



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2016-0081230
(43) 공개일자 2016년07월08일

(51) 국제특허분류(Int. Cl.)

G06F 17/30 (2006.01)

(21) 출원번호 10-2014-0194729

(22) 출원일자 2014년12월31일

심사청구일자 2014년12월31일

(71) 출원인

경희대학교 산학협력단

경기도 용인시 기흥구 덕영대로 1732 (서천동, 경희대학교 국제캠퍼스내)

(72) 발명자

이영구

대전광역시 서구 청사서로 11, 106동 302호 (월평동, 무지개아파트)

김태연

경기도 수원시 영통구 봉영로1517번길 76, 631동 506호 (영통동, 동보.신명 아파트)

박기성

경기도 부천시 원미구 원미로 123-4, E동 4호 (원미동, 백송연립)

(74) 대리인

인비전 특허법인

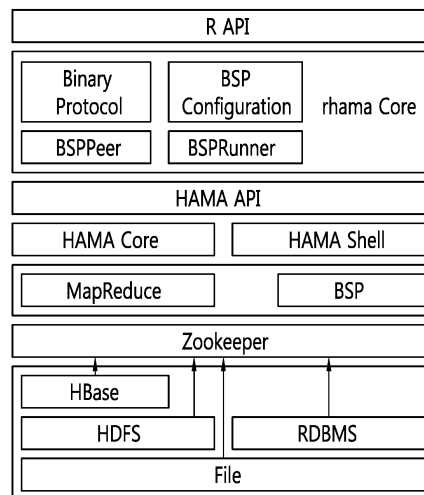
전체 청구항 수 : 총 19 항

(54) 발명의 명칭 R과 프리젤에 기반한 분산 프레임워크 장치 및 방법

(57) 요약

본 발명의 일 실시예에 따르면, HAMA로부터 그래프 데이터를 수신하는 단계, R 언어에서 프리젤을 수행할 수 있도록, 상기 그래프 데이터에 대해 HAMA와 R 언어의 동기화를 수행하는 단계 및 상기 그래프 데이터에 대해 R 언어를 이용하여 통계적 분석을 수행하는 단계를 포함하는 프레임워크 방법을 제공한다.

대표도 - 도4



이 발명을 지원한 국가연구개발사업

과제고유번호 1711014131

부처명 미래부

연구관리전문기관 정보통신산업진흥원

연구사업명 실시간 모바일 클라우드 서비스 플랫폼 연구개발

연구과제명 2014 대학 IT 연구센터육성 지원사업

기 여 율 1/1

주관기관 경희대학교 산학협력단

연구기간 2014.01.01 ~ 2014.12.31

명세서

청구범위

청구항 1

HAMA로부터 그래프 데이터를 수신하는 단계;

R 언어에서 프리젠크를 수행할 수 있도록, 상기 그래프 데이터에 대해 HAMA와 R 언어의 동기화를 수행하는 단계;
및

상기 그래프 데이터에 대해 R 언어를 이용하여 통계적 분석을 수행하는 단계;
를 포함하는 프레임워크 방법.

청구항 2

제1항에 있어서,

상기 HAMA에서는 프리젠크를 수행하는 것을 특징으로 하는 프레임워크 방법.

청구항 3

제1항에 있어서,

상기 HAMA로부터 그래프 데이터를 수신하는 단계에서는, 프리젠크를 R 언어에서도 사용할 수 있도록 연결해주는 매개 함수인 BSPRunner가 HAMA로부터 그래프 데이터를 수신하는 것을 특징으로 하는 프레임워크 방법.

청구항 4

제1항에 있어서,

상기 그래프 데이터에 대해 HAMA와 R 언어의 동기화를 수행하는 단계에서는, 버텍스 단위로 동기화가 수행되는 것을 특징으로 하는 프레임워크 방법.

청구항 5

제4항에 있어서,

상기 버텍스는 다른 버텍스와 메시지를 송수신 하는 것을 특징으로 하는 프레임워크 방법.

청구항 6

제1항에 있어서,

상기 통계적 분석을 수행하는 단계에서는, 버텍스 단위로 그래프 데이터에 대한 통계적 처리를 수행하는 것을 특징으로 하는 프레임워크 방법.

청구항 7

제6항에 있어서,

상기 통계적 분석이 수행된 그래프 데이터는 표준화된 문자열 형식인 것을 특징으로 하는 프레임워크 방법.

청구항 8

제6항에 있어서,

상기 버텍스는 BSPConfiguration과 BinaryProtocol을 통해 사용자가 원하는 작업을 표준 출력의 문자열 형식으로 변환되고,

상기 BSPConfiguration은 HAMA의 기본적인 연산인 setup, bsp, cleanup 연산으로 구성되어 있고,

상기 BinaryProtocol은 텍스트 기반 프로토콜으로서, HAMA로 데이터를 송수신할 때 사용되는 것을 특징으로 하는 프레임워크 방법.

청구항 9

그래프 데이터를 수신하는 단계;

상기 그래프 데이터에 대해 분산 처리를 하는 단계; 및

분산 처리된 상기 그래프 데이터에 대해 R 언어를 이용하여 통계적 분석을 수행하는 단계를 포함하는 데이터 연산 방법.

청구항 10

제9항에 있어서,

상기 그래프 데이터에 대해 분산 처리를 하는 단계에서는, 프리겔을 이용하는 것을 특징으로 하는 데이터 연산 방법.

청구항 11

제10항에 있어서,

상기 프리겔은 버텍스 단위로 데이터에 대해 분산 처리를 수행하는 것을 특징으로 하는 데이터 연산 방법.

청구항 12

제10항에 있어서,

상기 프리겔은 슈퍼스텝단위로 그래프 데이터에 대한 분산 처리를 하고,

상기 슈퍼스텝은 국부 계산(local computation), 커뮤니케이션(communication) 및 베리어 동기화(barrier synchronization)과정을 포함하는 것을 특징으로 하는 데이터 연산 방법.

청구항 13

제9항에 있어서, R 언어를 이용하여 통계적 분석을 수행하는 단계에서는

R 언어에서 프리겔을 수행할 수 있도록, 상기 그래프 데이터에 대해 프리겔과 R 언어의 동기화를 수행하는 단계; 및

상기 그래프 데이터에 대해 R 언어를 이용하여 통계적 분석을 수행하는 단계

를 더 포함하는 것을 특징으로 하는 데이터 연산 방법.

청구항 14

제13항에 있어서,

상기 그래프 데이터에 대해 프리젠크와 R 언어의 동기화를 수행하는 단계에서는, 버텍스 단위로 동기화가 수행되는 것을 특징으로 하는 데이터 연산 방법.

청구항 15

제14항에 있어서,

상기 버텍스는 다른 버텍스와 메시지를 송수신 하는 것을 특징으로 하는 데이터 연산 방법.

청구항 16

제13항에 있어서,

상기 통계적 분석을 수행하는 단계에서는, 버텍스 단위로 그래프 데이터에 대한 통계적 처리를 수행하는 것을 특징으로 하는 데이터 연산 방법.

청구항 17

제16항에 있어서,

상기 통계적 분석이 수행된 그래프 데이터는 표준화된 문자열 형식인 것을 특징으로 하는 데이터 연산 방법.

