



(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2022-0147856
(43) 공개일자 2022년11월04일

(51) 국제특허분류(Int. Cl.)
G06F 9/455 (2018.01) G06F 9/48 (2018.01)
(52) CPC특허분류
G06F 9/45558 (2013.01)
G06F 9/4837 (2013.01)
(21) 출원번호 10-2021-0054901
(22) 출원일자 2021년04월28일
심사청구일자 2021년04월28일

(71) 출원인
계명대학교 산학협력단
대구광역시 달서구 달구벌대로 1095, 계명대학교
산학협력관 201호(신당동)
(72) 발명자
박세진
대구광역시 동구 팔공로 260(봉무동, 협성휴포레
이시아폴리스)
권오은
대구광역시 남구 앞산순환로 655, 207동 1003호(
봉덕동, 보성대덕2차아파트)
(74) 대리인
특허법인 무한

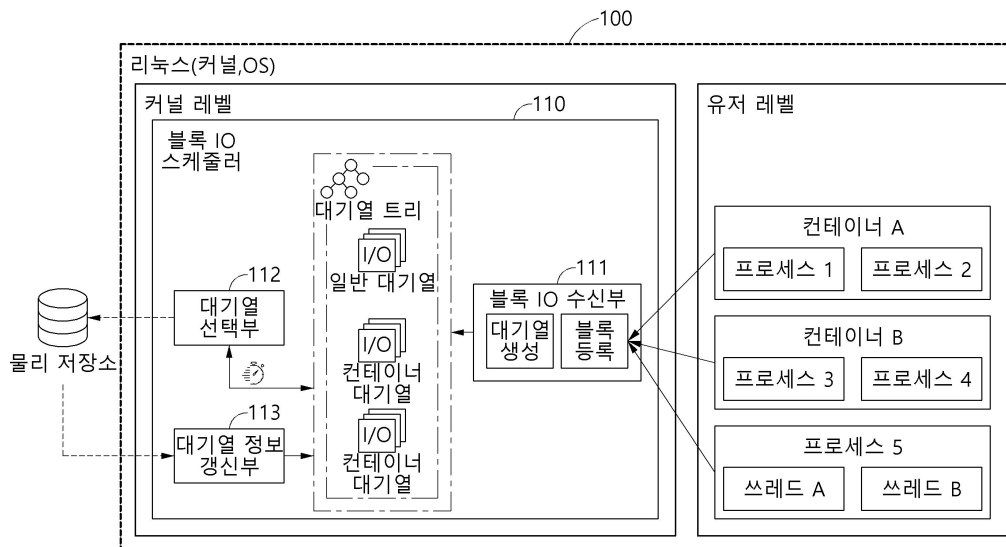
전체 청구항 수 : 총 16 항

(54) 발명의 명칭 블록 IO 스케줄링 방법 및 장치

(57) 요약

블록 IO 스케줄링 방법 및 장치가 개시된다. 커널 레벨의 블록 IO 스케줄러가 수행하는 컨테이너 단위 블록 IO 처리 방법은 어플리케이션을 실행하는 유저 레벨의 프로세스로부터 수신된 블록 IO의 컨테이너 정보에 기초하여 상기 수신된 블록 IO를 복수의 대기열들 중 어느 하나의 대기열에 등록하는 단계; 상기 블록 IO가 등록된 복수의 대기열들의 리소스 제공을 위한 시간 할당량에 기초하여 어느 하나의 대기열을 선택하는 단계; 및 상기 선택된 대기열에 등록되어 있는 블록 IO를 커널에 전달하는 단계를 포함할 수 있다.

대표도



(52) CPC특허분류

G06F 9/4881 (2013.01)

G06F 2009/45562 (2013.01)

G06F 2009/45579 (2019.08)

이 발명을 지원한 국가연구개발사업

과제고유번호	1711109765
과제번호	2019R1G1A1100305
부처명	과학기술정보통신부
과제관리(전문)기관명	한국연구재단
연구사업명	생애 첫 연구사업
연구과제명	모바일 플랫폼 지원을 위한 고성능 경량 블록체인 기술연구
기 여 율	1/1
과제수행기관명	계명대학교
연구기간	2020.03.01 ~ 2021.02.28

명세서

청구범위

청구항 1

커널 레벨의 블록 IO 스케줄러가 수행하는 컨테이너 단위 블록 IO 처리 방법에 있어서,
어플리케이션을 실행하는 유저 레벨의 프로세스로부터 수신된 블록 IO의 컨테이너 정보에 기초하여 상기 수신된 블록 IO를 복수의 대기열들 중 어느 하나의 대기열에 등록하는 단계;
상기 블록 IO가 등록된 복수의 대기열들의 리소스 제공을 위한 시간 할당량에 기초하여 어느 하나의 대기열을 선택하는 단계; 및
상기 선택된 대기열에 등록되어 있는 블록 IO를 커널에 전달하는 단계
를 포함하는 컨테이너 단위 블록 IO 처리 방법.

청구항 2

제1항에 있어서,
상기 등록하는 단계는,
상기 수신된 블록 IO를 발생시킨 프로세스 정보를 이용하여 해당 프로세스가 포함된 컨테이너 정보를 식별하는 단계; 및
상기 식별된 컨테이너 정보에 대응하는 컨테이너 대기열에 상기 수신된 블록 IO를 등록하는 단계
를 포함하는 컨테이너 단위 블록 IO 처리 방법.

청구항 3

제2항에 있어서,
상기 등록하는 단계는,
상기 식별된 컨테이너 정보에 대응하는 컨테이너 대기열이 존재하지 않는 경우, 신규 컨테이너 대기열을 생성하고, 생성된 신규 컨테이너 대기열에 상기 수신된 블록 IO를 등록하는 컨테이너 단위 블록 IO 처리 방법.

