

(12) 특허협력조약에 의하여 공개된 국제출원

(19) 세계지식재산권기구
국제사무국



(10) 국제공개번호

WO 2020/184982 A1

(43) 국제공개일

2020년 9월 17일 (17.09.2020)

WIPO | PCT

(51) 국제특허분류:

G06F 9/50 (2006.01)

Jung Ho); 08725 서울시 관악구 성현로 80, 112동 1801호, Seoul (KR).

(21) 국제출원번호:

PCT/KR2020/003396

(74) 대리인: 안제성 등 (AHN, Je Sung et al.); 06253 서울시 강남구 도곡로 117, 옥신타워 14층, AnK 특허법률사무소, Seoul (KR).

(22) 국제출원일:

2020년 3월 11일 (11.03.2020)

(25) 출원언어:

한국어

(26) 공개언어:

한국어

(30) 우선권정보:

10-2019-0027672 2019년 3월 11일 (11.03.2019) KR

10-2020-0030158 2020년 3월 11일 (11.03.2020) KR

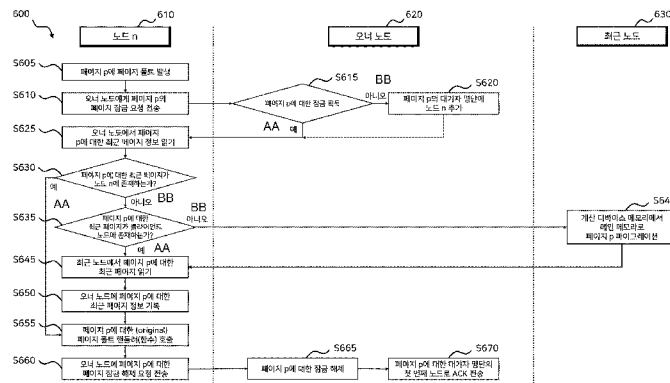
(71) 출원인: 서울대학교산학협력단 (SEOUL NATIONAL UNIVERSITY R&DB FOUNDATION) [KR/KR]; 08826 서울시 관악구 관악로 1, Seoul (KR).

(72) 발명자: 이재진 (LEE, Jae Jin); 06610 서울시 서초구 서운로 200, 112동 604호, Seoul (KR). 박정호 (PARK,

(81) 지정국 (별도의 표시가 없는 한, 가능한 모든 종류의 국내 권리의 보호를 위하여): AE, AG, AL, AM, AO, AT, AU, AZ, BA, BB, BG, BH, BN, BR, BW, BY, BZ, CA, CH, CL, CN, CO, CR, CU, CZ, DE, DJ, DK, DM, DO, DZ, EC, EE, EG, ES, FI, GB, GD, GE, GH, GM, GT, HN, HR, HU, ID, IL, IN, IR, IS, JO, JP, KE, KG, KH, KN, KP, KW, KZ, LA, LC, LK, LR, LS, LU, LY, MA, MD, ME, MG, MK, MN, MW, MX, MY, MZ, NA, NG, NI, NO, NZ, OM, PA, PE, PG, PH, PL, PT, QA, RO, RS, RU, RW, SA, SC, SD, SE, SG, SK, SL, ST, SV, SY, TH, TJ, TM, TN, TR, TT, TZ, UA, UG, US, UZ, VC, VN, WS, ZA, ZM, ZW.

(54) Title: METHOD FOR EXECUTING PROGRAM TO BE EXECUTED IN HETEROGENEOUS CLUSTER SYSTEM, AND COMPUTER PROGRAM

(54) 발명의 명칭: 이종클러스터 시스템에서 실행되는 프로그램을 실행시키는 방법 및 컴퓨터 프로그램



610 ... Node
620 ... Owner node
630 ... Recent node
S605 ... Generate page fault in page p
S610 ... Transmit page lock request for page p to owner node
S615 ... Acquire lock for page p
S620 ... Add node n to waiting list for page p
S625 ... Read recent page information about page p in owner node
S630 ... Does recent page about page p exist in node n?
S635 ... Does recent page about page p exist in client node?
S640 ... Migrate page p from calculation device memory to main memory
S645 ... Read recent page about page p in recent node
S650 ... Record recent page information about page p in owner node
S655 ... Call (original) page fault handler (function) on page p
S660 ... Transmit page unlock request for page p to owner node
S665 ... Unlock page p
S670 ... Transmit ACK to first node of waiting list for page p
AA ... Yes
BB ... No

(57) Abstract: A method for executing a program in a heterogeneous cluster system is disclosed. The method for executing a program in a heterogeneous cluster system comprises the steps in which: each of a plurality of calculation devices included in a heterogeneous cluster system receives a plurality of work groups linked with a program allocated from a client node of the heterogeneous cluster system; and each of the plurality of calculation devices processes the plurality of allocated work groups, wherein a page accessed by the plurality of work groups is managed through a virtual page included in a virtual memory for providing a single memory address space in the heterogeneous cluster system.

[다음 쪽 계속]

(84) 지정국 (별도의 표시가 없는 한, 가능한 모든 종류의 역내 권리의 보호를 위하여): ARIPO (BW, GH, GM, KE, LR, LS, MW, MZ, NA, RW, SD, SL, ST, SZ, TZ, UG, ZM, ZW), 유라시아 (AM, AZ, BY, KG, KZ, RU, TJ, TM), 유럽 (AL, AT, BE, BG, CH, CY, CZ, DE, DK, EE, ES, FI, FR, GB, GR, HR, HU, IE, IS, IT, LT, LU, LV, MC, MK, MT, NL, NO, PL, PT, RO, RS, SE, SI, SK, SM, TR), OAPI (BF, BJ, CF, CG, CI, CM, GA, GN, GQ, GW, KM, ML, MR, NE, SN, TD, TG).

공개:

— 국제조사보고서와 함께 (조약 제21조(3))

(57) 요약서: 본 개시는 이중클러스터 시스템에서 프로그램을 실행시키는 방법에 관한 것이다. 이중클러스터 시스템에서 프로그램을 실행시키는 방법은, 이중클러스터 시스템에 포함된 복수의 계산 디바이스의 각각에서, 이중클러스터 시스템의 클라이언트 노드로부터 할당된 프로그램과 연관된 복수의 워크 그룹을 수신하는 단계 및 복수의 계산 디바이스의 각각에서 할당된 복수의 워크 그룹을 처리하는 단계를 포함하고, 복수의 워크 그룹에 의해 접근되는 페이지는 이중클러스터 시스템 상에서 단일 메모리 주소 공간을 제공하도록 구성된 가상 메모리에 포함된 가상 페이지를 통해 관리된다.

명세서

발명의 명칭: 이중클러스터 시스템에서 실행되는 프로그램을 실행시키는 방법 및 컴퓨터 프로그램

기술분야

- [1] 본 발명은 이중 클러스터 시스템에서 소프트웨어를 실행시키는 방법 및 소프트웨어에 관한 것으로, 더 구체적으로, 이중클러스터 시스템에 포함된 복수의 계산 디바이스의 각각에서, 가상 메모리를 이용하여 이중클러스터 시스템의 클라이언트 노드로부터 수신된 프로그램과 연관된 복수의 워크 그룹을 처리하는 방법 및 컴퓨터 프로그램에 관한 것이다.

[2]

배경기술

- [3] 프로그래밍 복잡성 문제를 해결하기 위해서 복수의 가속기가 장착된 이중 시스템에서 애플리케이션에 단일한 가상 계산 디바이스 또는 가속기를 제공하는 기술에 대한 다양한 연구가 있어왔다. 즉, 이러한 기술은 단일한 계산 디바이스를 타겟으로 작성된 애플리케이션이 그대로 복수의 계산 디바이스로 나누어 실행하는 기술을 지칭한다. 이러한 종래의 기술에 따르면, 호스트 프로그램이 단일한 가상의 계산 디바이스에 커널 실행 명령을 내리면 해당 커널이 행하는 작업을 시스템에 장착된 복수의 실제 디바이스에 나누어 실행시킨다.
- [4] 종래의 기술에 따르면, GPU 등과 같은 가속기에 임의의 명령(command)이 입력되면 해당 명령을 여러 GPU 대상으로 나누어 실행할 수 있다. 예를 들면, 1000개의 워크 아이템(work items; CUDA의 경우 thread)가 있는 워크가 입력되었을 때 GPU 4개로 각각 250개의 워크 아이템이 분배되어 실행되면 이론적으로는 전체 계산시간이 분배되기 전보다 4배 정도 실행 시간이 단축될 수 있다. 조금 더 정확히는, 이중 병렬 프로그래밍 모델에서 워크 그룹(work group; CUDA의 경우 스레드 블록) 사이에 동기화가 불가능하고 실행 중에 atomic operation 등의 특수한 경우를 제외하고는 일관성이 보장될 필요가 없다. 다시 말해서, 서로 다른 워크 그룹(스레드 블록)에 속한 워크 아이템(스레드)들이 서로 다른 디바이스에 할당되어도 할당된 디바이스에서 문제없이 실행될 수 있다. 이에 따라, 워크 아이템을 분배하는 것은 워크 그룹 단위로 이루어질 수 있다.
- [5] 각 작업을 복수의 가속기의 각각에 분배하는 것과 함께 각 작업을 실행하기 위한 데이터를 복수의 가속기의 각각에 분배하는 것이 요구된다. 분배된 작업을 실행하기 위해서는 그 작업이 접근하는 데이터가 해당 가속기에 있어야 한다. 일반적으로 컴파일러 분석 등을 통해서 분배된 각 작업이 접근하는 데이터 영역을 분석하여 그 작업이 실행되기 전에 그 작업이 실행될 가속기로 데이터를

복사하도록 한다. 가장 쉽게는 모든 데이터를 각 계산 디바이스(가속기)에 복사하고 작업을 실행한 후에 결과를 모으는 방식도 있을 수 있다. 이 경우는 데이터를 복사하는 오버헤드가 클 수 있고, 결과를 취합할 때도 추가적인 오버헤드가 발생하므로 작업을 나누어서 얻는 이점보다 데이터를 복사하는 오버헤드가 커서 성능 향상이 미미하거나 없을 수도 있다.

- [6] 종래의 가속기 분배 기술은 데이터 분배를 위해 불필요한 호스트-디바이스 통신이 많이 요구될 수 있다. 메모리 접근이 완벽하게 분석되지 못하는 커널의 경우는 모든 데이터를 중복하여 복사해야 하기 때문에 불필요한 통신이 많이 발생될 수 있다. 이러한 과도한 통신은 성능저하를 초래할 수 있다. 또한, 종래 기술에서의 계산 디바이스의 디바이스 메모리는 비효율적으로 사용된다는 단점도 있다. 이는 특정 계산 디바이스에 할당된 워크 아이템들이 전혀 사용하지 않을 불필요한 데이터들도 모두 복사되어 계산 디바이스의 디바이스 메모리를 차지하고 있기 때문이다. 이에 따라 해당 시스템에서 실행할 수 있는 애플리케이션의 최대 데이터 크기는 계산 디바이스 하나의 디바이스 메모리 크기를 넘지 못하게 되고 이는 매우 큰 프로그램을 실행할 수 없다는 것을 의미한다. 일반적으로 이종 클러스터를 사용할 정도로 대규모의 계산이 필요한 애플리케이션의 경우 그 계산에 필요한 메모리가 매우 클 수 있다. 즉, 데이터가 클수록 계산량이 많아질 수 있다. 이러한 종래 기술 환경 하에서, 이렇게 큰 용량의 데이터를 요구하는 프로그램은 실행될 수 없으므로 이종클러스터의 활용도를 저하될 수 있다. 이에 따라, 이러한 불필요한 데이터 통신과 디바이스 메모리의 불필요한 점유가 디바이스 개수를 늘리는 만큼 계산 성능 개선을 가지고 오지 못하는 원인이 될 수 있다. 즉, 이종 시스템의 성능 확장성(scalability)을 저해하는 요소가 된다.

[7]

발명의 상세한 설명

기술적 과제

- [8] 본 개시는 상기와 같은 목적을 해결하기 위한 이종클러스터 시스템에서 프로그램을 실행시키는 방법 및 컴퓨터 프로그램을 제공한다.
- [9] 이종 클러스터 시스템에서 프로그램 또는 애플리케이션을 실행할 때 단일한 가상의 계산 디바이스를 프로그래머에게 제공하는데 있다. 나아가, 단일한 가상 디바이스를 대상으로 작성된 프로그램은 이종 클러스터상의 여러 계산 디바이스 또는 가속기로 분배되어 실행될 수 있다. 이때 분배되는 작업을 위해 불러오는 데이터 중에서, 분배된 작업에 반드시 요구되는 데이터만이 계산 디바이스 또는 가속기에 복사될 수 있다.
- [10] 이종 클러스터 시스템에서 가상의 단일 디바이스와 주소 공간을 공유하는 가상의 단일 디바이스 메모리를 프로그래머에게 제공하여 프로그래밍의 복잡성을 없애거나 최소화할 수 있다. 또한, 프로그램과 연관된 워크 그룹 또는

워크 아이템의 메모리 접근 패턴이 분석되어 계산 작업 분배 및 데이터 분배의 효율성이 향상될 수 있다.

- [11] 복수의 가속기가 장착된 여러 노드들이 있는 이중 클러스터 시스템에서 애플리케이션을 작성하는 복잡도가 낮아질 수 있다. 즉, 프로그래머는 복수의 노드, 복수의 가속기에 대한 고려없이 단일 계산 디바이스, 단일한 메모리에 대한 프로그램을 작성하거나 실행할 수 있기 때문에, 프로그램의 생산성이 매우 높아지고 유지보수도 매우 쉬워질 수 있다.

[12]

과제 해결 수단

- [13] 본 개시는 방법, 장치, 시스템, 컴퓨터 프로그램 또는 명령어들을 저장하는 컴퓨터 판독가능 저장 매체를 포함한 다양한 방식으로 구현될 수 있다.

- [14] 본 개시의 일 실시예에 따른 이중클러스터 시스템에서 프로그램을 실행시키는 방법은, 이중클러스터 시스템에 포함된 복수의 계산 디바이스의 각각에서, 이중클러스터 시스템의 클라이언트 노드로부터 할당된 프로그램과 연관된 복수의 워크 그룹을 수신하는 단계 및 복수의 계산 디바이스의 각각에서 할당된 복수의 워크 그룹을 처리하는 단계를 포함하고, 복수의 워크 그룹에 의해 접근되는 페이지는 이중클러스터 시스템 상에서 단일 메모리 주소 공간을 제공하도록 구성된 가상 메모리에 포함된 가상 페이지를 통해 관리된다.

- [15] 일 실시예에 따르면, 할당된 복수의 워크 그룹을 처리하는 단계는, 상기 복수의 계산 디바이스 중 제1 계산 디바이스에서, 제1 계산 디바이스에 할당된 복수의 워크 그룹을 실행하는 중에 페이지 폴트가 발생하는 경우, 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지에 대한 정보를 불러오는 단계 및 제1 계산 디바이스에서, 최근 페이지에 포함된 데이터를 수신하는 단계를 포함한다.

- [16] 일 실시예에 따르면, 복수의 계산 디바이스는 이중클러스터 시스템에 포함된 복수의 서버 노드의 각각에 포함된다.

- [17] 일 실시예에 따르면, 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지에 대한 정보를 불러오는 단계는, 페이지 폴트가 발생한 가상 페이지의 최근 페이지 정보 요청을 오너 노드에 송신하는 단계 및 오너 노드로부터 가상 페이지의 최근 페이지에 대한 정보를 수신하는 단계, 수신된 최근 페이지에 대한 정보가, 제1 계산 디바이스를 포함한 제1 노드 내에서, 제1 계산 디바이스와 상이한 계산 디바이스의 메모리에 포함된 페이지를 가리키는지 여부를 판정하는 단계를 포함한다.

- [18] 일 실시예에 따르면, 제1 계산 디바이스에서, 최근 페이지에 포함된 데이터를 수신하는 단계는, 오너 노드로부터 수신된 최근 페이지에 대한 정보가 제1 노드 내의 상이한 계산 디바이스의 메모리에 포함된 페이지를 가리키는 경우, 제1 노드에서 제공되는 페이지 폴트 핸들러를 실행하는 단계를 포함한다.

- [19] 일 실시예에 따르면, 제1 계산 디바이스에서 최근 페이지에 포함된 데이터를

수신하는 단계는, 오너 노드로부터 수신된 최근 페이지에 대한 정보가 제1 노드 내의 상이한 계산 디바이스의 메모리에 포함된 페이지가 아닌 경우, 가상 페이지와 연관된 최근 페이지에 포함된 데이터가 클라이언트 노드에 있는지 문의하는 단계 및 가상 페이지와 연관된 최근 페이지에 포함된 데이터가 클라이언트 노드에 있는 경우, 클라이언트 노드로부터 가상 페이지와 연관된 최근 페이지에 포함된 데이터를 수신하는 단계를 포함한다.

[20] 일 실시예에 따르면, 제1 계산 디바이스에서 최근 페이지에 포함된 데이터를 수신하는 단계는, 가상 페이지와 연관된 최근 페이지에 포함된 데이터가 클라이언트 노드에 포함되지 않은 경우, 최근 페이지에 대한 정보가 나타내는 제2 계산 디바이스로부터 가상 페이지와 연관된 최근 페이지에 포함된 데이터를 수신하는 단계를 포함한다.

[21] 일 실시예에 따르면, 오너 노드의 페이지 관리 테이블의 업데이트를 수행하기 위하여, 제1 계산 디바이스에 의해, 수신된 최근 페이지에 포함된 데이터를 처리한, 제1 계산 디바이스의 페이지에 대한 정보를 오너 노드로 송신하는 단계를 더 포함한다.

[22] 일 실시예에 따르면, 복수의 워크 그룹과 복수의 워크 그룹이 접근하는 복수의 가상 페이지 사이의 관계를 나타내는 테이블을 기초로, 프로그램과 연관된 복수의 워크 그룹은 상기 복수의 계산 디바이스의 각각에 할당된다.