청구항 18

제16항에 있어서,

상기 버텍스는 BSPConfiguration과 BinaryProtocol을 통해 사용자가 원하는 작업을 표준 출력의 문자열 형식으로 변환되고,

상기 BSPConfiguration은 프리젠크의 기본적인 연산인 setup, bsp, cleanup 연산으로 구성되어 있고,

상기 BinaryProtocol은 텍스트 기반 프로토콜으로서, 프리젠크로 데이터를 송수신할 때 사용되는 것을 특징으로 하는 데이터 연산 방법.

청구항 19

제9항에 있어서,

상기 그래프 데이터에 대해 분산 처리를 하는 단계는 클라우드(Cloud)에서 이루어지는 것을 특징으로 하는 데이터 연산 방법.

발명의 설명

기술 분야

본 발명은 데이터를 분석하기 위한 장치 및 방법에 관한 것으로, 보다 구체적으로는 분산 처리를 통해 그래프를

데이터 통계를 분석하는 장치 및 방법에 관한 것이다.

배경 기술

- [0002] 정보통신 산업의 발달은 인터넷을 이용한 정보의 검색 및 정보의 저장을 촉진시켰다. 이로 인하여, 사용자들은 인터넷 세계에 축적된 수많은 정보들로부터, 정보의 방향성과 정보의 분산 처리를 제공받을 수 있게 되었으며, 인터넷 세계에 축적된 수많은 정보들은 빅 데이터로 불리게 되었다.
- [0003] 스마트 기기의 발달은 인터넷 세계에 더욱 더 많은 정보들과 데이터들을 축적시키게 되었으며, 스마트 기기의 발달은 빅 데이터의 발달을 더욱 더 야기하였다. 특히, 소셜 네트워크 서비스(Social Network Service, SNS)는 그래프에 관한 수많은 정보를 인터넷 세계에 축적시켰으며, 이로 인하여 그래프 데이터는 대용량화, 빅 데이터화 되기에 이르렀다. 하지만, 종래의 기술에 따른 그래프 데이터를 분석하는 장치 또는 방법은 충분한 그래프 분석량 및 분석 속도를 제공하지 못하였으며, 프로그래밍 언어에 따라 적응적으로 그래프 데이터의 분석을 하지 못하는 문제점이 있었다.

발명의 내용

해결하려는 과제

- [0004] 본 발명이 해결하고자 하는 과제는 그래프 데이터를 분산적으로 처리하는 장치 및 방법을 제공하는 것이다.
- [0005] 본 발명이 해결하고자 하는 과제는 빅 데이터화된 그래프 데이터를 분산적으로 처리하는 장치 및 방법을 제공하는 것이다.
- [0006] 본 발명이 해결하고자 하는 과제는 분석 환경에 따라 적응적으로, 빅 데이터화된 그래프 데이터를 분산적으로 처리하는 장치 및 방법을 제공하는 것이다.
- [0007] 본 발명이 해결하고자 하는 과제는 분석 환경이 바뀔 경우에도, 빅 데이터화된 그래프 데이터를 분산적으로 처리하는 장치 및 방법을 제공하는 것이다.
- [0008] 본 발명이 해결하고자 하는 과제는 분석 환경에 따라 적응적으로 빅 데이터화된 그래프 데이터를 통계적으로 분석하는 장치 및 방법을 제공하는 것이다.

과제의 해결 수단

- [0009] 본 발명의 일 실시예에 따르면, HAMA로부터 그래프 데이터를 수신하는 단계, R 언어에서 프리젠크를 수행할 수 있도록, 상기 그래프 데이터에 대해 HAMA와 R 언어의 동기화를 수행하는 단계 및 상기 그래프 데이터에 대해 R 언어를 이용하여 통계적 분석을 수행하는 단계를 포함하는 프레임워크 방법을 제공한다.
- [0010] 이때, 상기 HAMA에서는 프리젠크를 수행할 수 있다.
- [0011] 이때, 상기 HAMA로부터 그래프 데이터를 수신하는 단계에서는, 프리젠크를 R 언어에서도 사용할 수 있도록 연결해주는 매개 함수인 BSPRunner가 HAMA로부터 그래프 데이터를 수신할 수 있다.
- [0012] 이때, 상기 그래프 데이터에 대해 HAMA와 R 언어의 동기화를 수행하는 단계에서는, 버텍스 단위로 동기화가 수행될 수 있다.
- [0013] 이때, 상기 버텍스는 다른 버텍스와 메시지를 송수신 할 수 있다.
- [0014] 이때, 상기 통계적 분석을 수행하는 단계에서는, 버텍스 단위로 그래프 데이터에 대한 통계적 처리를 수행할 수 있다.
- [0015] 이때, 상기 통계적 분석이 수행된 그래프 데이터는 표준화된 문자열 형식일 수 있다.
- [0016] 이때, 상기 버텍스는 BSPConfiguration과 BinaryProtocol을 통해 사용자가 원하는 작업을 표준 출력의 문자열 형식으로 변환되고, 상기 BSPConfiguration은 HAMA의 기본적인 연산인 setup, bsp, cleanup 연산으로 구성되어

있고, 상기 BinaryProtocol은 텍스트 기반 프로토콜으로서, HAMA로 데이터를 송수신할 때 사용될 수 있다.

- [0017] 본 발명의 다른 실시예에 따르면, 그래프 데이터를 수신하는 단계, 상기 그래프 데이터에 대해 분산 처리를 하는 단계 및 분산 처리된 상기 그래프 데이터에 대해 R 언어를 이용하여 통계적 분석을 수행하는 단계를 포함하는 데이터 연산 방법을 제공한다.
- [0018] 이때, 상기 그래프 데이터에 대해 분산 처리를 하는 단계에서는, 프리겔을 이용할 수 있다.
- [0019] 이때, 상기 프리겔은 버텍스 단위로 데이터에 대해 분산 처리를 수행할 수 있다.
- [0020] 이때, 상기 프리겔은 슈퍼스텝단위로 그래프 데이터에 대한 분산 처리를 하고, 상기 슈퍼스텝은 국부 계산(local computation), 커뮤니케이션(communication) 및 베리어 동기화(barrier synchronization)과정을 포함할 수 있다.
- [0021] 이때, R 언어를 이용하여 통계적 분석을 수행하는 단계에서는 R 언어에서 프리겔을 수행할 수 있도록, 상기 그래프 데이터에 대해 프리겔과 R 언어의 동기화를 수행하는 단계; 및 상기 그래프 데이터에 대해 R 언어를 이용하여 통계적 분석을 수행하는 단계를 더 포함할 수 있다.
- [0022] 이때, 상기 그래프 데이터에 대해 프리겔과 R 언어의 동기화를 수행하는 단계에서는, 버텍스 단위로 동기화가 수행될 수 있다.
- [0023] 이때, 상기 버텍스는 다른 버텍스와 메시지를 송수신 할 수 있다.
- [0024] 이때, 상기 통계적 분석을 수행하는 단계에서는, 버텍스 단위로 그래프 데이터에 대한 통계적 처리를 수행할 수 있다.
- [0025] 이때, 상기 통계적 분석이 수행된 그래프 데이터는 표준화된 문자열 형식일 수 있다.
- [0026] 이때, 상기 버텍스는 BSPConfiguration과 BinaryProtocol을 통해 사용자가 원하는 작업을 표준 출력의 문자열 형식으로 변환되고, 상기 BSPConfiguration은 HAMA의 기본적인 연산인 setup, bsp, cleanup 연산으로 구성되어 있고, 상기 BinaryProtocol은 텍스트 기반 프로토콜으로서, HAMA로 데이터를 송수신할 때 사용될 수 있다.
- [0027] 이때, 상기 그래프 데이터에 대해 분산 처리를 하는 단계는 클라우드(Cloud)에서 이루어질 수 있다.

