



(19) 대한민국특허청(KR)
(12) 등록특허공보(B1)

(45) 공고일자 2018년02월02일
(11) 등록번호 10-1825307
(24) 등록일자 2018년01월29일

- (51) 국제특허분류(Int. Cl.)
H04N 21/231 (2011.01) H04N 21/433 (2011.01)
- (52) CPC특허분류
H04N 21/23106 (2013.01)
H04N 21/2183 (2013.01)
- (21) 출원번호 10-2016-0122864
- (22) 출원일자 2016년09월26일
심사청구일자 2016년09월26일
- (56) 선행기술조사문헌
KR100866833 B1*
- 김선우 외 1명, “멀티미디어 스트리밍 서버를 위한 SSD 와 HDD 혼합 스토리지 구조에서의 캐시 관리 정책”, 한국통신학회 종합 학술 발표회 논문집 (동계) 2012, pp.818-819 (2012.02.08.)*
- 이종득, “무선 스트리밍 QoS를 위한 이득 기반 세그먼트 캐싱”, 한국항행학회, <한국항행학회논문지> 16권3호 (2012), pp.463-470 (2012.06.30.)*
- *는 심사관에 의하여 인용된 문헌

- (73) 특허권자
인하대학교 산학협력단
- 인천광역시 남구 인하로 100, 인하대학교 (용현동)
- (72) 발명자
송민석
- 서울특별시 서초구 남부순환로 2311-12, 107동 402호(방배동, 래미안방배아트힐)
- 이정우
- 인천광역시 남동구 문화서로65번길 28-13, 1층 (구월동)
- (74) 대리인
양성보

전체 청구항 수 : 총 5 항

심사관 : 김광식

(54) 발명의 명칭 DRAM과 SSD의 혼합 사용을 통한 비디오 서버의 캐시 관리 방법

(57) 요약

본 발명은 디스크 기반 비디오 서버의 캐시로 SSD와 DRAM을 혼합하여 사용하고 효율적으로 관리함으로써 비디오 서버의 디스크 대역폭 이용률을 줄일 수 있는 방법을 제안한다. 일 실시예에 따른 캐시 관리 방법은, 비디오 영상의 프레임들을 복수의 세그먼트로 분리하고, 상기 분리된 복수의 세그먼트를 인기도에 기반하여 SSD에 캐시될 상기 비디오 영상의 초기 세그먼트를 선택하는 단계; 및 상기 SSD의 크기와 대역폭에 기초하여 DRAM에 캐시될 인터벌을 결정하는 단계를 포함할 수 있다.

대표도 - 도1

```

1: Temporary variables: i, m and k;
2: for i = 1 to N' do
3:   for m = 1 to Sssd do
4:     Ci,mssd ← -∞;
5:     Ci,mssd ← 0;
6:   end for
7: end for
8: for i = 1 to Sssd do
9:   if m ≥ Si(i) then
10:    Ci,mssd ← maxk ∈ [Si(i) ≤ m] Gi(k);
11:    Ci,mssd ← arg maxk ∈ [Si(i) ≤ m] Gi(k);
12:   end if
13: end for
14: for i = 2 to N' do
15:   for m = 1 to Sssd do
16:     Ci,mssd ← Ci,m-1ssd;
17:     for k = 1 to N2 do
18:       if m - Si(k) > 0 then
19:         if Ci,mssd > Ci-1,m-Si(k)} + Gi(k) then
20:           Ci,mssd ← Ci-1,m-Si(k)} + Gi(k);
21:           Ci,mssd ← k;
22:         end if
23:       end if
24:     end for
25:   end for
26: end for
27: i ← N', m ← Sssd;
28: while i > 0 do
29:   Fi ← Ci,mssd;
30:   m ← m - Si(Fi);
31:   i ← i - 1;
32: end while

```

(52) CPC특허분류

H04N 21/23103 (2013.01)

H04N 21/433 (2013.01)

H04N 21/4331 (2013.01)

이 발명을 지원한 국가연구개발사업

과제고유번호 NRF-2015R1D1A1A01058646

부처명 교육부

연구관리전문기관 한국연구재단

연구사업명 이공분야기초연구사업 (기본연구지원사업)