[23] 일 실시예에 따르면, 프로그램의 컴파일 동안의 static 분석 결과, 프로그램의 실행 시의 Runtime 분석 결과 또는 프로그램의 프로파일(profile) 분석 결과 중 적어도 하나를 기초로, 프로그램과 연관된 복수의 워크 그룹은 복수의 계산 디바이스의 각각에 할당된다.

[24] 일 실시예에 따르면, 복수의 워크 그룹의 메모리 접근 패턴을 분석한 결과를 기초로, 복수의 워크 그룹과 연관된 데이터는, 복수의 워크 그룹이 복수의 계산 디바이스에 의해 처리되기 전에, 복수의 워크 그룹과 연관된 데이터는 상기 복수의 계산 디바이스의 각각에 미리 프리패칭(pre-fetching)될 수 있다. 본 개시의 일 실시예에 따른 상술한 이중클러스터 시스템에서 프로그램을 실행시키는 방법을 컴퓨터에서 실행하기 위해 컴퓨터 판독 가능한 기록 매체에 저장된 컴퓨터 프로그램이 제공된다.

[25]

발명의 효과

[26] 본 개시의 일부 실시예에 따르면, 런타임 시스템에서 가상의 단일 디바이스와 가상의 단일 디바이스 메모리를 제공함으로써, 가상의 단일 디바이스는 이중 클러스터 시스템에 탑재된 복수의 실제 디바이스에 나누어 실행될 수 있다. 이에 따라, 복수의 가속기가 장착된 여러 노드들이 있는 이중 클러스터 시스템에서 프로그램 또는 애플리케이션을 작성하는 복잡도가 매우 낮아지게 될 수 있다. 또한, 프로그래머 측에서는 복수의 노드, 복수의 가속기에 대한 고려없이 단일

계산 디바이스, 단일한 메모리에 대한 프로그래밍만을 작성하면 되기 때문에 프로그램의 생산성이 매우 높아지고 유지보수도 매우 쉬워질 수 있다. 또한, 계산 디바이스에서 접근하는 메모리만이 사용되기 때문에 계산 디바이스의 메모리를 넘어서는 메모리가 필요한 프로그램이 실행될 수 있다.

[27] 본 개시의 일부 실시예에 따르면, 기본적으로 나누어진 계산 작업이 이중 클러스터의 각 계산 디바이스에서 실행될 때 해당 계산 디바이스에서 발생하는 페이지 폴트를 이용하여 그 실행에 필요한 페이지만이 복사됨으로써 데이터 통신과 실제 계산 디바이스의 메모리 사용이 최소화될 수 있다.

[28] 본 개시의 일부 실시예에 따르면, 프로그램과 연관된 복수의 워크 그룹의 메모리 접근 패턴을 분석하여 분석된 결과를 기초로 복수의 워크 그룹 및/또는 이와 연관된 데이터를 복수의 계산 디바이스에 분배함으로써, 노드 사이의 데이터 통신 및/또는 호스트와 계산 디바이스 사이의 데이터 통신이 감소하여 전체적인 통신량이 최소화될 수 있으며, 이에 따라 발생된 통신 오버헤드가 최소화될 수 있다. 즉, 데이터의 분배 성능이 향상될 수 있다.

[29]

도면의 간단한 설명

[30] 본 개시의 실시예들은, 이하 설명하는 첨부 도면들을 참조하여 설명될 것이며, 여기서 유사한 참조 번호는 유사한 요소들을 나타내지만, 이에 한정되지는 않는다.

[31] 도 1은 본 개시의 일 실시예에 따른 이중클러스터 시스템에서 실행되는 프로그램을 실행시키기 위하여, 복수의 계산 디바이스를 포함하는 복수의 서버 노드를 가상 시스템을 이용해 프로그램을 실행 가능하도록 연결된 구성을 나타내는 개요도이다.

[32] 도 2는 본 개시의 일 실시예에 따른 클라이언트 노드 및 서버 노드의 내부 구성을 나타내는 블록도이다.

[33] 도 3은 본 개시의 일 실시예에 따른 이중클러스터 시스템에서 실행되는 프로그램을 실행시키기 위한 페이지 폴트 처리 방법을 나타내는 흐름도이다.

[34] 도 4는 본 개시의 일 실시예에 따른 복수의 워크 그룹을 복수의 계산 디바이스 각각에 할당하는 예시를 나타내는 도면이다.

[35] 도 5는 본 개시의 일 실시예에 따른 복수의 워크 그룹과 복수의 워크 그룹이 접근하는 복수의 가상 페이지 사이의 관계를 나타내는 테이블의 예시를 나타내는 도면이다.

[36] 도 6은 본 개시의 일 실시예에 따른 이중클러스터 시스템에서 실행되는 프로그램을 실행시키기 위해, 페이지 폴트를 처리하는 방법을 나타내는 흐름도이다.

[37] 도 7은 본 개시의 일 실시예에 따른 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지가 페이지 폴트가 발생한 페이지와 같은 노드에 있을 경우,

페이지 폴트를 처리하는 예시를 나타내는 도면이다.

[38] 도 8은 본 개시의 일 실시예에 따른 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지가 클라이언트 노드의 가상 메모리에 존재할 경우, 페이지 폴트를 처리하는 예시를 나타내는 도면이다.

[39] 도 9는 본 개시의 일 실시예에 따른 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지가 페이지 폴트가 발생한 페이지와 다른 노드에 있을 경우, 페이지 폴트를 처리하는 예시를 나타내는 도면이다.

[40]

발명의 실시를 위한 형태

[41] 이하, 본 개시의 실시를 위한 구체적인 내용을 첨부된 도면을 참조하여 상세히 설명한다. 다만, 이하의 설명에서는 본 개시의 요지를 불필요하게 흐릴 우려가 있는 경우, 널리 알려진 기능이나 구성에 관한 구체적 설명은 생략하기로 한다.

[42] 첨부된 도면에서, 동일하거나 대응하는 구성요소에는 동일한 참조부호가 부여되어 있다. 또한, 이하의 실시예들의 설명에 있어서, 동일하거나 대응되는 구성요소를 중복하여 기술하는 것이 생략될 수 있다. 그러나 구성요소에 관한 기술이 생략되어도, 그러한 구성요소가 어떤 실시예에 포함되지 않는 것으로 의도되지는 않는다.

[43] 개시된 실시예의 이점 및 특징, 그리고 그것들을 달성하는 방법은 첨부되는 도면과 함께 후술되어 있는 실시예들을 참조하면 명확해질 것이다. 그러나 본 개시는 이하에서 개시되는 실시예들에 한정되는 것이 아니라 서로 다른 다양한 형태로 구현될 수 있으며, 단지 본 실시예들은 본 개시가 완전하도록 하고, 본 개시가 통상의 기술자에게 발명의 범주를 완전하게 알려주기 위해 제공되는 것일 뿐이다.

[44] 본 명세서에서 사용되는 용어에 대해 간략히 설명하고, 개시된 실시예에 대해 구체적으로 설명하기로 한다. 본 명세서에서 사용되는 용어는 본 개시에서의 기능을 고려하면서 가능한 현재 널리 사용되는 일반적인 용어들을 선택하였으나, 이는 관련 분야에 종사하는 기술자의 의도 또는 관례, 새로운 기술의 출현 등에 따라 달라질 수 있다. 또한, 특정한 경우는 출원인이 임의로 선정한 용어도 있으며, 이 경우 해당되는 발명의 설명 부분에서 상세히 그 의미를 기재할 것이다. 따라서 본 개시에서 사용되는 용어는 단순한 용어의 명칭이 아닌, 그 용어가 가지는 의미와 본 개시의 전반에 걸친 내용을 토대로 정의되어야 한다.

[45] 본 명세서에서의 단수의 표현은 문맥상 명백하게 단수인 것으로 특정하지 않는 한, 복수의 표현을 포함한다. 또한, 복수의 표현은 문맥상 명백하게 복수인 것으로 특정하지 않는 한, 단수의 표현을 포함한다. 명세서 전체에서 어떤 부분이 어떤 구성요소를 "포함"한다고 할 때, 이는 특별히 반대되는 기재가 없는 한 다른 구성요소를 제외하는 것이 아니라 다른 구성요소를 더 포함할 수 있음을

의미한다.

- [46] 본 개시에서, '프로그램'은 이종시스템을 대상으로 하는 프로그램을 지칭할 수 있다. 또한, 프로그램은 프로세서에 의해 실행되는 클라이언트 프로그램 및 계산 디바이스에 의해 실행되는 커널(kernel)을 포함할 수 있다. 클라이언트 프로그램 및 커널은 OpenCL과 같이 완전히 분리되어 있을 수도 있고, CUDA, OpenMP 4.0, OpenACC 등과 같이 커널이 클라이언트 프로그램 중간에 포함(embed)되어 있을 수도 있다. 본 개시에서, 프로그램은 위의 클라이언트 프로그램 및 커널을 포함할 수 있다.
- [47] 본 개시에서, '워크 그룹(work-group)'은, 워크 아이템들(work-item)의 집합을 지칭할 수 있다. 워크 아이템은 클라이언트 프로그램이 계산 디바이스에 커널 실행 명령을 보내면, 계산 디바이스에서는 하나 이상의 커널 인스턴스(kernel instance)가 실행된다. 여기서, 커널 인스턴스는 커널을 실행하는 스레드를 지칭할 수 있다. OpenCL에서는 각각의 커널 인스턴스를 워크 아이템이라고 지칭할 수 있다. CUDA에서는 각각의 커널 인스턴스를 스레드(thread)라고 지칭한다. 또한, CUDA에서는 워크 그룹을 스레드블록(thread-block)이라고 지칭할 수 있다. 또한, 워크 그룹은 계산 디바이스에 포함된 복수의 계산 유닛들로 나뉘어져 실행될 수 있다. 또한, 워크 그룹에 속한 워크 아이템들은 다시, 계산 유닛에 포함된 복수의 PE(Processing Element)로 나누어져 실행될 수 있다. 즉, 워크 그룹은 계산 디바이스에 속한 많은 수의 PE에서 동시에 실행될 수 있다.
- [48] 본 개시에서, '페이지'는, 메모리 영역을 특정한 크기로 나뉘어서 관리하는 유닛을 지칭할 수 있다, 예를 들어, 일반적인 운영체제에서는 4K byte 단위로 관리가 된다. 이에 따르면, 1024 KB 메모리 영역을 할당하면 이는 256개의 페이지로 나뉘어져 관리가 될 수 있다.
- [49] 본 개시에서, '오너 노드'는, 가상의 메모리에 포함된 복수의 가상 페이지 중 적어도 일부 페이지를 관리하는 노드를 지칭할 수 있다. 예를 들어, 오너 노드는 오너 노드에 의해 관리되는 가상 페이지와 그러한 가상 페이지에 대한 최근 페이지(예: 계산 디바이스에 포함된 메모리 내의 실제 페이지)에 대한 정보를 저장하고 있으며, 최근 페이지에 대한 정보가 수정될 때 마다, 그러한 정보를 업데이트할 수 있다. 이러한 정보는 테이블의 형태로 오너 노드에 저장될 수 있다.
- [50] 본 개시에서, '이종시스템(Heterogeneous System)'은, 통상적으로 범용 계산을 위한 CPU(예를 들어, x86 CPU)와 고속의 병렬 처리에 특화된 계산 디바이스 또는 가속기를 함께 장착한 시스템을 포함할 수 있다. 예를 들어, 계산 디바이스는 GPU(Graphics Processing Unit), Intel Xeon Phi, FPGA, DSP 등을 포함할 수 있다. 병렬로 수행이 가능하고 계산량이 많은 코드 영역을 병렬 계산에 특화되어 있는 프로그램은 복수의 계산 디바이스에 오프로딩(offloading)되어 처리되면 프로그램의 성능과 전력효율을 크게 향상될

수 있다. 예를 들어, 이종시스템에서 이러한 오프로딩을 위한 프로그래밍 모델로는 CUDA와 OpenCL이 널리 사용되고 있다.

- [51] 본 개시에서, '이종클러스터 시스템'은, 네트워크를 통해 연결된 계산 디바이스 또는 가속기(예를 들어, GPU 또는 FPGA 등의 계산 디바이스)가 장착된 복수의 노드(예를 들어, 컴퓨터)를 포함할 수 있다. 이종클러스터 시스템에 사용되는 네트워크는 기가비트 이더넷(Gigabit Ethernet), 인피니밴드(Infiniband) 등을 포함할 수 있다. 이종클러스터 시스템에는 복수의 노드가 있고, 각 노드는 1개 이상의 가속기를 포함할 수 있다. 이러한 이종 클러스터 환경에서 응용프로그램의 특정 부분을 오프로딩 하여 실행하기 위해서는 노드간 통신이 필요하게 되고 노드 간 통신을 위해서 RDMA(Remote Direct Memory Access), MPI(Message Passing Interface), 소켓 통신 라이브러리 등의 통신 라이브러리가 사용될 수 있다.
- [52] 또한, 본 개시에서의 '이종클러스터 시스템'에서의 프로그래밍은 일반적으로 마스터-슬레이브(Master-Slave) 형태의 모델이 사용될 수 있다. 마스터-슬레이브 형태의 모델은 클라이언트-서버 모델로 이해할 수도 있다. 마스터-슬레이브 모델은 하나의 호스트 노드(마스터; 클라이언트)와 1개 이상의 계산 노드(슬레이브; 서버)로 구성된 환경을 가정한다. 호스트 노드에서는 호스트 프로그램이 동작하며 일반적으로 커널(Kernel)이라고 불리는 병렬 프로그램의 인스턴스(instance)를 각 계산 노드로 분산하여 실행을 하고, 메모리를 관리할 수 있다. 계산 노드는 호스트 노드로부터 받은 데이터와 커널 인스턴스를 자기가 가지고 있는 가속기(컴퓨터 디바이스)를 활용하여 실행할 수 있다. 이종클러스터 시스템에서의 프로그래밍은 오프로딩 하여 수행할 데이터를 해당 노드의 메인 메모리 혹은 이차 스토리지에 소유하고 있어야 한다. 만약 해당하는 데이터가 존재하지 않으면 노드간 통신을 통해서 해당 데이터를 그 데이터가 필요한 노드로 옮기는 작업이 우선되어야 한다. 클라이언트 노드는 소유한 데이터를 각 계산 노드로 나눠 주기 위해서 MPI, RDMA, 소켓 통신과 같은 노드간 통신 라이브러리를 활용하고, 계산 노드에는 오프로딩 된 계산 작업을 처리하기 위해 CUDA와 OpenCL같은 프로그래밍 모델을 사용할 수 있다.
- [53] 본 개시에서, 계산 디바이스에 의해 처리, 수신 또는 송신되는 것은 계산 디바이스가 포함된 노드의 CPU에서 처리, 수신 또는 송신되는 것을 포함할 수 있다.
- [54] 본 개시에서, '페이지 폴트'는, 프로세서 또는 계산 디바이스가 운영체제를 통해 특정 페이지를 메모리에 요청하고, 물리 메모리 상에 해당 페이지가 존재하면, 해당 페이지에 접근할 수 있다. 그러나, 해당 페이지가 물리 메모리 상에 존재하지 않는 경우, 가상 메모리 공간에 해당 페이지를 요청해야 하는데, 이를 페이지 폴트라 지칭할 수 있다.
- [55] 본 개시에서, '페이지 폴트 핸들러'는, 페이지 폴트가 발생하면, 가상 메모리로부터 물리 메모리 상에 해당 페이지를 로드할 수 있다. 또한, 페이지를 물리 메모리 상에 로드할 때, 물리 메모리의 공간이 부족하다면, 물리 메모리

상에 있는 페이지를 가상 메모리로 로드하고, 가상 메모리 상에 있는 접근하고자 하는 페이지를 물리 메모리 상에 로드할 수 있다. 프로세서의 메인 메모리에서 페이지 폴트가 발생하는 경우, 페이지 폴트 핸들러는 운영체제에서 제공하는 페이지 폴트 메커니즘을 의미할 수 있다. 또한, 계산 디바이스에서 페이지 폴트가 발생하는 경우, 페이지 폴트 핸들러는 ISR(Interrupt Service Routine)을 통해서 처리하는 것을 의미할 수 있다.

[56] 본 개시에서, 'n'은 1이상의 자연수를 지칭할 수 있으며, 'm'은 1 이상의 자연수를 지칭할 수 있으며, 'n'과 'm'은 동일한 수일 수 있고, 상이한 수일 수 있다.

[57] 도 1은 본 개시의 일 실시예에 따른 이중클러스터 시스템(130)에서 실행되는 프로그램을 실행시키기 위하여, 복수의 계산 디바이스(170_1, ..., 170_m)를 포함하는 복수의 서버 노드(150_1, ..., 150_n)를 가상 시스템(100)을 이용해 프로그램을 실행 가능하도록 연결된 구성을 나타내는 개요도이다. 여기서, 서버 노드(150_1, ..., 150_n)는 계산 노드 또는 슬레이브 노드를 지칭할 수 있다. 또한, 복수의 계산 디바이스(170_1, ..., 170_m)는 이중클러스터 시스템(130)에 포함된 복수의 서버 노드(150_1, ..., 150_n)의 각각에 포함될 수 있다. 예를 들어, 복수의 계산 디바이스(170_1, ..., 170_m)는 복수의 서버 노드(150_1, ..., 150_n)의 적어도 일부 노드에 포함될 수 있다. 또 다른 예로서, 복수의 서버 노드(150_1, ..., 150_n)의 각각은 복수의 계산 디바이스(170_1, ..., 170_m) 중 적어도 일부를 포함할 수 있다. 또한, 이중클러스터 시스템(130)은 클라이언트 노드(미도시), 복수의 서버 노드(150_1, ..., 150_n) 및 네트워크(140)를 포함할 수 있다. 또한, 클라이언트 노드(미도시) 및 복수의 서버 노드(150_1, ..., 150_n)의 각각은 프로세서(160), 복수의 계산 디바이스(170_1, ..., 170_m), 복수의 계산 디바이스 메모리(175_1, ..., 175_m), Interconnection Bus(180), 통신부(185) 및 메인 메모리(190)를 포함할 수 있다.