발명의 효과

- [0028] 본 발명은 그래프 데이터를 분산적으로 처리하는 장치 및 방법을 제공한다.
- [0029] 본 발명은 빅 데이터화된 그래프 데이터를 분산적으로 처리하는 장치 및 방법을 제공한다.
- [0030] 본 발명은 분석 환경에 따라 적응적으로, 빅 데이터화된 그래프 데이터를 분산적으로 처리하는 장치 및 방법을 제공한다.
- [0031] 본 발명은 분석 환경이 바뀔 경우에도, 빅 데이터화된 그래프 데이터를 분산적으로 처리하는 장치 및 방법을 제공한다.
- [0032] 본 발명은 분석 환경에 따라 적응적으로 빅 데이터화된 그래프 데이터를 통계적으로 분석하는 장치 및 방법을 제공한다.

도면의 간단한 설명

- [0033] 도 1은 프리겔(pregel)을 이용하여 분산 처리를 하는 과정의 일례를 개략적으로 도시한 것이다.
- 도 2는 BSP(Bulk Synchronous parallel model)에 대한 개념을 개략적으로 도시한 것이다.
- 도 3은 버텍스의 연산 과정의 예시로서, 최대 값을 찾는 과정의 일례다.
- 도 4는 R과 프리겔을 결합하기 위한 프레임워크의 개념을 개략적으로 도시한 것이다.
- 도 5는 R과 프리겔을 결합하기 위한 프레임워크 시스템을 개략적으로 도시한 것이다.

도 6은 본 발명의 일 실시예에 따른 R과 프리젤을 결합하기 위한 프레임워크 방법의 순서도이다.

발명을 실시하기 위한 구체적인 내용

- [0034] 이하, 도면을 참조하여 본 발명의 실시 형태에 대하여 구체적으로 설명한다. 본 명세서의 실시예를 설명함에 있어, 관련된 공지 구성 또는 기능에 대한 구체적인 설명이 본 명세서의 요지를 흐릴 수 있다고 판단되는 경우에는 그 상세한 설명은 생략한다.
- [0035] 어떤 구성 요소가 다른 구성 요소에 “연결되어” 있다거나 “접속되어” 있다고 언급된 때에는, 그 다른 구성 요소에 직접적으로 연결되어 있거나 또는 접속되어 있을 수도 있으나, 중간에 다른 구성 요소가 존재할 수도 있다고 이해되어야 할 것이다. 아울러, 본 발명에서 특정 구성을 “포함” 한다고 기술하는 내용은 해당 구성 이외의 구성을 배제하는 것이 아니며, 추가적인 구성이 본 발명의 실시 또는 본 발명의 기술적 사상의 범위에 포함될 수 있음을 의미한다.
- [0036] 제1, 제2 등의 용어는 다양한 구성요소들을 설명하는데 사용될 수 있지만, 상기 구성요소들은 상기 용어들에 의해 한정되어서는 안 된다. 상기 용어들은 하나의 구성요소를 다른 구성요소로부터 구별하는 목적으로만 사용된다. 예를 들어, 본 발명의 권리 범위를 벗어나지 않으면서 제1 구성요소는 제2 구성요소로 명명될 수 있고, 유사하게 제2 구성요소도 제1 구성요소로 명명될 수 있다.
- [0037] 또한, 본 발명의 실시예에 나타나는 구성부들은 서로 다른 특징적인 기능들을 나타내기 위해 독립적으로 도시되는 것으로, 각 구성부들이 분리된 하드웨어나 하나의 소프트웨어 구성단위로 이루어짐을 의미하지 않는다. 즉, 각 구성부는 설명의 편의상 각각의 구성부로 나열하여 포함한 것으로 각 구성부 중 적어도 두 개의 구성부가 합쳐져 하나의 구성부로 이루어지거나, 하나의 구성부가 복수 개의 구성부로 나뉘어져 기능을 수행할 수 있고 이러한 각 구성부의 통합된 실시예 및 분리된 실시예도 본 발명의 본질에서 벗어나지 않는 한 본 발명의 권리범위에 포함된다.
- [0038] 또한, 일부의 구성 요소는 본 발명에서 본질적인 기능을 수행하는 필수적인 구성 요소는 아니고 단지 성능을 향상시키기 위한 선택적 구성 요소일 수 있다. 본 발명은 단지 성능 향상을 위해 사용되는 구성 요소를 제외한 본 발명의 본질을 구현하는데 필수적인 구성부만을 포함하여 구현될 수 있고, 단지 성능 향상을 위해 사용되는 선택적 구성 요소를 제외한 필수 구성 요소만을 포함한 구조도 본 발명의 권리범위에 포함된다.
- [0039] 그래프를 분산 처리 하는 방법 중 하나로, 프리젤을 통해 그래프의 분산 처리를 할 수 있다. 프리젤은 그래프에 대한 정보를 다수의 연산 장치에 분배하여 그래프에 대한 정보를 다수의 연산 장치에서 분산처리 하는 방법을 의미하며, 프리젤은 대용량을 가지는 그래프에 대한 정보를 분산 처리할 때, 기존의 그래프 정보에 대한 연산 장치보다 빠른 연산 속도와 큰 연산 용량을 가진다. 이하, 그래프에 대한 정보가 프리젤을 통해 분산 처리 되는 구체적인 과정을 설명한다.
- [0040] 도 1은 프리젤(pregel)을 이용하여 분산 처리를 하는 과정의 일례를 개략적으로 도시한 것이다.
- [0041] 도 1을 참조하면, 프리젤을 이용하여 분산 처리를 하는 과정에는 마스터(Master) 노드(Node)와, 워커(Worker) 노드(Node)를 포함하며, 워커 노드는 여러 개일 수 있다. 도 1에서는 워커 1(Worker 1) 노드, 워커 2(Worker 2) 노드, 워커 3(Worker 3) 노드를 포함한다. 이때, 프리젤은 클라우드(Cloud)에서 수행될 수 있으며, 클라우드는 온라인에 분산되어 존재하는 데이터 센터(Data Center)를 의미할 수 있다. 즉, 프리젤의 데이터 연산에 대한 대다수의 작업은 온라인에 분산되어 존재하는 클라우드에서 수행되고, 사용자의 컴퓨터는 클라우드에 접속하여 상호작용하는 단말기 역할만을 할 수도 있다.
- [0042] 마스터 노드는 각각의 워커 노드들과 연결되어 있으며, 마스터 노드는 각각의 워커 노드에게 버텍스(vertex)를 할당해준다. 버텍스는 정점을 의미하며, 버텍스는 그래프에 관한 정보를 포함할 수 있다. 버텍스에 포함된 그래프에 관한 정보는 좌표를 포함할 수 있으며, 좌표는 변수를 복수 개 가질 수 있다. 예컨대, 좌표는 변수를 3개 가질 수 있으며, 이때의 좌표는 3차원 좌표를 의미할 수 있다. 또한 버텍스는 그래프의 설정 값을 가질 수도 있다.
- [0043] 워커 노드는 마스터 노드로부터 버텍스를 수신할 수 있으며, 각각의 워커 노드는 수신받은 버텍스에 대한 연산을 수행할 수 있다. 각각의 버텍스는 BSP 모델에 의해 동작하며, 각각의 버텍스가 BSP 모델에 의해 동작하는 구

체적인 예는 후술하도록 한다.

