



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2015-0062643

(43) 공개일자 2015년06월08일

(51) 국제특허분류(Int. Cl.)

G06F 9/46 (2006.01) G06F 9/54 (2006.01)

(21) 출원번호 10-2013-0147505

(22) 출원일자 2013년11월29일

심사청구일자 2013년11월29일

(71) 출원인

서강대학교산학협력단

서울특별시 마포구 백범로 35 (신수동, 서강대학교)

(72) 발명자

박성용

서울특별시 마포구 마포대로 195, 102동 401호 (아현동, 마포 래미안 푸르지오)

이권용

서울특별시 마포구 백범로 199, 2704호 (신공덕동, 메트로디오빌)

(뒷면에 계속)

(74) 대리인

특허법인충현

전체 청구항 수 : 총 11 항

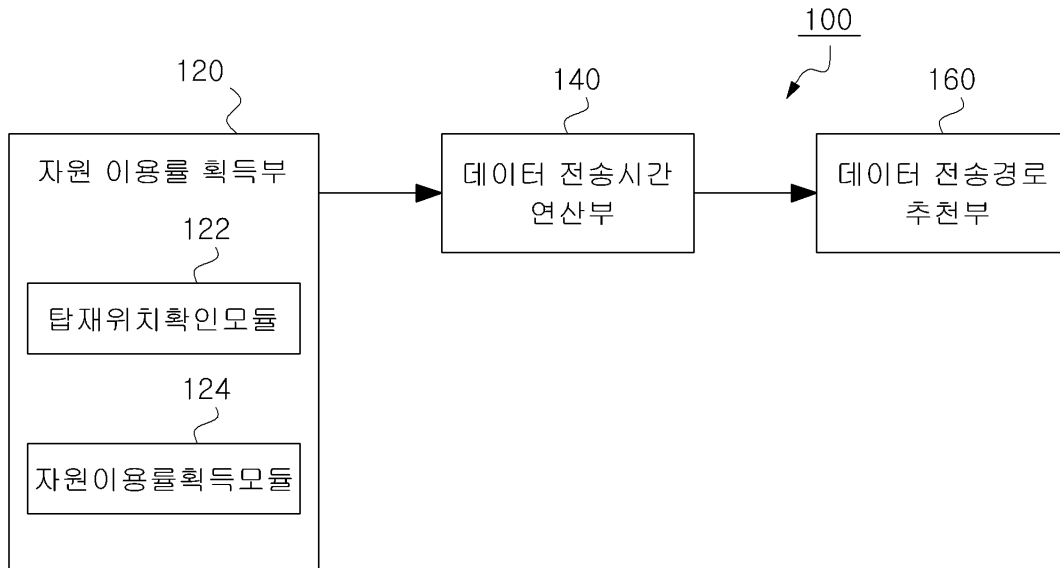
(54) 발명의 명칭 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법 및 시스템

(57) 요약

본 발명은 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법 및 시스템에 관한 것으로, 보다 구체적으로는 물리머신에 탑재된 맵퍼 가상머신 및 리듀서 가상머신간에 데이터가 전송되는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법에 있어서, 자원 이용률 획득부가 상기 가상머신 및 드라이버 도메인의 자원 이용률을 획득하

(뒷면에 계속)

대표도 - 도1



는 단계; 데이터 전송시간 연산부가 상기 가상머신 및 드라이버 도메인의 자원 이용률 및 전송하고자 하는 데이터 크기에 기초하여 상기 맵퍼 가상머신 및 리듀서 가상머신간 복수 개의 데이터 전송경로를 통한 데이터 전송시간을 예측하여 연산하는 단계; 및 데이터 전송경로 추천부가 연산한 데이터 전송시간 중 최소시간에 해당하는 데이터 전송경로를 검색하고, 검색된 데이터 전송경로를 상기 맵퍼 가상머신 및 리듀서 가상머신간에 최적의 데이터 전송경로로 추천하는 단계;를 포함하는 것을 특징으로 한다. 이러한 구성에 의해, 본 발명의 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법 및 시스템은 가상머신에 부착된 블록 장치를 동적으로 탈착한 후, 런타임(runtime) 동안 다른 가상머신에 재부착할 수 있는 것을 이용하여, 가상머신간에 어떠한 네트워크 전송도 없이 데이터를 용이하고 신속하게 전송할 수 있는 효과가 있다.

(72) 발명자

남윤성

서울특별시 마포구 와우산로38길 28, 303호 (동교동)

김택희

부산광역시 남구 분포로 111, 111동 2001호 (용호동, 엘지메트로시티아파트)

이 발명을 지원한 국가연구개발사업

과제고유번호 1415128733(H0301-13-3006)

부처명 미래창조과학부

연구관리전문기관 정보통신산업진흥원

연구사업명 대학IT연구센터 육성지원사업

연구과제명 소셜 미디어 서비스를 위한 클라우드 플랫폼 및 응용서비스 기술 개발

기 여 율 1/1

주관기관 건국대학교 산학협력단

연구기간 2013.01.01 ~ 2013.12.31

명세서

청구범위

청구항 1

물리머신에 탑재된 맵퍼 가상머신 및 리듀서 가상머신간에 데이터가 전송되는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법에 있어서,

자원 이용률 획득부가 상기 가상머신 및 드라이버 도메인의 자원 이용률을 획득하는 단계;

데이터 전송시간 연산부가 상기 가상머신 및 드라이버 도메인의 자원 이용률 및 전송하고자 하는 데이터 크기에 기초하여 상기 맵퍼 가상머신 및 리듀서 가상머신간 복수 개의 데이터 전송경로를 통한 데이터 전송시간을 예측하여 연산하는 단계; 및

데이터 전송경로 추천부가 연산한 데이터 전송시간 중 최소시간에 해당하는 데이터 전송경로를 검색하고, 검색된 데이터 전송경로를 상기 맵퍼 가상머신 및 리듀서 가상머신간에 최적의 데이터 전송경로로 추천하는 단계;

를 포함하는 것을 특징으로 하는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법.

청구항 2

제1항에 있어서,

상기 자원 이용률 획득부가 상기 가상머신 및 드라이버 도메인의 자원 이용률을 획득하는 단계는

데이터를 전송하고자 하는 맵퍼 가상머신 및 리듀서 가상머신이 동일한 물리머신 내 탑재되어 있는지 여부를 판단하는 과정; 및

상기 맵퍼 가상머신 및 리듀서 가상머신이 동일한 물리머신 내 탑재되는 경우에는 맵 태스크를 수행하는 맵퍼 가상머신의 자원 이용률, 리듀서 가상머신의 자원 이용률 및 리듀서 가상머신의 드라이버 도메인의 자원 이용률을 획득하는 과정;

을 포함하는 것을 특징으로 하는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법.

청구항 3

제2항에 있어서,

상기 자원 이용률 획득부가 상기 가상머신 및 드라이버 도메인들의 자원 이용률을 획득하는 단계는

상기 맵퍼 가상머신 및 리듀서 가상머신이 서로 다른 물리머신 내 탑재되는 경우에는 상기 맵 태스크를 수행하는 맵퍼 가상머신의 드라이버 도메인의 자원 이용률을 획득하는 과정;

을 더 포함하는 것을 특징으로 하는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법.

청구항 4

제1항에 있어서,

상기 데이터 전송시간 연산부가 데이터 전송시간을 연산하는 단계는

상기 가상머신 및 드라이버 도메인의 자원 이용률 및 전송하고자 하는 데이터 크기를 다중 회귀 분석(multiple regression analysis)방법에 적용하여 데이터 전송시간을 예측하여 연산하는 것을 특징으로 하는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법.

청구항 5

제4항에 있어서,

상기 데이터 전송시간 연산부가 데이터 전송시간을 연산하는 단계는

상기 맵퍼 가상머신 및 리듀서 가상머신간 복수 개의 데이터 전송경로 중 네트워크 내 데이터 전송경로의 각 구간별 데이터 전송시간을 연산하는 과정;

상기 맵퍼 가상머신 및 리듀서 가상머신간 복수 개의 데이터 전송경로 중 블록 장치 재구성에 따른 데이터 전송경로의 데이터 전송시간을 연산하는 과정;

상기 네트워크 내 데이터 전송경로의 각 구간별 데이터 전송시간과 상기 블록 장치 재구성에 따른 데이터 전송경로의 데이터 전송시간을 상호 비교하여, 적은 데이터 전송시간을 해당 구간에 할당하는 과정; 및

하나의 데이터 전송경로를 구성하는 적어도 하나의 구간에 할당된 데이터 전송시간을 각 구간별로 모두 합산하여 상기 복수 개의 데이터 전송경로별 데이터 전송시간을 연산하는 과정;

을 포함하는 것을 특징으로 하는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법.

청구항 6

제1항에 있어서,

상기 가상 맵리듀스는

하둡 맵리듀스(Hadoop MapReduce) 환경에서 수행되는 것을 특징으로 하는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법.

청구항 7

제1항 내지 제6항 중 어느 한 항에 따른 방법을 컴퓨터로 실행하기 위한 프로그램이 기록된 컴퓨터 판독가능 기록매체.

청구항 8

물리머신에 탑재된 맵퍼 가상머신 및 리듀서 가상머신간에 데이터가 전송되는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 시스템에 있어서,

상기 가상머신 및 드라이버 도메인의 자원 이용률을 획득하는 자원 이용률 획득부;

상기 가상머신 및 드라이버 도메인의 자원 이용률 및 전송하고자 하는 데이터 크기에 기초하여 상기 맵퍼 가상머신 및 리듀서 가상머신간 복수 개의 데이터 전송경로를 통한 데이터 전송시간을 예측하여 연산하는 데이터 전송시간 연산부; 및

상기 데이터 전송시간 중 최소시간에 해당하는 데이터 전송경로를 검색하고, 검색된 데이터 전송경로를 상기 맵퍼 가상머신 및 리듀서 가상머신간에 최적의 데이터 전송경로로 추천하는 데이터 전송경로 추천부;

를 포함하는 것을 특징으로 하는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 시스템.

청구항 9

제8항에 있어서,

상기 자원 이용률 획득부는

데이터를 전송하고자 하는 맵퍼 가상머신 및 리듀서 가상머신이 동일한 물리머신 내 탑재되어 있는지 여부를 판단하는 탑재위치확인모듈; 및

상기 맵퍼 가상머신 및 리듀서 가상머신이 동일한 물리머신 내 탑재되는 경우에는 맵 태스크를 수행하는 맵퍼 가상머신의 자원 이용률, 리듀서 가상머신의 자원 이용률 및 리듀서 가상머신의 드라이버 도메인의 자원 이용률을 획득하는 자원이용률획득모듈;

을 포함하는 것을 특징으로 하는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 시스템.

청구항 10

제9항에 있어서,

상기 자원이용률획득모듈은

상기 맵퍼 가상머신 및 리듀서 가상머신이 서로 다른 물리머신 내 탑재되는 경우에는 상기 맵 태스크를 수행하는 맵퍼 가상머신의 드라이버 도메인의 자원 이용률을 더 획득하는 것을 특징으로 하는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 시스템.

청구항 11

제8항에 있어서,

상기 가상 맵리듀스는

하둡 맵리듀스(Hadoop MapReduce) 환경에서 수행되는 것을 특징으로 하는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 시스템.

발명의 설명

기술 분야

[0001]

본 발명은 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법 및 시스템에 관한 것으로, 특히 물리머신에 탑재된 맵퍼 가상머신과 리듀서 가상머신간에 신속한 데이터 전송을 통해 맵리듀스의 어플리케이션 성능을 향상시킬 수 있는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법 및 시스템에 관한 것이다.

배경 기술

[0002]

최근 들어, 가상화 기술의 발달과 성능향상 및 대용량의 하드웨어 인프라 구축 기술이 구현됨에 따라, 아마존을 비롯한 대표적인 글로벌 기업에서는 가상화 기반의 클라우드 컴퓨팅 서비스를 상용화 하고 있는 추세이다.