청구항 4

제2항에 있어서,
상기 등록하는 단계는,
상기 수신된 블록 IO를 발생시킨 프로세스 정보를 이용하여 해당 프로세스가 컨테이너에 포함되지 않은 것으로 식별되면, 일반 대기열에 상기 수신된 블록 IO를 등록하는 컨테이너 단위 블록 IO 처리 방법.

청구항 5

제1항에 있어서,
상기 등록하는 단계는,
동일한 컨테이너 내에 복수의 프로세스들이 포함된 경우, 상기 동일한 컨테이너 내의 프로세스들에서 발생된 블록 IO들을 하나의 컨테이너 대기열에 시간 순서에 따라 등록하는 컨테이너 단위 블록 IO 처리 방법.

청구항 6

제1항에 있어서,
상기 선택하는 단계는,

상기 블록 IO가 등록된 복수의 대기열들 각각에 대해 리소스 제공을 위한 누적 시간 할당량을 식별하는 단계;
및

상기 복수의 대기열들 중 상기 식별된 누적 시간 할당량이 가장 적은 대기열을 선택하는 단계
를 포함하는 컨테이너 단위 블록 IO 처리 방법.

청구항 7

제1항에 있어서,

상기 선택하는 단계는,

상기 선택된 대기열 내에 블록 IO가 존재하지 않은 것으로 식별된 경우, 해당 대기열을 삭제하는 단계
를 더 포함하는 컨테이너 단위 블록 IO 처리 방법.

청구항 8

제1항에 있어서,

상기 커널에 전달된 블록 IO의 처리가 완료된 경우, 해당 블록 IO에 대응하는 대기열의 누적 시간 할당량을 수정하는 단계

를 더 포함하는 컨테이너 단위 블록 IO 처리 방법.

청구항 9

컨테이너 단위 블록 IO 처리 방법을 수행하는 커널 레벨의 블록 IO 스케줄러에 있어서,

상기 블록 IO 스케줄러는, 프로세서를 포함하고,

상기 프로세서는,

어플리케이션을 실행하는 유저 레벨의 프로세스로부터 수신된 블록 IO의 컨테이너 정보에 기초하여 상기 수신된 블록 IO를 복수의 대기열들 중 어느 하나의 대기열에 등록하고, 상기 블록 IO가 등록된 복수의 대기열들의 리소스 제공을 위한 시간 할당량에 기초하여 어느 하나의 대기열을 선택하며, 상기 선택된 대기열에 등록되어 있는 블록 IO를 커널에 전달하는 블록 IO 스케줄러.

청구항 10

제9항에 있어서,

상기 프로세서는,

상기 수신된 블록 IO를 발생시킨 프로세스 정보를 이용하여 해당 프로세스가 포함된 컨테이너 정보를 식별하고, 상기 식별된 컨테이너 정보에 대응하는 컨테이너 대기열에 상기 수신된 블록 IO를 등록하는 블록 IO 스케줄러.

청구항 11

제10항에 있어서,

상기 프로세서는,

상기 식별된 컨테이너 정보에 대응하는 컨테이너 대기열이 존재하지 않는 경우, 신규 컨테이너 대기열을 생성하고, 생성된 신규 컨테이너 대기열에 상기 수신된 블록 IO를 등록하는 블록 IO 스케줄러.

청구항 12

제10항에 있어서,

상기 프로세서는,

상기 수신된 블록 IO를 발생시킨 프로세스 정보를 이용하여 해당 프로세스가 컨테이너에 포함되지 않은 것으로 식별되면, 일반 대기열에 상기 수신된 블록 IO를 등록하는 블록 IO 스케줄러.

청구항 13

제9항에 있어서,

상기 프로세서는,

동일한 컨테이너 내에 복수의 프로세스들이 포함된 경우, 상기 동일한 컨테이너 내의 프로세스들에서 발생된 블록 IO들을 하나의 컨테이너 대기열에 시간 순서에 따라 등록하는 블록 IO 스케줄러.

청구항 14

제9항에 있어서,

상기 프로세서는,

상기 블록 IO가 등록된 복수의 대기열들 각각에 대해 리소스 제공을 위한 누적 시간 할당량을 식별하고, 상기 복수의 대기열들 중 상기 식별된 누적 시간 할당량이 가장 적은 대기열을 선택하는 블록 IO 스케줄러.

청구항 15

제9항에 있어서,

상기 프로세서는,

상기 선택된 대기열 내에 블록 IO가 존재하지 않은 것으로 식별된 경우, 해당 대기열을 삭제하는 블록 IO 스케줄러.

청구항 16

제9항에 있어서,

상기 프로세서는,

상기 커널에 전달된 블록 IO의 처리가 완료된 경우, 해당 블록 IO에 대응하는 대기열의 누적 시간 할당량을 수정하는 블록 IO 스케줄러.

발명의 설명

기술 분야

[0001] 본 발명은 블록 IO 스케줄링 방법 및 장치에 관한 것으로, 보다 구체적으로는 가상화 컨테이너 환경에서 발생하는 블록 IO의 스레드 수와 무관하게 전체 하드웨어 자원을 컨테이너 단위로 할당하는 방법 및 장치에 관한 것이다.

배경 기술

[0002] 도커(Docker)는 단일의 커널, 컨트롤 호스트에서 운영체제 수준의 가상화 방법이다. 기존 하이퍼바이저(Hypervisor)의 하드웨어 수준 가상화 방법에 상대적으로 낮은 부하를 가져 도커의 실행 단위인 컨테이너에서 어플리케이션이 보다 빠르고 효율적으로 동작한다. 또한 컨테이너 각각은 외부와 격리된 단일의 환경을 제공할 수 있다.

[0003] 많은 사용자가 동시에 그리고 독립적인 환경이 제공되어야 하는 클라우드 컴퓨팅에서 위의 이점을 취하고자 도커를 채용하고 있다. 다양한 서비스와 사용자를 지원하는 클라우드 컴퓨팅에서 제한된 하드웨어 자원, 특히 디스크 자원의 경쟁이 발생한다.