- [0044] 프리퀀을 이용하여 분산 처리를 하는 과정의 일례로, 마스터 노드는 워커 1 노드에게 버텍스 4개를 할당하고, 워커 2 노드에는 버텍스 3개를 할당할 수 있으며, 마스터 노드는 워커 3 노드에게 버텍스 5개를 할당할 수 있다. 각각의 워커 노드들은 각각의 워커 노드에 할당된 버텍스에 대한 연산을 각각 수행할 수 있다. 버텍스는 다른 버텍스와 연결되어 의존적으로 연산을 수행할 수 있으며, 이때, 버텍스는 다른 워커에 존재하는 버텍스와도 연결되어 의존적으로 연산을 수행할 수도 있다.
- [0045] 도 2는 BSP(Bulk Synchronous parallel model)에 대한 개념을 개략적으로 도시한 것이다.
- [0046] 도 2를 참조하면, BSP는 국부 계산(local computation), 커뮤니케이션(communication) 및 배리어 동기화(barrier synchronization)을 포함할 수 있다.
- [0047] 국부 계산은 입력되는 데이터를 사용자가 정의한 함수로 계산하는 것을 의미한다. 국부 계산시 복수의 프로세서(processor)가 데이터 연산에 참여하며, 각각의 프로세서에 입력되는 데이터는 버텍스일 수 있다. 이때, 각각의 프로세서에 입력되는 각각의 데이터의 길이는 상이할 수 있으며, 각각의 프로세서에 입력되는 각각의 데이터의 길이는 동일할 수도 있다.
- [0048] 이때, 프리퀀은 버텍스의 계산을 할 때 필요한 정보를 설정할 수 있으며, 버텍스의 계산이 수행되기 전에 필요한 정보를 설정하는 함수는 'Setup' 함수로 정의될 수 있다. 또한, 버텍스는 'BSP Peer(Bulk Synchronous parallel model Peer)'로 정의될 수 있다. 프리퀀은 버텍스에서 계산이 다 끝나거나, 계산이 실패 했을 경우에는 각각의 프로세서에 할당된 버텍스를 해제할 필요가 있으며, 각각의 프로세서에 할당된 버텍스를 해제하는 함수는 'cleanup' 함수로 정의될 수 있다.
- [0049] 커뮤니케이션 과정에서는 국부 계산을 통해 프로세서에서 연산된 각각의 데이터가 다른 버텍스에게 전송돼야 할 경우, 프로세서에서 연산된 데이터를 다른 버텍스에게 전송한다. 이때, 커뮤니케이션 과정은 메시지 전달 방식을 이용하여, 계산 값을 다른 버텍스에게 전달할 수 있다.
- [0050] 각각의 버텍스가 모든 연산을 완료 할 경우, 각각의 버텍스는 배리어에 도달했다는 정보를 발생시킨다. 이때, 배리어 동기화 과정에서는 모든 버텍스가 모든 연산을 완료할 때까지 각각의 버텍스는 다른 버텍스의 연산이 완료되기를 기다린다.
- [0051] 상술한 국부 계산(local computation), 커뮤니케이션(communication) 및 배리어 동기화(barrier synchronization)과정이 한번 완료되는 과정을 슈퍼스텝(Superstep)라고 한다.
- [0052] 도 3은 버텍스의 연산 과정의 예시로서, 최대 값을 찾는 과정의 일례다.
- [0053] 도 3에서는 버텍스가 4개의 프로세서에 각각 한 개씩 할당되어 있고, 각각의 버텍스는 3, 6, 2, 1의 값을 가질 경우, 최대의 값을 가지는 버텍스를 구하는 과정의 예시를 나타내며, 실선은 각각의 버텍스가 값을 비교할 대상을 의미한다. 이때, 버텍스의 연산 과정은 각각의 슈퍼스텝 단위로 이루어진다. 이하, 버텍스의 연산 과정에 대한 구체적인 예를 각각의 슈퍼스텝에 따라 설명한다.
- [0054] 슈퍼스텝 0에서의 각각의 버텍스는 버텍스 1, 버텍스 2, 버텍스 3, 버텍스 4를 의미할 수 있다. 이때, 버텍스 1은 3의 값을 가지고, 버텍스 2는 6을 가지며, 버텍스 3은 2의 값을 가지고, 버텍스 4는 1의 값일 수 있다.
- [0055] 슈퍼스텝 1에서, 버텍스 1은 버텍스 2와 비교를 수행한다. 이때, 버텍스 1은 버텍스 2보다 작은 값을 가지므로, 버텍스 1은 버텍스 2에 자신의 값을 삽입할 수 없다. 이후, 버텍스 2는 화살표를 따라, 버텍스 1 및 버텍스 4와 비교를 수행한다. 버텍스 2는 버텍스 1 및 버텍스 4보다 큰 값을 가지고 있기 때문에, 버텍스 2의 값을 버텍스 1 및 버텍스 4에 삽입한다. 버텍스 3은 화살표를 따라, 버텍스 2 및 버텍스 4와 비교를 수행한다. 이때, 버텍스 3은 버텍스 2보다는 작은 값을 가지고 있기 때문에, 버텍스 3의 값을 버텍스 2에 삽입할 수는 없으며, 버텍스 3은 버텍스 4보다 큰 값을 가지고는 있지만, 이미 버텍스 2의 값을 버텍스 4에 삽입하였으므로, 버텍스 3의 값을 버텍스 4에 삽입하지는 않는다. 버텍스 4는 화살표를 따라 버텍스 3과의 비교를 수행한다. 하지만 버텍스 4는 버텍스 3보다 작은 값을 가지고 있기 때문에, 버텍스 4의 값을 버텍스 3에 삽입하지는 않는다. 상술한 과정에서, 외부로부터 새로운 값이 삽입된 버텍스는 활성화된 버텍스이며, 외부로부터 새로운 값이 삽입되지 않은 버텍스는 비활성화된 버텍스를 의미한다.