[0003]

특히, 클라우드 환경에서는 가상화를 통해 서비스 제공자로부터 사용자가 원하는 만큼의 컴퓨팅 자원을 할당받을 수 있으며, 사용자에게 할당된 컴퓨팅 자원에 대한 포괄적인 제어를 통해 사용자의 요구사항에 맞춰 컴퓨팅 자원 할당을 신속하게 확장하거나, 축소할 수 있다.

[0004]

따라서, 물리머신 기반이던 클러스터가 클라우드의 이점을 얻기 위해, 점차 가상화 클러스터로 구축 환경이 변화되었다.

[0005]

이러한, 클러스터 컴퓨팅 환경은 대규모 병렬 또는 분산 어플리케이션을 실행하기 위해 큰 성장을 지속해왔는데, 고성능 서버 클러스터들과 고가용성 클러스터들 같은 클러스터 서비스들을 제공하기 위해, 서로 연결된 다수의 물리머신으로 구성된다.

[0006]

특히, 아마존 EC2 및 GoGrid 와 같은 클라우드 서비스의 등장으로 인하여, 클라우드 상에서 가상머신들을 이용하여 가상 클러스터를 구성하는 것이 많은 각광을 받고 있다.

[0007]

하지만 네트워크 입출력의 가상화는 클라우드 내의 가상머신들이 서로 한정된 네트워크 자원을 공유하고 대개

단일 프리빌리지드 도메인(privileged domain)에 의해 중개되기 때문에, 클라우드 환경에서의 명확한 성능 병목 지점으로 알려져 있다. 예를 들어, 클라우드를 구축하기 위한 가상머신모니터(VMM, Virtual Machine Monitor)로서 널리 이용되는 Xen의 경우, 드라이버 도메인(driver domain)이 입출력 장치들을 직접 접근하기 위해 모든 네이티브 디바이스 드라이버들(Native Device Drivers)을 가지고 있는 반면, 가상머신들은 가상 입출력 장치들을 통해 모든 입출력 요청을 드라이버 도메인에게 전송한다. 이에 따라, 드라이버 도메인은 가상머신의 네트워크 활동을 중개하기 위해 많은 CPU 자원을 소모함으로써, 전체 시스템의 성능이 심각하게 저하되는 문제점이 발생했다.

[0008] 상술한 바와 같이, 본 발명의 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법 및 시스템을 살펴보면 다음과 같다.

[0009] 선행기술 1은 한국공개특허공보 제2012-0071979호(2012.07.03)로서, 클라우드 컴퓨팅 시스템의 자원관리장치 및 방법에 관한 것이다. 이러한 선행기술 1은 네트워크 상의 컴퓨팅 자원을 가상화한 가상머신들을 복수의 사용자에게 제공하는 클라우드 컴퓨팅 시스템의 자원관리장치에 있어서, 사용자로부터 사용할 컴퓨팅 자원의 사용시점 및 자원사용량을 포함하는 개인 스케줄링 정보를 입력받는 스케줄러입력부; 상기 입력받은 개인 스케줄링 정보를 기초로 상기 사용자가 사용할 가상머신이 생성될 위치를 배정하는 위치배정부; 및 상기 사용시점에 상기 자원사용량만큼 컴퓨팅 자원을 사용할 수 있는 가상머신을 상기 배정된 위치에 생성하여 상기 사용자에게 제공하는 가상머신제공부를 포함함으로써, 클라우드 컴퓨팅 시스템은 사용자의 시간대별로 사용할 컴퓨팅 자원에 관한 개인 스케줄링 정보를 고려하여, 컴퓨팅 자원의 할당 및 관리를 최적화할 수 있고, 컴퓨팅 자원에 대한 비용절감 및 사용자에게 대한 최적의 서비스를 제공할 수 있게 된다.

[0010] 또한, 선행기술 2는 한국공개특허공보 제2013-0022091호(2013.03.06)로서, 클라우드 컴퓨팅 서버 시스템의 가상머신 제어 장치 및 방법에 관한 것이다. 이러한 선행기술 2는 가상화된 컴퓨팅 자원을 할당한 복수의 가상머신들을 통해 가상화된 컴퓨팅 환경을 클라이언트에 제공하는 클라우드 컴퓨팅 서버 시스템의 가상머신 제어 장치에 있어서, 상기 가상머신들에 각각 할당된 복수의 가상 스토리지들에 접근할 수 있는 인터페이스를 제공하는 스토리지 인터페이스를 통해 상기 복수의 가상 스토리지들의 이용 가능 여부를 지속적으로 모니터링하는 모니터링부; 및 상기 모니터링된 결과 상기 복수의 가상 스토리지들 중에서 이용가능하지 않은 서비스 불가 상태의 가상 스토리지를 검출하고, 상기 복수의 가상머신들 중 상기 검출된 가상 스토리지에 대응되는 가상머신에서 이용 중인 컴퓨팅 자원을 반납하지 않은 상태에서 수행 중인 프로세스를 일시중지시키는 제어부를 포함함으로써, 반가상화 기반의 클라우드 컴퓨팅 서버 시스템에서 하이퍼바이저와 스토리지 사이의 연결 네트워크 상에 문제가 발생하거나 스토리지 자체에서 일시적으로 가상머신에 대한 파일 입출력 서비스를 제공하지 못하는 상황에서도 해당 가상머신의 파일 시스템의 일관성 손상을 방지할 수 있다.

발명의 내용

해결하려는 과제

[0011] 상기와 같은 종래 기술의 문제점을 해결하기 위해, 본 발명은 가상머신간에 신속한 데이터 전송을 통해 맵리듀스 어플리케이션의 성능을 향상시킬 수 있는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법 및 시스템을 제공하고자 한다.

과제의 해결 수단

[0012] 위와 같은 과제를 해결하기 위한 본 발명의 한 실시 예에 따른 물리머신에 탑재된 맵퍼 가상머신 및 리듀서 가상머신간에 데이터가 전송되는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법은 자원 이용률 획득부가 상기 가상머신 및 드라이버 도메인의 자원 이용률을 획득하는 단계; 데이터 전송시간 연산부가 상기 가상머신 및 드라이버 도메인의 자원 이용률 및 전송하고자 하는 데이터 크기에 기초하여 상기 맵퍼 가상머신 및 리듀서 가상머신간 복수 개의 데이터 전송경로를 통한 데이터 전송시간을 예측하여 연산하는 단계; 및 데이터 전송경로 추천부가 연산한 데이터 전송시간 중 최소시간에 해당하는 데이터 전송경로를 검색하고, 검색된 데이터 전송경로를 상기 맵퍼 가상머신 및 리듀서 가상머신간에 최적의 데이터 전송경로로 추천하는 단계;를 포함한다.

[0013] 보다 바람직하게는 데이터를 전송하고자 하는 맵퍼 가상머신 및 리듀서 가상머신이 동일한 물리머신 내 탑재되

어 있는지 여부를 판단하는 과정; 및 상기 맵퍼 가상머신 및 리듀서 가상머신이 동일한 물리머신 내 탑재되는 경우에는 맵 태스크를 수행하는 맵퍼 가상머신의 자원 이용률, 리듀서 가상머신의 자원 이용률 및 리듀서 가상머신의 드라이버 도메인의 자원 이용률을 획득하는 과정;을 포함하는 상기 자원 이용률 획득부가 상기 가상머신 및 드라이버 도메인의 자원 이용률을 획득하는 단계를 포함할 수 있다.

[0014] 특히, 상기 맵퍼 가상머신 및 리듀서 가상머신이 서로 다른 물리머신 내 탑재되는 경우에는 상기 맵 태스크를 수행하는 맵퍼 가상머신의 드라이버 도메인의 자원 이용률을 획득하는 과정;을 더 포함하는 상기 자원 이용률 획득부가 상기 가상머신 및 드라이버 도메인들의 자원 이용률을 획득하는 단계를 포함할 수 있다.

[0015] 특히, 상기 가상머신 및 드라이버 도메인의 자원 이용률 및 전송하고자 하는 데이터 크기를 다중 회귀 분석(multiple regression analysis)방법에 적용하여 데이터 전송시간을 예측하여 연산하는 상기 데이터 전송시간 연산부가 데이터 전송시간을 연산하는 단계를 포함할 수 있다.

[0016] 보다 바람직하게는 상기 맵퍼 가상머신 및 리듀서 가상머신간 복수 개의 데이터 전송경로 중 네트워크 내 데이터 전송경로의 각 구간별 데이터 전송시간을 연산하는 과정; 상기 맵퍼 가상머신 및 리듀서 가상머신간 복수 개의 데이터 전송경로 중 블록 장치 재구성에 따른 데이터 전송경로의 데이터 전송시간을 연산하는 과정; 상기 네트워크 내 데이터 전송경로의 각 구간별 데이터 전송시간과 상기 블록 장치 재구성에 따른 데이터 전송경로의 데이터 전송시간을 상호 비교하여, 적은 데이터 전송시간을 해당 구간에 할당하는 과정; 및 하나의 데이터 전송경로를 구성하는 적어도 하나의 구간에 할당된 데이터 전송시간을 각 구간별로 모두 합산하여 상기 복수 개의 데이터 전송경로별 데이터 전송시간을 연산하는 과정; 을 포함하는 상기 데이터 전송시간 연산부가 데이터 전송시간을 연산하는 단계를 포함할 수 있다.

[0017] 특히, 하둡 맵리듀스(Hadoop MapReduce) 환경에서 수행되는 가상 맵리듀스를 포함할 수 있다.

[0018] 위와 같은 과제를 해결하기 위한 본 발명의 다른 실시 예에 따른 물리머신에 탑재된 맵퍼 가상머신 및 리듀서 가상머신간에 데이터가 전송되는 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 시스템은 상기 가상머신 및 드라이버 도메인의 자원 이용률을 획득하는 자원 이용률 획득부; 상기 가상머신 및 드라이버 도메인의 자원 이용률 및 전송하고자 하는 데이터 크기에 기초하여 상기 맵퍼 가상머신 및 리듀서 가상머신간 복수 개의 데이터 전송경로를 통한 데이터 전송시간을 예측하여 연산하는 데이터 전송시간 연산부; 및 상기 데이터 전송시간 중 최소시간에 해당하는 데이터 전송경로를 검색하고, 검색된 데이터 전송경로를 상기 맵퍼 가상머신 및 리듀서 가상머신간에 최적의 데이터 전송경로로 추천하는 데이터 전송경로 추천부;를 포함하는 것을 특징으로 한다.

[0019] 보다 바람직하게는 데이터를 전송하고자 하는 맵퍼 가상머신 및 리듀서 가상머신이 동일한 물리머신 내 탑재되어 있는지 여부를 판단하는 탑재위치확인모듈; 및 상기 맵퍼 가상머신 및 리듀서 가상머신이 동일한 물리머신 내 탑재되는 경우에는 맵 태스크를 수행하는 맵퍼 가상머신의 자원 이용률, 리듀서 가상머신의 자원 이용률 및 리듀서 가상머신의 드라이버 도메인의 자원 이용률을 획득하는 자원이용률획득모듈;을 포함하는 자원 이용률 획득부를 포함할 수 있다.

[0020] 특히, 상기 맵퍼 가상머신 및 리듀서 가상머신이 서로 다른 물리머신 내 탑재되는 경우에는 상기 맵 태스크를 수행하는 맵퍼 가상머신의 드라이버 도메인의 자원 이용률을 더 획득하는 자원이용률획득모듈을 포함할 수 있다.

[0021] 특히, 하둡 맵리듀스(Hadoop MapReduce) 환경에서 수행되는 가상 맵리듀스를 포함할 수 있다.

발명의 효과

[0022] 본 발명의 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법 및 시스템은 가상머신에 부착된 블록 장치를 동적으로 탈착 후, 런타임(runtime) 동안 다른 가상머신에 재부착하여, 가상머신간에 데이터 전송을 용이하고 신속하게 수행할 수 있는 효과가 있다.