[0004] 종래의 리눅스 기반 블록 IO 스케줄링 방법은 컨테이너를 인지하지 않고 단순히 프로세스 단위로 처리하여 컨테이너 단위로 최저 디스크 자원을 확보할 수 없는 한계가 있다. 따라서 프로세스를 컨테이너 단위로 대기 및 처리할 수 있는 방법이 요구된다.

발명의 내용

해결하려는 과제

[0005] 본 발명은 가상화 컨테이너 환경에서 다중 컨테이너 내 다중 프로세스로부터 발생하는 블록 IO 요청을 리눅스 커널 레벨에서 컨테이너 단위로 분류하여 처리하는 컨테이너 단위 블록 IO 스케줄링 방법 및 장치를 제공한다.

과제의 해결 수단

- [0006] 본 발명의 일실시예에 따른 커널 레벨의 블록 IO 스케줄러가 수행하는 컨테이너 단위 블록 IO 처리 방법은 어플리케이션을 실행하는 유저 레벨의 프로세스로부터 수신된 블록 IO의 컨테이너 정보에 기초하여 상기 수신된 블록 IO를 복수의 대기열들 중 어느 하나의 대기열에 등록하는 단계; 상기 블록 IO가 등록된 복수의 대기열들의 리소스 제공을 위한 시간 할당량에 기초하여 어느 하나의 대기열을 선택하는 단계; 및 상기 선택된 대기열에 등록되어 있는 블록 IO를 커널에 전달하는 단계를 포함할 수 있다.
- [0007] 상기 등록하는 단계는 상기 수신된 블록 IO를 발생시킨 프로세스 정보를 이용하여 해당 프로세스가 포함된 컨테이너 정보를 식별하는 단계; 및 상기 식별된 컨테이너 정보에 대응하는 컨테이너 대기열에 상기 수신된 블록 IO를 등록하는 단계를 포함할 수 있다.
- [0008] 상기 등록하는 단계는 상기 식별된 컨테이너 정보에 대응하는 컨테이너 대기열이 존재하지 않는 경우, 신규 컨테이너 대기열을 생성하고, 생성된 신규 컨테이너 대기열에 상기 수신된 블록 IO를 등록할 수 있다.
- [0009] 상기 등록하는 단계는 상기 수신된 블록 IO를 발생시킨 프로세스 정보를 이용하여 해당 프로세스가 컨테이너에 포함되지 않은 것으로 식별되면, 일반 대기열에 상기 수신된 블록 IO를 등록할 수 있다.
- [0010] 상기 등록하는 단계는 동일한 컨테이너 내에 복수의 프로세스들이 포함된 경우, 상기 동일한 컨테이너 내의 프로세스들에서 발생된 블록 IO들을 하나의 컨테이너 대기열에 시간 순서에 따라 등록할 수 있다.
- [0011] 상기 선택하는 단계는 상기 블록 IO가 등록된 복수의 대기열들 각각에 대해 리소스 제공을 위한 누적 시간 할당량을 식별하는 단계; 및 상기 복수의 대기열들 중 상기 식별된 누적 시간 할당량이 가장 적은 대기열을 선택하는 단계를 포함할 수 있다.
- [0012] 상기 선택하는 단계는 상기 선택된 대기열 내에 블록 IO가 존재하지 않은 것으로 식별된 경우, 해당 대기열을 삭제하는 단계를 더 포함할 수 있다.
- [0013] 상기 커널에 전달된 블록 IO의 처리가 완료된 경우, 해당 블록 IO에 대응하는 대기열의 누적 시간 할당량을 수정하는 단계를 더 포함할 수 있다.
- [0014] 본 발명의 일실시예에 따른 컨테이너 단위 블록 IO 처리 방법을 수행하는 커널 레벨의 블록 IO 스케줄러는 프로세서를 포함하고, 상기 프로세서는 어플리케이션을 실행하는 유저 레벨의 프로세스로부터 수신된 블록 IO의 컨테이너 정보에 기초하여 상기 수신된 블록 IO를 복수의 대기열들 중 어느 하나의 대기열에 등록하고, 상기 블록 IO가 등록된 복수의 대기열들의 리소스 제공을 위한 시간 할당량에 기초하여 어느 하나의 대기열을 선택하며, 상기 선택된 대기열에 등록되어 있는 블록 IO를 커널에 전달할 수 있다.
- [0015] 상기 프로세서는 상기 수신된 블록 IO를 발생시킨 프로세스 정보를 이용하여 해당 프로세스가 포함된 컨테이너 정보를 식별하고, 상기 식별된 컨테이너 정보에 대응하는 컨테이너 대기열에 상기 수신된 블록 IO를 등록할 수 있다.
- [0016] 상기 프로세서는 상기 식별된 컨테이너 정보에 대응하는 컨테이너 대기열이 존재하지 않는 경우, 신규 컨테이너 대기열을 생성하고, 생성된 신규 컨테이너 대기열에 상기 수신된 블록 IO를 등록할 수 있다.
- [0017] 상기 프로세서는 상기 수신된 블록 IO를 발생시킨 프로세스 정보를 이용하여 해당 프로세스가 컨테이너에 포함되지 않은 것으로 식별되면, 일반 대기열에 상기 수신된 블록 IO를 등록할 수 있다.
- [0018] 상기 프로세서는 동일한 컨테이너 내에 복수의 프로세스들이 포함된 경우, 상기 동일한 컨테이너 내의 프로세스들에서 발생된 블록 IO들을 하나의 컨테이너 대기열에 시간 순서에 따라 등록할 수 있다.
- [0019] 상기 프로세서는 상기 블록 IO가 등록된 복수의 대기열들 각각에 대해 리소스 제공을 위한 누적 시간 할당량을 식별하고, 상기 복수의 대기열들 중 상기 식별된 누적 시간 할당량이 가장 적은 대기열을 선택할 수 있다.
- [0020] 상기 프로세서는 상기 선택된 대기열 내에 블록 IO가 존재하지 않은 것으로 식별된 경우, 해당 대기열을 삭제할

수 있다.