[58] 가상 시스템(100)은, 이중클러스터 시스템을 이용하는 프로그램을 실행시키도록 구성될 수 있다. 가상 시스템(100)은, 클라이언트 노드(미도시)에 의해 제어될 수 있다. 또한, 가상 시스템(100)은 복수의 서버 노드(150_1, ..., 150_n)와 통신 가능하도록 구성되고, 복수의 서버 노드(150_1, ..., 150_n)에 포함된 복수의 계산 디바이스(170_1, ..., 170_m)의 동작을 제어하도록 구성될 수 있다. 예를 들어, 가상 시스템(100)은 이중클러스터 시스템(130)에 포함된 전체 노드를 대상으로 전체 작업을 분배하고, 각각의 노드에서 복수의 계산 디바이스로 워크로드를 분배하여 실행할 수 있다. 그리고, 모든 계산 작업이 완료되면, 분배된 결과를 다시 클라이언트 노드에서 취합할 수 있다.

[59] 클라이언트 프로세서(110)는, 클라이언트 노드(미도시)에 포함될 수 있다. 또한, 클라이언트 프로세서(110)는, 이중클러스터 시스템(130)을 제어하기 위한, 가상 계산 디바이스(115) 및 가상 메모리(120)를 제어할 수 있다. 클라이언트 프로세서(110)는, 예를 들어, CPU(Central Processing Unit, 중앙 처리 장치)와 같은 연산 처리를 위한 범용 프로세서로 구성될 수 있으며, 클라이언트

- 프로세서(110)는 가상 계산 디바이스(115)와 연결되어 복수의 계산 디바이스(170_1, ..., 170_m)의 동작을 제어할 수 있다. 또한, 클라이언트 프로세서(110)는 가상 메모리(120) 및/또는 메인 메모리(190)와 연결될 수 있다. 예를 들어, 클라이언트 프로세서(110)는 PCI-E(Peripheral component interconnect-Express) 버스를 통해 복수의 계산 디바이스(170_1, ..., 170_m) 및/또는 메인 메모리(190)와 서로 연결될 수 있으며 복수의 계산 디바이스(170_1, ..., 170_m) 및/또는 메인 메모리(190)의 제어를 위한 데이터를 송수신할 수 있다.
- [60] 가상 계산 디바이스(115)는, 클라이언트 프로세서(110)에 의해 제어될 수 있다. 또한, 가상 계산 디바이스(115)는 단일 계산 디바이스를 타겟으로 작성된 애플리케이션을 복수의 계산 디바이스(170_1, ..., 170_m)에 나누어 실행할 수 있다. 가상 계산 디바이스(115)를 구현하기 위하여, 가상 GPU(예를 들어, NVIDIA의 vGPU 등)에서 제공하는 기술이 이용될 수 있다.
- [61] 가상 메모리(120)는, 본 발명의 구현을 위해 이종클러스터 시스템(130) 상에서 단일한 메모리 주소공간을 제공하기 위한 통합된 가상 메모리를 지칭할 수 있다. 본 개시에서는 이를 이기종 가속기 클러스터를 위한 통합 메모리(UMHC, Unified Memory for Heterogeneous-Accelerator Clusters)라고 지칭할 수 있다. 일 실시예에 따르면, 가상 메모리(120)는 이종 클러스터 내의 모든 메인 메모리와 복수의 계산 디바이스의 메모리가 단일 메모리 주소 공간 상에서 할당, 관리, 및/또는 동작되도록 구성될 수 있다. 예를 들어, 가상 메모리(120)는 CUDA의 Unified memory, OpenCL의 OpenCL SVM 등을 이용해 구현될 수 있다. 여기서, 가상 메모리(120)는 가상의 디바이스 메모리를 지칭할 수 있다. 이러한 가상의 디바이스 메모리는 전체 이종 클러스터 상에서 단일한 주소 공간을 공유하며, 운영체제에서 제공하는 페이지 폴트(page fault) mechanism 및/또는 계산 디바이스 또는 가속기에서 제공하는 페이지 폴트를 처리하기 위한 ISR(Interrupt Service Routine)을 이용하여 구현될 수 있다.
- [62] RHAC(Runtime system for Heterogeneous Accelerator Cluster)(125)는, 이기종 가속기 클러스터를 위한 런타임 시스템을 지칭할 수 있다. 일 실시예에 따르면, RHAC(125)는 복수의 계산 디바이스(170_1, ..., 170_m)를 이용하는 이종클러스터 시스템(130)에서 단일한 계산 디바이스를 이용하는 것처럼 프로그램을 실행시키도록 구성될 수 있다. 예를 들어, RHAC(125)는 복수의 계산 디바이스(170_1, ..., 170_m)를 가상 계산 디바이스(115)로 관리하고, 복수의 서버 노드(150_1, ..., 150_n) 각각이 포함하는 메인 메모리(190) 및 계산 디바이스 메모리(175_1, ... 175_m)를 단일 주소를 갖는 가상 메모리(120)로 관리하기 위한 런타임 시스템을 지칭할 수 있다. 또한, RHAC(125)는, 단일 주소 공간을 제공하기 위해, 페이지 폴트 핸들링을 실행할 수 있으며, 도 7 내지 도 9를 통해 보다 상세히 후술한다.
- [63] 이종클러스터 시스템(130)은, 네트워크(140)를 통해 연결된 클라이언트 노드(미도시) 및 복수의 서버 노드(150_1, ..., 150_n)를 포함할 수 있다. 또한,

이종클러스터 시스템(130)을 통해, 클라이언트 노드(미도시)는 이종클러스터 시스템(130)을 하나의 컴퓨터처럼 이용할 수 있다.

- [64] 네트워크(140)는, 이종클러스터 시스템(130)에 포함됨 클라이언트 노드(미도시) 및 복수의 서버 노드(150_1, ..., 150_n) 사이의 통신이 가능하도록 구성될 수 있다. 네트워크(140)는 설치 환경에 따라, 예를 들어, 이더넷(Ethernet), 유선 홈 네트워크(Power Line Communication), 전화선 통신 장치 및 RS-serial 통신 등의 유선 네트워크, 이동통신망, WLAN(Wireless LAN), Wi-Fi, Bluetooth 및 ZigBee 등과 같은 무선 네트워크 또는 그 조합으로 구성될 수 있다. 다시 말해, 통신 방식은 제한되지 않으며, 네트워크(140)가 포함할 수 있는 통신망(일례로, 이동통신망, 유선 인터넷, 무선 인터넷, 방송망, 위성망 등)을 활용하는 통신 방식뿐만 아니라 클라이언트 노드(미도시) 및 복수의 서버 노드(150_1, ..., 150_n) 사이의 근거리 무선 통신 역시 포함될 수 있다. 예를 들어, 네트워크(140)는 PAN(personal area network), LAN(local area network), CAN(campus area network), MAN(metropolitan area network), WAN(wide area network), BBN(broadband network), 인터넷 등의 네트워크 중 하나 이상의 임의의 네트워크를 포함할 수 있다. 또한, 네트워크(140)는 버스 네트워크, 스타 네트워크, 링 네트워크, 메쉬 네트워크, 스타-버스 네트워크, 트리 또는 계층적(hierarchical) 네트워크 등을 포함하는 네트워크 토폴로지 중 임의의 하나 이상을 포함할 수 있으나, 이에 제한되지 않는다.

- [65] 복수의 서버 노드(150_1, ..., 150_n)는, 이종클러스터 시스템(130) 상에서 정보처리, 계산, 연산 및/또는 통신을 수행하도록 구성된 컴퓨팅 장치를 포함할 수 있다. 복수의 서버 노드(150_1, ..., 150_n)는 컴퓨터 또는 원격처리 장치와 같은 단말의 형태로 구성될 수 있다. 또한, 각 복수의 서버 노드(150_1, ..., 150_n)는 독립적으로 정보처리 등을 수행할 수 있으나, 병렬 프로그래밍을 통하여 다른 복수의 서버 노드(150_1, ..., 150_n)들과 협력하면서 정보처리 등을 수행할 수 있다. 각 복수의 서버 노드(150_1, ..., 150_n)는 네트워크(140)를 통해 프로그램 동작을 위한 통신 및 페이지 폴트를 처리하기 위한 통신을 실행할 수 있다. 이러한 복수의 서버 노드(150_1, ..., 150_n)는 데이터의 송신원, 수신처 또는 중계점 중 어느 하나에 해당할 수 있다. 또는 이러한 복수의 서버 노드(150_1, ..., 150_n) 중 적어도 일부는 오너 노드로서 수행될 수 있다. 도시된 바와 같이, 복수의 서버 노드(150_1, ..., 150_n)의 각각은 프로세서(160), 계산 디바이스(170_1, ..., 170_m), 계산 디바이스 메모리(175_1, ..., 175_m), Interconnection Bus(180), 메인 메모리(190) 및 통신부(185)를 포함할 수 있다. 또한, 이러한 구성은 클라이언트 노드(미도시)에도 포함될 수 있다.

- [66] 프로세서(160)는, 예를 들어, CPU(Central Processing Unit, 중앙 처리 장치)와 같은 연산 처리를 위한 범용 프로세서로 구성될 수 있으며, 프로세서(160)는 복수의 계산 디바이스(170_1, ..., 170_m)와 연결되어 복수의 계산 디바이스(170_1, ..., 170_m)의 동작을 제어할 수 있다. 또한, 프로세서(160)는

메인 메모리(190)가 연결될 수 있다. 예를 들어, 프로세서(160)는 Interconnection Bus(180)를 통해 복수의 계산 디바이스 메모리(175_1, ..., 175_m) 및/또는 메인 메모리(190)와 서로 연결될 수 있으며 복수의 계산 디바이스 메모리(175_1, ..., 175_m) 및/또는 메인 메모리(190)의 제어를 수행하고, 이를 위한 데이터를 송수신할 수 있다.

- [67] 복수의 계산 디바이스(170_1, ..., 170_m)는, 범용의 CPU와는 달리 특정 패턴의 연산에 특화된 프로세서로 구성될 수 있다. 예를 들어, 복수의 계산 디바이스(170_1, ..., 170_m)의 각각은 GPU, FPGA, DSP, Intel Xeon Phi, TPU, NPU, 멀티코어 CPU 등을 포함할 수 있다. 또한, 복수의 계산 디바이스(170_1, ..., 170_m) 각각에는 메인 메모리(190)와는 별도로 계산 디바이스 메모리(175_1, ..., 175_m)가 연결될 수 있다.
- [68] 복수의 계산 디바이스(170_1, ..., 170_m)는, 각각 하나 이상의 계산 유닛(CU; compute unit)을 포함하고, 각각의 계산 유닛은 다시 하나 이상의 PE(processing element)를 포함할 수 있다.
- [69] 복수의 계산 디바이스(170_1, ..., 170_m) 각각의 모든 PE는 각각의 계산 디바이스 메모리를 공유하며, 각각의 계산 디바이스 메모리에서 데이터를 읽고 쓸 수 있다. 복수의 계산 디바이스(170_1, ..., 170_m)에 포함된 각각의 구성요소들은 실제 물리적으로 구분되는 것일 수도 있고, 물리적으로는 구분되지 않지만 논리적으로(예를 들면, 소프트웨어에 의해서) 구분되는 것일 수도 있다. 특히 메인 메모리(190) 혹은 계산 디바이스 메모리(175_1, ..., 175_m)는 실제 물리적인 메모리 칩 하나를 가리킬 수도 있지만, 복수의 메모리 칩 및 다른 구성요소가 연결되어 단일한 메모리 주소 공간(address space)를 제공할 때, 그 단일한 메모리 주소 공간을 가리킬 수도 있다.
- [70] 복수의 계산 디바이스 메모리(175_1, ..., 175_m)는, 복수의 계산 디바이스(170_1, ..., 170_m)에 연결될 수 있다. 예를 들어, 복수의 계산 디바이스(170_1, ..., 170_m)가 프로세서(160)에 연결된 메인 메모리(190)에 직접 접근은 할 수 없기 때문에, 복수의 계산 디바이스(170_1, ..., 170_m)는 메인 메모리(190)에서 복수의 계산 디바이스 메모리(175_1, ..., 175_m)로 입력 데이터를 복사하고, 복수의 계산 디바이스(170_1, ..., 170_m)에 오프로딩된 또는 분배된 코드를 실행할 수 있다. 이러한 오프로딩 또는 분배된 코드는 클라이언트 노드(미도시)로부터 수신될 수 있으며, 자신의 계산 디바이스에 포함된 계산 디바이스 메모리에 저장될 수 있다. 또한, 오프로딩 또는 분배된 코드를 처리한 이후, 복수의 계산 디바이스(170_1, ..., 170_m)는 코드를 처리한 결과 데이터를 복수의 계산 디바이스 메모리(175_1, ..., 175_m)로부터 프로세서(160)의 메인 메모리(190)로 복사할 수 있다. 복수의 계산 디바이스 메모리(175_1, ..., 175_m)는 Interconnection Bus(180)를 통해 메인 메모리(190)와 연결될 수 있다.
- [71] Interconnection Bus(180)는, 복수의 계산 디바이스 메모리(175_1, ..., 175_m), 통신부(185) 및 메인 메모리(190)와 연결되어 통신을 수행할 수 있다.

- Interconnection Bus(180)는 예를 들어, PCI-E(Peripheral component interconnect-Express) 버스와 같은 통신을 수행하기 위한 장치로 구성될 수 있다.
- [72] 통신부(185)는, Interconnection Bus(180)와 연결되어, 복수의 계산 디바이스 메모리(175_1, ..., 175_m) 및 메인 메모리(190)에 저장된 데이터를 송수신할 수 있다. 통신부(185)는, 클라이언트 노드 및 복수의 서버 노드(150_1, ..., 150_n)의 각각에 포함될 수 있으며, 다른 노드와 데이터 송수신할 수 있다. 통신부(185)는, 예를 들어, 클라이언트 노드 및 복수의 서버 노드(150_1, ..., 150_n)를 인피니밴드나 기가비트 이더넷으로 연결할 수 있다. 또한, 노드간 통신은 RDMA나 MPI등 다양한 통신 라이브러리 및 기술을 이용해 수행될 수 있다.
- [73] 메인 메모리(190)는, 전자 정보를 저장 가능한 임의의 전자 컴포넌트를 포함하도록 넓게 해석되어야 한다. 예를 들어, 메인 메모리(190)는 임의 액세스 메모리(RAM), 판독-전용 메모리(ROM), 비-휘발성 임의 액세스 메모리(NVRAM), 프로그램가능 판독-전용 메모리(PROM), 소거-프로그램가능 판독 전용 메모리(EPROM), 전기적으로 소거가능 PROM(EEPROM), 플래쉬 메모리, 자기 또는 광학 데이터 저장장치, 레지스터들 등과 같은 프로세서-판독가능 매체의 다양한 유형들을 지칭할 수도 있다. 프로세서(160) 및/또는 복수의 계산 디바이스(170_1, ..., 170_m)로부터 정보를 판독하거나 메모리에 정보를 기록할 수 있다면 메인 메모리(190)는 프로세서(160) 및/또는 복수의 계산 디바이스(170_1, ..., 170_m)와 전자 통신 상태에 있다고 불린다. 본 개시에서, 메인 메모리(190)는 프로세서(160) 및/또는 복수의 계산 디바이스(170_1, ..., 170_m)에 의해 실행되는 프로그램과 연관된 임의의 데이터 및/또는 정보를 저장할 수 있다.
- [74] 도 2는 본 개시의 일 실시예에 따른 클라이언트 노드(210) 및 복수의 서버 노드(220_1, 220_2)의 내부 구성을 나타내는 블록도이다. 클라이언트 노드(210) 및 복수의 서버 노드(220_1, 220_2)는 이종클러스터 시스템에 포함될 수 있다. 도 2에서는 서버 노드가 2개 이상으로 도시되어 있으나, 이에 한정되지 않으며, 복수의 서버 노드는 n개(여기서, n은 1 이상의 자연수)일 수 있다. 여기서, 클라이언트 노드(210)는 호스트 노드 또는 마스터 노드를 지칭할 수 있다.
- [75] 또한, 복수의 서버 노드(220_1, 220_2)의 각각은 도 1의 서버 노드(150_1, ..., 150_n)의 각각과 동일할 수 있다. 또는, 도 1의 서버 노드(150_1, ..., 150_n)의 내부 구성은 복수의 서버 노드(220_1, 220_2)의 내부 구성에 포함되거나, 복수의 서버 노드(220_1, 220_2)의 내부 구성은 도 1의 서버 노드(150_1, ..., 150_n)의 내부 구성에 포함될 수 있다. 이에 따라, 도 2에서, 도 1에서 설명된 구성과 동일 또는 유사한 구성에 대한 설명은 반복을 피하기 위하여 생략된다.
- [76] 도시된 바와 같이, 각각의 클라이언트 노드(210) 및 복수의 서버 노드(220_1, 220_2)는 RHAC(225_0, 225_1, 225_2), 가상 메모리(230), 계산 디바이스 런타임(240_0, 240_1, 240_2), 가상 메모리 드라이버(250_0, 250_1, 250_2), 계산 디바이스 드라이버(260_0, 260_1, 260_2), 복수의 계산 디바이스(270_1, ...,

270_m, 271_1, ..., 271_m, 272_1, ..., 272_m) 및 통신부(285_0, 285_1, 285_2)를 포함할 수 있다. 본 개시에서, 복수의 계산 디바이스는 각 노드에서 m 개의 계산 디바이스를 포함한다고 편의상 도시되어 있으나, 이에 한정되지 않으며, 각 노드는 상이한 개수의 계산 디바이스를 포함하도록 구성될 수 있다.