연구과제명 동적 스트리밍 환경에서 전력 관리 기법에 관한 연구

기 여 율 7/10

주관기관 인하대학교 산학협력단

연구기간 2015.11.01 ~ 2016.10.30

이 발명을 지원한 국가연구개발사업

과제고유번호 HI14C0765

부처명 보건복지부

연구관리전문기관 보건산업진흥원

연구사업명 의료기기기술개발사업

연구과제명 발달장애아동을 위한 Tangible UX 기반의 완구형 교육시스템 개발

기 여 율 3/10

주관기관 인하대학교 산학협력단

연구기간 2016.05.01 ~ 2017.04.30

명세서

청구범위

청구항 1

캐시 관리 방법에 있어서,

비디오 영상의 프레임들을 복수의 세그먼트로 분리하고, 상기 분리된 복수의 세그먼트를 인기도에 기반하여 SSD에 캐시될 상기 비디오 영상의 초기 세그먼트를 선택하는 단계; 및

상기 SSD의 크기와 대역폭에 기초하여 DRAM에 캐시될 인터벌을 결정하는 단계

를 포함하고,

상기 SSD에 캐시될 상기 비디오 영상의 초기 세그먼트를 선택하는 단계는,

상기 SSD에 저장될 초기 세그먼트를 결정하기 위하여 SSD 프레임의 세그먼트 결정법을 사용하고, 첫번째 비디오부터 i (i 는 자연수) 비디오까지 전체 비디오에 대하여 상기 SSD에 기 설정된 크기로 저장하였을 때 캐시로 획득할 수 있는 최대 이익에 대한 변수를 점화식을 설립하여 계산하고, 상기 변수를 결정하여 상기 비디오 및 상기 SSD의 용량에 기반하여 역으로 테이블을 추적함으로써 각각의 비디오에 저장될 초기 세그먼트를 도출하는 단계

를 포함하는 캐시 관리 방법.

청구항 2

제1항에 있어서,

상기 DRAM에 캐시될 인터벌을 결정하는 단계는,

동일한 비디오에 대하여 연속된 복수 개의 요청이 주어짐에 따라 상기 복수 개의 요청 중 이전 요청에 대한 데이터가 캐시에 남아있는 경우, 상기 요청이 상기 캐시를 통하여 서비스되도록 하고,

상기 인터벌은,

단일의 영상에 대한 적어도 두 개 이상의 연속된 요청 사이의 거리를 나타내는

캐시 관리 방법.

청구항 3

제2항에 있어서,

상기 DRAM에 캐시될 인터벌을 결정하는 단계는,

상기 요청에 대하여 인터벌 크기의 내림차순으로 정렬된 순서로 상기 SSD의 대역폭의 한계까지 서비스하는, 캐시 관리 방법.

청구항 4

제3항에 있어서,

상기 DRAM에 캐시될 인터벌을 결정하는 단계는,

기 설정된 기준 이상의 인터벌의 요청들이 상기 SSD로부터 지원됨으로써 기 설정된 이하의 인터벌의 요청이 상기 DRAM에 캐시되도록 하는, 캐시 관리 방법.

청구항 5

삭제

청구항 6

삭제

청구항 7

캐시 관리 시스템에 있어서,

비디오 영상의 프레임들을 복수의 세그먼트로 분리하고, 상기 분리된 복수의 세그먼트를 인기도에 기반하여 SSD에 캐시될 상기 비디오 영상의 초기 세그먼트를 선택하는 선택부; 및

상기 SSD의 크기와 대역폭에 기초하여 DRAM에 캐시될 인터벌을 결정함으로써 디스크 대역폭의 이용률을 관리하는 결정부

를 포함하고,

상기 선택부는,

상기 SSD에 저장될 초기 세그먼트를 결정하기 위하여 SSD 프레임스 세그먼트 결정법을 사용하고, 첫번째 비디오부터 i (i 는 자연수) 비디오까지 전체 비디오에 대하여 상기 SSD에 기 설정된 크기로 저장하였을 때 캐시로 획득할 수 있는 최대 이익에 대한 변수를 점화식을 설립하여 계산하고, 상기 변수를 결정하여 상기 비디오 및 상기 SSD의 용량에 기반하여 역으로 테이블을 추적함으로써 각각의 비디오에 저장될 초기 세그먼트를 도출하는

캐시 관리 시스템.