- [0056] 슈퍼스텝 2에서, 버텍스 1은 화살표를 따라, 버텍스 2와 비교를 수행하며, 버텍스 1의 값과, 버텍스 2의 값은 일치하므로, 버텍스 1의 값을 버텍스 2에 삽입하지는 않는다. 이후, 버텍스 4는 화살표를 따라, 버텍스 3과 비교를 수행하며, 버텍스 4의 값은 버텍스 3의 값보다 크기 때문에, 버텍스 3에는 버텍스 4의 값을 삽입한다. 상술한 과정에서, 외부로부터 새로운 값이 삽입된 버텍스는 활성화된 버텍스이며, 외부로부터 새로운 값이 삽입되지 않은 버텍스는 비활성화된 버텍스를 의미한다.
- [0057] 슈퍼스텝 3에서, 버텍스 3은 화살표를 따라 버텍스 2 및 버텍스 4와 비교를 수행한다. 이때, 버텍스 3은 버텍스 2 및 버텍스 4와 동일한 값을 가지고 있기 때문에, 버텍스 3의 값을 버텍스 2 및 버텍스 4에 삽입하지는 않는다. 슈퍼스텝 3에서는 모든 버텍스가 비활성화 된 상태이므로, 버텍스의 연산 과정을 종료한다.
- [0058] 상술한 바와 같이 프리겔은 대용량 데이터를 가지는 그래프를 분산 처리하여 보다 빠른 데이터 연산을 하는데 적합하지만, 그래프에 대해 통계적 처리를 하여 그래프 데이터를 분석하는 것은 불가능하다. 따라서, 대용량 데이터를 가지는 그래프를 분산 처리 하면서, 동시에 그래프에 대한 통계적 처리를 할 수 있는 방법 및 장치가 요구된다.
- [0059] R 언어는 통계 함수를 포함하고 있다. 따라서, 그래프는 R 언어를 이용하여 통계적 처리될 수 있으며, R 언어는 그래프에 대한 정보를 분석할 수 있다. 따라서, 프리겔을 이용하여 그래프에 대한 데이터를 분산 처리하되, 동시에 R언어를 이용하여 그래프에 대한 정보를 분석하는 방법 및 장치가 요구된다.
- [0060] 프리겔과 R 언어는 각각 고유한 알고리즘과 언어를 가지고 있다. 따라서, 프리겔을 이용하여 그래프에 대한 데이터를 분산 처리하되, 동시에 R언어를 이용하여 그래프에 대한 정보를 분석하기 위해서는 R 언어와 프리겔을 모두 사용할 수 있게 하는 매개체가 요구된다. 이때, R 언어와 프리겔을 이어주는 매개체는 프레임 워크를 의미한다. 즉, 프레임워크는 동시에 하나 이상의 기능을 사용할 수 있게 해주는 방법 및 장치를 의미한다.
- [0061] 이하에서는 보다 구체적으로 프레임워크에 대해 설명하도록 한다.
- [0062] 도 4는 R과 프리겔을 결합하기 위한 프레임워크의 개념을 개략적으로 도시한 것이다.
- [0063] 도 4를 참조하면, R과 프리겔을 결합하기 위한 프레임워크는 R API, rHama Core, HAMA를 포함할 수 있다. 이때, R 언어와 프리겔을 결합하기 위한 프레임워크는 rhama로 정의될 수 있다.
- [0064] R API(application programming interface)는 R언어에 대한 함수들의 집합을 의미할 수 있으며, R API는 R언어를 통해 데이터의 통계적 분석을 제공할 수 있다.
- [0065] rHama Core는 BinaryProtocol, BSPConfiguration, BSPPeer 및 BSPRunner를 포함할 수 있다. rHama Core는 R 언어와, 프리겔의 연결고리 역할을 하며, rHama Core는 R 언어에서 분산 처리를 할 수 있도록, 프리겔에서 작동하는 함수들이 저장되어 있다.
- [0066] BinaryProtocol은 텍스트 기반 프로토콜로서, R에서 작성된 데이터를 HAMA로 송수신할 때 사용된다. BinaryProtocol에는 프리겔의 기본 연산들이 고유 식별자(OP_CODE)로 정의되어 있다. 이때, BinaryProtocol의 구체적인 실시 예는 아래 표 1과 같다.

표 1

```

1 BinaryProtocol = function(){
2
3     START = 0
4     SET_BSPJOB_CONF = 1
5     SET_INPUT_TYPES = 2
6     RUN_SETUP = 3
7     RUN_BSP = 4
8     RUN_CLEANUP = 5
9     READ_KEYVALUE = 6
10    WRITE_KEYVALUE = 7
11    GET_MSG = 8
12    GET_MSG_COUNT = 9
13    SEND_MSG = 10
14    SYNC = 11
15    GET_ALL_PEERNAME = 12
16    GET_PEERNAME = 13
17    GET_PEER_INDEX = 14
18    GET_PEER_COUNT = 15
19    GET_SUPERSTEP_COUNT = 16
20    REOPEN_INPUT = 17
21    CLEAR = 18
22    CLOSE = 19
23    ABORT = 20
24    DONE = 21
25    TASK_DONE = 22
26    REGISTER_COUNTER = 23
27    INCREMENT_COUNTER = 24
28    LOG = 25
29
30    getProtocolString = function(opcode){
31        return ( "%" + str(opcode) + "%=")
32    }
33    getAckProtocolString = function(opcode){
34        return ("%ACK_" + str(opcode) + "%=")
35    }
36 }

```

[0067]

[0068] BSPConfiguration은 HAMA의 기본적인 연산인 setup, bsp, cleanup 연산을 포함하고 있다. 이때, setup은 각각의 버텍스에 대한 속성 값들을 할당 하고, bsp는 분산 처리하는 사용자 정의 함수를 의미하며, cleanup은 버텍스와 같은 리소스를 해제하는 역할을 한다. 또한, BSPConfiguration은 알고리즘의 입력 파라미터를 관리하기 위한 연산을 제공한다.

[0069] BSPPeer는 HAMA와 동기화를 하거나, 버텍스 간 메시지를 송수신 하거나, 현재의 슈퍼스텝의 횟수를 확인하는 등의 작업을 수행한다. 즉, BSPPeer는 HAMA의 동작에 필요한 함수를 포함한다. 이때, HAMA API는 HAMA에서 사용되는 함수들의 집합을 의미할 수 있다.

[0070] BSPRunner는 사용자가 작성한 R프로그램을 프리젤 내의 테스크에 생성하는 작업을 한다.

[0071] HAMA는 HAMA API, HAMA Core, HAMA Shell, MapReduce BSP, Zookeeper, HBase, HDFS, RDBMS 및 File을 포함할 수 있다. 이때, HAMA는 상술한 프리젤에 해당하며, 프리젤에 해당하는 HAMA에서는 그래프에 대한 분산 처리를 수행할 수 있다. 프리젤에 대한 구체적인 설명은 상술한 바와 같다.