- [77] 일 실시예에 따르면, 계산 디바이스 런타임(240_0, 240_1, 240_2)은, 계산 디바이스 제조사가 제공하는 런타임을 지칭할 수 있다. 예를 들어, 계산 디바이스가 NVIDIA의 GPU일 경우, 계산 디바이스 런타임(240_0, 240_1, 240_2)은 CUDA 런타임을 포함할 수 있다. 또한, RHAC(225_0, 225_1, 225_2)는 계산 디바이스 런타임(240_0, 240_1, 240_2)을 이용하여 구현될 수 있다.
- [78] 가상 메모리 드라이버(250_0, 250_1, 250_2)는, 복수의 계산 디바이스 드라이버(260_0, 260_1, 260_2)를 이용하여 가상 메모리(230)(예를 들어, UMHC(Unified Memory for Heterogeneous-Accelerator Clusters))를 구현할 수 있다.
- [79] 계산 디바이스 드라이버(260_0, 260_1, 260_2)의 각각은 자신의 노드에 포함된 복수의 계산 디바이스를 제어할 수 있다. 예를 들어, 계산 디바이스가 NVIDIA의 GPU일 경우, 계산 디바이스 드라이버(260_0, 260_1, 260_2)는 NVIDIA에 의해 제공되는 NVIDIA GPU 드라이버를 포함할 수 있다.
- [80] 본 개시의 일 실시예에 따르면, 분배된 계산 작업(예를 들어, 프로그램, 커널 프로그램 등)이 실행되는 중에, 해당 노드 혹은 해당 계산 디바이스가 접근하는 메모리 영역에, 계산에 필요한 데이터가 없다면 등록된 페이지 폴트 핸들러가 호출되어 처리될 수도 있다. 일반적으로 프로세서의 메인 메모리에서 발생하는 페이지 폴트는 운영체제의 페이지 폴트 메커니즘을 이용하여 처리되거나, 계산 디바이스에서 발생하는 페이지 폴트는 ISR(Interrupt Service Routine)을 통해서 처리될 수 있다.
- [81] 가상 메모리(230)는 이종클러스터 시스템에 단일 주소 공간의 공유 메모리를 제공할 수 있다. 가상 메모리(230)는 이종클러스터 시스템의 클라이언트 노드(210) 및 복수의 서버 노드(220_1, 220_2)가 모두 동일한 주소로 읽고 쓸 수 있는 메모리이며, 그 데이터는 모든 노드에서 공유될 수 있다. 가상 메모리(120)를 구현을 위해서 각 노드 내에서는 계산 디바이스 제조사가 제공하는 계산 디바이스 드라이버의 통합 메모리(또는 공유 메모리)를 사용하여 구현될 수도 있다. 예를 들어, NVIDIA의 CUDA는 Unified Memory라는 명칭으로 공유 메모리가 제공되고, OpenCL은 OpenCL SVM이라는 명칭으로 해당 공유 메모리의 기능이 제공된다. 다만, 이러한 종래의 가상 메모리는 하나의 노드 내에서만 단일한 주소 공간을 가지고 공유되는 메모리이며, 이종클러스터 시스템 상의 복수의 서버 노드 사이에는 가상 메모리를 제공하지 않을 수 있다. 본 개시의 가상 메모리(230)는 이종 클러스터 시스템 상의 복수의 서버 노드 사이에서 제공되는 가상 메모리를 지칭할 수 있다. 예를 들어, 가상 메모리(230)는 이종 클러스터 시스템에 포함된 임의의 메모리를 단일의 주소 공간 상에 할당, 관리 및/또는 동작되도록 구성될 수 있다.

- [82] 이러한 가상 메모리(230)를 제공하기 위해서, 이종클러스터 시스템 내의 계산 디바이스에서 접근하는 메모리의 페이지가 계산 디바이스 메모리 상에 없을 때, 인터럽트를 통해 페이지 폴트를 처리하는 방식 또는 이와 유사한 방식으로 페이지 폴트 메커니즘이 구현될 수 있다. 예를 들어, 페이지 폴트가 발생하면, 페이지 폴트가 발생한 페이지가 있는 곳 또는 위치(예를 들어, 메인 메모리 또는 계산 디바이스 메모리)에서 페이지를 복사할 수 있고, 페이지에 저장된 데이터를 액세스할 수 있다. 일반적으로 페이지를 가장 최근에 접근한 디바이스에게 최근 페이지를 이전(migration)하는 방식으로 구현될 수 있다. 또 다른 예로서, 시스템 상에 동일한 페이지가 동시에 여러 별 존재할 수도 있으며, 이에 제한되지 않는다. 동일한 페이지가 여러 별 시스템 상에 존재하는 경우는 읽기 전용(read-only) 메모리의 경우일 수 있다. 이와 달리, 메모리 일관성을 위해서 쓰기(write)가 진행된 페이지의 경우, 동일한 복수 개의 페이지가 존재할 수 없다.
- [83] 본 개시의 일 실시예에 따르면, 이종클러스터 시스템(130)을 대상으로 하는 프로그램은 크게 클라이언트 노드의 프로세서에서 실행되는 클라이언트 프로그램(또는, 호스트 프로그램)과 계산 디바이스에서 실행되는 하나 이상의 커널(kernel)로 구성될 수 있다. 클라이언트 프로그램과 커널은 완전히 분리되어 있을 수도 있고(예를 들어, OpenCL의 경우), 커널이 클라이언트 프로그램의 중간에 포함(embed)되어 있을 수도 있다(예를 들어, CUDA, OpenMP 4.0, OpenACC의 경우).
- [84] 본 개시의 일 실시예에 따르면, 클라이언트 프로그램과 커널은 모두 예를 들어, C 언어 등의 통상적인 고수준 프로그래밍 언어를 사용해 작성될 수 있다. 클라이언트 프로그램은 병렬 프로그래밍 모델이 제공하는 API를 사용해 메인 메모리의 데이터를 계산 디바이스 메모리로 복사하고, 계산 디바이스에서 커널을 실행시키고, 계산 디바이스 메모리의 데이터를 메인 메모리로 복사하라는 명령(command)을 내릴 수 있다. 전술한 API를 구현(implementation)한 소프트웨어인 런타임 시스템(runtime system)이 클라이언트 프로그램과 동시에 실행되면서, 클라이언트 프로그램의 명령을 받아 계산 디바이스를 제어할 수 있다. 클라이언트 프로그램이 내리는 명령은, 쓰기 명령(write command), 커널 실행 명령(kernel execution command), 읽기 명령(command)을 포함할 수 있다.
- [85] 예를 들어, 쓰기 명령은, 메인 메모리의 데이터를 계산 디바이스의 메모리로 복사할 수 있다. 커널 실행 명령은 계산 디바이스에서 커널을 실행시킬 수 있다. 읽기 명령은 디바이스 메모리의 데이터를 메인 메모리로 복사할 수 있다.
- [86] 도 3은 본 개시의 일 실시예에 따른 이종클러스터 시스템에서 실행되는 프로그램을 실행시키기 위한 페이지 폴트 처리 방법(300)을 나타내는 흐름도이다. 도시된 바와 같이, 이종클러스터 시스템에서 프로그램을 실행시키는 방법(300)은, 이종클러스터 시스템에 포함된 복수의 계산 디바이스의 각각에서 이종클러스터 시스템의 클라이언트 노드로부터 할당된

프로그램과 연관된 복수의 워크 그룹을 수신하는 단계(S310)로 개시될 수 있다. 그리고 나서, 복수의 계산 디바이스의 각각에서 할당된 복수의 워크 그룹은 처리될 수 있다(S320). 복수의 워크 그룹에 의해 접근(읽기 및/또는 쓰기)되는 페이지는 이중클러스터 시스템 상에서 단일 메모리 주소 공간을 제공하도록 구성된 가상 메모리에 포함된 가상 페이지를 통해 관리될 수 있다.

- [87] 단계(S330)에서, 복수의 계산 디바이스 중 제1 디바이스가 페이지 폴트가 발생하였는지 여부를 판정할 수 있다. 제1 계산 디바이스에 할당된 복수의 워크 그룹을 실행하는 중에 페이지 폴트가 발생하는 경우, 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지에 대한 정보를 불러올 수 있다(S340). 그리고 나서, 최근 페이지에 대한 데이터가 수신될 수 있다(S350). 페이지 폴트가 발생되지 않는 경우, 할당된 복수의 워크 그룹의 처리를 계속하거나, 처리가 완료되었으면 종료될 수 있다. 예를 들어, 제1 계산 디바이스는 할당된 워크 그룹을 처리하고 나면, 다음 워크 그룹을 수신하여 처리할 수 있다. 여기서, 제1 계산 디바이스에 의해 처리되는 워크 그룹을 하기 위한 데이터는 이중클러스터 시스템 내의 다른 계산 디바이스 또는 다른 노드에 존재하거나 저장될 수 있다. 이 경우, 제1 계산 디바이스는 이중클러스터 시스템 내의 다른 계산 디바이스에서 워크 그룹을 처리한 결과 데이터를 입력 데이터로 수신하는 경우가 생길 수 있다. 이를 위해, 제1 계산 디바이스는 이러한 입력 데이터를 수신하기 위해, 해당 페이지의 최근 노드를 찾아야 할 수 있는데, 이러한 과정은 도 6을 참고하여 상세히 설명된다.
- [88] 도 4는 본 개시의 일 실시예에 따른 복수의 워크 그룹을 복수의 계산 디바이스 각각에 할당하는 예시를 나타내는 도면이다. 여기서, 복수의 노드의 각각은 복수의 계산 디바이스를 포함할 수 있다. 도 4에서는 복수의 노드의 각각이 동일한 수의 계산 디바이스를 포함하도록 도시되어 있으나, 이에 한정되지 않으며, 복수의 노드의 각각은 상이한 수의 계산 디바이스를 포함할 수 있다.
- [89] 이중클러스터 시스템에서, 복수의 계산 디바이스의 특성 및 성능에 따라 복수의 계산 디바이스의 각각에 할당되는 최적의 워크 그룹의 개수를 결정할 수 있다. 일 실시예에 따르면, 프로그램의 컴파일 동안의 static 분석 결과, 프로그램의 실행 시의 Runtime 분석 결과 또는 프로그램의 프로파일(profile) 분석 결과 중 적어도 하나를 기초로, 프로그램과 연관된 복수의 워크 그룹은 복수의 계산 디바이스의 각각에 할당될 수 있다. 예를 들어, 컴파일 동안의 정적 분석(Static Analysis) 기법은, 프로그램의 컴파일 타임 중에 복수의 계산 디바이스가 접근하는 메모리 영역의 패턴을 추출할 수 있다. 또한, 프로그램이 실행되기 위해 복수의 계산 디바이스 각각이 접근하는 메모리 영역에 해당하는 데이터를 복수의 계산 디바이스 각각에 미리 분배하는데 이용할 수 있으며, 이에 제한되지 않고 다양한 분석 기법들을 활용될 수 있다.
- [90] 도 4에 도시된 바와 같이, 노드 0의 제1 계산 디바이스(410), 제2 계산 디바이스(420), 및 노드 N의 제2 계산 디바이스(430) 각각에 워크 그룹을 분배할 수 있다. 복수의 계산 디바이스는 도시된 바와 같이, 서로 상이한 노드에

포함되어 있으며, 단일 메모리 주소공간을 갖는 가상 메모리를 공유할 수 있다. 또한, 복수의 계산 디바이스에 할당된 워크 그룹들은 도시된 바와 같이, 워크 아이템(440)으로 나뉘어, 복수의 계산 디바이스 각각에 포함된 계산 유닛 또는 복수의 PE(Processing Element)에 분배되어 처리될 수 있다.

- [91] 일 실시예에서, 가상 계산 디바이스로 입력된 워크 그룹은, 이종클러스터 시스템 상의 복수의 계산 디바이스에 분배되어 실행될 수 있다. 예를 들어, 워크 그룹의 분배는, 입력 받은 전체 워크 그룹을 균등하게 복수의 계산 디바이스에 분배할 수 있다. 다른 실시예에서, 특정 서버 노드 또는 특정 계산 디바이스에만 워크 그룹을 분배할 수도 있다. 또 다른 실시예에서, 복수의 계산 디바이스(410, 420, 430)들의 성능이 균등하지 않을 경우, 실행이 먼저 끝난 계산 디바이스가 처리되지 않은 워크 그룹을 할당받아 처리하는 형태로 워크 그룹이 분배될 수도 있다. 또 다른 실시예에서, 분배되는 복수개의 워크 그룹이 하나의 계산 디바이스에 할당되어 실행될 수도 있다.
- [92] 도 5는 본 개시의 일 실시예에 따른 복수의 워크 그룹과 복수의 워크 그룹이 접근하는 복수의 가상 페이지 사이의 관계를 나타내는 테이블(500)의 예시를 나타내는 도면이다. 여기서, 복수의 가상 페이지는 이종클러스터 시스템에서 제공되는 가상의 메모리에 포함된 임의의 페이지를 지칭할 수 있다. 일 실시예에 따르면, 복수의 워크 그룹과 복수의 워크 그룹이 접근하는 복수의 가상 페이지 사이의 관계를 나타내는 테이블(500)을 기초로, 프로그램과 연관된 복수의 워크 그룹은 복수의 계산 디바이스의 각각에 할당될 수 있다. 여기서, 테이블(500)은, 도 5에 도시된 바와 같이, 각 워크그룹이 접근하는 페이지를 2차원 매트릭스(예를 들어, 가로축은 워크그룹 id, 세로축은 page id)로 나타낼 수 있다. 일 실시예에서, 이러한 테이블(500)을 생성하기 위하여, 종래의 컴파일러 분석 기법, 런타임 분석 기법, 프로파일링 기법 및/또는 샘플링 기법 등의 다양한 기법이 이용될 수 있다. 예를 들어, 정적 분석(Static Analysis) 기법은, 프로그램의 컴파일 타임 중에 복수의 계산 디바이스가 접근하는 메모리 영역의 패턴을 추출할 수 있다. 또한, 런타임 분석 기법은 프로그램의 런타임 중에 복수의 계산 디바이스에 접근하는 메모리 영역의 패턴을 분석할 수 있다. 이에 더하여, 프로파일링 기법은, 프로그램과 연관된 복수의 워크 그룹이 복수의 계산 디바이스에 의해 워크 그룹 내의 동작을 실제 실행하지 않고, 메모리 영역에만 접근하도록 실행시키는 기법을 지칭할 수 있다. 이러한 메모리 영역의 접근을 분석함으로써, 메모리 영역의 패턴이 분석될 수 있다. 또한, 샘플링 기법은 프로그램과 연관된 복수의 워크 그룹 중 일부를 복수의 계산 디바이스에 의해 실행함으로써 분석된 복수의 워크 그룹 중 일부에 의한 메모리 영역의 접근 패턴을 기초로, 복수의 워크 그룹 전체의 메모리 영역의 접근 패턴이 추론될 수 있다. 이러한 다양한 분석 방식을 통해 추출되거나 추론된 패턴을 기초로, 프로그램과 연관된 복수의 워크 그룹 및/또는 복수의 워크 그룹의 데이터(예: 초기 데이터)는 복수의 계산 디바이스 각각에 미리 분배될 수 있다. 예를 들어,

복수의 워크 그룹의 데이터는 페이지 폴트가 발생되기 전에 해당 워크 그룹이 실행될 계산 디바이스의 메모리에 미리 프리페칭(pre-fetching)될 수 있다. 이러한 복수의 워크 그룹 및/또는 복수의 워크 그룹의 데이터의 분배에 대한 정보는 도 5의 테이블(500)에 포함될 수 있다.

- [93] 생성된 테이블(500)에 포함된 정보를 기초로, 이종클러스터 시스템 내에서 산술적으로 전체 프로그램이 실행되기 위한 최적의 스케줄은 산출될 수 있다. 예를 들어, 전술한 최적의 스케줄은 페이지 폴트를 최소한으로 발생되도록 구성된 스케줄을 포함할 수 있다. 이러한 최적의 스케줄을 이용하여, 복수의 워크 그룹 및 복수의 워크 그룹의 데이터를 복수의 계산 디바이스에 할당하면, 페이지 폴트를 최소화하도록, 즉 최고의 효율로 프로그램을 복수의 계산 디바이스로 나누어 실행할 수 있다.
- [94] 일 실시예에 따르면, 이러한 최적의 스케줄은, 일 실시예에서, 최적의 솔루션이 도출 가능한 ILP(integer linear programming)를 이용해 추출할 수 있으며, 이에 제한되지 않고, 다른 실시예에서, 최적에 가까운 솔루션을 위해 그리디(Greedy) 알고리즘으로 휴리스틱 하게 추출할 수도 있다. 이러한 최적의 스케줄에 포함된 정보는 테이블(500)에 포함될 수 있다.
- [95] 일 실시예에 따르면, 상술된 분석 및/또는 최적의 스케줄을 이용하여 생성된 테이블(500)은, 도 5에 도시된 바와 같이, 페이지_0에 읽고 쓰는(Read, Write) 작업을 수행하는 워크 그룹(wg)_0, 페이지_0에 읽는 작업을 수행하고 페이지_6에 읽는(R) 작업을 수행하는 워크 그룹_1 및 페이지_1에 읽는(R) 작업을 수행하고 페이지_6에 읽는(R) 작업을 수행하는 워크 그룹_2를 하나의 계산 디바이스 0에 할당하도록 구성된 정보를 포함할 수 있다. 또한, 테이블(500)에 도시된 바와 같이, 페이지 2에 읽고 쓰는(RW) 작업을 수행하는 워크 그룹_3, 페이지 2에 쓰는(W) 작업을 수행하는 워크 그룹_4, 및 페이지 2, 페이지 3 및 페이지 4에 읽는(R) 작업을 수행하는 워크 그룹 5는 계산 디바이스 1에 할당될 수 있다. 여기서, 페이지 0 부터 페이지 2까지는 오너 노드 0에 의해 관리되고, 페이지 3부터 페이지 5까지는 오너 노드 1에 의해 관리될 수 있다.
- [96] 도 6은 본 개시의 일 실시예에 따른 이종클러스터 시스템에서 실행되는 프로그램을 실행시키기 위해, 페이지 폴트를 처리하는 방법(600)을 나타내는 흐름도이다. 페이지 폴트를 처리하는 방법(600)은, 노드 n(610)의 제1 계산 디바이스에서 페이지 p에 페이지 폴트가 발생하는 단계(S605)로 개시될 수 있다. 그리고 나서, 단계(S610)에서, 제1 계산 디바이스는 오너 노드(620)에게 페이지 p의 페이지 잠금 요청을 전송할 수 있다. 이 때, 노드 n(610)의 제1 계산 디바이스는 페이지 폴트가 발생한 페이지 p의 최근 페이지 정보 요청을 오너 노드(620)에 송신할 수 있다(S610). 여기서, 제1 계산 디바이스는 노드 n에 포함된 임의의 계산 디바이스를 지칭할 수 있다. 이에 더하여, 페이지 p는 노드 n(610), 클라이언트 노드 및 오너 노드(620)를 포함한 이종클러스터 시스템에 동작가능한 가상 메모리에 포함된 임의의 가상 페이지를 지칭할 수 있다. 또한,

오너 노드는 가상 메모리에 포함된 복수의 페이지 중 일부 페이지를 관리하는 노드를 지칭할 수 있다. 즉, 오너 노드는 자신이 관리하는 페이지에 대한 최근 페이지에 대한 정보를 저장할 수 있다. 예를 들어, 오너 노드(620)는 가상 메모리에 포함된 복수의 가상 페이지 중에서 페이지 p에 대한 최근 페이지에 대한 정보를 저장할 수 있다.