[0021] 상기 프로세서는 상기 커널에 전달된 블록 IO의 처리가 완료된 경우, 해당 블록 IO에 대응하는 대기열의 누적 시간 할당량을 수정할 수 있다.

발명의 효과

[0022] 본 발명의 일실시예에 의하면, 가상화 컨테이너 환경에서 다중 컨테이너 내 다중 프로세스로부터 발생하는 블록 IO 요청을 리눅스 커널 수준에서 컨테이너 단위로 분류하여 처리함으로써 컨테이너 단위로 최저 하드웨어 자원을 확보할 수 있다.

도면의 간단한 설명

[0023] 도 1은 본 발명의 일실시예에 따른 블록 IO 스케줄링 시스템을 나타낸 도면이다.

도 2는 본 발명의 일실시예에 따른 블록 IO 스케줄러의 구성도를 나타낸 도면이다.

도 3은 본 발명의 일실시예에 따른 블록 IO 스케줄러의 블록 IO 수신 과정을 플로우차트로 나타낸 도면이다.

도 4는 본 발명의 일실시예에 따른 블록 IO 스케줄러의 대기열 선택 과정을 플로우차트로 나타낸 도면이다.

도 5는 본 발명의 일실시예에 따른 대기열 정보 갱신 과정을 플로우차트로 나타낸 도면이다.

발명을 실시하기 위한 구체적인 내용

[0024] 이하, 본 발명의 실시예를 첨부된 도면을 참조하여 상세하게 설명한다.

[0025] 도 1은 본 발명의 일실시예에 따른 블록 IO 스케줄링 시스템을 나타낸 도면이다.

[0026] 도 1을 참고하면, 블록 IO 스케줄링 시스템(100)은 어플리케이션이 실행될 유저 레벨의 프로세스들이 가상화 컨테이너 환경에서 컨테이너 별로 구분되어 포함될 수 있다. 일례로, 유저 레벨의 프로세서 1과 프로세스 2는 컨테이너 A에 포함되고, 프로세스 3과 프로세스 4는 컨테이너 B에 포함되어 가상화 컨테이너 사이드(Virtual container side)의 어플리케이션이 실행될 수 있다. 이와는 달리 유저 레벨의 프로세스 5는 어느 컨테이너도 포함되지 않으며, 호스트 사이드(Host side)의 어플리케이션이 실행될 수 있다.

[0027] 이때, 커널 레벨의 블록 IO 스케줄러(110)는 커널 레벨에서 유저 레벨에 포함된 컨테이너 단위로 프로세스들의 블록 IO 요청을 분류하여 처리함으로써 컨테이너 단위로 공정하게 하드웨어 자원을 확보할 수 있는 방법을 제공할 수 있다.

[0029] 도 2는 본 발명의 일실시예에 따른 블록 IO 스케줄러의 구성도를 나타낸 도면이다.

[0030] 도 2를 참고하면, 커널 레벨의 블록 IO 스케줄러(110)는 블록 IO 수신부(111)를 통해 어플리케이션을 실행하는 유저 레벨의 프로세스로부터 수신된 블록 IO를 컨테이너 정보에 기초하여 복수의 대기열들 중 어느 하나의 대기열에 등록할 수 있다.

[0031] 보다 구체적으로 블록 IO 수신부(111)는 수신된 블록 IO를 발생시킨 프로세스 정보를 이용하여 해당 프로세스가 포함된 컨테이너 정보를 식별하고, 식별된 컨테이너 정보에 대응하는 컨테이너 대기열에 해당 블록 IO를 등록할 수 있다. 예를 들어, 수신된 블록 IO의 컨테이너 정보가 컨테이너 A로 식별된 경우, 블록 IO 수신부(111)는 컨테이너 A 대기열에 해당 블록 IO를 등록할 수 있다.

[0032] 이때, 블록 IO 수신부(111)는 식별된 컨테이너 정보에 대응하는 컨테이너 대기열이 존재하지 않는 경우, 해당 식별된 컨테이너 정보에 대응하는 신규 컨테이너 대기열을 생성하고, 생성된 신규 컨테이너 대기열에 수신된 블록 IO를 등록할 수 있다.

[0033] 이와는 달리 만약 수신된 블록 IO를 발생시킨 프로세스 정보를 이용하여 해당 프로세스가 컨테이너에 포함되지 있지 않다고 식별되면, 블록 IO 수신부(111)는 해당 블록 IO를 컨테이너 대기열이 아닌 별도의 일반 대기열에 등록할 수 있다.

[0034] 이후 블록 IO 스케줄러(110)는 대기열 선택부(112)를 통해 블록 IO가 등록된 복수의 대기열들 각각의 하드웨어 자원을 제공하기 위한 시간 할당량에 기초하여 복수의 대기열들 중 어느 하나의 대기열을 선택할 수 있다. 보다

구체적으로 대기열 선택부(112)는 복수의 대기열들 각각에 대해 하드웨어 자원을 제공하기 위한 누적 시간 할당량을 식별하고, 복수의 대기열들 중 식별된 누적 시간 할당량이 가장 적은 대기열을 선택할 수 있다.