발명의 설명

기술 분야

[0001] 아래의 설명은 서버의 캐시 관리 기술에 관한 것으로, DRAM과 SSD의 혼합 사용을 통한 비디오 서버의 캐시 관리 방법 및 시스템에 관한 것이다.

배경 기술

[0002] 최근 멀티미디어 및 네트워크 기술의 발전은 인터넷을 통해 콘텐츠의 전송 및 서비스를 대중화시켰다. 그러나 높은 스토리지와 대역폭을 필요로 하기 때문에, 비디오의 상당량을 유저에게 전달하는 것은 큰 서버 구조를 필요로 한다. 일례로, YouTube 서버는 매일 4000만개의 영상들을 제공하고 있으며, 그 총량은 200TB에 달한다.

[0003] 비디오 서비스를 위한 서버는 일반적으로 디스크 배열 기반으로 이루어져 있다. 비록 하드 디스크의 스토리지 용량은 꾸준히 증가하였지만, 회전시간과 탐색시간에 의한 대역폭은 여전히 한계가 있다. 디스크 대역폭은 비디오 서비스 성능의 가장 중요한 결정인자 중 하나인 동시간 요청의 수를 제한한다.

[0004] 플래시 메모리 기반의 SSD는 적은 전력 소모와 낮은 접근 지연시간, 그리고 충격에 강한 특성 등의 많은 기술적인 장점을 가지고 있어 SSD를 서버의 저장매체로 사용하려는 여러 연구들이 진행되어 왔다. 특히, SSD는 작은 읽기 지연시간과 큰 임의 읽기 처리량을 가지고 있어 동시에 많은 양의 스트림을 읽어야 하는 비디오 서버에 좋은 성능을 나타내었다. 그러나 SSD의 높은 가격 때문에 큰 용량이 요구되는 비디오 서버에서 단독으로 사용되는 것은 거의 불가능하다. 그렇기 때문에 SSD를 비디오 서버의 캐시로 사용하는 것이 합리적이다.

[0005] 일반적으로, 디스크 버퍼 캐시는 DRAM으로 구성되고 해당 캐시는 디스크 요청을 줄이기 위해 유지된다. 연속된 데이터 요청 쌍의 인터벌들에 대하여 인터벌 사이즈가 작은 요청을 캐시하는 인터벌 캐시는 구현 비용이 낮고 성능이 좋아 비디오 서버의 잘 알려진 버퍼 캐시 방법이다.

[0006] SSD는 쓰기 횟수가 SSD 수명에 상당한 영향을 주기 때문에 인터벌 캐시에는 적절하지 않다. 이에 따라 SSD를 비디오 서버의 캐시로 활용하고, DRAM을 통하여 버퍼 캐시를 이루는 방법이 제안될 필요가 있다.

발명의 내용

해결하려는 과제

- [0007] 본 발명은 SSD와 DRAM을 혼합하여 디스크 기반 비디오 서버의 캐시로 사용하고 관리하는 방법을 제안한다. 더욱 상세하게는, 디스크의 대역폭 이용률을 최소화하기 위해 비디오 내부에서 변화하는 인기도를 기반으로 초기 세그먼트를 SSD에 저장할 것인지 결정하고, 연속된 요청 쌍 사이에서 DRAM에 캐시 될 인터벌을 결정함으로써 디스크 대역폭 사용을 효율적으로 관리할 수 있도록 제공한다.