- [97] 이에 응답하여, 오너 노드(620)는, 단계(S615)에서, 오너 노드(620)에서 페이지 p에 대한 잠금 획득 여부를 검증할 수 있다. 페이지 p에 대한 잠금을 획득한 경우, 페이지 p 잠금 획득에 대응하는 ACK를 노드 n(610)에 전송하고, 단계(S625)로 진행할 수 있다. 페이지 p에 대한 잠금을 획득하지 못한 경우, 단계(S620)로 진행할 수 있다. 다음으로, 단계(S620)에서, 오너 노드(620)에 포함된 페이지 p의 대기자 명단에 노드 n(610) 또는 노드 n(610)의 계산 디바이스를 추가할 수 있다. 페이지 p의 대기자 명단에 노드 n(610) 또는 노드 n(610)의 계산 디바이스를 추가한 경우, 페이지 p에 대한 잠금을 획득할 때까지 대기한 후, 페이지 p에 대한 잠금을 획득하면, 단계(S615)로 돌아가 페이지 p에 대한 잠금 획득에 대응하는 ACK를 노드 n(610) 또는 노드 n(610)의 계산 디바이스에 전송하고, 단계(S625)로 진행할 수 있다.
- [98] 다음으로, 단계(S625)에서, 노드 n의 제1 계산 디바이스는 오너 노드(620)에서 페이지 p에 대한 최근 페이지 정보를 수신하거나 읽을 수 있다. 다음으로, 단계(S630)에서, 페이지 p에 대한 최근 페이지가 노드 n(610) 내에 존재하는지 검증할 수 있다. 즉, 페이지 p에 대한 최근 페이지에 대한 정보가 노드 n(610)의 제1 계산 디바이스와 상이한 계산 디바이스의 메모리에 포함된 페이지를 가리키는지 여부가 판정될 수 있다.
- [99] 페이지 p에 대한 최근 페이지가 노드 n(610)에 존재하는 경우, 단계(S655)로 진행할 수 있다. 페이지 p에 대한 최근 페이지가 노드 n(610)에 존재하지 않는 경우, 단계(S635)로 진행할 수 있다. 다음으로, 단계(S635)에서, 페이지 p에 대한 최근 페이지가 클라이언트 노드에 존재하는지 검증할 수 있다. 즉, 노드 n(610)은, 페이지 p에 대한 최근 페이지에 포함된 데이터가 클라이언트 노드에 있는 문의할 수 있다. 페이지 p에 대한 최근 페이지가 클라이언트 노드에 존재하는 경우, 단계(S645)로 진행할 수 있다. 페이지 p에 대한 최근 페이지가 클라이언트 노드에 존재하지 않는 경우, 단계(S640)로 진행할 수 있다. 다음으로, 단계(S640)에서, 페이지 p는 계산 디바이스 메모리에서 메인 메모리로 마이그레이션될 수 있다.
- [100] 다음으로, 단계(S645)에서, 노드 n(610)은 최근 노드(630)에서 페이지 p에 대한 최근 페이지를 읽을 수 있다. 예를 들어, 클라이언트 노드에 최근 페이지에 대한 데이터가 포함된 경우, 노드 n(610)은 클라이언트 노드로부터 페이지 p에 대한 최근 페이지에 대한 데이터를 수신할 수 있다. 또 다른 예로서, 페이지 p에 대한 최근 페이지가 클라이언트 노드에 포함되지 않은 경우, 최근 페이지 정보가 나타내는 최신 노드(620) 또는 최신 노드(620)의 제2 계산 디바이스로부터

페이지 p와 연관된 최근 페이지에 대한 데이터를 수신할 수 있다. 여기서, 이러한 데이터 수신을 위해 제1 계산 디바이스와 제2 계산 디바이스 사이의 통신이 이용될 수 있다. 이와 달리, 이러한 데이터 수신을 위해 노드 n(510)의 CPU와 최근 노드(630)의 CPU 사이의 통신이 이용될 수 있다. 또는, 클라이언트 노드를 통해 제1 계산 디바이스로부터 제2 계산 디바이스로 데이터가 수신될 수 있다.

- [101] 단계(S650)에서, 오너 노드(620)에서 페이지 p에 대한 최근 페이지 정보를 기록(write)할 수 있고, 단계(S655)로 진행될 수 있다. 일 실시예에 따르면, 오너 노드(620)의 가상 페이지 관리 테이블의 업데이트를 수행하기 위하여, 노드 n(610)의 제1 계산 디바이스에 의해 최근 페이지에 포함된 데이터를 처리한, 페이지 p에 대한 정보를 오너 노드(620)로 송신할 수 있다.
- [102] 단계(S655)에서, 노드 n(610)은 페이지 p에 대한 페이지 폴트 핸들러를 호출할 수 있다. 일 실시예에 따르면, 노드 n(610)은 오너 노드로부터 수신된 최근 페이지에 대한 정보가 노드 n(610)의 상이한 계산 디바이스의 메모리에 포함된 페이지를 가리키는 경우, 노드 n(610)에서 제공하는 페이지 폴트 핸들러를 실행할 수 있다. 그리고 나서, 오너 노드(620)에서 페이지 p에 대한 페이지 잠금 해제 요청을 전송할 수 있다(S660).
- [103] 다음으로, 단계(S665)에서, 오너 노드(620)는 페이지 p에 대한 잠금을 해제할 수 있고, 페이지 p에 대한 대기자 명단의 첫 번째 노드로 ACK를 전송할 수 있다(S670).
- [104] 상술한 단계를 통해 이중클러스터 시스템에서 가상 메모리가 구현되어, 이중클러스터 시스템에 포함된 모든 노드가 단일 메모리 주소 공간을 공유하는 메모리를 구현할 수 있으며, 이에 제한되지 않고, 이와 유사한 다른 방식으로 가상 메모리를 구현할 수 있다.
- [105] 일 실시예에서, 가상 메모리(120)를 이용하여 각각의 노드에 포함된 복수의 계산 디바이스는 프로그램의 실행에 필요한 데이터를 필요할 때마다, 복수의 계산 디바이스 메모리로 가져와 프로그램(예를 들어, 커널 프로그램)을 실행할 수 있다. 이 때, 프로그램의 실행에 필요한 데이터는 페이지 단위로 관리될 수 있으나, 이에 제한되지 않고, 더 큰 메모리 단위로 관리될 수도 있다.
- [106] 요컨대, 노드 n(610)의 페이지 p에 페이지 폴트가 발생하면 atomic한 페이지 폴트 처리를 위해 페이지 p의 오너 노드(620)에 잠금(lock)을 요청하고, 노드 n(610)이 페이지 p에 대한 잠금 획득에 대응하는 ACK(Acknowledgement, 예를 들어, 확인응답)를 수신하면, 오너 노드(620)로부터 페이지 정보를 읽고, 최근 페이지가 존재하는 노드에서 페이지의 데이터를 읽어올 수 있다. 그리고, 오너 노드(620)가 보유한 페이지 정보에 페이지 폴트가 발생한 노드 n(610)을, 최근 페이지를 보유한 노드로 수정할 수 있다. 그리고 나서, 계산 디바이스 제조사에서 제공하는 계산 디바이스의 페이지 폴트 핸들러(예를 들어, 페이지 폴트 핸들러 함수)를 호출할 수 있다. 마지막으로 모든 페이지 폴트 처리가 끝나면, 페이지 폴트에 대한 잠금을 해제할 수 있다.

- [107] 일 실시예에 따르면, 페이지 정보를 관리하기 위한 페이지 정보 테이블이 각 노드에 있으며, 이는 페이지 폴트가 발생한 메모리가 할당(allocation)될 때 생성될 수 있다. 예를 들어, 페이지 정보는 해당 페이지가 위치한 최근 노드의 정보를 포함할 수 있다.
- [108] 일 실시예에 따르면, 모든 페이지 정보 테이블을 하나의 노드가 가지고 있을 수 있으나, 이에 제한되지 않고, 여러 노드에 분산되어 페이지 정보 테이블을 가질 수 있다. 예를 들어, 중앙집중형으로 모든 페이지 정보를 하나의 노드가 가지고 있을 경우, 페이지 폴트가 발생할 때마다 최근 페이지의 위치를 확인하기 위해, 하나의 노드에 통신이 발생하여 성능 저하가 발생할 수 있다.
- [109] 다른 실시예에 따르면, 페이지 정보 테이블을 여러 노드에 분산하여 관리할 수도 있다. 예를 들어, 페이지 폴트가 발생할 때, 최근 페이지의 위치를 확인하기 위한 통신이 분산되어, 통신으로 인한 성능 저하를 줄일 수 있다.
- [110] 일 실시예에 따르면, 페이지 정보 테이블을 가지고 있는 페이지를 오너 노드라고 지칭할 수 있다. 예를 들어, 오너 노드는 복수의 서버 노드 중 일정 개수의 서버 노드들의 페이지 정보 테이블을 관리할 수도 있다. 또한, 해당 페이지의 최신 위치를 최근 노드라고 지칭할 수 있다.
- [111] 일 실시예에 따르면, 본 발명에 따른 이중클러스터 시스템(130)에서 처리하는 페이지 폴트는 세 가지의 경우를 포함할 수 있다. 페이지 폴트의 세 가지 경우는, 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지가 페이지 폴트가 발생한 페이지와 같은 노드에 있는 경우, 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지가 클라이언트 노드의 가상 메모리에 존재할 경우 및 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지가 페이지 폴트가 발생한 페이지와 다른 노드에 있을 경우를 포함할 수 있으며, 각각 도 7 내지 도 9를 통해 보다 상세히 후술한다.
- [112] 도 7 내지 도 9를 통해 본 개시의 일 실시예에 따른 페이지 폴트를 처리하는 예시를 설명한다. 도 7 내지 도 9에 도시된 바와 같이, 클러스터 시스템에서 단일 주소공간의 가상 메모리(예를 들어, 공유 메모리)를 제공하기 위해서, 가상 메모리 드라이버는, 페이지 폴트 메커니즘에 통신부에 의해 수행되는 노드간 통신을 추가할 수 있다. 예를 들어, 특정 노드에 포함된 특정 계산 디바이스에서 페이지 폴트가 발생한 경우, 해당 페이지를 소유하고 있는 노드를 찾아낸 뒤, 해당 페이지를 네트워크를 통해 페이지 폴트가 발생한 노드로 가져와, 페이지 폴트가 발생한 특정 계산 디바이스의 특정 계산 디바이스 메모리에 전송하여 페이지 폴트를 처리할 수 있다.
- [113] 도 7은 본 개시의 일 실시예에 따른 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지가 페이지 폴트가 발생한 페이지와 같은 노드에 있을 경우, 페이지 폴트를 처리하는 예시를 나타내는 도면이다. 일 실시예에 따르면, 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지가 페이지 폴트가 발생한 페이지와 같은 노드에 존재하는 경우에는, 하나의 노드에서 복수의 계산

디바이스를 이용해 프로그램을 처리할 때 발생한 페이지 폴트를 처리하는 방법과 유사하며, 각각의 노드에 설치된 운영체제가 제공하는 페이지 폴트 메커니즘 및 계산 디바이스 제조사가 제공하는 페이지 폴트 핸들러 함수(예를 들어, ISR, Interrupt Service Routine)를 이용해 페이지 폴트를 처리할 수 있다. 예를 들어, 제1 계산 디바이스에서, 오퍼 노드로부터 수신된 최근 페이지에 대한 정보가 제1 계산 디바이스가 포함된 노드 내의 상이한 계산 디바이스(여기서, 제2 계산 디바이스)의 메모리에 포함된 페이지를 가리키는 경우, 노드에서 제공되는 페이지 폴트 핸들러를 실행할 수 있다. 이에 따라, 제1 계산 디바이스는 제2 계산 디바이스에 포함된 최근 페이지에 포함된 데이터를 제1 계산 디바이스의 노드 메인 메모리를 통해 수신할 수 있다.

[114] 도 8은 본 개시의 일 실시예에 따른 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지가 클라이언트 노드의 가상 메모리에 존재할 경우, 페이지 폴트를 처리하는 예시를 나타내는 도면이다. 일 실시예에 따르면, 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지가 클라이언트 노드의 가상 메모리에 존재하는 경우에는, 클라이언트 노드의 가상 메모리에 저장된 페이지 p에 대한 최근 페이지에 대한 데이터를 읽어올 수 있다. 예를 들어, 도시된 바와 같이, 오퍼 노드로부터 수신된 최근 페이지에 대한 정보가 서버 노드 내의 상이한 계산 디바이스의 메모리에 포함된 페이지가 아닌 경우, 가상 페이지와 연관된 최근 페이지에 포함된 데이터가 클라이언트 노드에 있는지 문의할 수 있다. 이에 응답하여, 가상 페이지와 연관된 최근 페이지에 포함된 데이터가 클라이언트 노드에 있는 경우, 제1 계산 디바이스는 클라이언트 노드로부터 가상 페이지와 연관된 최근 페이지에 포함된 데이터를 서버 노드 메인 메모리를 거쳐 수신할 수 있다.

[115] 도 9는 본 개시의 일 실시예에 따른 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지가 페이지 폴트가 발생한 페이지와 다른 노드에 있을 경우, 페이지 폴트를 처리하는 예시를 나타내는 도면이다. 일 실시예에 따르면, 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지가 페이지 폴트가 발생한 페이지와 다른 노드에 존재하는 경우에는, 최근 노드의 계산 디바이스 메모리에서 최근 노드의 메인 메모리로 페이지 p를 마이그레이션 한 뒤, 노드 n(610)에서 페이지 p에 대한 최근 노드의 최근 페이지를 읽어올 수 있다. 예를 들어, 가상 페이지와 연관된 최근 페이지에 포함된 데이터가 클라이언트 노드에 포함되지 않은 경우, 제1 계산 디바이스는 최근 페이지에 대한 정보가 나타내는 제2 계산 디바이스로부터 가상 페이지와 연관된 최근 페이지에 포함된 데이터를 수신할 수 있다. 도시된 바와 같이, 제2 계산 디바이스에 포함된 데이터는 클라이언트 노드를 통해 제1 계산 디바이스로 이동되거나 복사될 수 있다. 이와 달리, 페이지 폴트와 연관된 데이터의 통신은, 클라이언트 노드를 거치지 않고, 각각의 서버 노드에 포함된 프로세서에 의해 노드간 통신을 통해 수행될 수 있다.

- [116] 또는, 페이지 폴트와 연관된 데이터의 통신은, 각각의 서버 노드에 포함된 계산 디바이스 메모리들의 직접 통신을 통해 수행될 수 있다. 예를 들어, RDMA(Remote Direct Memory Access), MPI(Message Passing Interface) 또는, 소켓 통신 라이브러리 등을 포함하는 노드간 통신 라이브러리를 이용하여 계산 디바이스 메모리들의 직접 통신을 수행할 수 있다.
- [117] 상술된 페이지 폴트 처리 방법을 이용해, 페이지 폴트 발생 시, 프로그램을 실행하기 위해 계산 디바이스에서 필요한 페이지만을 복사함으로써, 데이터 통신 및 실제 계산 디바이스 메모리의 사용을 최소화할 수 있다.
- [118] 상술된 이종클러스터 시스템에서 프로그램을 실행시키는 방법은, 컴퓨터로 읽을 수 있는 기록매체에 컴퓨터가 읽을 수 있는 코드로서 구현될 수도 있다. 컴퓨터가 읽을 수 있는 기록매체는 컴퓨터 시스템에 의해 판독될 수 있는 데이터가 저장되는 모든 종류의 기록장치를 포함한다. 컴퓨터가 읽을 수 있는 기록매체의 예로는 ROM, RAM, CD-ROM, 자기 테이프, 플로피 디스크, 광데이터 저장장치 등이 있다. 또한, 컴퓨터가 읽을 수 있는 기록매체는 네트워크로 연결된 컴퓨터 시스템에 분산되어, 분산방식으로 컴퓨터가 읽을 수 있는 코드가 저장되고 실행될 수 있다. 그리고, 전술된 실시예들을 구현하기 위한 기능적인(functional) 프로그램, 코드 및 코드 세그먼트들은 본 발명이 속하는 기술분야의 프로그래머들에 의해 용이하게 추론될 수 있다.
- [119] 본 개시의 방법, 동작 또는 기법들은 다양한 수단에 의해 구현될 수도 있다. 예를 들어, 이러한 기법들은 하드웨어, 펌웨어, 소프트웨어, 또는 이들의 조합으로 구현될 수도 있다. 본원의 개시와 연계하여 설명된 다양한 예시적인 논리적 블록들, 모듈들, 회로들, 및 알고리즘 단계들은 전자 하드웨어, 컴퓨터 소프트웨어, 또는 양자의 조합들로 구현될 수도 있음을 통상의 기술자들은 이해할 것이다. 하드웨어 및 소프트웨어의 이러한 상호 대체를 명확하게 설명하기 위해, 다양한 예시적인 구성요소들, 블록들, 모듈들, 회로들, 및 단계들이 그들의 기능적 관점에서 일반적으로 위에서 설명되었다. 그러한 기능이 하드웨어로서 구현되는지 또는 소프트웨어로서 구현되는지의 여부는, 특정 애플리케이션 및 전체 시스템에 부과되는 설계 요구사항들에 따라 달라진다. 통상의 기술자들은 각각의 특정 애플리케이션을 위해 다양한 방식들로 설명된 기능을 구현할 수도 있으나, 그러한 구현들은 본 개시의 범위로부터 벗어나게 하는 것으로 해석되어서는 안된다.
- [120] 하드웨어 구현에서, 기법들을 수행하는 데 이용되는 프로세싱 유닛들은, 하나 이상의 ASIC들, DSP들, 디지털 신호 프로세싱 디바이스들(digital signal processing devices; DSPD들), 프로그램가능 논리 디바이스들(programmable logic devices; PLD들), 필드 프로그램가능 게이트 어레이들(field programmable gate arrays; FPGA들), 프로세서들, 제어기들, 마이크로제어기들, 마이크로프로세서들, 전자 디바이스들, 본 개시에 설명된 기능들을 수행하도록 설계된 다른 전자 유닛들, 컴퓨터, 또는 이들의 조합 내에서 구현될 수도 있다.