- [0035] 이때, 선택된 대기열에 등록되어 있는 블록 IO는 커널의 장치 드라이버를 통해 하드웨어 수준의 장치, 즉, 물리 저장소(ex. 디스크 등)에 전달될 수 있으며, 커널을 통해 물리저장소에서 특정 작업(ex. 읽기 또는 쓰기 작업)을 처리할 수 있다. 여기서 장치 드라이버는 커널과 물리저장소의 중간에서 블록 IO 요청을 물리저장소에서 처리할 수 있도록 인터페이스(Interface) 역할을 하는 소프트웨어일 수 있다.
- [0036] 한편, 물리저장소는 일례로, 하드디스크 드라이브 또는 솔리드 스테이트 드라이브에 해당할 수 있으며, 본 발명의 블록 IO 스케줄러(110)에서 스케줄링된 블록 IO를 처리하는 실제 주체일 수 있다.
- [0037] 대기열 선택부(112)는 만약 선택된 대기열에 내에 블록 IO가 존재하지 않은 것으로 식별될 경우, 해당 선택된 대기열을 삭제한 후 대기열 선택 과정을 다시 수행할 수 있다.
- [0038] 대기열 정보 갱신부(113)는 커널에 전달된 블록 IO의 처리가 완료된 경우, 커널로부터 해당 블록 IO의 처리 결과를 수신하고, 해당 블록 IO에 대응하는 대기열의 누적 시간 할당량을 수정할 수 있다.
- [0040] 도 3은 본 발명의 일실시예에 따른 블록 IO 스케줄러의 블록 IO 수신 과정을 플로우차트로 나타낸 도면이다.
- [0041] 단계(310)에서, 블록 IO 스케줄러(110)는 어플리케이션을 실행하는 유저 레벨의 프로세스로부터 블록 IO를 수신할 수 있다. 이때, 블록 IO 스케줄러(110)는 다중 컨테이너 내 다중 프로세스로부터 블록 IO들을 수신할 수 있으며, 수신된 블록 IO 들은 수신 시간에 기초하여 차례대로 처리될 수 있다.
- [0042] 단계(312)에서, 블록 IO 스케줄러(110)는 수신된 블록 IO를 발생시킨 프로세스 정보로부터 해당 프로세스가 포함된 컨테이너 정보를 식별할 수 있다.
- [0043] 단계(314)에서, 블록 IO 스케줄러(110)는 수신된 블록 IO가 컨테이너 환경에서 동작 중인 프로세스에서 발생되었는지를 판단할 수 있다.
- [0044] 만약 수신된 블록 IO가 컨테이너 환경에서 동작 중인 프로세스에서 발생된 것이 아닌 것으로 판단되면, 블록 IO 스케줄러(110)는 단계(316)에서, 해당 수신된 블록 IO를 일반 대기열에 등록할 수 있다.
- [0045] 이와는 달리 수신된 블록 IO가 컨테이너 환경에서 동작 중인 프로세스에서 발생된 것으로 판단되면, 블록 IO 스케줄러(110)는 단계(318)에서, 해당 컨테이너에 대응하는 컨테이너 대기열을 대기열 트리에서 검색할 수 있다.
- [0046] 만약 단계(320)에서, 대기열 트리를 통해 해당 컨테이너에 대응하는 컨테이너 대기열이 검색되면, 단계(322)에서 블록 IO 스케줄러(110)는 검색된 컨테이너 대기열에 해당 수신된 블록 IO를 등록할 수 있다.
- [0047] 이와는 달리 대기열 트리를 통해 해당 컨테이너에 대응하는 컨테이너 대기열이 검색되지 않으면, 단계(324)에서, 블록 IO 스케줄러(110)는 해당 컨테이너에 대응하는 컨테이너 대기열을 신규 생성하고, 초기화를 수행할 수 있다. 이때, 초기화는 컨테이너 대기열이 서비스 받기 위해 대기열내 값인 누적 할당 시간, 최대 할당 가능 시간 및 현재 물리 저장소에 서비스 중인 블록 IO 개수를 0또는 특정 값으로 설정하는 것을 의미할 수 있다.
- [0048] 이후 블록 IO 스케줄러(110)는 단계(326)에서, 신규 생성된 컨테이너 대기열에 대응 블록 IO를 등록할 수 있다. 이때, 동일한 컨테이너 내에 복수의 프로세스들이 포함된 경우, 블록 IO 스케줄러(110)는 동일한 컨테이너 내의 프로세스들에서 발생된 블록 IO 들은 하나의 컨테이너 대기열에 시간 순서에 따라 등록할 수 있다.
- [0049] 이후 단계(328)에서, 블록 IO 스케줄러(110)는 커널에 정상 처리 결과를 리턴할 수 있다.
- [0051] 도 4는 본 발명의 일실시예에 따른 블록 IO 스케줄러의 대기열 선택 과정을 플로우차트로 나타낸 도면이다.
- [0052] 단계(410)에서, 블록 IO 스케줄러(110)는 커널로부터 다음 처리될 블록 IO의 선택 요청을 수신한 후 단계(412)에서, 대기열 트리를 검색하여 컨테이너 대기열이 존재하는지 여부를 판단할 수 있다.
- [0053] 만약 대기열 트리를 검색하여 컨테이너 대기열이 존재하지 않는 것으로 판단되면, 단계(414)에서, 블록 IO 스케줄러(110)는 일반 대기열에 등록된 블록 IO를 선택하고, 단계(426)에서, 선택된 일반 대기열의 블록 IO에 대한 대기를 해제한 후 커널에 리턴할 수 있다.