과제의 해결 수단

- [0008] 일 실시예에 따르면, 캐시 관리 방법은, 비디오 영상의 프레임들을 복수의 세그먼트로 분리하고, 상기 분리된 복수의 세그먼트를 인기도에 기반하여 SSD에 캐시될 상기 비디오 영상의 초기 세그먼트를 선택하는 단계; 및 상기 SSD의 크기와 대역폭에 기초하여 DRAM에 캐시될 인터벌을 결정하는 단계를 포함할 수 있다.
- [0009] 상기 DRAM에 캐시될 인터벌을 결정하는 단계는, 동일한 비디오에 대하여 연속된 복수 개의 요청이 주어짐에 따라 상기 복수 개의 요청 중 이전 요청에 대한 데이터가 캐시에 남아있는 경우, 상기 요청이 상기 캐시를 통하여 서비스되도록 하고, 상기 인터벌은, 단일의 영상에 대한 적어도 두 개 이상의 연속된 요청 사이의 거리를 나타낼 수 있다.
- [0010] 상기 DRAM에 캐시될 인터벌을 결정하는 단계는, 상기 요청에 대하여 인터벌 크기의 내림차순으로 정렬된 순서로 상기 SSD의 대역폭의 한계까지 서비스할 수 있다.
- [0011] 상기 DRAM에 캐시될 인터벌을 결정하는 단계는, 기 설정된 기준 이상의 인터벌의 요청들이 상기 SSD로부터 지원됨으로써 기 설정된 이하의 인터벌의 요청이 상기 DRAM에 캐시되도록 할 수 있다.
- [0012] 상기 SSD에 캐시될 상기 비디오 영상의 초기 세그먼트를 선택하는 단계는, 상기 SSD에 저장될 초기 세그먼트를 결정하기 위하여 SSD 프레임워크 세그먼트 결정법을 사용할 수 있다.
- [0013] 상기 SSD에 캐시될 상기 비디오 영상의 초기 세그먼트를 선택하는 단계는, 첫번째 비디오부터 i (i 는 자연수) 비디오까지 전체 비디오에 대하여 상기 SSD에 기 설정된 크기로 저장하였을 때 캐시로 획득할 수 있는 최대 이익에 대한 변수를 점화식을 설립하여 계산하고, 상기 변수를 결정하여 상기 비디오 및 상기 SSD의 용량에 기반하여 역으로 테이블을 추적함으로써 각각의 비디오에 저장될 초기 세그먼트를 도출할 수 있다.
- [0014] 일 실시예에 따르면, 캐시 관리 시스템은, 비디오 영상의 프레임들을 복수의 세그먼트로 분리하고, 상기 분리된 복수의 세그먼트를 인기도에 기반하여 SSD에 캐시될 상기 비디오 영상의 초기 세그먼트를 선택하는 선택부; 및 상기 SSD의 크기와 대역폭에 기초하여 DRAM에 캐시될 인터벌을 결정함으로써 디스크 대역폭의 이용률을 관리하는 결정부를 포함할 수 있다.

발명의 효과

- [0015] 본 발명은 다중 디스크 배열 기반의 비디오 서버에서 SSD와 DRAM을 혼합하여 사용하는 캐시 방법을 제공할 수 있다. 더욱 상세하게는, 비디오 세그먼트의 인기도를 분석하여 동적 프로그래밍 알고리즘을 수행함으로써 캐시 효과를 최대화 할 수 있는 세그먼트를 SSD에 저장하고, DRAM 캐시를 통한 인터벌 캐싱 알고리즘을 통합하여 SSD 대역폭을 효과적으로 사용할 수 있다.
- [0016] 또한, SSD와 DRAM의 활용도를 높이고, 디스크의 대역폭 이용률을 효율적으로 사용할 수 있다.

도면의 간단한 설명

- [0017] 도 1 및 도 2는 일 실시예에 따른 캐시 관리 방법을 설명하기 위한 수도코드를 나타낸 것이다.
- 도 3은 일 실시예에 따른 캐시 관리 시스템의 구성을 설명하기 위한 블록도이다.
- 도 4는 일 실시예에 따른 캐시 관리 방법을 설명하기 위한 도면이다.

도 5 내지 도 7은 디스크 대역폭 사용의 절약율을 비교하는 그래프를 나타낸 것이다.

발명을 실시하기 위한 구체적인 내용

[0018] 이하, 실시예를 첨부한 도면을 참조하여 상세히 설명한다.

[0019] 도 1 및 도 2는 일 실시예에 따른 캐시 관리 방법을 설명하기 위한 수도코드를 나타낸 것이다.