[0023] 또한 본 발명의 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법 및 시스템은 맵퍼 가상머신과 리듀서 가상머신간 데이터 전송 시 발생하는 오버헤드를 최소화할 수 있는 효과가 있다.

[0024] 더불어, 본 발명의 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법 및 시스템은 맵리듀스 어플리케이션의 실행시간을 단축시킬 수 있는 효과가 있다.

도면의 간단한 설명

- [0025] 도 1은 본 발명의 일 실시 예에 따른 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 시스템의 블록도이다.
- 도 2는 본 발명의 다른 실시 예에 따른 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법의 순서도이다.
- 도 3은 블록 장치 재구성을 이용한 맵리듀스 프레임워크를 나타낸 블록도이다.
- 도 4는 맵퍼 가상머신들과 리듀서 가상머신간 데이터 전송시간을 나타낸 그래프이다.
- 도 5는 맵퍼 가상머신들과 리듀서 가상머신간 데이터 전송 시 자원 이용률을 나타낸 그래프이다.
- 도 6은 맵퍼 가상머신과 리듀서 가상머신간 데이터 전송 시 각 물리머신 별 가상머신의 누적 디스크 입출력 대기시간을 나타낸 그래프이다.
- 도 7은 데이터 크기별 전송시간을 나타낸 그래프이다.
- 도 8은 데이터를 수신하는 가상머신의 가용 자원 이용률별 전송 시간을 나타낸 그래프이다.
- 도 9는 드라이버 도메인의 가용 자원 이용률별 전송 시간을 나타낸 그래프이다.
- 도 10은 맵퍼 가상머신과 리듀서 가상머신간 데이터 전송경로를 나타낸 도면이다.
- 도 11은 가상머신별 데이터 전송과정을 나타낸 도면이다.
- 도 12는 동적 블록 재구성 장치를 통한 가상 맵리듀스 클러스터에서 선택된 전송경로의 비율을 나타낸 그래프이다.
- 도 13은 네 개의 물리 CPU 사용 시, 블록 크기별 맵리듀스 수행시간을 나타낸 그래프이다.
- 도 14는 세 개의 물리 CPU 사용 시, 블록 크기별 맵리듀스 수행시간을 나타낸 그래프이다.
- 도 15는 두 개의 물리 CPU 사용 시, 블록 크기별 맵리듀스 수행시간을 나타낸 그래프이다.

발명을 실시하기 위한 구체적인 내용

- [0026] 이하, 본 발명을 바람직한 실시 예와 첨부한 도면을 참고로 하여 본 발명이 속하는 기술 분야에서 통상의 지식을 가진 자가 용이하게 실시할 수 있도록 상세히 설명한다. 그러나 본 발명은 여러 가지 상이한 형태로 구현될 수 있으며, 여기에서 설명하는 실시 예에 한정되는 것은 아니다.
- [0027] 먼저, 본 발명을 설명하기에 앞서, 본 발명이 구현되는 가상환경에 대하여 간략히 설명하도록 한다.
- [0028] 가상환경의 구현 시 사용되는 가상머신은 특정한 프로그램을 실행하되, 하드웨어와 직접적인 통신이 이루어지지 않는 가상의 컴퓨터를 말한다. 이러한 가상머신은 시스템 가상머신과 프로세스 가상머신으로 크게 나누어지는데, 본 발명에서는 시스템 가상머신에 대한 가상환경을 중심으로 설명하도록 한다.
- [0029] 시스템 가상머신은 완전한 시스템 플랫폼 즉, 완전한 운영체제(Operating System)의 실행을 지원하며, 각각의 운영체제를 실행하는 가상 머신 사이의 기초가 되는 물리 컴퓨터를 다중화(multiplex)한다. 이처럼, 가상화를 제공하는 소프트웨어 계층은 가상머신 모니터 또는 하이퍼바이저(Hypervisor)라고 하며, 상기 하이퍼바이저는 하드웨어 또는 호스트 운영체제의 위에서 실행할 수 있다.
- [0030] 하둡(Hadoop)은 분산처리 플랫폼을 제공하는 아파치 오픈소스 프로젝트로, 신뢰성(Reliability)과 확장성(Scalability)을 보장하는 환경을 제공하여 현재 아마존(Amazon), 야후(Yahoo!), 페이스북(Facebook) 등의 많은 업체에서 사용되고 있다.
- [0031] 이러한 하둡을 구성하는 중요한 요소 중 하나인 맵리듀스는 구글(Google)에서 제안한 맵리듀스 프로그래밍 모델을 하둡에 맞춰 구현한 소프트웨어 프레임워크로서, 분산환경에서 대용량 데이터를 처리하는데 있어서, 간단하고 직관적인 인터페이스를 제공한다.
- [0032] 이러한 맵리듀스는 그 간편성, 확장성, 성능으로 인하여 생물 정보학(bioinformatics)이나 자연 언어 처리(natural language processing) 등과 같은 대규모 데이터 처리 어플리케이션을 위한 효과적인 모델 중 하나이다. 맵리듀스의 핵심 개념은 하나의 job을 다수의 작은 태스크로 분할하여 클러스터 된 자원 상에서 처리

하는 것이다. 이 맵리듀스 모델은 맵(map), 셔플(shuffle), 리듀스(reduce)인 세 개의 주 단계로 구성된다. 각각의 맵리듀스 job은 맵 또는 리듀스 단계에서 처리되는 몇 개의 태스크(task)로 나누어진다. 셔플 단계는 상기 맵 단계 및 리듀스 단계를 중개하는 중간 단계로서의 역할을 한다.

[0033] 분산 파일 시스템의 데이터는 기본적으로 다수의 작은 블록(block)으로 나누어진다. 각각의 맵퍼(mapper)는 분할된 작은 데이터를 이용하여 맵 태스크를 수행하고 그것의 출력(output)을 로컬 디스크(local disk)에 저장한다. 셔플 단계에서 리듀서는 맵 태스크들의 출력들을 복사하고 병합하여 리듀스 단계에서 병합된 데이터를 입력으로 리듀스 태스크를 수행한다.

[0034] 이에 따라, 물리머신 내 데이터를 송신하는 맵퍼 가상머신과 송신된 데이터를 수신하는 리듀서 가상머신 간에 데이터 전송 시 이용하는 데이터 전송경로를 추천하는 시스템에 대하여 이하 자세히 살펴보도록 한다.

[0035] 도 1은 본 발명의 일 실시 예에 따른 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 시스템의 블록도이다.

[0036] 도 1에 도시된 바와 같이, 본 발명의 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 시스템(100)은 자원 이용률 획득부(120), 데이터 전송시간 연산부(140) 및 데이터 전송경로 추천부(160)를 포함한다.

[0037] 자원 이용률 획득부(120)는 가상머신 및 드라이버 도메인의 자원 이용률을 획득한다. 이러한 자원 이용률 획득부(120)는 탑재위치확인모듈(122) 및 자원이용률획득모듈(124)을 포함한다.

[0038] 탑재위치확인모듈(122)은 데이터를 전송하고자 하는 맵퍼 가상머신 및 리듀서 가상머신이 동일한 물리머신 내 탑재되어 있는지 여부를 판단한다.

[0039] 자원이용률획득모듈(124)은 상기 맵퍼 가상머신 및 리듀서 가상머신이 동일한 물리머신 내 탑재되는 경우에는 맵 태스크를 수행하는 맵퍼 가상머신의 자원 이용률, 리듀서 가상머신의 자원 이용률 및 리듀서 가상머신의 드라이버 도메인의 자원 이용률을 획득한다. 또한, 자원이용률획득모듈(124)은 상기 맵퍼 가상머신 및 리듀서 가상머신이 서로 다른 물리머신 내 탑재되는 경우에는 상기 맵 태스크를 수행하는 맵퍼 가상머신의 드라이버 도메인의 자원 이용률을 더 획득할 수 있다.

[0040] 데이터 전송시간 연산부(140)는 가상머신 및 드라이버 도메인의 자원 이용률 및 전송하고자 하는 데이터 크기에 기초하여 상기 맵퍼 가상머신 및 리듀서 가상머신간 복수 개의 데이터 전송경로를 통한 데이터 전송시간을 예측하여 연산한다.

[0041] 데이터 전송경로 추천부(160)는 상기 데이터 전송시간 중 최소시간에 해당하는 데이터 전송경로를 검색하고, 검색된 데이터 전송경로를 상기 맵퍼 가상머신 및 리듀서 가상머신간에 최적의 데이터 전송경로로 추천한다.

[0042] 이러한 가상 맵리듀스는 하둡 맵리듀스(Hadoop MapReduce) 환경에서 수행될 수 있으며, 하둡 이외에도 다양한 종류의 가상 맵리듀스 환경에서 적용될 수 있다.

[0043] 이하, 도 2를 참조하여 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법에 대하여 살펴보도록 한다.

[0044] 도 2는 본 발명의 다른 실시 예에 따른 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법의 순서도이다.

[0045] 도 2에 도시된 바와 같이, 본 발명의 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법은 자원 이용률 획득부(120)가 상기 가상머신 및 드라이버 도메인의 자원 이용률을 획득한다(S210). 이때, 상기 자원 이용률 획득부(120)가 먼저, 데이터를 전송하고자 하는 맵퍼 가상머신 및 리듀서 가상머신이 동일한 물리머신 내 탑재되어 있는지 여부를 판단한다.

[0046] 만약, 상기 맵퍼 가상머신 및 리듀서 가상머신이 동일한 물리머신 내 탑재되는 경우에는 상기 자원 이용률 획득부(120)가 맵 태스크를 수행하는 맵퍼 가상머신의 자원 이용률, 리듀서 가상머신의 자원 이용률 및 리듀서 가상머신의 드라이버 도메인의 자원 이용률을 획득한다.

[0047] 하지만, 이와 달리, 상기 맵퍼 가상머신 및 리듀서 가상머신이 서로 다른 물리머신 내 탑재되는 경우에는 상기 자원 이용률 획득부(120)가 동일한 물리머신 내 탑재 시 획득한 맵 태스크를 수행하는 맵퍼 가상머신의 자원 이용률, 리듀서 가상머신의 자원 이용률 및 리듀서 가상머신의 드라이버 도메인 뿐만 아니라, 상기 맵 태스크를 수행하는 맵퍼 가상머신의 드라이버 도메인의 자원 이용률을 더 획득한다.

[0048] 데이터 전송시간 연산부(140)가 상기 가상머신 및 드라이버 도메인의 자원 이용률 및 전송하고자 하는 데이터 크기에 기초하여 상기 맵퍼 가상머신 및 리듀서 가상머신간 복수 개의 데이터 전송경로를 통한 데이터 전송시간을 예측하여 연산한다(S220). 이때, 상기 데이터 전송시간 연산부(140)가 상기 가상머신 및 드라이버 도메인의

자원 이용률 및 전송하고자 하는 데이터 크기를 다중 회귀 분석(multiple regression analysis)방법에 적용하여 데이터 전송시간을 예측하여 연산할 수 있다.