- [0054] 이와는 달리 대기열 트리를 검색하여 컨테이너 대기열이 존재하는 것으로 판단되면, 단계(416)에서, 블록 IO 스케줄러(110)는 대기열 트리를 순회하며 가장 낮은 누적 시간 할당량을 가지는 대기열을 선택할 수 있다. 이때, 블록 IO 스케줄러(110)는 대기열 트리에 포함된 컨테이너 대기열 및 일반 대기열을 모두 고려하여 가장 낮은 누적 시간 할당량을 가지는 대기열을 선택할 수 있다.
- [0055] 단계(418)에서, 선택된 대기열에 등록된 블록 IO가 존재하지 않는 것으로 판단되면, 블록 IO 스케줄러(110)는 단계(420)에서, 해당 선택된 대기열을 삭제하고, 대기열 선택 과정을 다시 수행할 수 있다.
- [0056] 이와는 달리 선택된 대기열에 등록된 블록 IO가 존재하는 것으로 판단되면, 블록 IO 스케줄러(110)는 단계(422)에서, 해당 선택된 대기열을 현재 서비스 대기열로 설정하고, 단계(424)에서, 서비스 시작 시간을 기록할 수 있다.
- [0057] 이후 단계(426)에서, 블록 IO 스케줄러(110)는 선택된 대기열의 블록 IO에 대한 대기를 해제한 후 커널에 리턴할 수 있다.
- [0059] 도 5는 본 발명의 일실시예에 따른 대기열 정보 갱신 과정을 플로우차트로 나타낸 도면이다.
- [0060] 단계(510)에서, 블록 IO 스케줄러(110)는 커널로부터 전달된 블록 IO의 처리 결과를 수신할 수 있다.
- [0061] 단계(512)에서, 블록 IO 스케줄러(110)는 대기열 트리를 검색하여 단계(514)에서, 처리 결과가 수신된 블록 IO가 소속된 컨테이너 대기열이 존재하는지 여부를 판단할 수 있다.
- [0062] 만약, 해당 블록 IO가 소속된 컨테이너 대기열이 존재하지 않는 것으로 판단되면, 블록 IO 스케줄러(110)는 단계(516)에서, 오류 시간 및 오류 내용을 로깅(Logging)한 후 단계(522)와 같이 함수를 종료할 수 있다.
- [0063] 이와는 달리 해당 블록 IO가 소속된 컨테이너 대기열이 존재하는 것으로 판단되면, 블록 IO 스케줄러(110)는 단계(518)에서 해당 컨테이너 대기열에 기록된 블록 IO 처리 시작 시간과 종료 시간을 이용하여 누적 시간 할당량을 수정할 수 있다.
- [0064] 단계(520)에서, 블록 IO 스케줄러(110)는 누적 시간 할당량이 수정된 컨테이너 대기열을 현재 서비스 대기열에서 해제하고, 단계(522)와 같이 함수를 종료할 수 있다.
- [0066] 한편, 본 발명에 따른 방법은 컴퓨터에서 실행될 수 있는 프로그램으로 작성되어 마그네틱 저장매체, 광학적 판독매체, 디지털 저장매체 등 다양한 기록 매체로도 구현될 수 있다.
- [0067] 본 명세서에 설명된 각종 기술들의 구현들은 디지털 전자 회로조직으로, 또는 컴퓨터 하드웨어, 펌웨어, 소프트웨어로, 또는 그들의 조합들로 구현될 수 있다. 구현들은 데이터 처리 장치, 예를 들어 프로그램가능 프로세서, 컴퓨터, 또는 다수의 컴퓨터들의 동작에 의한 처리를 위해, 또는 이 동작을 제어하기 위해, 컴퓨터 프로그램 제품, 즉 정보 캐리어, 예를 들어 기계 판독가능 저장 장치(컴퓨터 판독가능 매체) 또는 전파 신호에서 유형적으로 구체화된 컴퓨터 프로그램으로서 구현될 수 있다. 상술한 컴퓨터 프로그램(들)과 같은 컴퓨터 프로그램은 컴파일된 또는 인터프리트된 언어들을 포함하는 임의의 형태의 프로그래밍 언어로 기록될 수 있고, 독립형 프로그램으로서 또는 모듈, 구성요소, 서브루틴, 또는 컴퓨팅 환경에서의 사용에 적절한 다른 유닛으로서 포함하는 임의의 형태로 전개될 수 있다. 컴퓨터 프로그램은 하나의 사이트에서 하나의 컴퓨터 또는 다수의 컴퓨터들 상에서 처리되도록 또는 다수의 사이트들에 걸쳐 분배되고 통신 네트워크에 의해 상호 연결되도록 전개될 수 있다.
- [0068] 컴퓨터 프로그램의 처리에 적절한 프로세서들은 예로서, 범용 및 특수 목적 마이크로프로세서들 둘 다, 및 임의의 종류의 디지털 컴퓨터의 임의의 하나 이상의 프로세서들을 포함한다. 일반적으로, 프로세서는 판독 전용 메모리 또는 랜덤 액세스 메모리 또는 둘 다로부터 명령어들 및 데이터를 수신할 것이다. 컴퓨터의 요소들은 명령어들을 실행하는 적어도 하나의 프로세서 및 명령어들 및 데이터를 저장하는 하나 이상의 메모리 장치들을 포함할 수 있다. 일반적으로, 컴퓨터는 데이터를 저장하는 하나 이상의 대량 저장 장치들, 예를 들어 자기, 자기-광 디스크들, 또는 광 디스크들을 포함할 수 있거나, 이것들로부터 데이터를 수신하거나 이것들에 데이터를 송신하거나 또는 양쪽으로 되도록 결합될 수도 있다. 컴퓨터 프로그램 명령어들 및 데이터를 구체화하는데 적절한 정보 캐리어들은 예로서 반도체 메모리 장치들, 예를 들어, 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(Magnetic Media), CD-ROM(Compact Disk Read Only Memory), DVD(Digital Video Disk)와 같은 광 기록 매체(Optical Media), 플롭티컬 디스크(Floptical Disk)와 같은 자기-광 매체(Magneto-Optical Media), 롬

(ROM, Read Only Memory), 램(RAM, Random Access Memory), 플래시 메모리, EPROM(Erasable Programmable ROM), EEPROM(Electrically Erasable Programmable ROM) 등을 포함한다. 프로세서 및 메모리는 특수 목적 논리 회로조직에 의해 보충되거나, 이에 포함될 수 있다.

[0069] 또한, 컴퓨터 판독가능 매체는 컴퓨터에 의해 액세스될 수 있는 임의의 가용매체일 수 있고, 컴퓨터 저장매체 및 전송매체를 모두 포함할 수 있다.