[0020] 아래의 설명에서는 비디오 서버의 요청에 대하여 SSD와 DRAM을 사용하여 디스크 대역폭을 효과적으로 관리하며 처리하는 방법을 제안하기로 한다.

[0021] 비디오 서버의 총 단위시간당 요청수를 λ , 클라이언트로부터 요청되는 비디오의 수를 N^v , b_i 를 비디오 i ($i=1, \dots, N^v$)의 비트율, $\lambda_{i,j}$ 를 비디오 i 의 j 세그먼트의 단위시간당 요청수를 의미할 수 있다. 이때, 비디오 세그먼트의 인기도는 지프 분포를 따른다고 가정한다. 그러므로 세그먼트의 인기도는 $\lambda_{i,1} > \dots > \lambda_{i,n}$ 일 수 있다. $p_{i,j}$ 를 비디오 i 의 j 세그먼트의 요청 확률이라고 할 때, $(\lambda_{i,j} = p_{i,j} \lambda)$ 이 성립한다.

[0022] 각 세그먼트는 u^{MB} 의 데이터를 포함할 수 있다. 이때, 비디오 i 의 한 세그먼트당 시간은 $\frac{u}{b_i}$ 가 되며, L_i 로 대체하여 설명하기로 한다. $G_i(k)$ 는 캐시를 통하여 획득되는 이익이다. 캐시를 통하여 획득되는 이익은 비디오 i 의 초기 k 세그먼트를 SSD에 저장하였을 때, 캐시를 통해 디스크 대역폭의 감소량이다.

[0023] Littlewell Law에 의하면, j 세그먼트의 L_i 초 동안 동시간 요청의 수는 $\lambda_{i,j} L_i$ 로 표현될 수 있다. $\lambda_{i,j} = p_{i,j} \lambda$ 이기 때문에, 포함된 세그먼트 동안 획득된 대역폭의 $G_i(k)$ 는 비디오의 인기도($\sum_{j=1}^k p_{i,j}$)에 의존된다. 이에 대한 수학적식은 수학적식 1과 같다.

[0024] 수학적식 1:

$$G_i(k) = \lambda b_i L_i \sum_{j=1}^k p_{i,j}.$$

[0025] S^{ssd} 를 u^{MB} 로 나누어진 SSD의 크기라고 하자. 비디오 i 의 초기 k 세그먼트를 저장하였을 때의 크기를 $S_i(k)$ 라고 할 때, $S_i(k) = k b_i L_i$ 가 된다. 비디오 i 에 대하여 SSD에 저장될 초기 세그먼트를 F_i ($F_i=0, \dots, N_i^s$)라고 할 때, F_i 를 결정하기 위해 프레픽스 세그먼트 결정법(prefix selection problem: PSP)을 수학적식 2의 조건에서 수학적식 3을 최대화하는 것으로 공식화할 수 있다.

[0027] 수학적식 2:

$$\sum_{i=1}^{N^v} S_i(F_i) \leq S^{ssd},$$

$$F_i = 0, \dots, N_i^s$$

[0028]

[0029] 수학적식 3:

$$\sum_{i=1}^{N^v} G_i(F_i)$$

[0030]

[0031] 프레픽스 세그먼트 결정법은 다중선택 배낭(multiple-choice-knapsack) 문제의 일종이다. 도 1을 참고하면, SSD 프레픽스 세그먼트 결정 알고리즘(SSD prefix caching (SPC) algorithm)을 다음과 같이 제안할 수 있으며, 3개의 단계로 구성될 수 있다.

[0032] $C_{i,m}^{gain}$ 는 첫번째 비디오부터 i 번째 비디오까지 모든 비디오에 대하여, SSD에 m (단위 uMB) 크기를 저장하였을 때, 캐시로 획득할 수 있는 최대 이익을 의미할 수 있다. 이때, $\forall i, m, C_{i,m}^{gain}$ 를 모두 0으로 초기화한다.