- [0049] 이러한 다중 회귀분석이란, 복수 개의 독립변수들 간에 인과관계를 분석하는 방법을 말한다.
- [0050] 즉, 데이터 전송시간 연산부(140)가 상기 맵퍼 가상머신 및 리듀서 가상머신간 복수 개의 데이터 전송경로 중 네트워크 내 데이터 전송경로의 각 구간별 데이터 전송시간을 연산한다.
- [0051] 이어서, 상기 맵퍼 가상머신 및 리듀서 가상머신간 복수 개의 데이터 전송경로 중 블록 장치 재구성에 따른 데이터 전송경로의 데이터 전송시간을 연산한다.
- [0052] 네트워크를 이용한 데이터 전송경로의 각 구간별 데이터 전송시간과 상기 블록 장치 재구성에 따른 데이터 전송경로의 데이터 전송시간을 상호 비교하여, 적은 데이터 전송시간을 해당 구간에 할당한다.
- [0053] 하나의 데이터 전송경로를 구성하는 적어도 하나의 구간에 할당된 데이터 전송시간을 각 구간별로 모두 합산하여 상기 복수 개의 데이터 전송경로별 데이터 전송시간을 각각 연산한다.
- [0054] 이후, 데이터 전송경로 추천부(160)가 연산한 데이터 전송시간 중 최소시간을 갖는 데이터 전송경로를 검색하고, 검색된 데이터 전송경로를 상기 맵퍼 가상머신 및 리듀서 가상머신간에 최적의 데이터 전송경로로 추천한다(S230).
- [0055] 특히, 클라우드 환경에서 가상화된 자원들은 동적으로 재구성될 수 있고 할당된 자원량을 증가시키거나 감소시키고 가상머신에 가상 장치를 추가하거나 제거함으로써 관리될 수 있다.
- [0056] 이하에서는 블록 장치 재구성의 방법 및 맵리듀스 어플리케이션의 전송 오버헤드를 감소시키는 과정에 대하여 보다 구체적으로 살펴보도록 한다.
- [0057] 블록 장치 재구성은 가상머신에 블록 장치를 부착하거나 탈착함으로써 수행된다. 즉, 동일한 물리머신에 탑재된 발신(sending) 가상머신으로부터 수신(receiving) 가상머신으로 데이터가 전송될 때, 상기 데이터는 네트워크 전송 없이 블록 장치 재구성을 통해 용이하게 전송될 수 있다.
- [0058] 예를 들어, 발신 가상머신이 가상 블록 장치(VBD, Virtual Block Device)에 데이터를 기록한 후, 블록 장치 재구성으로 상기 가상 블록 장치에 해당하는 블록 장치 파일(BDF, Block Device File)을 발신 가상머신으로부터 탈착하고 수신 가상머신에 재부착하면, 수신 가상머신은 발신 가상머신이 기록한 데이터를 읽을 수 있다.
- [0059] 즉, 네트워크를 통해 데이터가 전송되는 동안 CPU 시간을 소모하는데 비하여, 블록 장치 재구성은 수행하는 시간 동안 상대적으로 아주 적은 자원만을 사용하게 된다.
- [0060] 뿐만 아니라, 서로 다른 물리머신에서 동작하는 가상머신간의 데이터 전송 경우에도 동일한 원리가 적용된다. 서로 다른 물리머신에서 동작하는 가상머신간 데이터 전송의 경우, 데이터가 드라이버 도메인을 통해 전송되므로, 블록 장치 재구성을 이용하여 가상머신과 상기 가상머신의 물리머신에 존재하는 드라이버 도메인 사이의 전송 오버헤드를 제거할 수 있다.
- [0061] 특히, 가상 맵리듀스 클러스터에서 블록 장치 재구성 방법을 통한 성능 이득 여부를 확인하기 위하여, Xen 하이퍼바이저 상에서 블록 장치 재구성을 이용하는 간단한 맵리듀스 프레임워크(framework)를 구현하고 이에 대한 성능을 측정하였다.
- [0062] 도 3은 블록 장치 재구성을 이용한 맵리듀스 프레임워크를 나타낸 블록도이다.
- [0063] 도 3에 도시된 바와 같이, 맵리듀스 프레임워크에서 맵퍼 가상머신과 리듀서 가상머신의 두 가지 형태의 가상머신을 확인할 수 있다. 상기 맵퍼 가상머신은 맵 태스크들만을 수행하는 반면에, 상기 리듀서 가상머신은 리듀스 태스크와 더불어 맵 태스크들을 함께 수행할 수 있다.
- [0064] 이에 따라, 상기 맵퍼 가상머신에서 동작하는 맵 태스크의 수행이 종료되면, 블록 장치를 재구성하여 상기 맵 태스크의 출력을 리듀스 태스크의 로컬 디스크로 옮길 수 있다.
- [0065] 도 3의 (a) 내지 (e)까지의 다섯 단계는 맵퍼 가상머신과 리듀서 가상머신이 동일한 물리머신에 탑재되었을 때의 상세 데이터 전송 단계를 나타낸 것이다.
- [0066] 먼저, 맵퍼 가상머신에서 동작하는 맵 태스크는 (a) 부착된 가상 블록 장치를 자신의 출력 디렉터리에 마운트(mount)시키고, 맵 함수의 출력을 기록한다. 이후, 맵 함수가 종료되면 (b) 마운트된 해당 출력 디렉터리를 언

마운트(unmount) 시킨다. 블록 장치 재구성 모듈은 (c) 맵퍼 가상머신으로부터 가상 블록 장치를 탈착하고, (d) 리듀서 가상머신에 리듀스 함수의 입력으로써 해당 가상 블록 장치의 블록 장치 파일을 재부착시킨다. 이후, 리듀서 가상머신에서 동작하는 리듀스 태스크는 (e) 부착된 가상 블록 장치를 입력 디렉터리에 마운트시키고 해당 가상 블록 장치로부터 데이터를 읽는다. 상기 리듀스 태스크는 모든 맵 태스크들의 출력이 자신에게 도착할 때까지 기다린 후, 리듀스 함수를 수행한다.

[0067] 하지만 상술한 바와 달리, 맵퍼 가상머신과 다른 물리머신에 탑재된 리듀서 가상머신의 경우에는 도 3의 (1) 내지 (8)까지의 총 여덟 단계의 데이터 전송단계가 수행된다. 이러한 각 단계들은 각 물리머신에 위치한 드라이버 도메인이 네트워크를 이용해 데이터 전송을 중개하는 (4) 내지 (6)의 단계를 제외하고, 상술한 맵퍼 가상머신과 리듀서 가상머신이 동일한 물리머신에 탑재되어 있는 경우와 동일하게 수행된다.

[0068] 이하, 도 4 내지 도 6을 참조하여 네트워크를 통해 모든 데이터를 전송하는 원래의 맵리듀스 프레임워크와 블록 장치 재구성 방법이 적용된 맵리듀스 프레임워크의 성능을 분석하고 비교한 결과를 살펴보고자 한다.

[0069] 본 실험을 수행하기 위해, 두 개의 물리머신들을 이용하였고 각 물리머신은 6개 또는 7개의 가상머신을 각각 탑재하였다. 제1 물리머신(PM1)에 탑재된 하나의 가상머신은 리듀서 가상머신으로 지정하고, 제1 가상머신(VM1) 내지 제3 가상머신(VM3)까지의 세 가상머신은 각 물리머신에서 맵퍼 가상머신들로 지정하였으며 로 표시된 나머지 세 개의 가상머신들은 입출력 I/O 생성기(generator) 가상머신들로서 데이터 전송으로 인한 성능 영향을 분석하기 위해 디스크 입출력 작업을 수행하게 하였다. 이러한 가상머신(들)은 디스크 입출력 벤치마크 툴인 debench에 디폴트(default) 구성과 하나의 dbench 클라이언트(client)를 이용하여 실행하였다.

[0070] 도 4는 맵퍼 가상머신들과 리듀서 가상머신간 데이터 전송시간을 나타낸 그래프이다.

[0071] 도 4에 도시된 바와 같이, 동일한 제1 물리머신(PM1)에서 동작하는 세 개의 맵퍼 가상머신에서 리듀서 가상머신으로의 데이터 전송 시간이 블록 장치 재구성 방법을 사용하는 경우에 약 97% 감소되는 것을 알 수 있다. 또한, 제2 물리머신(PM2)에 탑재된 세 맵퍼 가상머신에서 상기 제1 물리머신(PM1)에 탑재된 리듀서 가상머신으로의 데이터 전송 시간도 블록 장치 재구성 방법을 사용하는 경우에 약 15% 감소하는 것을 알 수 있다.

[0072] 뿐만 아니라, 누적 자원(CPU) 이용률과 디스크 입출력 대기시간 측면에서도 블록 장치 재구성 방법을 사용한 맵리듀스 프레임워크의 성능이 원래의 맵리듀스 프레임워크보다 더 향상된 것을 알 수 있다.

[0073] 도 5는 맵퍼 가상머신들과 리듀서 가상머신간 데이터 전송 시 자원 이용률을 나타낸 그래프이다.

[0074] 도 5에 도시된 바와 같이, 블록 장치 재구성을 이용한 데이터 전송이 더 적은 CPU 자원량을 필요로 하는 것을 알 수 있다. 또한 원래의 맵리듀스 프레임워크가 드라이버 도메인(Domain-0)과 모든 맵퍼 가상머신들의 CPU 자원량을 소모하는 반면에, 블록 장치 재구성이 적용되면 상기 드라이버 도메인(Domain-0)의 CPU 자원만을 소모하는 것을 알 수 있다. 이러한 이유는 네트워크를 통한 데이터 전송이 전송 시간 동안 CPU 자원을 소모하지만, 블록 장치 재구성 전송은 블록 장치를 재구성하는 경우에만 CPU 자원을 필요로 하기 때문이다.

[0075] 또한, 도 5의 (c) 및 (d)에 도시된 바와 같이, 블록 장치 재구성 전송에서 CPU 이용률이 데이터 전송의 시작과 끝에서 일정시간 동안 급증하는 것을 확인할 수 있는데, 이러한 이유는 세 개의 블록 장치를 동시에 부착하고 탈착하는 블록 장치 재구성 오버헤드로 인한 것이다.

[0076] 도 6은 맵퍼 가상머신과 리듀서 가상머신간 데이터 전송 시 각 물리머신 별 가상머신의 누적 디스크 입출력 대기시간을 나타낸 그래프이다.

[0077] 도 6에 도시된 바와 같이, 블록 장치 재구성에 따른 데이터 전송의 경우인 가상머신의 누적 디스크(들)이 네트워크를 이용한 데이터 전송의 경우인 가상머신의 누적 디스크(들)보다 향상된 디스크 입출력 성능을 가지는 것을 알 수 있다. 이러한 이유는 네트워크를 이용한 데이터 전송의 경우인 가상머신의 누적 디스크(들)이 맵퍼 가상머신들과 자원 경쟁으로 인해 성능이 악화되는데 반해, 블록 장치 재구성은 입출력 성능에 영향을 미치지 않기 때문이다.

[0078] 하지만 네트워크와 블록 장치 재구성 방법을 이용한 데이터 전송 시간은 가용 CPU 자원과 전송하는 데이터 크기에 의해 결정된다. 즉, 데이터 전송 조건에 따라서 상이한 성능 결과가 나타날 수 있다.

[0079] 도 7 내지 도 9는 한 물리머신에서 다른 물리머신에 탑재된 가상머신으로의 데이터 전송 시간을 데이터 크기와 가용 CPU 자원량을 변화시키며 측정한 결과이다.

[0080] 본 실험에는 세 개의 가상머신이 탑재된 각각의 두 물리머신을 사용하였다. 물리머신은 8개의 CPU를 가지며, 각

가상머신은 두 개의 가상 CPU를 갖도록 구성하였다. 하둡의 wordcount 예제를 실험에 사용하였으며, 데이터 크기를 10 MB 에서 20 MB까지 변화시키며 측정한 성능을 비교하였다

[0081]

도 7은 데이터 크기별 전송시간을 나타낸 그래프이다.

[0082]

도 7에 도시된 바와 같이, 데이터 크기가 작은 경우에는 네트워크 전송이 블록 장치 재구성보다 상대적으로 빠르다는 것을 알 수 있다. 이러한 이유는 블록 장치 재구성이 데이터 크기와는 상관없이 블록 장치를 재구성하는 시간이 기본적으로 필요하기 때문이다.

[0083]

하지만, 데이터 크기가 커질수록 네트워크를 통한 데이터 전송이 블록 장치 재구성보다 오랜 시간이 소요될 수 있다.