[0070] 본 명세서는 다수의 특정한 구현물의 세부사항들을 포함하지만, 이들은 어떠한 발명이나 청구 가능한 것의 범위에 대해서도 제한적인 것으로서 이해되어서는 안되며, 오히려 특정한 발명의 특정한 실시형태에 특유할 수 있는 특징들에 대한 설명으로서 이해되어야 한다. 개별적인 실시형태의 문맥에서 본 명세서에 기술된 특정한 특징들은 단일 실시형태에서 조합하여 구현될 수도 있다. 반대로, 단일 실시형태의 문맥에서 기술한 다양한 특징들 역시 개별적으로 혹은 어떠한 적절한 하위 조합으로도 복수의 실시형태에서 구현 가능하다. 나아가, 특징들이 특정한 조합으로 동작하고 초기에 그와 같이 청구된 바와 같이 묘사될 수 있지만, 청구된 조합으로부터의 하나 이상의 특징들은 일부 경우에 그 조합으로부터 배제될 수 있으며, 그 청구된 조합은 하위 조합이나 하위 조합의 변형물로 변경될 수 있다.

[0071] 마찬가지로, 특정한 순서로 도면에서 동작들을 묘사하고 있지만, 이는 바람직한 결과를 얻기 위하여 도시된 그 특정한 순서나 순차적인 순서대로 그러한 동작들을 수행하여야 한다거나 모든 도시된 동작들이 수행되어야 하는 것으로 이해되어서는 안 된다. 특정한 경우, 멀티태스킹과 병렬 프로세싱이 유리할 수 있다. 또한, 상술한 실시형태의 다양한 장치 컴포넌트의 분리는 그러한 분리를 모든 실시형태에서 요구하는 것으로 이해되어서는 안되며, 설명한 프로그램 컴포넌트와 장치들은 일반적으로 단일의 소프트웨어 제품으로 함께 통합되거나 다중 소프트웨어 제품에 패키징 될 수 있다는 점을 이해하여야 한다.

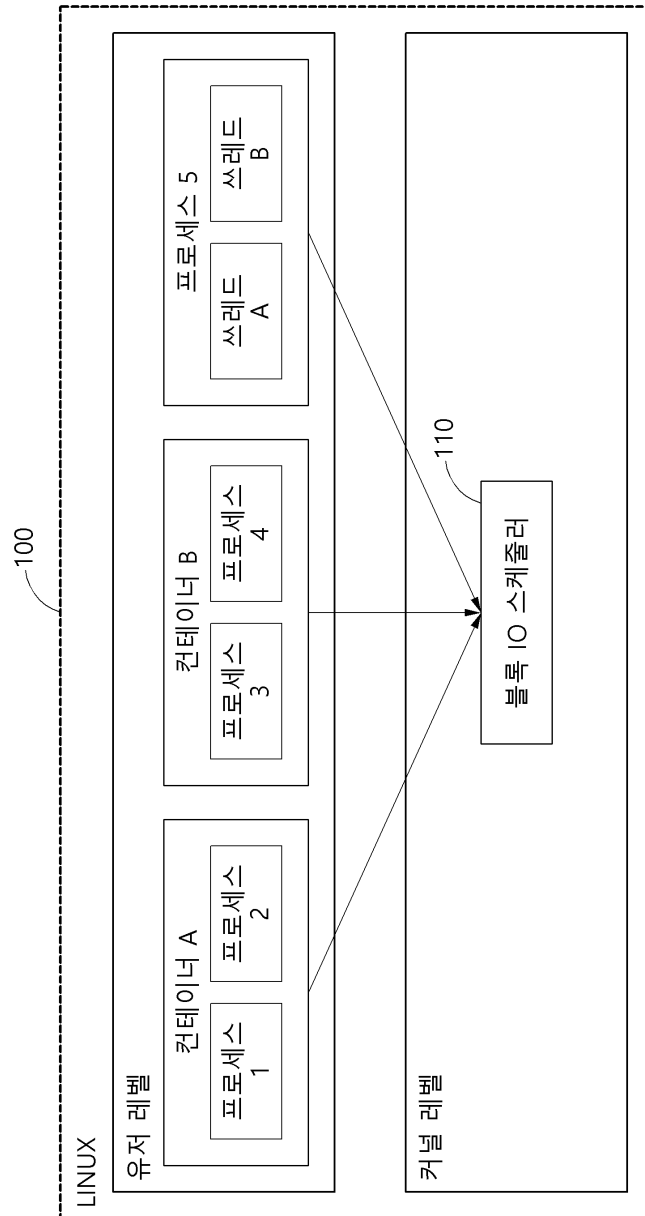
[0072] 한편, 본 명세서와 도면에 개시된 본 발명의 실시 예들은 이해를 돕기 위해 특정 예를 제시한 것에 지나지 않으며, 본 발명의 범위를 한정하고자 하는 것은 아니다. 여기에 개시된 실시 예들 이외에도 본 발명의 기술적 사상에 바탕을 둔 다른 변형 예들이 실시 가능하다는 것은, 본 발명이 속하는 기술분야에서 통상의 지식을 가진 자에게 자명한 것이다.