[0033] $C_{i,m}^{gain}$ 는 다음과 같은 점화식으로 계산될 수 있다.

$$C_{i,m}^{gain} = \max G_i(k) \{k | k \in S_1(k) \leq m\}$$

[0034]

[0035] 만약, $C_{i-1,m-S_i(k)}^{gain} > 0$ 이면, $C_{i,m}^{gain}, (i=2, \dots, N^v)$ 는 다음과 같은 점화식으로 갱신될 수 있다.

$$C_{i,m}^{gain} = \max \{ \max (C_{i,m-1}^{gain}, C_{i-1,m-S_i(k)}^{gain} + G_i(k)) \} \{k | k = 1, \dots, N_i^s\}$$

[0036]

[0037] $C_{N^v, S^{ssd}}^{gain}$ 는 SSD 캐시를 통하여 최대로 획득하게 되는 이익을 의미할 수 있다. 비디오 N^v , SSD 용량 S^{ssd} 부터 시작하여 역으로 테이블을 추적하며 각 비디오 i 에 저장될 초기 세그먼트 F_i 를 탐색할 수 있다.

이를 위하여 변수 $C_{i,k}^{prefix}$ 를 결정할 수 있다. $C_{i,k}^{prefix}$ 는 $C_{N^v, S^{ssd}}^{gain}$ 를 획득할 수 있도록 하는 각 비디오 i 에 대하여 SSD의 용량이 k 일 때, SSD에 저장되는 초기 세그먼트의 수이다.

[0038] 각 비디오에 대하여 비디오 전체를 캐시하거나 하지 않는지를 선택하는 문제로 수정될 때, SSD 프레픽스 세그먼트 결정 알고리즘은 상기 문제를 해결할 수 있다. 루프 하나를 제외하여 각 영상을 세그먼트 크기별로 나누는 것이 아닌 영상 전체에 대한 크기로 계산함으로써 해결할 수 있다. 도 1을 참고하면, 17번째 줄의 for 루프를 제외함으로써 해결할 수 있다. 이러한 문제점을 해결한 알고리즘을 SEVC(SSD entire video caching)이라 명명하기로 한다.

[0039] 하루와 같이 일정시간 동안, 비디오 i 의 j 세그먼트에 대한 요청률 $P_{i,j}$ 는 각 비디오의 인기를 관촬 및 분석

함으로써 예측될 수 있다. 예를 들면, VOD 서비스에서 인기도 정보는 장기간 동안 안정적으로 보이기 때문에 SSD 프레픽스 세그먼트 결정을 통한 구성은 유효할 것으로 예상된다. 비디오 인기도에 기 설정된 기준 이상의 변화가 생길 경우, SSD 프레픽스 세그먼트 결정은 어떤 비디오 세그먼트가 SSD에 교체될 것인지 선택하게 된다.

[0040] 도 2를 참고하면, DRAM 캐싱 알고리즘을 나타낸 것이다. DRAM 캐시 알고리즘은 언제든지 요청들이 DRAM에 캐시될 것인지를 결정한다. 이에 따라 VOD 시스템과 낮은 복잡도와 좋은 성능이 증명된 인터벌 캐싱을 기반으로 하는 DRAM 캐싱 알고리즘을 제안하고자 한다. 이때, 인터벌 캐싱은 같은 비디오에 대하여 연속된 두 개의 요청이 주어졌을 경우, 둘 중 이전 요청의 데이터가 캐시에 남아있는 경우, 나중 요청은 캐시를 통해 서비스될 수 있도록 허락한다.

[0041] 인터벌은 하나의 영상에 대하여 두 개의 연속된 요청 사이의 거리를 나타낸다. $r(R)$ 과 $I(R)$ 을 각각의 요청 R에 대한 비트율과 인터벌 크기라 하고, S^{dram} 을 DRAM 캐시의 크기라 하자. 만약, 같은 비디오 세그먼트에 대하여 이전 요청이 존재하지 않았다면, $I(R)$ 은 ∞ 가 된다. 인터벌 캐싱은 적은 양의 메모리를 사용하면서도 캐시를 통해 서비스되는 요청들의 수를 최대화하고자 하므로, 모든 인터벌에 대하여 크기 순으로 정렬되었을 때, $\sum I(R) \leq S^{dram}$ 을 만족하는 동안 작은 크기의 캐시들 순으로 캐시될 수 있다.