[0084]

도 8은 데이터를 수신하는 가상머신의 가용 자원 이용률별 15MB의 데이터 전송 시간을 나타낸 그래프이다. 도 8(a)는 드라이버 도메인이 약 70 % CPU를 사용하도록 구성할 때의 데이터 전송시간을 나타내고, 도 8(b)는 드라이버 도메인이 약 50%의 CPU를 사용하도록 구성할 때의 데이터 전송시간을 나타내었다.

[0085]

도 8에 도시된 바와 같이, 블록 장치 재구성에 소요되는 시간이 수신 가상머신의 가용 CPU 자원량에 영향을 받지 않는 반면에, 네트워크 전송 시간은 점차 증가하는 것을 알 수 있다.

[0086]

비록 네트워크 전송이 드라이버 도메인에 의해 CPU를 소모하여 중개되지만 블록 장치 재구성 방법의 시간은 드라이버 도메인의 가용 CPU 자원량에 더욱 영향을 받는다.

[0087]

도 9는 드라이버 도메인의 가용 자원 이용률별 전송 시간을 나타낸 그래프이다. 도 9(a)에 도시된 바와 같이, 수신 가상머신이 약 20 %의 CPU 자원을 이용할 수 있는 경우에 드라이버 도메인의 가용 CPU 이용률을 변화시키며 네트워크를 이용한 데이터 전송시간과 블록 장치 재구성을 이용한 데이터 전송시간을 측정하였다. 또한, 도 9(b)에 도시된 바와 같이, 수신 가상머신이 약 10%의 CPU 자원을 이용할 수 있는 경우에, 드라이버 도메인의 가용 CPU 이용률을 변화시키며 네트워크 전송과 블록 장치 재구성 시간을 측정하였다.

[0088]

일반적으로 스케줄러와 하부의 파일 시스템에 의해 하둡 맵리듀스와 같은 맵리듀스 구현물들에서 맵과 리듀스 태스크의 개수와 위치가 결정된다. 즉, 가상 맵리듀스 클러스터에서 맵과 리듀스 태스크들이 동일한 물리머신에 생성될 수도 있고, 서로 다른 물리머신에 생성될 수도 있는 것이다.

[0089]

도 10은 맵퍼 가상머신과 리듀서 가상머신간 데이터 전송경로를 나타낸 도면이다.

[0090]

도 10에 도시된 바와 같이, 셔플 단계에서 맵퍼 가상머신과 리듀서 가상머신 사이의 데이터 전송이 가능한 총 여섯 개의 데이터 전송 경로를 확인할 수 있으며, 이때, 상기 여섯 개의 데이터 전송 경로는 네트워크를 이용한 데이터 전송과, 블록 장치 재구성을 이용한 데이터 전송이 상호 조합되어 이루어진다.

[0091]

맵리듀스 job은 다수의 태스크로 구성되고 태스크 집합 T 는 맵 태스크들과 리듀스 태스크들로 구성된다.

즉, M 이 맵 태스크들의 집합이고 R 이 리듀스 태스크들의 집합일 때, 태스크 집합 $T = M \cup R$ 이다. 이때, 각각의 태스크는 가상 맵리듀스 클러스터의 가상머신 중 하나에서 수행된다. 가상 맵리듀스 클러스터에서 가상머신은 사용 가능한 슬롯(slot)들이 존재하는 경우, 동시에 맵 또는 리듀스 태스크들을 수행한다.

[0092]

이하에서는 태스크 할당 함수에 대한 정의를 설정하기로 한다.

[0093]

먼저, 태스크 할당 함수의 첫 번째 정의를 살펴보도록 한다.

[0094]

태스크 할당 함수는 함수 $f_A: T \rightarrow VM$ 이고, 상기 함수에 의해 태스크 t 가 가상머신 $f_A(t)$ 에 할당되었다고 할 수 있다.

[0095]

또한 맵퍼 가상머신은 맵 태스크를 수행하는 가상머신을 의미하고, 리듀서 가상머신은 리듀스 태스크를 수행하는 가상머신을 의미한다. 맵퍼 가상머신의 모든 슬롯이 맵 태스크에 할당되는 반면에, 리듀스 가상머신에는 최소한 하나의 슬롯에 리듀스 태스크가 할당되고 나머지 슬롯은 맵 태스크를 위해 사용된다.

[0096]

이어서, 태스크 할당 함수의 두 번째 정의를 살펴보면, 리듀서 가상머신들의 집합은 하기의 수학적 식 1과 같이 표

현된다.

[0097] [수학식 1]

$$VM_R = \{vm | \exists t \in R \text{ that } f_A(t) = vm\}$$

[0098]

이때, 상기 VM_{MR} 이 가상 맵리듀스 클러스터의 가상머신들의 집합을 의미할 때, 맵퍼 가상머신들의 집합은 하기의 수학식 2와 같이 표현된다.

[0100] [수학식 2]

$$VM_M = VM_{MR} - VM_R$$

[0101]

[0102] 클라우드 환경에서 물리머신은 대개 다수의 가상머신들을 탑재하므로, 가상 맵리듀스 클러스터는 다수의 물리머신에 걸쳐 생성될 수 있다. 따라서 특정 가상머신을 운영하는 물리머신을 확인하기 위해 가상머신 탑재 함수를 정의해야 한다. 이 함수는 리듀서 가상머신과 같은 물리머신에서 동작하는 특정 가상머신을 명시하기 위해 사용될 것이다.

[0103] 또한 태스크 할당 함수의 세 번째 정의를 살펴보면 다음과 같다.

[0104] 다수의 가상머신을 탑재한 물리머신들의 집합을 PM 이라 할 때, 가상머신 탑재 함수는 함수

$$f_H: VM \rightarrow PM$$

이고, 상기 함수에 의해 가상머신 vm 이 물리머신 $f_H(vm)$ 에 탑재되었다고 할 수 있다.

[0105] 도 10에 도시된 바와 같이, 여섯 개의 데이터 전송경로를 가중치 유형 그래프로 표현하기 위하여 그래프 상의 정점(vertex)의 종류를 정의해야 한다.

[0106] 즉, 모두 네 종류의 정점이 존재한다. 첫 번째 정점의 종류는 리듀서 가상머신, 그 다음은 드라이버

도메인이다. 물리머신 pm 의 드라이버 도메인들의 집합을 DD_{pm} 이라고 한다면, 전체 드라이버 도메인의 집

합은 $DD = \bigcup_{pm \in PM} DD_{pm}$ 으로 표현할 수 있다. 다른 두 종류의 정점은 로컬 맵퍼 가상머신과 리모트

(remote) 맵퍼 가상머신이다. 본 발명에서 로컬 가상머신은 리듀스 태스크 r 를 수행하는 리듀서 가상머신과

동일한 물리머신에 탑재된 가상머신을 의미한다. 리모트 가상머신은 리듀스 태스크 r 에 대해 로컬 가상머신이 아닌 가상머신을 의미한다.

[0107] 이어서, 태스크 할당 함수의 네 번째 정의를 살펴보면 다음과 같다.

[0108] 리듀스 태스크 r 에 대한 맵퍼 가상머신들의 집합은 $VM_M^{(r)}$ 와 같이 표현된다. 이 중 리듀스 태스크 r 에 대한 로컬 맵퍼 가상머신들의 집합은 하기의 수학식 3과 같이 표현될 수 있다.

[0109] [수학식 3]

$$VM_M^{local(r)} = \{vm | \exists r \in R \text{ that } vm \in f_H^{-1}(f_A(r))\}$$

[0110]

이때, 상기 리듀스 태스크 r 에 대한 리모트 맵퍼 가상머신들의 집합은 하기의 수학식 4와 같이 표현될 수 있다.

[수학식 4]

$$VM_M^{remote(r)} = VM_M^{(r)} - VM_M^{local(r)}$$

이어서, 태스크 할당 함수의 다섯 번째 정의를 살펴보면 다음과 같다.

데이터 전송 그래프는 비순환 유향 그래프로서 E 가 간선(edge)들의 집합이라 하면 $G_{MR} = (VM_{MR} \cup DD, E)$ 로 표현된다. 정점 s 에서 정점 d 로의 간선을 $e(s, d)$ 라고 할 때, 각 간선이 갖는 값은 하기의 수학식 5와 같이 표현될 수 있다.

[수학식 5]

$$e(s, d) = \begin{cases} 1 & , \text{if exists} \\ 0 & , \text{otherwise} \end{cases}$$

또한, 각 리듀스 태스크 r 에 대한 G_{MR} 의 부분 그래프(sub-graph)를

$G_{MR}(r)$ 이라 할 때, 상기 G_{MR} 은 하기의 수학식 6과 같이 나타낼 수 있다.

[수학식 6]

$$G_{MR} = \bigcup_{r \in R} G_{MR}(r)$$

도 11은 가상머신별 데이터 전송과정을 나타낸 도면이다.

특히, 상술한 태스크 할당 함수의 다섯 번째로 정의한 데이터 전송 그래프 G_{MR} 의 예를 나타낸다.

도 11에 도시된 바와 같이, 리듀스 태스크 r 에 대한 리모트 가상머신으로서의 i 번째 맵퍼 가상머신을

의미하는 정점 $vm_{Mi}^{remote(r)} \in VM_M^{remote(r)}$ 는 $pm_{Mi} = f_H(vm_{Mi}^{remote(r)})$ 일 때, 각각 $e(vm_{Mi}^{remote(r)}, dd_{pm_{Mi}})$, $e(vm_{Mi}^{remote(r)}, dd_{f_H(vm_r)})$, $e(vm_{Mi}^{remote(r)}, vm_r)$ 의 세 간선을 갖는다.

마지막으로, 태스크 할당 함수의 여섯 번째 정의를 살펴보면, 그래프 G_{MR} 에 있는 간선들은 데이터가 한 정점에서 다른 정점으로 네트워크 또는 블록 장치 재구성성을 통해 전송될 수 있다는 것을 나타낸다. 두 방법에 대

한 전송 시간은 $T_e^{Network}$ 또는 T_e^{BDR} 로 나타내며 각각의 간선은 이들 중 최소값을 갖는다. 즉, 상기 간선의 최소값은 하기의 수학적 식 7과 같이 표현될 수 있다.

[수학적 식 7]

$$T_e = \min\{T_e^{Network}, T_e^{BDR}\}$$

하지만, 서로 다른 물리머신에 탑재된 가상머신들은 블록 장치 재구성 방법만으로는 데이터를 전송할 수 없다.

따라서, $pm_{Mi} = f_H(vm_{Mi}^{remote(r)})$ 라고 할 때, 간선 $e(vm_{Mi}^{remote(r)}, vm_r)$, $e(vm_{Mi}^{remote(r)}, dd_{f_H(vm_r)})$, $e(dd_{pm_{Mi}}, vm_r)$, $e(dd_{pm_{Mi}}, dd_{f_H(vm_r)})$ 들은 리듀스 태스크 r 에

대해 블록 장치 재구성으로는 데이터를 전송할 수 없고 이때의 T_e^{BDR} 은 ∞ 가 된다.

결국, 데이터 전송 그래프 $G_{MR}(r)$ 에서 출발지 매퍼 가상머신 $vm_{Mi}^{(r)}$ 과 목적지 리듀서 가상머신 vm_r 이 주어졌을 때, 최소 데이터 전송 시간을 갖는 최단 전송 경로를 찾는 것이 매우 중요하다. $P(s, d)$ 를 출발지

정점 s 에서 목적지 정점 d 에 도달하는 가능한 경로의 집합으로 할 때, 각 경로 $p \in P(s, d)$ 는 그 그래프 G_{MR} 에서 적어도 하나의 간선으로 구성된다. 선택된 경로 p 를 통한 전송 시간이

$$T_p = \sum_{e \in p} T_e \quad p^* = \operatorname{argmin}_{p \in P} T_p$$

일 때, 최단 전송 경로는 p^* 에 의해 획득된다.

이하에서는 매퍼 가상머신과 리듀서 가상머신간 최소의 데이터 전송 시간을 갖는 데이터 전송경로를 결정하는 과정에 대하여 자세히 살펴보도록 한다.