부호의 설명

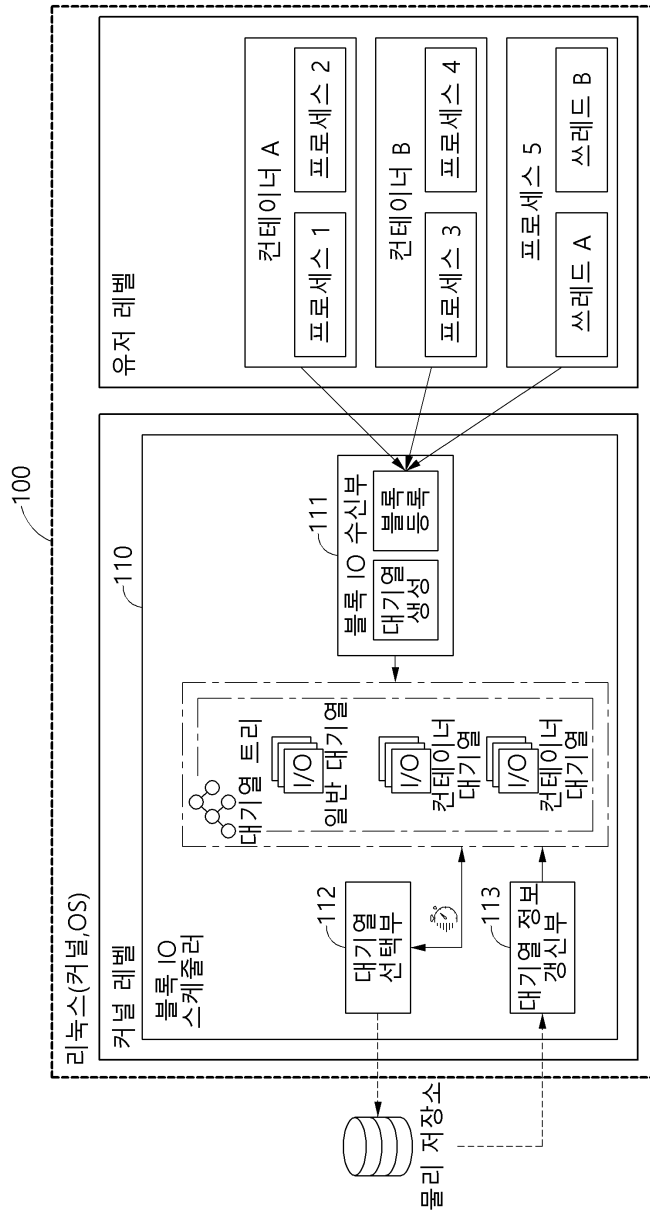
- [0073]
- 100 : 블록 IO 스케줄링 시스템
 - 110 : 블록 IO 스케줄러
 - 111 : 블록 IO 수신부
 - 112 : 대기열 선택부
 - 113 : 대기열 정보 갱신부

도면

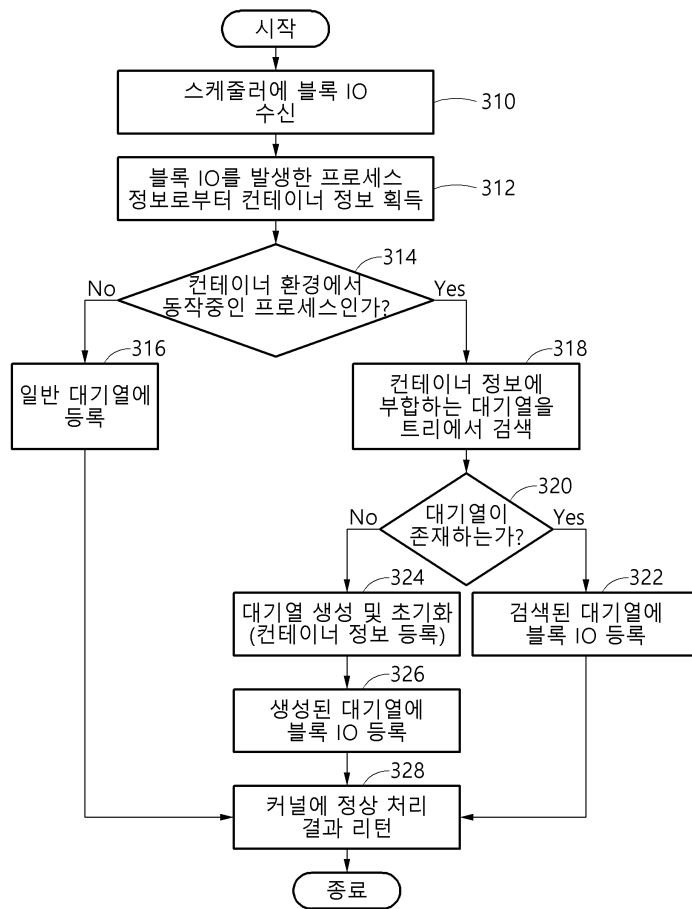
도면1



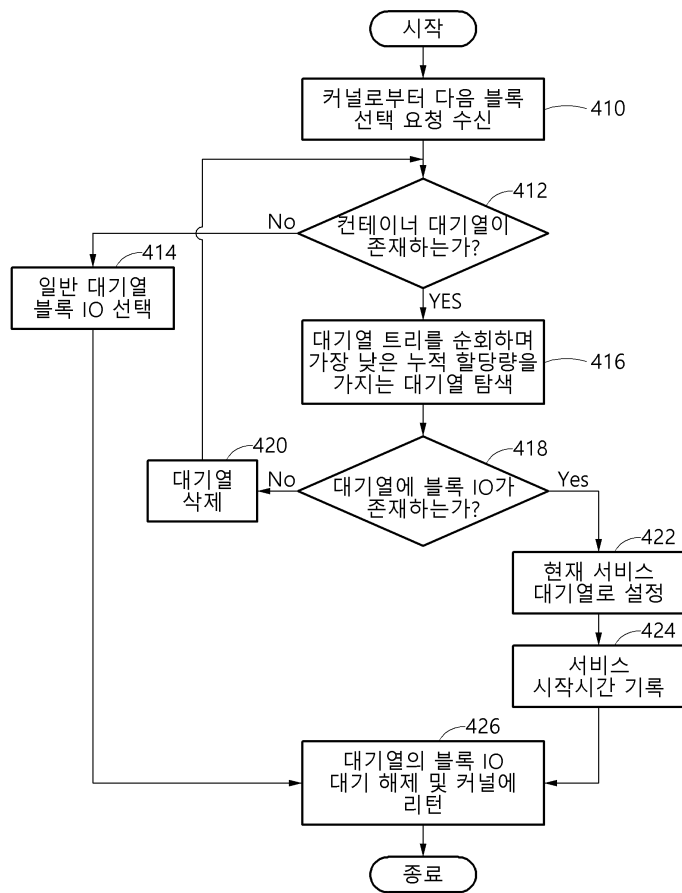
도면2



도면3



도면4



도면5

