감소시키는 것이 가능하다. 성능평가 결과, 제안하는 기법의 데이터 재배치 블록 비율이 약 75% 감소하였다. 이를 통해 제안하는 기법에서 데이터 블록의 재배치를 위한 네트워크의 부하가 감소되는 것을 확인하였다. 향후 연구로는 데이터 재배치 후 맵-리듀스의 처리 속도를 향상시키기 위한 알고리즘을 접목하는 것이다.

■ 참 고 문 헌 ■

[1] Hadoop, http://hadoop.apache.org
[2] H. Jin, X. Yang, X.-H. Sun, I. Raicu, "ADAPT: Availability-Aware MapReduce Data Placement for Non-Dedicated Distributed Computing," Proc. of IEEE Int. Conf. on Distributed Computing Systems, pp. 516-525, 2012.