- [121] 따라서, 본 개시와 연계하여 설명된 다양한 예시적인 논리 블록들, 모듈들, 및 회로들은 범용 프로세서, DSP, ASIC, FPGA나 다른 프로그램 가능 논리 디바이스, 이산 게이트나 트랜지스터 로직, 이산 하드웨어 컴포넌트들, 또는 본원에 설명된 기능들을 수행하도록 설계된 것들의 임의의 조합으로 구현되거나 수행될 수도 있다. 범용 프로세서는 마이크로프로세서일 수도 있지만, 대안으로, 프로세서는 임의의 종래의 프로세서, 제어기, 마이크로제어기, 또는 상태 머신일 수도 있다. 프로세서는 또한, 컴퓨팅 디바이스들의 조합, 예를 들면, DSP와 마이크로프로세서, 복수의 마이크로프로세서들, DSP 코어와 연계한 하나 이상의 마이크로프로세서들, 또는 임의의 다른 구성의 조합으로서 구현될 수도 있다.
- [122] 펌웨어 및/또는 소프트웨어 구현에 있어서, 기법들은 랜덤 액세스 메모리(random access memory; RAM), 판독 전용 메모리(read-only memory; ROM), 비휘발성 RAM(non-volatile random access memory; NVRAM), PROM(programmable read-only memory), EPROM(erasable programmable read-only memory), EEPROM(electrically erasable PROM), 플래시 메모리, 콤팩트 디스크(compact disc; CD), 자기 또는 광학 데이터 스토리지 디바이스 등과 같은 컴퓨터 판독가능 매체 상에 저장된 명령들로서 구현될 수도 있다. 명령들은 하나 이상의 프로세서들에 의해 실행 가능할 수도 있고, 프로세서(들)로 하여금 본 개시에 설명된 기능의 특정 양태들을 수행하게 할 수도 있다.
- [123] 소프트웨어로 구현되는 경우, 상기 기법들은 하나 이상의 명령들 또는 코드로서 컴퓨터 판독 가능한 매체 상에 저장되거나 또는 컴퓨터 판독 가능한 매체를 통해 전송될 수도 있다. 컴퓨터 판독가능 매체들은 한 장소에서 다른 장소로 컴퓨터 프로그램의 전송을 용이하게 하는 임의의 매체를 포함하여 컴퓨터 저장 매체들 및 통신 매체들 양자를 포함한다. 저장 매체들은 컴퓨터에 의해 액세스될 수 있는 임의의 이용 가능한 매체들일 수도 있다. 비제한적인 예로서, 이러한 컴퓨터 판독가능 매체는 RAM, ROM, EEPROM, CD-ROM 또는 다른 광학 디스크 스토리지, 자기 디스크 스토리지 또는 다른 자기 스토리지 디바이스들, 또는 소망의 프로그램 코드를 명령들 또는 데이터 구조들의 형태로 이송 또는 저장하기 위해 사용될 수 있으며 컴퓨터에 의해 액세스될 수 있는 임의의 다른 매체를 포함할 수 있다. 또한, 임의의 접속이 컴퓨터 판독가능 매체로 적절히 칭해진다.
- [124] 예를 들어, 소프트웨어가 동축 케이블, 광섬유 케이블, 연선, 디지털 가입자 회선(DSL), 또는 적외선, 무선, 및 마이크로파와 같은 무선 기술들을 사용하여 웹사이트, 서버, 또는 다른 원격 소스로부터 전송되면, 동축 케이블, 광섬유 케이블, 연선, 디지털 가입자 회선, 또는 적외선, 무선, 및 마이크로파와 같은 무선 기술들은 매체의 정의 내에 포함된다. 본원에서 사용된 디스크(disk)와 디스크(disc)는, CD, 레이저 디스크, 광 디스크, DVD(digital versatile disc), 플로피디스크, 및 블루레이 디스크를 포함하며, 여기서 디스크들(disks)은 보통

자기적으로 데이터를 재생하고, 반면 디스크들(disks)은 레이저를 이용하여 광학적으로 데이터를 재생한다. 위의 조합들도 컴퓨터 판독가능 매체들의 범위 내에 포함되어야 한다.

- [125] 소프트웨어 모듈은, RAM 메모리, 플래시 메모리, ROM 메모리, EPROM 메모리, EEPROM 메모리, 레지스터들, 하드 디스크, 이동식 디스크, CD-ROM, 또는 공지된 임의의 다른 형태의 저장 매체 내에 상주할 수도 있다. 예시적인 저장 매체는, 프로세서가 저장 매체로부터 정보를 판독하거나 저장 매체에 정보를 기록할 수 있도록, 프로세서에 연결될 수 있다. 대안으로, 저장 매체는 프로세서에 통합될 수도 있다. 프로세서와 저장 매체는 ASIC 내에 존재할 수도 있다. ASIC은 유저 단말 내에 존재할 수도 있다. 대안으로, 프로세서와 저장 매체는 유저 단말에서 개별 구성요소들로서 존재할 수도 있다.
- [126] 이상 설명된 실시예들이 하나 이상의 독립형 컴퓨터 시스템에서 현재 개시된 주제의 양태들을 활용하는 것으로 기술되었으나, 본 개시는 이에 한정되지 않고, 네트워크나 분산 컴퓨팅 환경과 같은 임의의 컴퓨팅 환경과 연계하여 구현될 수도 있다. 또 나아가, 본 개시에서 주제의 양상들은 복수의 프로세싱 칩들이나 장치들에서 구현될 수도 있고, 스토리지는 복수의 장치들에 걸쳐 유사하게 영향을 받게 될 수도 있다. 이러한 장치들은 PC들, 네트워크 서버들, 및 휴대용 장치들을 포함할 수도 있다.
- [127] 본 명세서에서는 본 개시가 일부 실시예들과 관련하여 설명되었지만, 본 개시의 발명이 속하는 기술분야의 통상의 기술자가 이해할 수 있는 본 개시의 범위를 벗어나지 않는 범위에서 다양한 변형 및 변경이 이루어질 수 있다. 또한, 그러한 변형 및 변경은 본 명세서에 첨부된 특허청구의 범위 내에 속하는 것으로 생각되어야 한다.

청구범위

- [청구항 1] 이종클러스터 시스템에서 프로그램을 실행시키는 방법에 있어서,
 상기 이종클러스터 시스템에 포함된 복수의 계산 디바이스의 각각에서,
 상기 이종클러스터 시스템의 클라이언트 노드로부터 할당된 상기
 프로그램과 연관된 복수의 워크 그룹을 수신하는 단계; 및
 상기 복수의 계산 디바이스의 각각에서 상기 할당된 복수의 워크 그룹을
 처리하는 단계를 포함하고,
 상기 복수의 워크 그룹에 의해 접근되는 페이지는 상기 이종클러스터
 시스템 상에서 단일 메모리 주소 공간을 제공하도록 구성된 가상
 메모리에 포함된 가상 페이지를 통해 관리되는,
 이종클러스터 시스템에서 프로그램을 실행시키는 방법.
- [청구항 2] 제1항에 있어서,
 상기 할당된 복수의 워크 그룹을 처리하는 단계는,
 상기 복수의 계산 디바이스 중 제1 계산 디바이스에서, 상기 제1 계산
 디바이스에 할당된 복수의 워크 그룹을 실행하는 중에 페이지 폴트가
 발생하는 경우, 상기 페이지 폴트가 발생한 가상 페이지와 연관된 최근
 페이지에 대한 정보를 불러오는 단계; 및
 상기 제1 계산 디바이스에서, 상기 최근 페이지에 포함된 데이터를
 수신하는 단계를 포함하는,
 이종클러스터 시스템에서 프로그램을 실행시키는 방법.
- [청구항 3] 제2항에 있어서,
 상기 복수의 계산 디바이스는 상기 이종클러스터 시스템에 포함된
 복수의 서버 노드의 각각에 포함되는,
 이종클러스터 시스템에서 프로그램을 실행시키는 방법.
- [청구항 4] 제3항에 있어서,
 상기 페이지 폴트가 발생한 가상 페이지와 연관된 최근 페이지에 대한
 정보를 불러오는 단계는,
 상기 페이지 폴트가 발생한 가상 페이지의 최근 페이지 정보 요청을 오너
 노드에 송신하는 단계; 및
 상기 오너 노드로부터 상기 가상 페이지의 최근 페이지에 대한 정보를
 수신하는 단계;
 상기 수신된 최근 페이지에 대한 정보가, 상기 제1 계산 디바이스를
 포함한 제1 노드 내에서, 상기 제1 계산 디바이스와 상이한 계산
 디바이스의 메모리에 포함된 페이지를 가리키는지 여부를 판정하는
 단계를 포함하는,
 이종클러스터 시스템에서 프로그램을 실행시키는 방법.
- [청구항 5] 제4항에 있어서,

상기 제1 계산 디바이스에서, 상기 최근 페이지에 포함된 데이터를 수신하는 단계는, 상기 오너 노드로부터 수신된 최근 페이지에 대한 정보가 상기 제1 노드 내의 상이한 계산 디바이스의 메모리에 포함된 페이지를 가리키는 경우, 상기 제1 노드에서 제공되는 페이지 폴트 핸들러를 실행하는 단계를 포함하는,
이종클러스터 시스템에서 프로그램을 실행시키는 방법.

[청구항 6]

제4항에 있어서,
상기 제1 계산 디바이스에서 상기 최근 페이지에 포함된 데이터를 수신하는 단계는,
상기 오너 노드로부터 수신된 최근 페이지에 대한 정보가 상기 제1 노드 내의 상이한 계산 디바이스의 메모리에 포함된 페이지가 아닌 경우, 상기 가상 페이지와 연관된 최근 페이지에 포함된 데이터가 상기 클라이언트 노드에 있는지 문의하는 단계; 및
상기 가상 페이지와 연관된 최근 페이지에 포함된 데이터가 상기 클라이언트 노드에 있는 경우, 상기 클라이언트 노드로부터 상기 가상 페이지와 연관된 최근 페이지에 포함된 데이터를 수신하는 단계를 포함하는,
이종클러스터 시스템에서 프로그램을 실행시키는 방법.

[청구항 7]

제6항에 있어서,
상기 제1 계산 디바이스에서 상기 최근 페이지에 포함된 데이터를 수신하는 단계는, 상기 가상 페이지와 연관된 최근 페이지에 포함된 데이터가 상기 클라이언트 노드에 포함되지 않은 경우, 상기 최근 페이지에 대한 정보가 나타내는 제2 계산 디바이스로부터 상기 가상 페이지와 연관된 최근 페이지에 포함된 데이터를 수신하는 단계를 포함하는,
이종클러스터 시스템에서 프로그램을 실행시키는 방법.

[청구항 8]

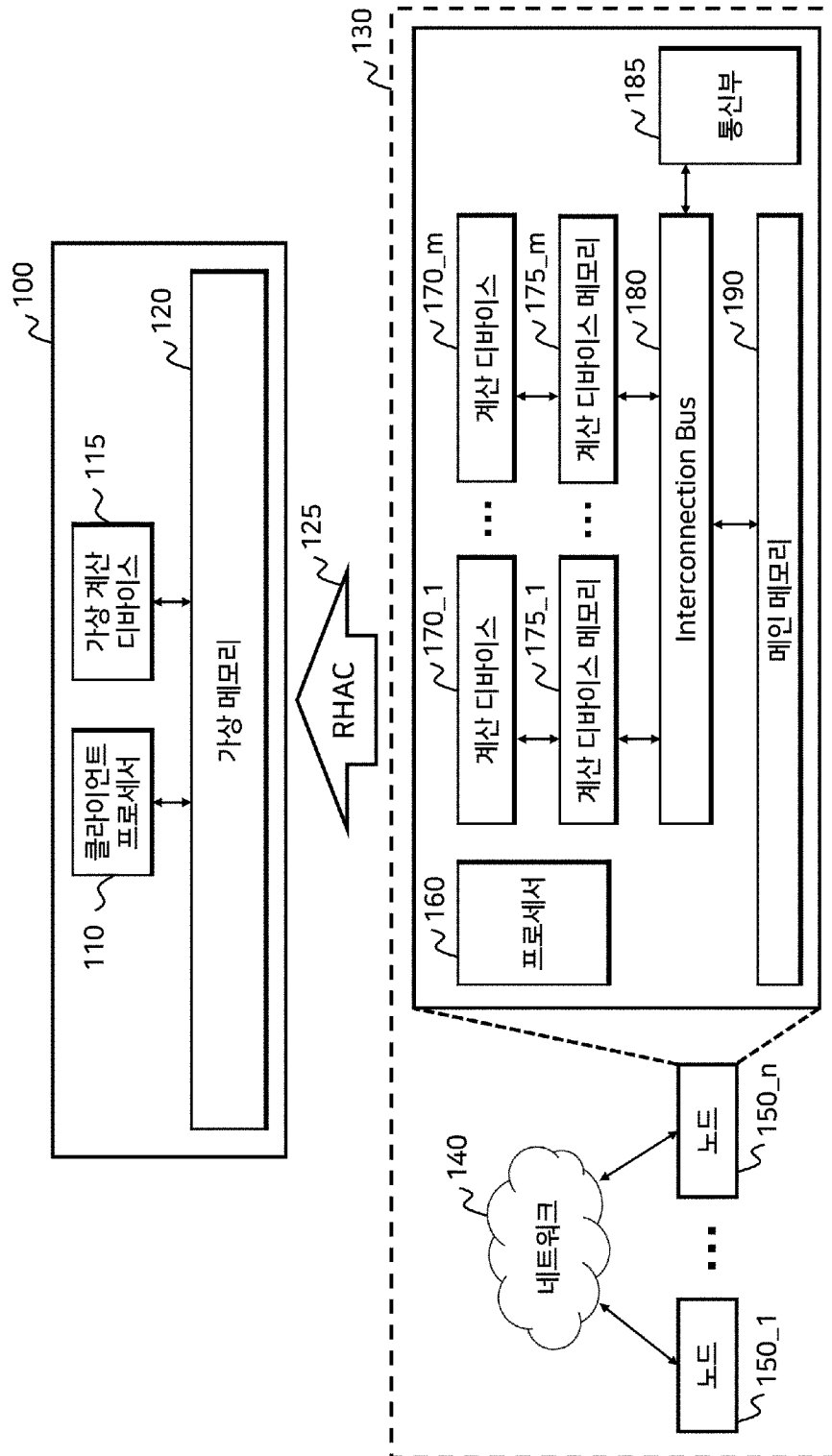
제2항에 있어서,
오너 노드의 가상 페이지 관리 테이블의 업데이트를 수행하기 위하여,
상기 제1 계산 디바이스에 의해, 상기 수신된 최근 페이지에 포함된 데이터를 처리한, 페이지에 대한 정보를 상기 오너 노드로 송신하는 단계를 더 포함하는,
이종클러스터 시스템에서 프로그램을 실행시키는 방법.