- [0072] 도 5는 R과 프리젤을 결합하기 위한 프레임워크 시스템을 개략적으로 도시한 것이다.
- [0073] 도 5를 참조하면, 프레임워크 시스템은 클라이언트, HAMA, 파이프 어플리케이션, BSPRunner, BSP_HelloWorld, BSPPeer, BSPConfiguration, BinaryProtocol, StreamingProtocol을 포함할 수 있다.
- [0074] 클라이언트(Client)는 사용자를 의미하고, HAMA와 정보를 송수신하며, 클라이언트는 HAMA로부터 분산 처리된 그래프 데이터를 송수신할 수 있다. 또한, 클라이언트는 R에 기반한 그래프에 대한 통계적 정보에 대해서도 수신할 수 있다.
- [0075] HAMA는 프리젤을 의미하며, 상술한 바와 같이 HAMA는 그래프를 분산 처리한다. HAMA가 그래프를 분산 처리하는 구체적인 방법은 상술한 바와 같으며, HAMA를 통해 그래프의 분산 처리를 함과 동시에 그래프에 대한 통계 처리를 하기 위해서는 프레임워크가 필요하다. 프레임 워크에는 R과 프리젤을 결합하는 프레임 워크인 rHama가 사용될 수 있으며, rHama의 구체적 내용은 상술한 바와 같다.
- [0076] 이때, HAMA는 파이프 어플리케이션을 통하여, 표준화된 문자열로 이루어진 통계적 정보를 수신할 수 있으며, 표준화된 문자열로 이루어진 통계적 정보의 생성 방법은 후술하도록 한다.
- [0077] 파이프 어플리케이션은 HAMA와 프레임워크를 연결시켜주는 역할을 한다. 즉, 파이프 어플리케이션은 서로 다른 언어로 작성된 프로그램을 연결시켜주는 가교 역할을 하며, 파이프 어플리케이션은 HAMA로부터 그래프에 대한 데이터를 송수신 하고, Streaming Protocol로부터는 표준화된 문자열로 이루어진 통계적 정보를 수신할 수 있다.
- [0078] BSPRunner는 파이프 어플리케이션을 통해 HAMA와 연결되며, BSPRunner는 R 언어로 구현된 함수를 동작시키기 위한 매개체 역할을 한다. 이때, R 언어는 R 인터프리터(Interpreter)를 통해 정의될 수 있으며, BSPRunner는 사용자가 작성한 R프로그램을 프리젤 내의 태스크에 생성하는 작업을 한다.
- [0079] 이후, 사용자가 정의한 함수는 BSPRunner로부터 정보를 수신하며, 사용자가 정의한 함수는 BSPPeer에 정의되어 있는 함수를 이용하여 작성될 수 있다. 이때, 사용자가 정의한 함수는 'BSP_HelloWorld' 로 정의될 수 있다.
- [0080] BSPPeer는 BSP_HelloWorld로부터 정보를 수신하며, BSPPeer는 HAMA와 동기화를 하거나, 버텍스 간 메시지를 송수신 하할 수 있다. 또한, BSPPeer는 현재의 슈퍼스택의 횟수를 확인하는 등의 작업을 수행할 수 있으며, BSPPeer의 구체적인 내용은 상술한 바와 같다. BSPPeer는 BSPConfiguration과 BinaryProtocol을 이용하여 사용자가 원하는 작업을 표준 출력에 문자열 형식으로 생성하며, BSPConfiguration은 HAMA의 기본적인 연산인 setup, bsp, cleanup 연산으로 구성되어 있고, BinaryProtocol은 텍스트 기반 프로토콜로서, R에서 작성한 프로그램과 HAMA로 데이터를 송수신할 때 사용된다.
- [0081] Streaming Protocol은 생성된 표준 출력형식의 데이터를 읽어오며, 그 결과를 파이프 어플리케이션에 전달한다.
- [0082] 이후, HAMA는 파이프 어플리케이션을 통해 수신한 표준 출력형식의 데이터에 대해 프리젤을 수행하며, 프리젤의 수행 결과인 분산 처리된 그래프 데이터는 클라이언트에게 제공된다.
- [0083] 상술한 과정인, 프리젤 기반의 알고리즘에 대한 일 실시예를 R에 의해 구현할 경우, 표 2와 같다.

표 2

[0084]

```

assignCenters : Center를 구하기 위한 함수
sync : Vertex간의 동기화를 위한 함수
updateCenters : assignCenters를 통해 구해진 Center로 갱신하는 함수
superstepcount : This time superstepcount
1. assignCenters<-function(peer){
2.   newCenterArray <- array(dim=numCenters)
3.   summationCount <- array(dim=numCenters)
4.   lowersIndex = 0,lowest <- as.double(999999), i <- 1
5.   while(i <= numCenters){
6.     dist <- distance(vector,centers[i])
7.     if(lowest > dist){
8.       lowest = dist
9.       lowestIndex = i
10.    }i = i+1
11.  }
12.  while(i <= numCenters){
13.    for(peerName in getAllPeerNames(peer)){

```

```

14.      msg <-c(toString(i), toString(summationCount[i]))
15.      send(peer,peerName,toString(msg))
16.    }
17.    }i <-i+1
18.  }
19.  updateCenters<-function(peer){
20.    msgCenters = array(dim = numCenters)
21.    incrementSum <<- array(dim = numCenters)
22.    for(msg in getAllMessages(peer)){
23.      split = strsplit(msg, split ="[ ]")
24.      centerIndex = as.integer(split[[1]][1])
25.      oldCenter = msgCenters[centerIndex]
26.      incrementSum[centerIndex] = incrementSum[centerIndex] + as.integer(split[[1]][2])
27.      newCenter = toVector(split[[1]][3])
28.      if(is.na(oldCenter))msgCenters[centerIndex] = newCenter
29.      else{ msgCenters[centerIndex] = sum(oldCenter, newCenter)
30.    }
31.  } i <-1
32.  ma <<- matrix(0,nrow = numCenters)
33.  while(i <= numCenters){
34.    msgCenters[i] = msgCenters[i]/incrementSum[i]
35.    i = i+1
36.  }
37. }
38. setup<-function(peer){
39.  numCenters <-as.integer(get$config, "kmeans.num.centers" )
40.  maxIterations <- as.integer(get$config, "kmeans.max.iterations" )
41.  i<-0
42.  while(i<numCenters){
43.    centers<-append(centers,value) i = i+1
44.  }
45. }
46. bsp<-function(peer){
47.  while (TRUE){
48.    assignCenters(peer)
49.    sync(peer)
50.    updateCenters(peer)
51.    if(maxIterations < getSuperstepCount(peer)) break
52.  }
53. }

```

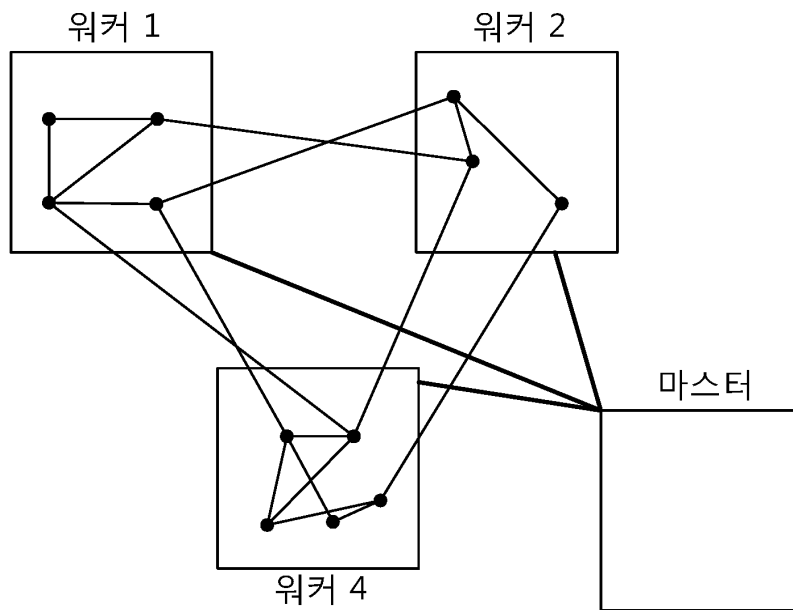
[0085] 표 2는 rHama를 이용한 k-평균 알고리즘의 일 실시예이며, 표 2에 대해 구체적인 내용은 아래와 같다.