먼저, 하나의 매퍼 가상머신 job은 하나의 리듀스 태스크를 갖는다고 가정하며, 이러한 설정은 하둡 매퍼 가상머신의 기본

설정과 동일하다. 따라서, $|R| = 1, |VM_R| = 1$ 이다. 또한 각 물리머신은 하나의 드라이버 도메인을 갖는다고 가정한다. 즉, $pm \in PM$ 일 때, $|DD_{pm}| = 1$ 이다.

첫 번째, 매퍼 가상머신과 리듀서 가상머신 사이의 총 전송 시간을 추정하기 위해, 해당 가상머신들과 그들의 드라이버 도메인들의 자원 이용률을 획득하여, $CPU_{vm_{Mi}}$, $CPU_{dd_{pm_{Mi}}}$, CPU_{vm_r} , $CPU_{dd_{f_H(vm_r)}}$ 을 각각 i 번째 맵 태스크를 수행하는 매퍼 가상머신, 매퍼 가상머신의 드라이버 도메인, 리듀서 가상머신, 리듀서 가상머신의 드라이버 도메인의 CPU 이용률 값이라고 하면, 매퍼 가상머신이 리듀서 가상

머신과 같은 물리머신에 탑재되어 있을 때, 리듀서 가상머신과 드라이버 도메인을 공유하므로 $CPU_{vm_{Mi}}$,

CPU_{vm_r} , $CPU_{dd_{fH}(vm_r)}$ 를 획득한다. 이에 더하여, 상기 매퍼 가상머신과 리듀서 가상머신이 서로 다

른 물리머신에 탑재된 경우에는 $CPU_{dd_{pm_{Mi}}}$ 또한 획득한다.

[0133]

두 번째, 매퍼 가상머신과 리듀서 가상머신 사이의 가능한 경로들의 전송 시간을 현재 CPU 이용률과 데이터 크기 같은 전송 파라미터를 이용하여 가능한 데이터 전송경로의 전송시간을 예측한다.

[0134]

CPU 이용률과 데이터 크기를 이용하여 전송 시간을 추정하기 위하여 가상 매퍼듀스 클러스터 상에서의 벤치마킹 결과에 기반한 다중 회귀 분석(multiple regression analysis)을 사용한다. 이러한 다중 회귀 분석 방법을 통해 네트워크를 통한 전송 시간이 각 가상머신이 적은 가용 CPU를 가질수록 지수 증가함에 비하여, 데이터 크기를 증가시킴에 따라 단조 증가함을 알 수 있었다. 하지만 블록 장치 재구성에서의 전송 시간은 데이터 크기에 영향을 받지 않았으며, 드라이버 도메인의 가용 CPU에만 관련이 있었다. 이러한 결과는 블록 장치 재구성이 드라이버 도메인에서 블록 장치를 부착 또는 탈착할 뿐, 전송하고자 하는 데이터 크기와는 무관하기 때문이다.

따라서, 다중 회귀 분석을 이용하여 전송 시간 T 에 대한 다항식은 하기의 수학적 식 8과 같이 표현될 수 있다.

[0135]

[수학적 식 8]

$$T_e^{Network} = \left(\sum_{x \in X} \alpha_x \cdot e^{-\beta_x \cdot AvailCPU_x} \right) \cdot \gamma \cdot DataSize + c$$

[0136]

$$T_e^{BDR} = \alpha \cdot e^{-\beta \cdot AvailCPU_{dd}} + c$$

[0137]

이때, 상기 $X = [vm_{Mi}, dd_{pm_{Mi}}, dd_{fH}(vm_r), vm_r]$ 이고, 상기 $DataSize$ 는 전송 데이터의 크기를 의미한다.

$AvailCPU_x$ 는 x 의 남은 CPU 이용률이고, $AvailCPU_{dd}$ 는 매퍼 가상머신 또는 리듀서 가상머

신의 드라이버 도메인의 남은 CPU 이용률이다. c 는 상수이고 α, β, γ 는 다항식의 계수이다.

[0138]

또한 다중 지수 회귀 분석에 의해 구한 계수들은 추정 결과를 향상시키기 위해 GRG(Generalized Reduced Gradient)를 통해 최적화 시킨다.

[0139]

하기의 표 1은 다중 회귀 분석과 GRG에 의해 만들어진 다항식의 계수들을 나타낸 것이다.

표 1

Coefficients		BDR		Network			
		Attach	Detach	vm_{Mi} to vm_r	vm_{Mi} to $dd_{fH}(vm_r)$	$dd_{pm_{Mi}}$ to vm_{Mi}	$dd_{pm_{Mi}}$ to $dd_{fH}(vm_r)$
α	vm_{Mi}	-	-	0.02	0.06	0.01	0.03
	$dd_{pm_{Mi}}$	-	8.04	0.04	0.06	0.11	0.09
	$dd_{fH}(vm_r)$	8.28	-	0.06	0.19	0.09	0.23
	vm_r	-	-	0.06	0.00	0.08	0.01
β	vm_{Mi}	-	-	0.32	0.15	0.00	0.11
	$dd_{pm_{Mi}}$	-	0.04	0.20	0.16	0.11	0.12
	$dd_{fH}(vm_r)$	0.04	-	0.15	0.11	0.14	0.10
	vm_r	-	-	0.14	0.00	0.12	0.00
γ		-	-	17.42	11.44	11.87	8.93
c		0.76	0.73	1.15	1.47	1.36	1.56

각 열은 데이터 전송 그래프의 간선들을 나타낸다. 상술한 바와 같이, 데이터 크기와 가용 CPU 양은 네트워크를 통해 데이터를 전송할 때의 성능에 영향을 미친다. 반면, 블록 장치 재구성 경우에는 드라이버 도메인의 가용 CPU 양만이 관련 있다.

예를 들어, vm_{Mi} 에서 $dd_{fH}(vm_r)$ 으로 네트워크를 통해 데이터가 전송될 때, 전송 시간은 vm_{Mi} , $dd_{pm_{Mi}}$, $dd_{fH}(vm_r)$ 의 가용 CPU 자원에 따라 변하고, vm_r 의 남은 CPU 이용률에 관계되지 않는다.

그러므로 다른 계수들은 0이 아닌 값을 갖는 반면, 계수 α_{vm_r} 과 β_{vm_r} 은 거의 0에 가깝게 되었다. 이 계수들을 이용하여 네트워크를 통한 데이터 전송과 블록 장치 재구성 방법의 수행 시간을 추정한다.

세 번째, 추정 시간 $T_e^{Network}$ 와 T_e^{BDR} 을 계산하고, 정의 6에 의해 그들 중 작은 값을 해당 간선에 할당한다.

네 번째, 데이터 전송에 대한 최단 경로를 결정한다. 맵퍼 가상머신에서 리듀서 가상머신으로의 간선들이 갖는 값들을 합산하여 모든 경로에 대한 추정 전송 시간을 계산한 후, 그들 중 최단 경로를 선택한다.

본 발명은 단계 4에서 vm_{Mi} 에서 vm_r 로의 최단 경로를 결정하고자 할 때, 하기의 표 2에 나타난 경로 중 하나를 하기의 수학적 9에 따라 결정한다.

[수학적 9]

$$p^* = \operatorname{argmin}_{p \in P(vm_{Mi}, vm_r)} T_p$$

표 2

vm_{Mi} 의 종류	경로 $p \in P(vm_{Mi}, vm_r)$ 의 전송 시간
$VM_M^{local(r)}$	$T_{e(vm_{Mi}, vm_r)}^{Network}$
	$T_{e(vm_{Mi}, dd_{f_H}(vm_r))}^{BDR} + T_{e(dd_{f_H}(vm_r), vm_r)}^{BDR}$
$VM_M^{remote(r)}$	$T_{e(vm_{Mi}, vm_r)}^{Network}$
	$T_{e(vm_{Mi}, dd_{f_H}(vm_r))}^{Network} + T_{e(dd_{f_H}(vm_r), vm_r)}^{BDR}$
	$T_{e(vm_{Mi}, dd_{pm_{Mi}})}^{BDR} + T_{e(dd_{pm_{Mi}}, vm_r)}^{Network}$
	$T_{e(vm_{Mi}, dd_{pm_{Mi}})}^{BDR} + T_{e(dd_{pm_{Mi}}, dd_{f_H}(vm_r))}^{Network} + T_{e(dd_{f_H}(vm_r), vm_r)}^{BDR}$

예를 들어, $vm_{Mi} \in VM_M^{local(r)}$ 이면, $e1 = e(vm_{Mi}, dd_{f_H}(vm_r))$ 이고, $e2 = e(dd_{f_H}(vm_r), vm_r)$ 일 때,

$$p^* = \operatorname{argmin}_{p \in P(vm_{Mi}, vm_r)} \{T_{e(vm_{Mi}, vm_r)}^{Network}, T_{e1}^{BDR} + T_{e2}^{BDR}\}$$

이 된다. 이때, 드라이버 도메인에 의해 중개되는 시간

$T_{e1}^{Network}$ 와 $T_{e2}^{Network}$ 를 계산하지 않는다. 이때, 받아서 전달하는 시간은 드라이버 도메인에서 의해 버퍼에 받아서 버퍼로부터 전달하는 시간을 나타낸다. 즉,

$$T_{e(vm_{Mi}, vm_r)}^{Network} < T_{e1}^{Network} + T_{e2}^{Network}$$

이다. 또한, 리모트 가상머신에 대한 하기의 수학식 10의 경우도 성립한다.

[수학식 10]

$$T_{e(vm_{Mi}, dd_{f_H}(vm_r))}^{Network} < T_{e(vm_{Mi}, dd_{pm_{Mi}})}^{Network} + T_{e(dd_{pm_{Mi}}, dd_{f_H}(vm_r))}^{Network}$$

$$T_{e(dd_{pm_{Mi}}, vm_r)}^{Network} < T_{e(dd_{pm_{Mi}}, dd_{f_H}(vm_r))}^{Network} + T_{e(dd_{f_H}(vm_r), vm_r)}^{Network}$$

본 발명에서 제안하는 동적 블록 장치 재구성 알고리즘을 평가하기 위하여, 자원 모니터(resource monitor), 블록 장치 재구성 모듈(module), 그리고 하둡 맵리듀스 1.0.4에 제안하는 과정을 구현하였다.

먼저, 자원 모니터와 블록 장치 재구성 모듈을 구현하기 위해 *libvirt* 라이브러리와 Xen의 기본 관리 툴인 *xm*을 사용하였다. 또한 블록 장치 풀(pool)을 구성하여 블록 장치들을 관리하고 블록 장치 재구성 모듈이 블록 장치를 요구할 때 제공하거나 더 이상 사용되지 않을 때 회수한다.

실험을 위해 두 대의 물리머신을 사용하였으며 각 물리머신은 두 개의 Intel Xeon 2.40 GHz 쿼드 코어(quad-core) 프로세서(processor)들을 탑재하여 총 8개의 물리 CPU(PCPU)를 제공한다. 16GB의 메모리를 갖고 있으며 우분투(Ubuntu) 13.04를 설치하였다. 가상화 하이퍼바이저로는 Xen 4.2.1을 사용하였으며 기본 크레딧 스케줄러를 사용하였다. 각 물리머신은 두 대의 가상머신을 탑재하였으며 각 가상머신은 두 개의 가상 CPU와 1GB의 메모리로 구성되었다. 최대 4개의 물리 CPU를 가상머신들의 가상 CPU를 수행하는데 사용하였다. 클라우드 환경에서 일반적으로 CPU 자원은 몇몇의 가상머신들에 의해 공유되므로, 네 개의 가상 CPU가 4, 3, 2개의 물리 CPU를 공

유하도록 설정하는 세 가지의 공유 구성을 사용하였다.