[청구항 9]

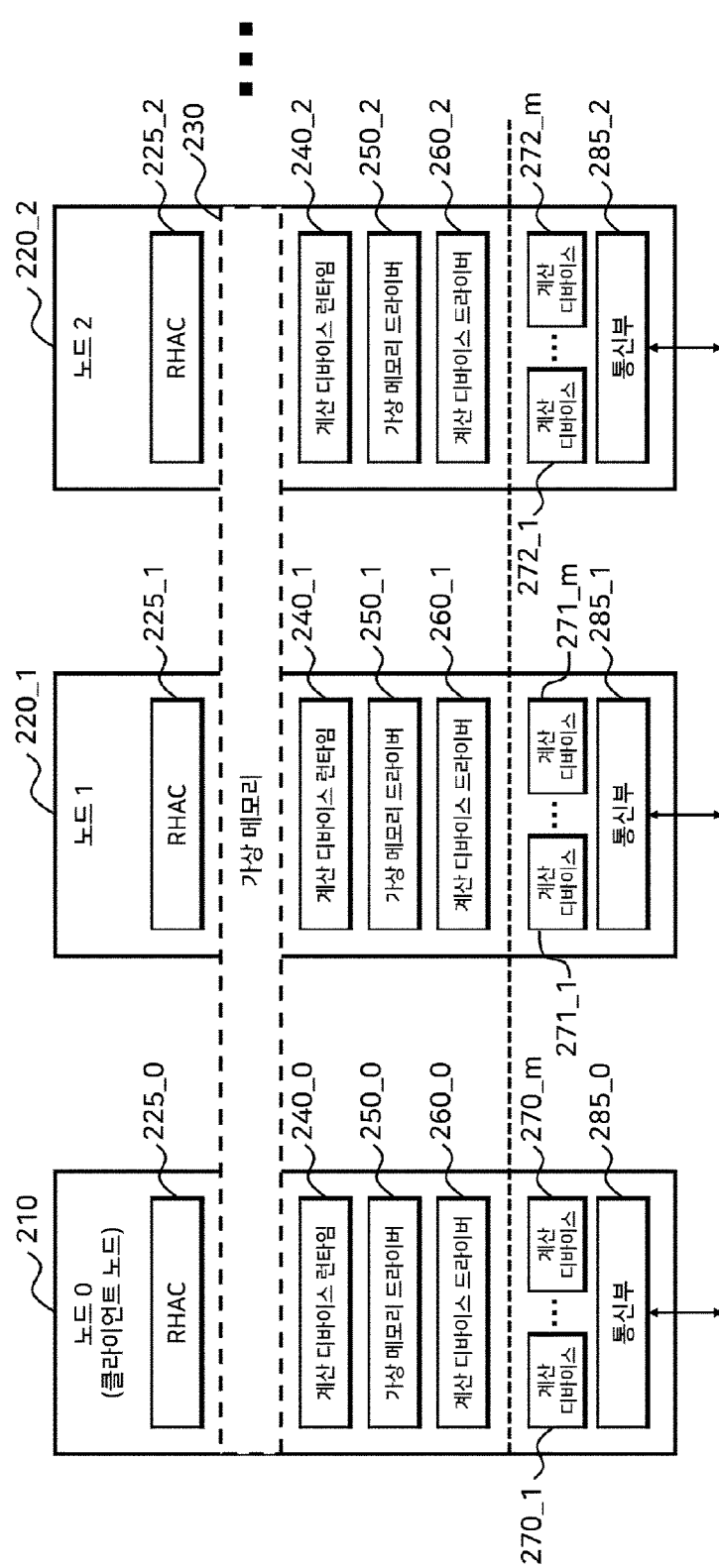
제1항에 있어서,
상기 복수의 워크 그룹과 상기 복수의 워크 그룹이 접근하는 복수의 가상 페이지 사이의 관계를 나타내는 테이블을 기초로, 상기 프로그램과 연관된 복수의 워크 그룹은 상기 복수의 계산 디바이스의 각각에 할당되는,
이종클러스터 시스템에서 프로그램을 실행시키는 방법.

- [청구항 10] 제1항에 있어서,
상기 프로그램의 컴파일 동안의 static 분석 결과, 상기 프로그램의 실행시의 Runtime 분석 결과 또는 상기 프로그램의 프로파일(profile) 분석 결과 중 적어도 하나를 기초로, 상기 프로그램과 연관된 복수의 워크 그룹은 상기 복수의 계산 디바이스의 각각에 할당되는,
이종클러스터 시스템에서 프로그램을 실행시키는 방법.
- [청구항 11] 제1항에 있어서,
상기 복수의 워크 그룹의 메모리 접근 패턴을 분석한 결과를 기초로, 상기 복수의 워크 그룹과 연관된 데이터는, 상기 복수의 워크 그룹이 상기 복수의 계산 디바이스의 각각에 의해 처리되기 전에, 상기 복수의 워크 그룹과 연관된 데이터는 상기 복수의 계산 디바이스의 각각에 미리 프리페칭(pre-fetching)되는,
이종클러스터 시스템에서 프로그램을 실행시키는 방법.
- [청구항 12] 제1항 내지 제11항 중 어느 한 항에 따른 이종클러스터 시스템에서 프로그램을 실행시키는 방법을 컴퓨터에서 실행하기 위해 컴퓨터 판독 가능한 기록 매체에 저장된 컴퓨터 프로그램.

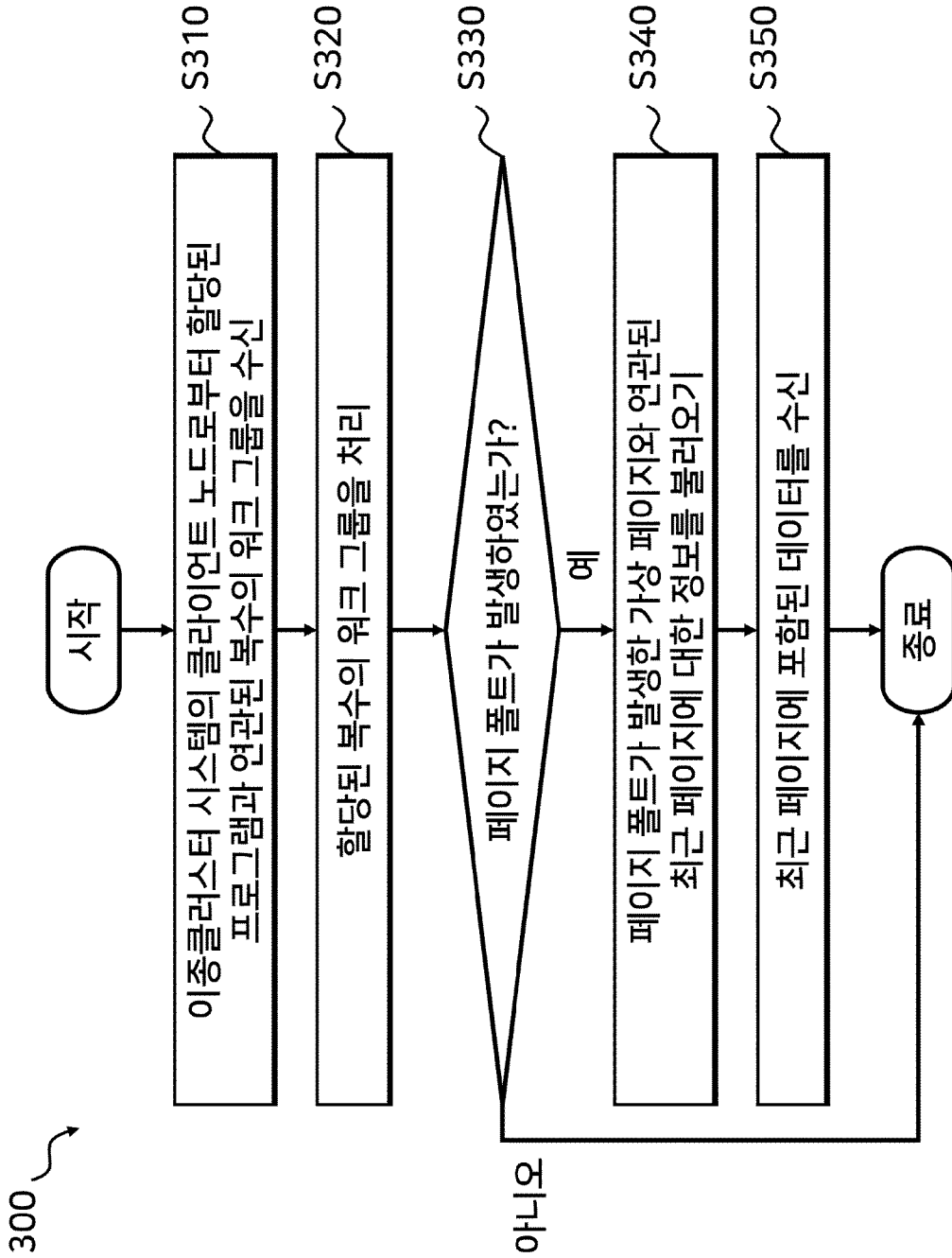
[도 1]



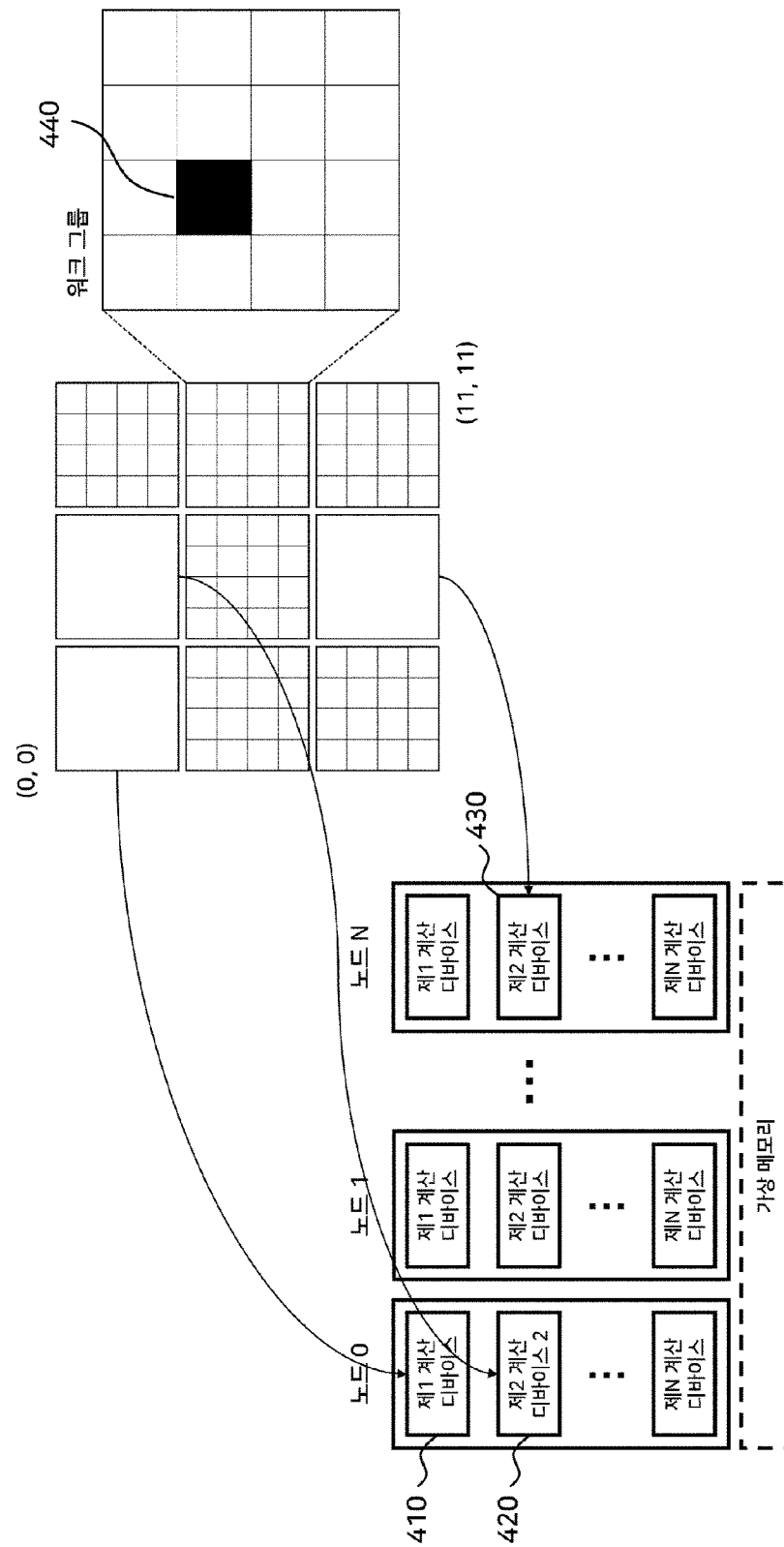
[도2]



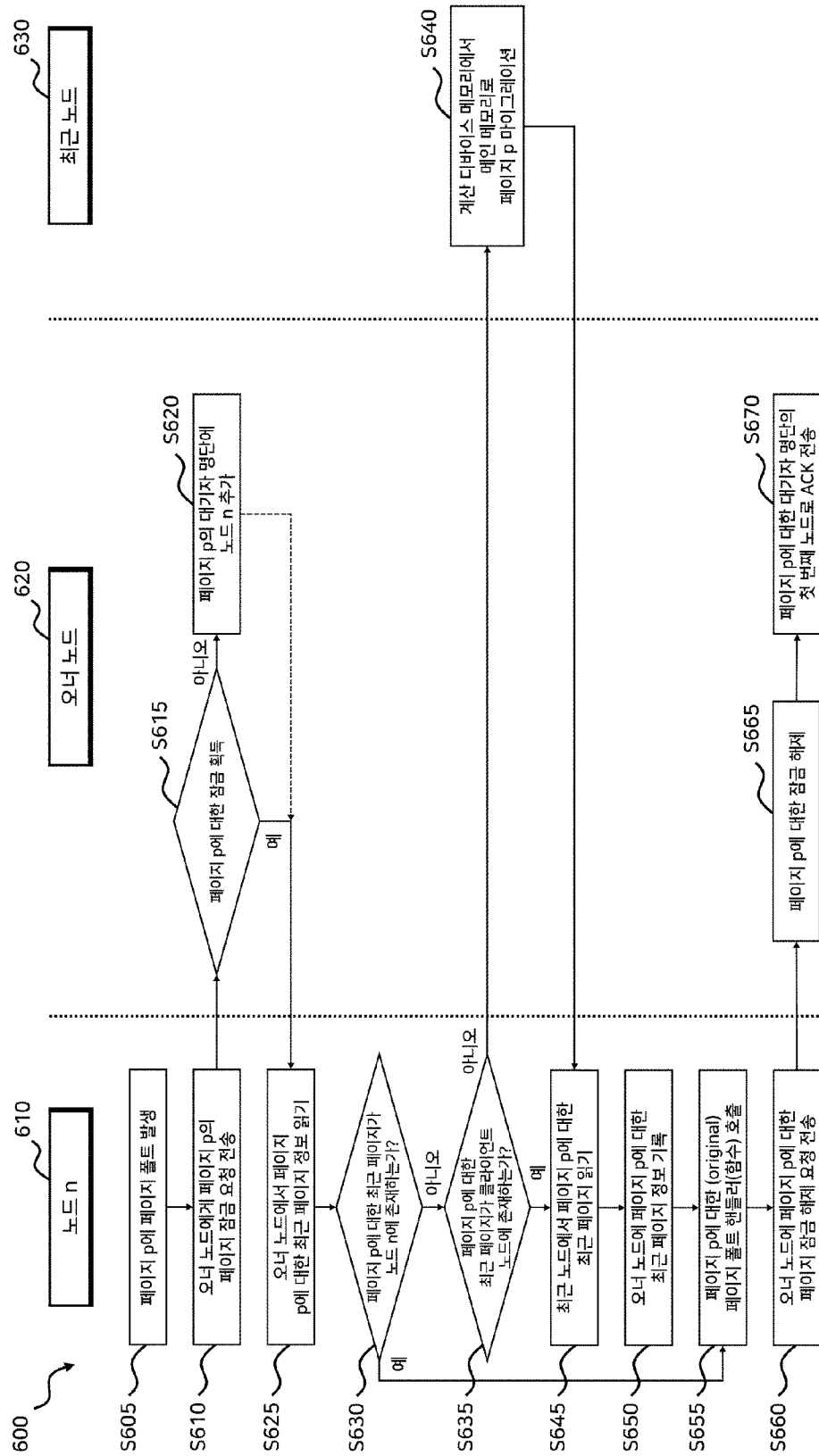
[도3]



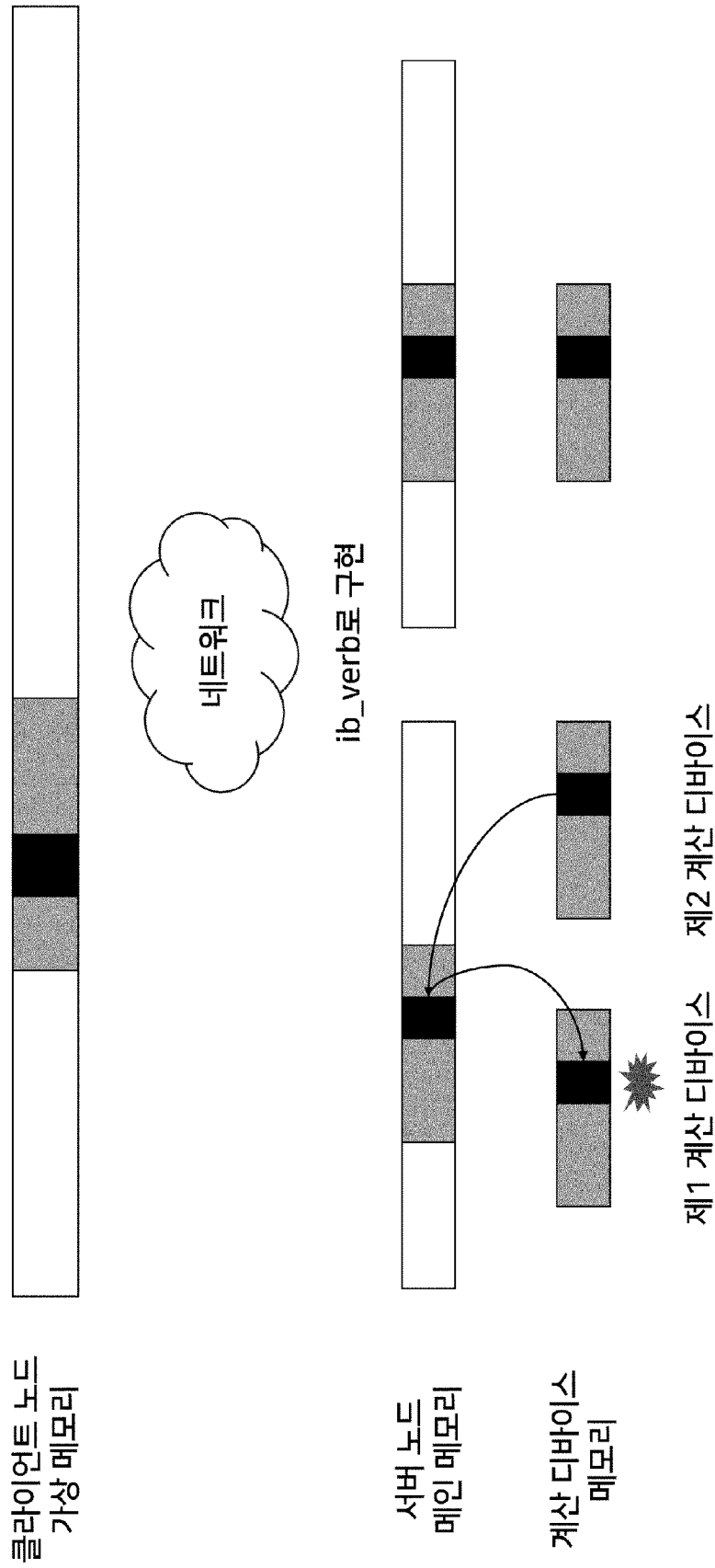
[도4]