[0042] 본 발명은 SSD 대역폭을 고려한 상황에서 인터벌 캐싱으로 확장시키는 것을 제안한다. B^{ssd} 는 SSD의 최대 대역폭, A^{ssd} 는 SSD에 저장된 세그먼트들의 배열, A^{disk} 는 오직 디스크에만 저장된 비디오 세그먼트들의 배열이라고 하자. 만약, SSD의 대역폭이 충분한 상황이라면, 모든 요청 $R(R \in A^{ssd})$ 은 SSD를 통해 지원될 수 있다. SSD의 대역폭이 충분하지 않을 상황일 경우, 몇몇 요청들은 A^{disk} 에서 지원되어야 한다.

[0043] 일 실시예에 따른 DRAM 캐싱 알고리즘은 요청들에 대하여 인터벌 크기의 내림차순 정렬된 순서로, SSD의 대역폭의 한계까지 서비스할 수 있다. 이를 통해 기 설정된 기준 이상의 인터벌의 요청들이 SSD로부터 지원됨으로써 기 설정된 기준 이하의 인터벌의 요청이 DRAM에 캐시될 수 있도록 하여 DRAM 캐시를 효과적으로 이끌 수 있다.

[0044] t^{bw} 는 SSD에서 지원되고 있는 대역폭을 계산하는 값으로 초기값은 0이다. t^{dram} 은 DRAM에 저장된 인터벌들의 크기를 계산하는 값으로 초기값은 0이다. A^{ssd} 는 SSD에 저장되어 있는 요청들의 리스트이다. A^{disk} 는 오직 디스크에만 저장되어 있는 요청들의 리스트이다. 이때, 본 발명에서 제안하는 캐시 관리 방법은 다음과 같다.

[0045] $A^{ssd} \neq \emptyset$ 동안, 다음을 실행할 수 있다.

[0046] 가장 인터벌이 긴 요청 R을 탐색할 수 있다.

[0047] $A^{ssd} \leftarrow A^{ssd} - R$

[0048] 만약, $t^{bw} + r(R) \leq B^{ssd}$ 이라면, $t^{bw} \leftarrow t^{bw} + r(R)$ 을 한다.

[0049] 아닐 경우, $A^{disk} \leftarrow A^{disk} \cup \{R\}$ 를 한다.

[0050] $A^{disk} \neq \emptyset$ 동안, 다음을 실행할 수 있다.

[0051] 가장 인터벌이 짧은 요청 R을 탐색할 수 있다.

[0052]
$$A^{disk} \leftarrow A^{disk} - R$$

[0053] 만약, $t^{dram} + I(R) \leq S^{dram}$ 이라면, $t^{dram} \leftarrow t^{dram} + I(R)$ 을 한다.

[0054] 이에 대한 상세한 알고리즘은 도 2에 수도코드로 표현되어 있다. 본 발명을 통하여 어떠한 요청이 디스크에서 지원받을 것인지를 결정하고 길이가 짧은 인터벌일수록 DRAM에 캐시되어 유지되도록 한다.

[0055] 도 3은 일 실시예에 따른 캐시 관리 시스템의 구성을 설명하기 위한 블록도이다.

[0056] 캐시 관리 시스템(300)은 비디오 서버에서 SSD와 DARM을 혼합하여 사용하도록 제공하기 위한 것이다.

[0057] SSD 는 하드디스크를 대체할 낸드 플래시 메모리 기반의 저장 장치로서 임의 요청에 대해 하드디스크보다 성능이 좋다는 장점이 있고, 내부 구조가 공개 되어 있어서 읽기 및 쓰기 연산에 대해 성능 예측이 가능하다는 장점을 가진다.

[0058] DRAM은 램의 한 종류로 정보를 구성하는 개개의 비트를 각기 분리된 축전기에 저장하는 기억 장치로서, 저장된 정보가 시간에 따라 소멸하기 때문에 주기적으로 재생시켜야 한다. 구조가 간단하여 집적이 용이하므로 대용량 임시기억장치로 사용된다.