- [0155] 본 발명을 평가하기 위해 하둡의 *wordcount* 예제를 맵리듀스 어플리케이션으로 사용하였고, 트위터에서 얻은 4GB의 텍스트(text)를 입력 데이터로 사용하였다. 이러한 맵리듀스 job의 4GB 입력 데이터는 64, 128, 256MB 크기의 64, 32, 16개 블록으로 나누어지고, 그 결과 맵리듀스 job에서는 64, 32, 16개의 맵 태스크가 생성된다. 하둡에서는 슬롯의 개수를 CPU의 개수보다 하나 많게 설정하기를 권장하기 때문에 각 가상머신당 슬롯의 개수는 세 개로 설정하였다. 그 외 다른 설정들은 HDFS의 블록 크기를 제외하고는 기본 설정을 사용하였다. HDFS의 기본 블록 사이즈는 64MB이지만, 고대역폭의 기가비트 네트워크를 일반적으로 사용하는 클라우드 기반 시설(infrastructure)에서는 맞지 않으므로, 실제 많이 사용되는 128MB의 블록 크기를 사용하였다.
- [0156] 우선 본 발명이 가능한 데이터 전송경로 후보들 중에서 최적의 경로를 선택하는지 여부를 확인하기 위한 실험을 수행하였다.
- [0157] 도 12는 동적 블록 재구성 장치를 통한 가상 맵리듀스 클러스터에서 선택된 전송경로의 비율을 나타낸 그래프이며, 특히, 맵리듀스가 4GB의 입력 데이터와 128MB의 블록 크기에 대해 동작하는 경우에 선택하는 경로의 개수를 나타낸 것이다. 이때, 동일한 물리머신에서 동작하는 각 가상머신은 세 개의 물리 CPU를 공유한다. 도 12에 도시된 local path 0은 블록 장치 재구성을 통해 로컬 맵퍼 가상머신에서 리듀서 가상머신으로 전송되는 경로를 의미하고, local path 1은 네트워크를 통한 동일 경로를 의미한다. 또한 Remote path 0은 리듀서 가상머신에서 다른 물리머신에 탑재된 맵퍼 가상머신으로의 블록 장치 재구성 경로를 의미한다.
- [0158] Remote path 1은 리모트 맵퍼 가상머신에서 맵퍼 가상머신의 드라이버 도메인으로 블록 장치 재구성을 통해 전송되고, 상기 드라이버 도메인에서 리듀서 가상머신으로 네트워크를 통해 전송되는 경로이다.
- [0159] 하지만 이와 반면에, remote path 2는 리모트 맵퍼 가상머신에서 리듀서 가상머신의 드라이버 도메인으로 네트워크를 통해 전송되고, 상기 드라이버 도메인에서 리듀서 가상머신으로 블록 장치 재구성을 통해 전송되는 경로이다.
- [0160] 마지막으로 remote path 3은 리모트 맵퍼 가상머신에서 리듀서 가상머신으로 네트워크를 통해 전송되는 경로이다.
- [0161] 도 12에 도시된 바와 같이, 같은 물리머신에 탑재된 맵퍼 가상머신과 리듀서 가상머신, 즉 로컬의 경우에는 대부분 local path 0이 선택됐다. 리모트의 경우, 즉 서로 다른 물리머신에 탑재된 맵퍼 가상머신과 리듀서 가상머신의 경우에는 remote path 0과 remote path 2를 선택했다. 네트워크를 통한 데이터 전송은 수신 측에서 더욱 많은 CPU를 소모하게 되고 리듀서 가상머신은 이를 처리하기에 충분한 CPU를 사용하지 못하므로, path 1과 path 3은 선택하지 않았다. 하지만 드라이버 도메인이 충분한 CPU를 갖지 못한다면 다른 경로를 선택할 수 있을 것으로 예측할 수 있다.
- [0162] 도 13은 네 개의 물리 CPU 사용 시, 블록 크기별 맵리듀스 수행시간을 나타낸 그래프이며, 특히, 네 개의 물리 CPU를 가상머신들이 공유하는 경우에 블록 크기를 64MB부터 256MB까지 변화시키며 동적 블록 장치 재구성을 사용하는 맵리듀스의 수행시간을 원래의 맵리듀스와 비교하였다.
- [0163] 도 13에 도시된 바와 같이, 동적 장치 재구성을 이용한 수행시간이 원래의 하둡 맵리듀스보다 적은 것을 알 수 있다. 이러한 성능 향상은 본 발명이 런타임에 동적으로 맵퍼 가상머신에서 리듀서 가상머신으로의 최단 전송 경로를 선택하여 데이터 전송 시간과 가상머신간 자원 경쟁을 감소시켰기 때문인 것으로 알 수 있다.
- [0164] 비록 데이터 전송이 셔플 단계에서만 이루어지지만, 맵퍼 가상머신과 리듀서 가상머신 사이에 더 큰 데이터를 전송할수록 성능 폭은 더욱 벌어질 것으로 예측할 수 있다.
- [0165] 도 14는 세 개의 물리 CPU 사용 시, 블록 크기별 맵리듀스 수행시간을 나타낸 그래프이며, 특히, 세 개의 물리 CPU를 사용한 경우에 동적 블록 장치를 사용한 맵리듀스와 원래의 하둡 맵리듀스의 실행 시간을 나타낸 것이다.
- [0166] 도 14에 도시된 내용은 도 12에 도시된 내용과 달리, 동적 블록 장치 재구성이 맵리듀스 어플리케이션의 수행시간을 최대 14%까지 감소시켰다. 도 13과 도 14에서 보여지는 결과의 차이는 세 개의 물리 CPU를 사용하여 가상머신간 자원 경쟁이 발생하였기 때문인 것을 알 수 있다.
- [0167] 이러한 결과는 네트워크를 통한 데이터 전송이 블록 장치 재구성의 실행보다 자원 경쟁에 더 큰 영향을 받는다는 것을 나타내고, 원래의 하둡 맵리듀스에서 생성된 맵 태스크의 수행이 더욱 오래 걸리게 되었기 때문이다. 따라서, 클라우드 환경에서 대체로 가상 CPU의 개수가 물리 CPU의 개수보다 많고 그로 인해 가상머신간 자원 경

쟁이 일반적인 상황에서 동적 블록 장치 재구성이 보다 효과적임을 알 수 있다.

[0168] 도 15는 두 개의 물리 CPU 사용 시, 블록 크기별 맵리듀스 수행시간을 나타낸 그래프이다.

[0169] 도 15에 도시된 바와 같이, 64MB의 블록 크기를 제외하고는 앞선 실험 결과와 마찬가지로 동적 블록 장치 재구성이 원래의 하둡보다 좋은 성능을 나타낸다. 결과적으로, 총 전송 시간이 거의 같음에도 불구하고 64MB의 블록 크기에 대해 비슷한 성능이 나오는 것은 맵 태스크의 출력 데이터가 매우 작아서 블록 장치 재구성하는 시간이 네트워크로 전송하는 시간보다 길 것으로 추정되어 동적 블록 장치 재구성 알고리즘이 대부분 네트워크를 통해 데이터 전송을 하도록 선택하였기 때문이다.

[0170] 상술한 도 13 내지 15에 도시된 그래프를 살펴보면, 128MB의 블록 크기에서 수행된 시간이 다른 블록 크기에서 수행된 시간보다 대체로 적다는 것을 알 수 있다. 이는 맵 태스크의 개수와 그들의 슬롯 개수에 관련이 있다. 가상머신당 슬롯의 개수가 3개이고 가상머신이 네 개이므로 총 12개의 슬롯이 존재한다. 그리고 64MB, 128MB, 256MB의 블록 크기에 대해 각각 64, 32, 16개의 맵 태스크가 생성되므로 맵 단계는 슬롯당 평균 6, 3, 2번의 맵 태스크를 수행하게 된다. 그 결과 각 블록 크기에 대해서 4, 8, 4개의 맵 태스크가 마지막에 이루어지게 되는데, 마지막 맵 태스크를 수행하는 슬롯이 많을수록 가상머신의 자원을 더욱 잘 활용한 것이 되기 때문에 128MB의 블록 크기일 때 성능이 다른 블록 크기인 경우보다 좋게 나타났다.

[0171] 본 발명의 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법 및 시스템은 가상머신에 부착된 블록 장치를 동적으로 탈착 후, 런타임(runtime) 동안 다른 가상머신에 재부착하여, 가상머신간에 데이터 전송을 용이하고 신속하게 수행할 수 있는 효과가 있다.

[0172] 또한 본 발명의 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법 및 시스템은 맵퍼 가상머신과 리듀서 가상머신간 데이터 전송 시 발생하는 오버헤드를 최소화할 수 있는 효과가 있다.

[0173] 더불어, 본 발명의 가상 맵리듀스 클러스터 내 데이터 전송경로 추천 방법 및 시스템은 맵리듀스 어플리케이션의 실행시간을 단축시킬 수 있는 효과가 있다.

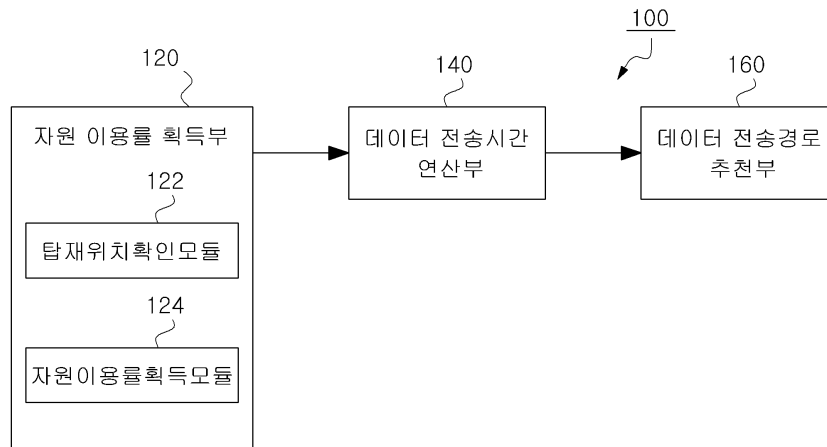
[0174] 상기에서는 본 발명의 바람직한 실시 예에 대하여 설명하였지만, 본 발명은 이에 한정되는 것이 아니고 본 발명의 기술 사상 범위 내에서 여러 가지로 변형하여 실시하는 것이 가능하고 이 또한 첨부된 특허청구범위에 속하는 것은 당연하다.