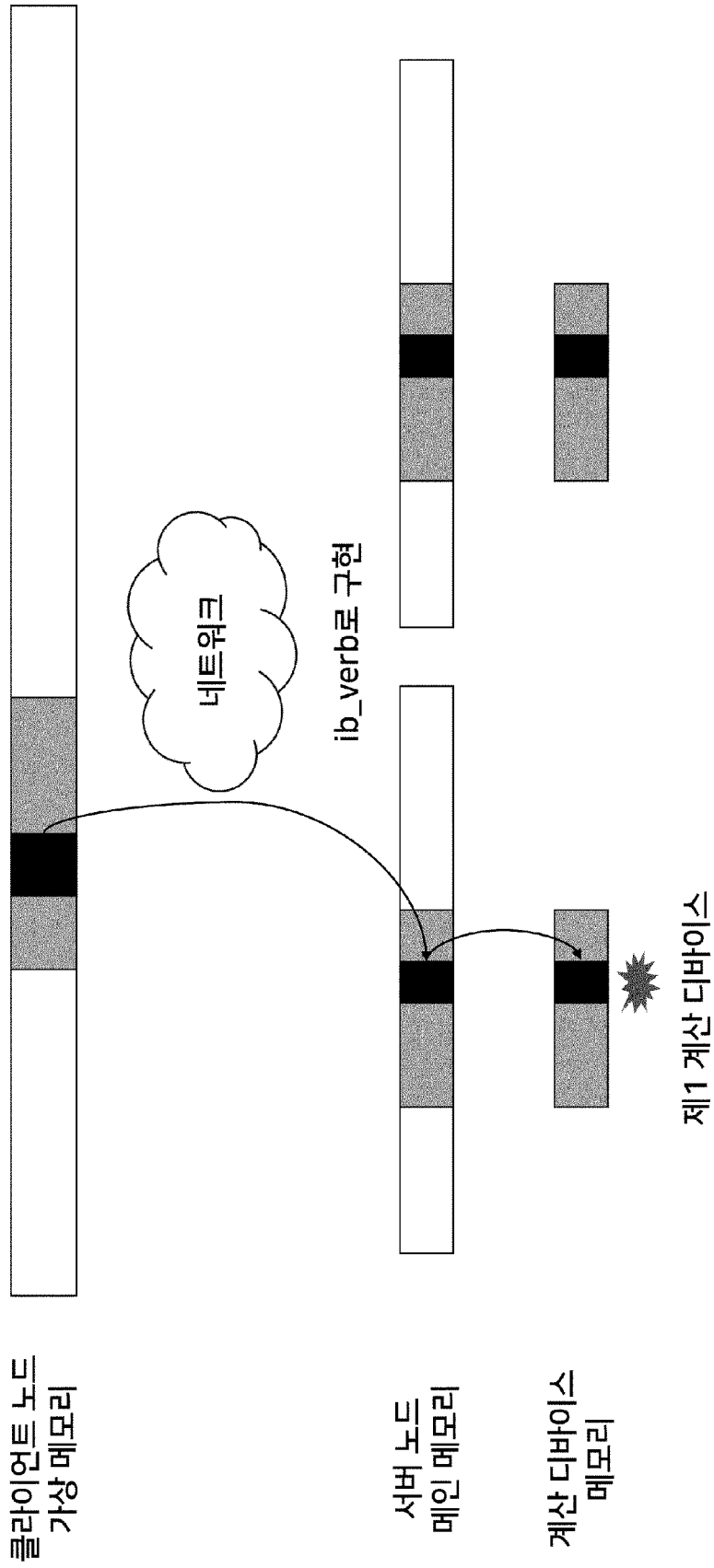
[도 6]



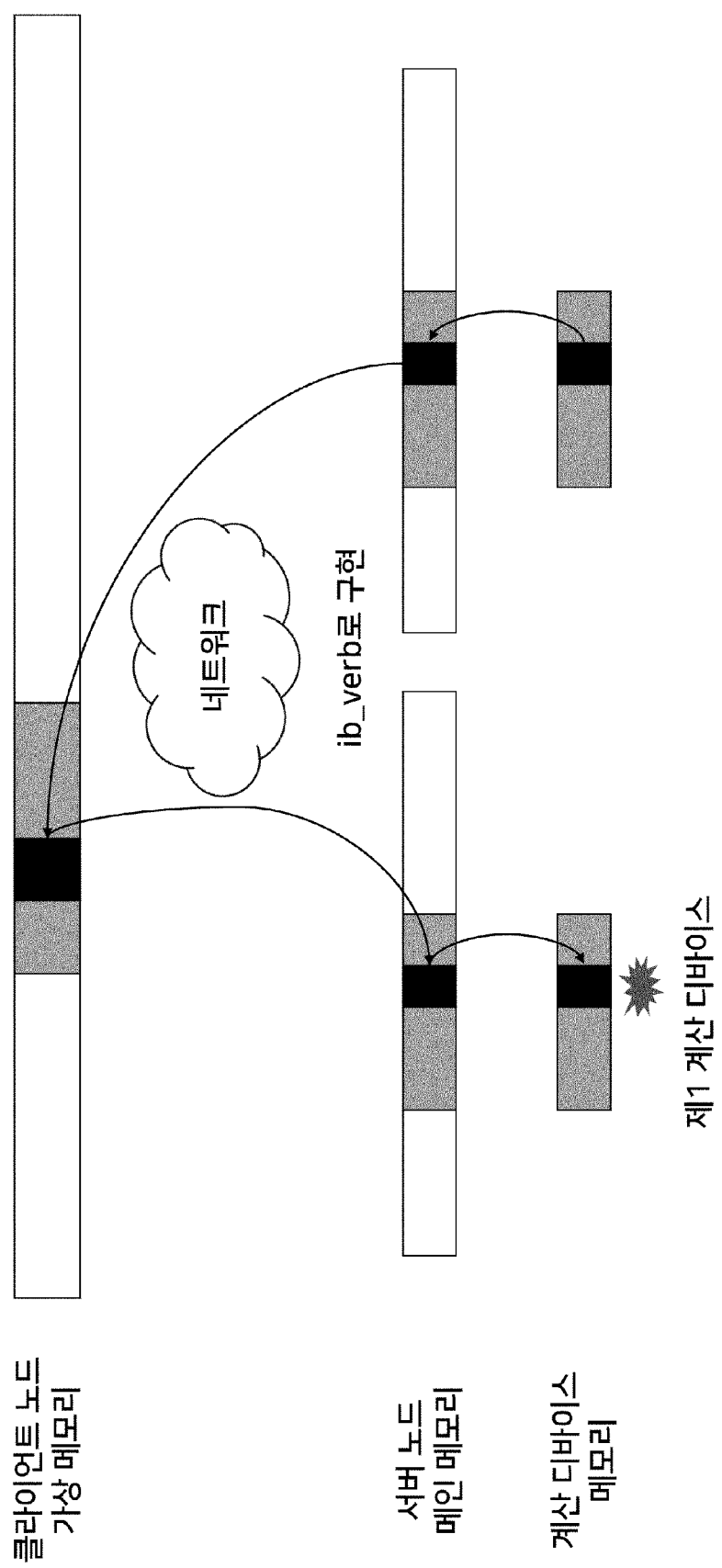
[도7]



[도8]



[도9]



INTERNATIONAL SEARCH REPORT

International application No.

PCT/KR2020/003396

A. CLASSIFICATION OF SUBJECT MATTER

G06F 9/50(2006.01)i

According to International Patent Classification (IPC) or to both national classification and IPC

B. FIELDS SEARCHED

Minimum documentation searched (classification system followed by classification symbols)

G06F 9/50; G06F 15/16; G06F 3/06; G06F 9/44; G06T 1/00

Documentation searched other than minimum documentation to the extent that such documents are included in the fields searched

Korean utility models and applications for utility models: IPC as above

Japanese utility models and applications for utility models: IPC as above

Electronic data base consulted during the international search (name of data base and, where practicable, search terms used)

eKOMPASS (KIPO internal) & Keywords: heterogeneous cluster, program, client, node, workgroup, virtual page, memory

C. DOCUMENTS CONSIDERED TO BE RELEVANT

Category*	Citation of document, with indication, where appropriate, of the relevant passages	Relevant to claim No.
X	WO 2018-065530 A1 (PARTEC CLUSTER COMPETENCE CENTER GMBH.) 12 April 2018 See page 4, line 18-page 10, line 18, claims 1, 2 and figure 1.	1-12
A	KR 10-2018-0084682 A (SEOUL NATIONAL UNIVERSITY R&DB FOUNDATION) 25 July 2018 See paragraphs [0027]-[0052] and figure 1.	1-12
A	US 2016-0170773 A1 (ALCATEL LUCENT) 16 June 2016 See paragraphs [0125]-[0147] and figure 1.	1-12
A	US 2012-0320070 A1 (ARVO, Jukka-pekka) 20 December 2012 See paragraphs [0034]-[0080] and figures 1-5.	1-12
A	US 2009-0213127 A1 (KRISHNAMURTHY, Rajaram B. et al.) 27 August 2009 See paragraphs [0020]-[0028] and figure 1.	1-12



Further documents are listed in the continuation of Box C.



See patent family annex.

* Special categories of cited documents:

“A” document defining the general state of the art which is not considered to be of particular relevance

“E” earlier application or patent but published on or after the international filing date

“L” document which may throw doubts on priority claim(s) or which is cited to establish the publication date of another citation or other special reason (as specified)

“O” document referring to an oral disclosure, use, exhibition or other means

“P” document published prior to the international filing date but later than the priority date claimed

“T” later document published after the international filing date or priority date and not in conflict with the application but cited to understand the principle or theory underlying the invention

“X” document of particular relevance; the claimed invention cannot be considered novel or cannot be considered to involve an inventive step when the document is taken alone

“Y” document of particular relevance; the claimed invention cannot be considered to involve an inventive step when the document is combined with one or more other such documents, such combination being obvious to a person skilled in the art

“&” document member of the same patent family

Date of the actual completion of the international search

19 JUNE 2020 (19.06.2020)

Date of mailing of the international search report

19 JUNE 2020 (19.06.2020)

Name and mailing address of the ISA/KR

Korean Intellectual Property Office
Government Complex Daejeon Building 4, 189, Cheongsu-ro, Seo-gu,
Daejeon, 35208, Republic of Korea

Facsimile No. +82-42-481-8578

Authorized officer

Telephone No.

INTERNATIONAL SEARCH REPORT
Information on patent family members

International application No.

PCT/KR2020/003396

Patent document cited in search report	Publication date	Patent family member	Publication date
WO 2018-065530 A1	12/04/2018	CA 3035459 A1 CN 109791509 A EP 3523723 A1 JP 2019-533249 A KR 10-2019-0058619 A US 2019-0243694 A1	12/04/2018 21/05/2019 14/08/2019 14/11/2019 29/05/2019 08/08/2019
KR 10-2018-0084682 A	25/07/2018	KR 10-2086044 B1	06/03/2020
US 2016-0170773 A1	16/06/2016	CN 105408864 A EP 2833265 A1 JP 2016-531353 A KR 10-2016-0025606 A WO 2015-014431 A1	16/03/2016 04/02/2015 06/10/2016 08/03/2016 05/02/2015
US 2012-0320070 A1	20/12/2012	CN 103608777 A CN 103608777 B EP 2721488 A1 EP 2721488 B1 JP 2014-523021 A JP 5752853 B2 KR 10-1590734 B1 KR 10-2014-0015631 A US 9092267 B2 WO 2012-177343 A1	26/02/2014 10/05/2017 23/04/2014 16/08/2017 08/09/2014 22/07/2015 01/02/2016 06/02/2014 28/07/2015 27/12/2012
US 2009-0213127 A1	27/08/2009	US 7953912 B2	31/05/2011

A. 발명이 속하는 기술분류(국제특허분류(IPC))

G06F 9/50(2006.01)i

B. 조사된 분야

조사된 최소문헌(국제특허분류를 기재)

G06F 9/50; G06F 15/16; G06F 3/06; G06F 9/44; G06T 1/00

조사된 기술분야에 속하는 최소문헌 이외의 문헌

한국등록실용신안공보 및 한국공개실용신안공보: 조사된 최소문헌란에 기재된 IPC

일본등록실용신안공보 및 일본공개실용신안공보: 조사된 최소문헌란에 기재된 IPC

국제조사에 이용된 전산 데이터베이스(데이터베이스의 명칭 및 검색어(해당하는 경우))

eKOMPASS(특허청 내부 검색시스템) & 키워드: 이종클러스터(heterogeneous cluster), 프로그램(program), 클라이언트(client), 노드(node), 워크 그룹(workgroup), 가상 페이지(virtual page), 메모리(memory)

C. 관련 문헌

카테고리*	인용문헌명 및 관련 구절(해당하는 경우)의 기재	관련 청구항
X	WO 2018-065530 A1 (PARTEC CLUSTER COMPETENCE CENTER GMBH) 2018.04.12 페이지 4, 라인 18 - 페이지 10, 라인 18, 청구항 1, 2 및 도면 1	1-12
A	KR 10-2018-0084682 A (서울대학교산학협력단) 2018.07.25 단락 [0027]-[0052] 및 도면 1	1-12
A	US 2016-0170773 A1 (ALCATEL LUCENT) 2016.06.16 단락 [0125]-[0147] 및 도면 1	1-12
A	US 2012-0320070 A1 (JUKKA-PEKKA ARVO) 2012.12.20 단락 [0034]-[0080] 및 도면 1-5	1-12
A	US 2009-0213127 A1 (RAJARAM B. KRISHNAMURTHY 등) 2009.08.27 단락 [0020]-[0028] 및 도면 1	1-12

☐ 추가 문헌이 C(계속)에 기재되어 있습니다.

☒ 대응특허에 관한 별지를 참조하십시오.

* 인용된 문헌의 특별 카테고리:

“A” 특별히 관련이 없는 것으로 보이는 일반적인 기술수준을 정의한 문헌

“D” 본 국제출원에서 출원인이 인용한 문헌

“E” 국제출원일보다 빠른 출원일 또는 우선일을 가지나 국제출원일 이후 “X”에 공개된 선출원 또는 특허 문헌

“L” 우선권 주장에 의문을 제기하는 문헌 또는 다른 인용문헌의 공개일 또는 다른 특별한 이유(이유를 명시)를 밝히기 위하여 인용된 문헌

“O” 구두 개시, 사용, 전시 또는 기타 수단을 언급하고 있는 문헌

“P” 우선일 이후에 공개되었으나 국제출원일 이전에 공개된 문헌

“T” 국제출원일 또는 우선일 후에 공개된 문헌으로, 출원과 상충하지 않으며 발명의 기초가 되는 원리나 이론을 이해하기 위해 인용된 문헌

“X” 특별한 관련이 있는 문헌. 해당 문헌 하나만으로 청구된 발명의 신규성 또는 진보성이 없는 것으로 본다.

“Y” 특별한 관련이 있는 문헌. 해당 문헌이 하나 이상의 다른 문헌과 조합하는 경우로 그 조합이 당업자에게 자명한 경우 청구된 발명은 진보성이 없는 것으로 본다.

“&” 동일한 대응특허문헌에 속하는 문헌

국제조사의 실제 완료일

2020년 06월 19일 (19.06.2020)

국제조사보고서 발송일

2020년 06월 19일 (19.06.2020)

ISA/KR의 명칭 및 우편주소



대한민국 특허청
(35208) 대전광역시 서구 청사로 189,
4동 (둔산동, 정부대전청사)

팩스 번호 +82-42-481-8578

심사관

김성희

전화번호 +82-42-481-3516



국제조사보고서에서 인용된 특허문헌	공개일	대응특허문헌	공개일
WO 2018-065530 A1	2018/04/12	CA 3035459 A1 CN 109791509 A EP 3523723 A1 JP 2019-533249 A KR 10-2019-0058619 A US 2019-0243694 A1	2018/04/12 2019/05/21 2019/08/14 2019/11/14 2019/05/29 2019/08/08
KR 10-2018-0084682 A	2018/07/25	KR 10-2086044 B1	2020/03/06
US 2016-0170773 A1	2016/06/16	CN 105408864 A EP 2833265 A1 JP 2016-531353 A KR 10-2016-0025606 A WO 2015-014431 A1	2016/03/16 2015/02/04 2016/10/06 2016/03/08 2015/02/05
US 2012-0320070 A1	2012/12/20	CN 103608777 A CN 103608777 B EP 2721488 A1 EP 2721488 B1 JP 2014-523021 A JP 5752853 B2 KR 10-1590734 B1 KR 10-2014-0015631 A US 9092267 B2 WO 2012-177343 A1	2014/02/26 2017/05/10 2014/04/23 2017/08/16 2014/09/08 2015/07/22 2016/02/01 2014/02/06 2015/07/28 2012/12/27
US 2009-0213127 A1	2009/08/27	US 7953912 B2	2011/05/31