[0086] 표 2에서는 k-평균 알고리즘(K-means algorithm)을 프리젠티 처리 하기 위하여, 데이터를 그래프로 모델링 하였다. 이때, K-평균 알고리즘은 주어진 데이터를 k개의 버텍스로 묶는 알고리즘으로, 각 버텍스와 거리 차이의 분산을 최소화하는 방식으로 동작한다. 같은 프로세스에 할당된 데이터들은 버텍스에 할당되고, 각 버텍스는 모두 연결되며, 버텍스를 연결할 때, 각각의 버텍스는 에지를 통해 연결된다. 이후, 각 버텍스 내의 데이터들에 대한 센터가 계산되며, 계산된 센터가 이웃 버텍스에게 전달된다. 계산된 센터 값을 수신한 이웃 버텍스는 수신된 센터 값으로 자신의 값을 갱신한다. 상술한 과정을 통해 k-평균 알고리즘은 각 버텍스와 거리의 차이에 대한 분산을 최소화할 수 있다. 이때, 사용자 정의 함수(line37-44) 과정에서는 센터를 계산하는 함수가 호출되고 계산이 끝나면 버텍스 간에 동기화를 한다. 사용자 정의 함수 과정은 더 이상 센터가 갱신되지 않거나, 현재 슈퍼 스텝의 횟수가 최대 반복회수보다 작을 때까지 반복하며 k개의 센터를 구할 수 있다.

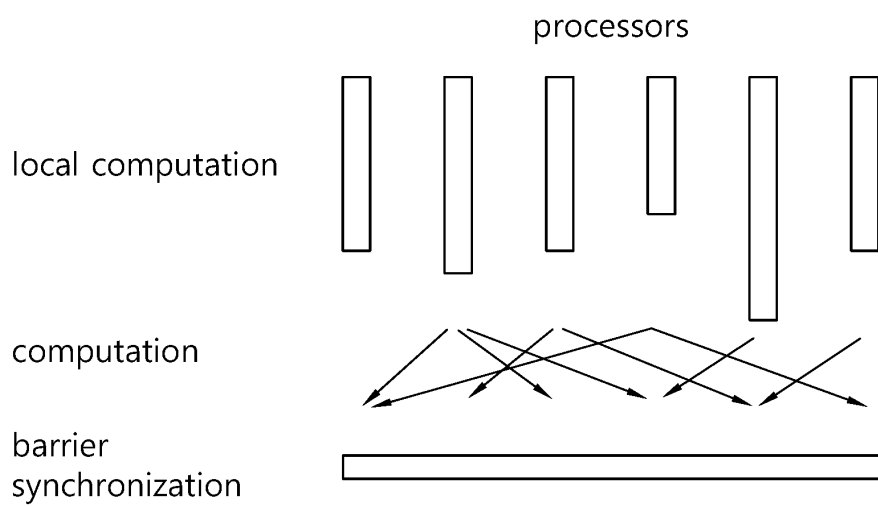
- [0087] 도 6은 본 발명의 일 실시예에 따른 R과 프리젤을 결합하기 위한 프레임워크 방법의 순서도이다.
- [0088] 프레임워크는 HAMA로부터 데이터를 수신한다(S600). 이때, 프레임워크가 수신하는 데이터는 프리젤에 기반하여 통계적 처리를 요청하는 정보를 의미할 수 있다. 또한, 프레임워크가 수신하는 데이터는 프리젤에 기반하여 통계적 처리가 수행되는 그래프 데이터를 의미할 수도 있다. 프레임워크가 HAMA로부터 데이터를 수신할 때, 프레임워크는 BSPRunner를 이용할 수 있으며, BSPRunner는 파이프 어플리케이션을 통해 HAMA와 연결될 수 있다. BSPRunner는 프리젤을 R 언어에서도 사용할 수 있도록 연결해주는 매개 함수를 의미하며, BSPRunner는 사용자가 작성한 R 프로그램을 프리젤 내의 테스크에 생성하는 작업을 한다. 이때, R 언어는 R 인터프리터(Interpreter)를 통해 정의될 수 있다.
- [0089] 이후, 프레임워크는 HAMA와 동기화를 수행한다(S610). HAMA와 동기화를 수행할 때, 프레임워크는 BSPPeer를 이용할 수 있으며, BSPPeer는 R 언어를 이용할 경우에도 HAMA의 연산을 수행할 수 있다. 이때, BSPPeer는 HAMA와 동기화를 하거나, 버텍스 간 메시지를 송수신 할 수 있으며, BSPPeer는 현재의 슈퍼스텝의 횟수를 확인하는 작업을 수행할 수도 있다.
- [0090] 프레임워크는 그래프 데이터에 대해 통계적 처리를 수행한다(S620). 프레임워크는 상술한 BSPPeer를 통하여 그래프 데이터에 대한 통계적 처리를 수행하며, 통계적 처리된 그래프 데이터는 표준화된 문자열 형식으로 생성될 수 있다. 이때, BSPPeer는 BSPConfiguration과 BinaryProtocol을 통해 사용자가 원하는 작업을 표준 출력의 문자열 형식으로 변환되며, BSPConfiguration은 하마의 기본적인 연산인 setup, bsp, cleanup 연산으로 구성되어 있고, BinaryProtocol은 텍스트 기반 프로토콜으로서, R에서 작성한 프로그램과 HAMA로 데이터를 송수신할 때 사용된다.
- [0091] 프레임워크는 Streaming Protocol을 통해, 생성된 표준 출력형식의 데이터를 읽을 수 있으며, 프레임 워크는 생성된 표준 출력 형식의 데이터를 파이프 어플리케이션에 전달할 수 있다.
- [0092] 상술한 실시예들에서, 방법들은 일련의 단계 또는 유닛으로서 순서도를 기초로 설명되고 있으나, 본 발명은 단계들의 순서에 한정되는 것은 아니며, 어떤 단계는 상술한 바와 다른 단계와 다른 순서로 또는 동시에 발생할 수 있다. 또한, 당해 기술 분야에서 통상의 지식을 가진 자라면 순서도에 나타난 단계들이 배타적이지 않고, 다른 단계가 포함되거나, 순서도의 하나 또는 그 이상의 단계가 본 발명의 범위에 영향을 미치지 않고 삭제될 수 있음을 이해할 수 있을 것이다.
- [0093] 상술한 실시예는 다양한 양태의 예시들을 포함한다. 다양한 양태들을 나타내기 위한 모든 가능한 조합을 기술할 수는 없지만, 해당 기술 분야의 통상의 지식을 가진 자는 다른 조합이 가능함을 인식할 수 있을 것이다. 따라서, 본 발명은 이하의 특허청구범위 내에 속하는 모든 다른 교체, 수정 및 변경을 포함한다고 할 것이다.
- [0094] 상술한 본 발명에 따른 방법은 컴퓨터에서 실행되기 위한 프로그램으로 제작되어 컴퓨터가 읽을 수 있는 기록 매체에 저장될 수 있으며, 컴퓨터가 읽을 수 있는 기록 매체의 예로는 ROM, RAM, CD-ROM, 자기 테이프, 플로피 디스크, 광 데이터 저장장치 등이 있으며, 또한 캐리어 웨이브(예를 들어 인터넷을 통한 전송)의 형태로 구현되는 것도 포함한다.
- [0095] 컴퓨터가 읽을 수 있는 기록 매체는 네트워크로 연결된 컴퓨터 시스템에 분산되어, 분산방식으로 컴퓨터가 읽을 수 있는 코드가 저장되고 실행될 수 있다. 그리고, 상기 방법을 구현하기 위한 기능적인(function) 프로그램, 코드 및 코드 세그먼트들은 본 발명이 속하는 기술분야의 프로그래머들에 의해 용이하게 추론될 수 있다.
- [0096] 또한, 본 발명의 바람직한 실시예에 대하여 도시하고 설명하였지만, 본 발명은 상술한 특정의 실시예에 한정되지 아니하며, 청구범위에서 청구하는 본 발명의 요지를 벗어남이 없이 당해 발명이 속하는 기술분야에서 통상의 지식을 가진 자에 의해 다양한 변형 실시가 가능한 것은 물론이고, 이러한 변형 실시들은 본 발명의 기술적 사상이나 전망으로부터 개별적으로 이해 되어서는 안 될 것이다.