부호의 설명

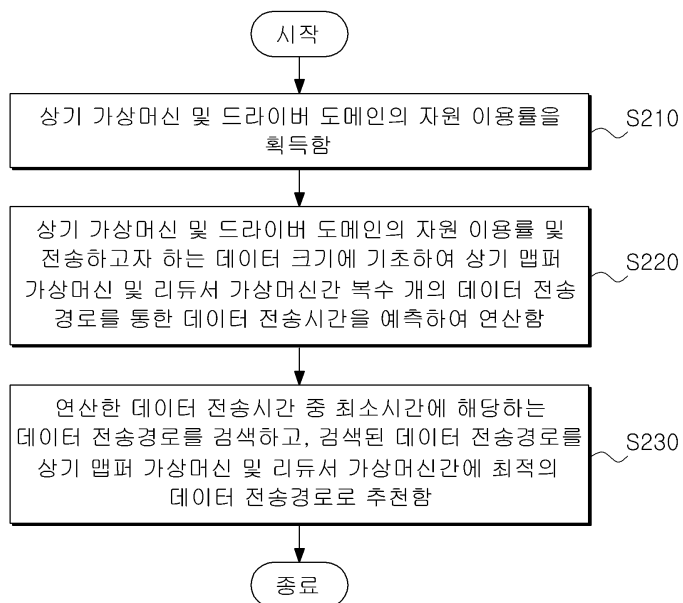
[0175] 120: 자원 이용률 획득부 122: 탑재위치확인모듈
124: 자원이용률획득모듈 140: 데이터 전송시간 연산부
160: 데이터 전송경로 추천부

도면

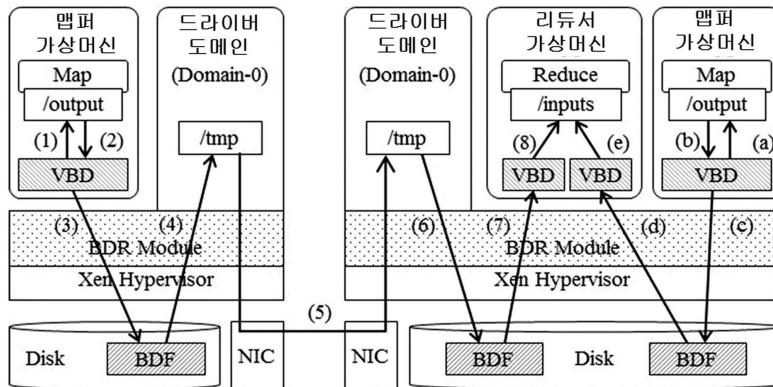
도면1



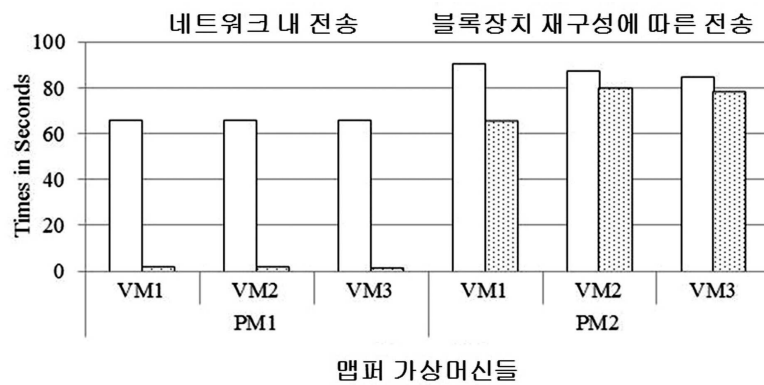
도면2



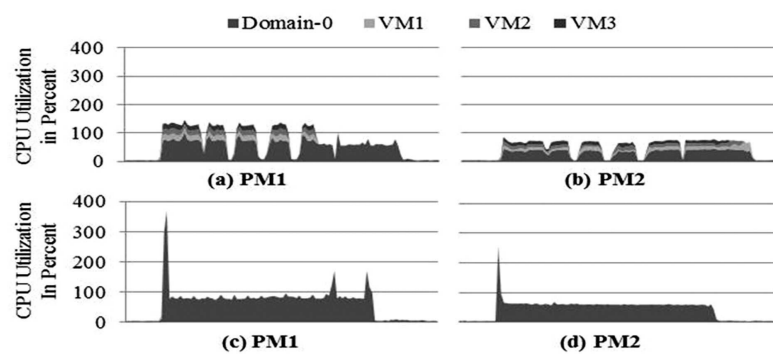
도면3



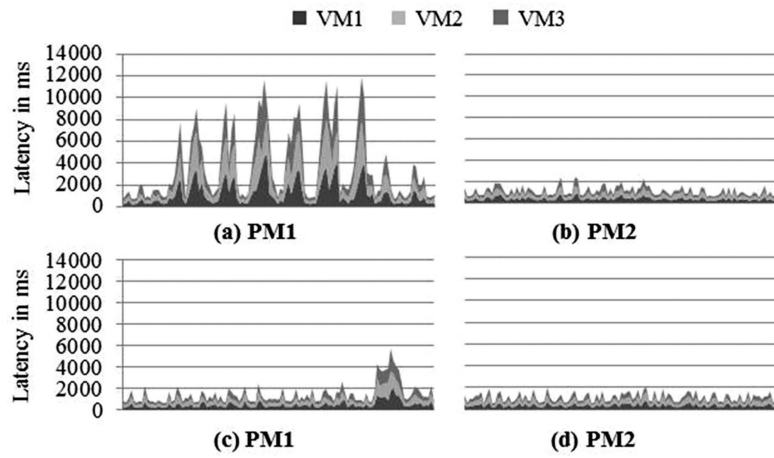
도면4



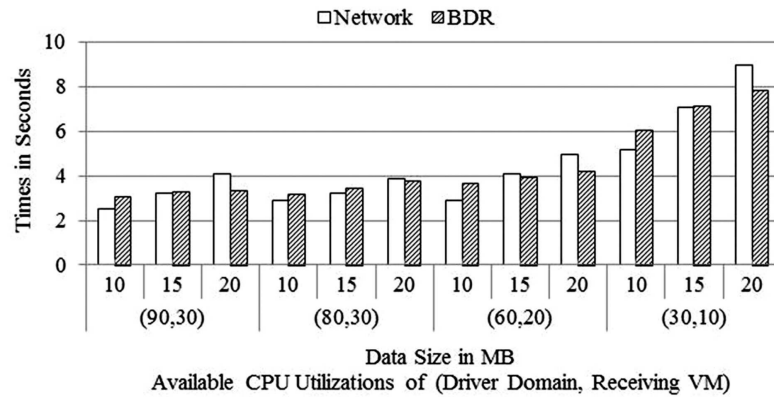
도면5



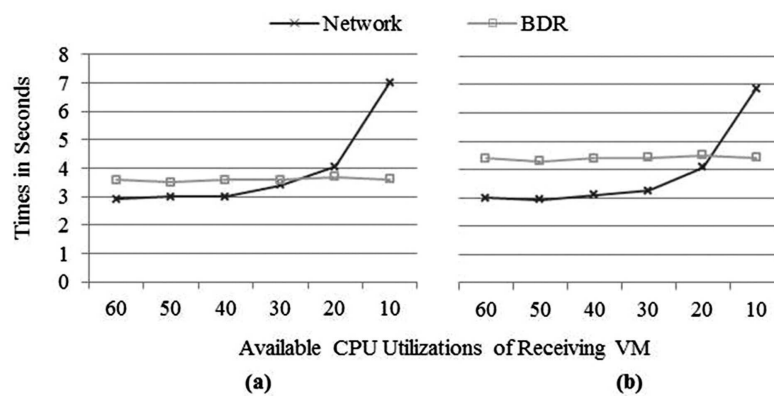
도면6



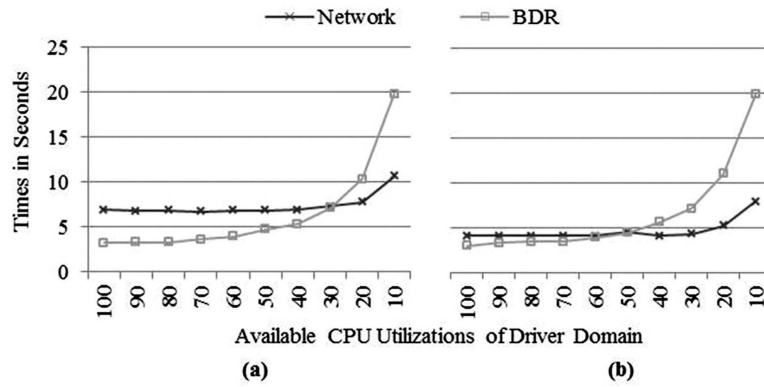
도면7



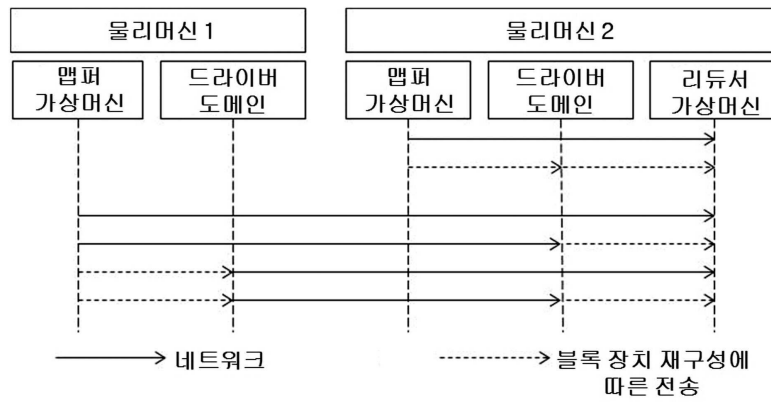
도면8



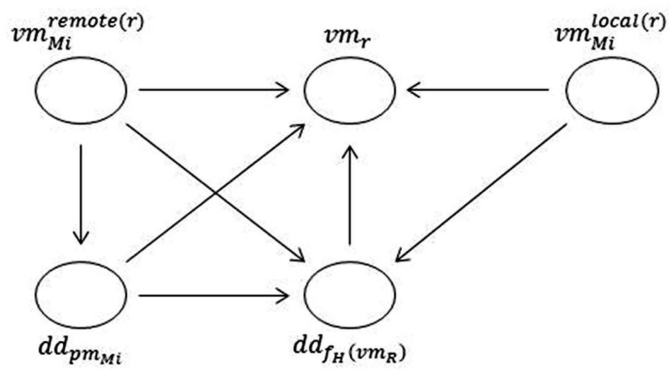
도면9



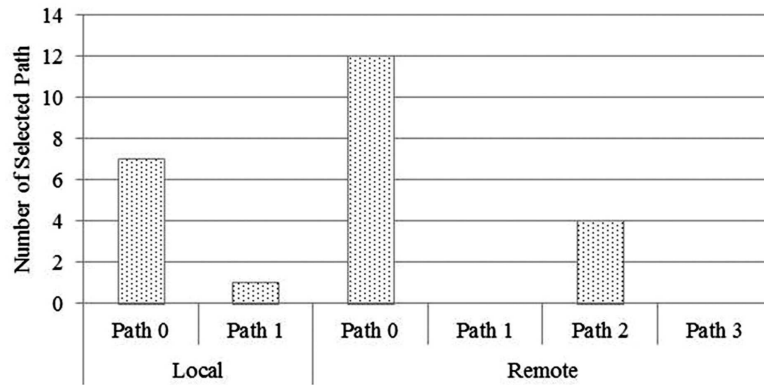
도면10



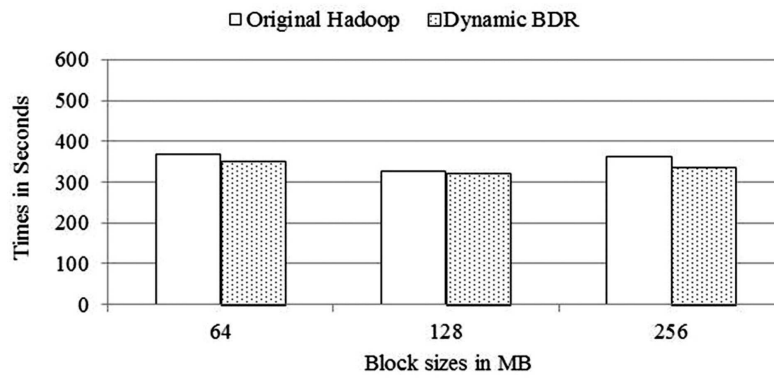
도면11



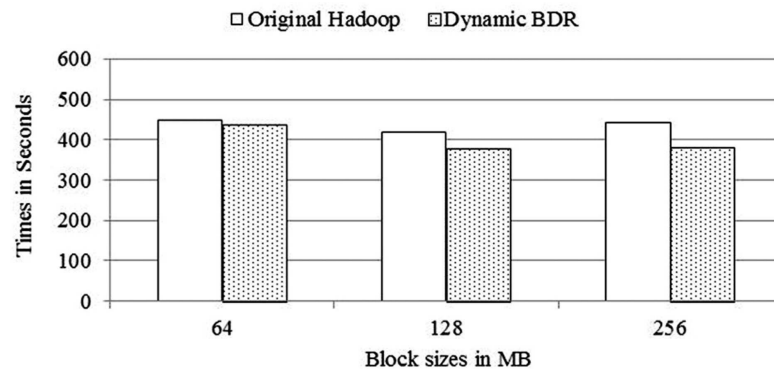
도면12



도면13



도면14



도면15