도면

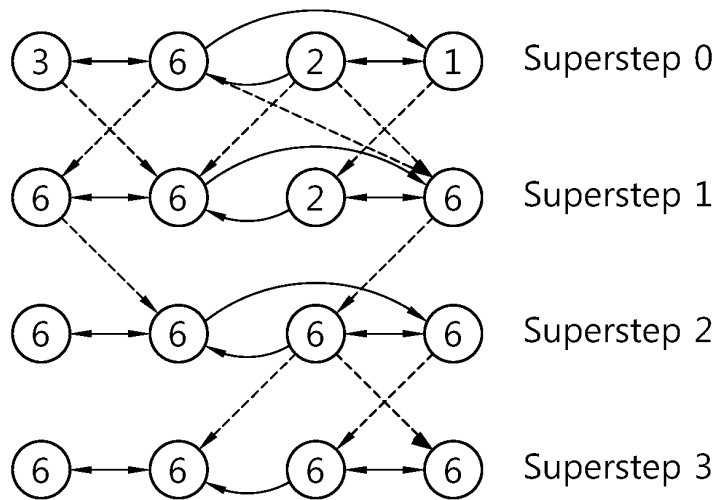
도면1



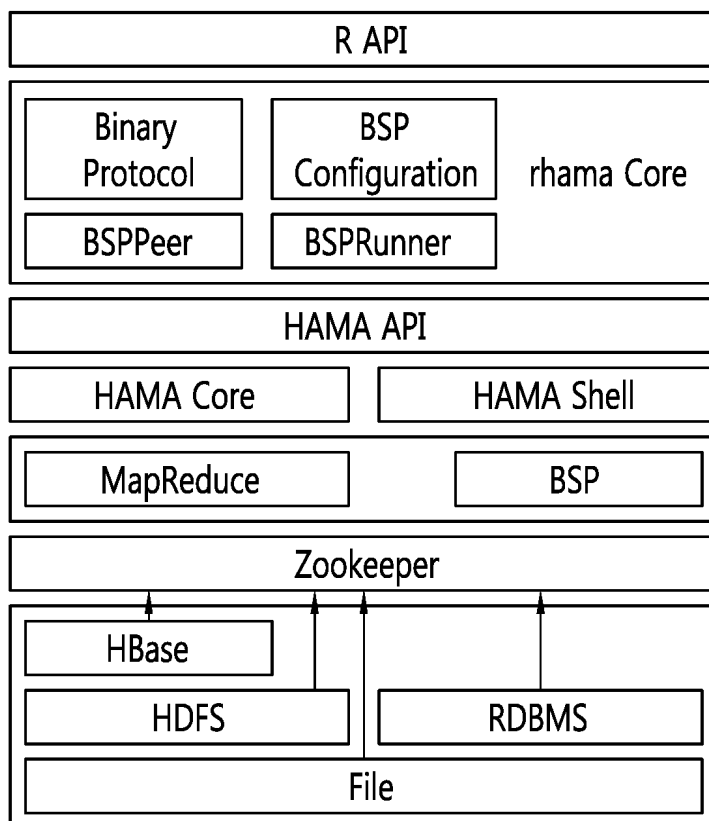
도면2



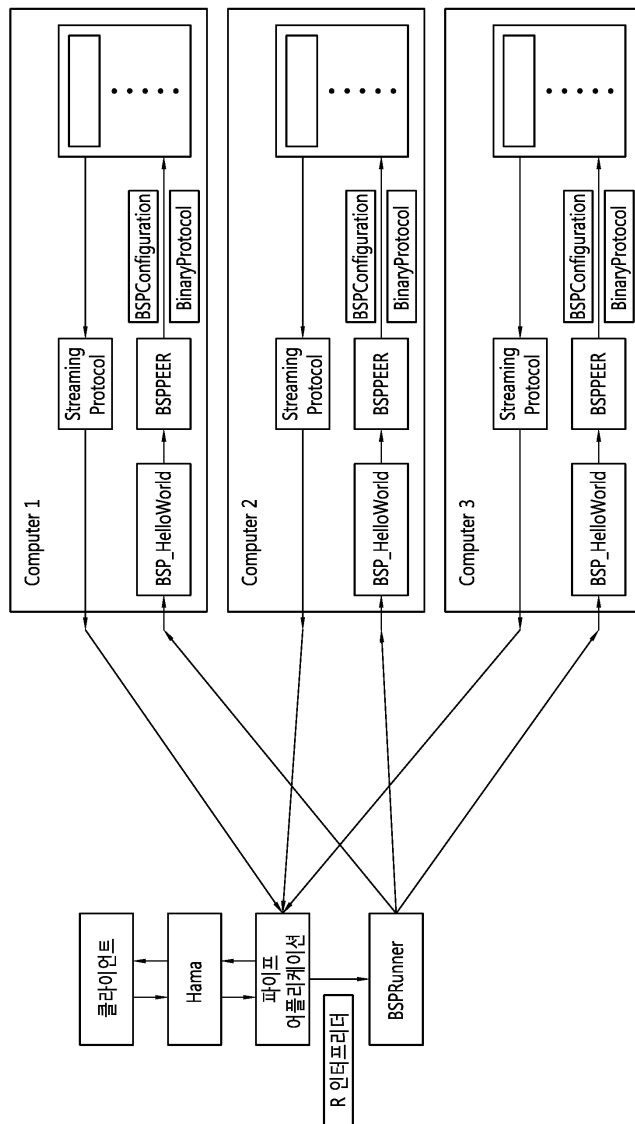
도면3



도면4



도면5



도면6