[0059] 선택부(310)는 비디오 영상의 프레임들을 복수의 세그먼트로 분리하고, 복수의 세그먼트를 인기도에 기반하여 SSD에 캐시될 비디오 영상의 초기 세그먼트를 선택할 수 있다. 선택부(310)는 SSD에 저장될 초기 세그먼트를 결정하기 위하여 SSD 프레픽스 세그먼트 결정법을 사용할 수 있다. 선택부(310)는 비디오부터 i(i는 자연수) 비디오까지 전체 비디오에 대하여 SSD에 기 설정된 크기로 저장하였을 때 캐시로 획득할 수 있는 최대 이익에 대한 변수를 점화식을 설립하여 계산하고, 상기 변수를 결정하여 비디오 및 SSD의 용량에 기반하여 역으로 테이블을 추적함으로써 각각의 비디오에 저장될 초기 세그먼트를 도출할 수 있다.

[0060] 결정부(320)는 SSD의 크기와 대역폭에 기초하여 DRAM에 캐시될 인터벌을 결정할 수 있다. 결정부(320)는 동일한 비디오에 대하여 연속된 복수 개의 요청이 주어짐에 따라 복수 개의 요청 중 이전 요청에 대한 데이터가 캐시에 남아있는 경우, 요청이 캐시를 통하여 서비스되도록 할 수 있다. 결정부(320)는 요청에 대하여 인터벌 크기의 내림차순으로 정렬된 순서로 SSD의 대역폭의 한계까지 서비스할 수 있다. 결정부(320)는 기 설정된 기준 이상의 인터벌의 요청들이 상기 SSD로부터 지원됨으로써 기 설정된 이하의 인터벌의 요청이 DRAM에 캐시되도록 할 수 있다.

[0061] 도 4는 일 실시예에 따른 캐시 관리 방법을 설명하기 위한 도면이다.

[0062] 캐시 관리 방법은 도 3에서 설명한 캐시 관리 시스템을 이용하여 구체적으로 설명할 수 있다.

[0063] 단계(410)에서 캐시 관리 시스템은 비디오 영상의 프레임들을 복수의 세그먼트로 분리하고, 복수의 세그먼트를 인기도에 기반하여 SSD에 캐시될 비디오 영상의 초기 세그먼트를 선택할 수 있다. 캐시 관리 시스템은 SSD에 저장될 초기 세그먼트를 결정하기 위하여 SSD 프레픽스 세그먼트 결정법을 사용할 수 있다. 캐시 관리 시스템은 첫번째 비디오부터 i(i는 자연수) 비디오까지 전체 비디오에 대하여 SSD에 기 설정된 크기로 저장하였을 때 캐시로 획득할 수 있는 최대 이익에 대한 변수를 점화식을 설립하여 계산하고, 상기 변수를 결정하여 비디오 및 SSD의 용량에 기반하여 역으로 테이블을 추적함으로써 각각의 비디오에 저장될 초기 세그먼트를 도출할 수 있다.

[0064] 단계(420)에서 캐시 관리 시스템은 SSD의 크기와 대역폭 내에서 DRAM에 캐시될 인터벌을 결정함으로써 디스크 대역폭의 이용률을 관리할 수 있다. 캐시 관리 시스템은 동일한 비디오에 대하여 연속된 복수 개의 요청이 주어짐에 따라 복수 개의 요청 중 이전 요청에 대한 데이터가 캐시에 남아있는 경우, 요청이 캐시를 통하여 서비스되도록 할 수 있다. 캐시 관리 시스템은 요청에 대하여 인터벌 크기의 내림차순으로 정렬된 순서로 SSD의 대역폭의 한계까지 서비스할 수 있다. 캐시 관리 시스템은 기 설정된 기준 이상의 인터벌의 요청들이 상기 SSD로부터 지원됨으로써 기 설정된 이하의 인터벌의 요청이 DRAM에 캐시되도록 할 수 있다.

[0065] 도 5 내지 도 7은 디스크 대역폭 사용의 절약율을 비교하는 그래프를 나타낸 것이다.

[0066] 예를 들면, 클라이언트의 요청의 도착은 포아송 분포를 따르고, 접근 확률은 $\theta = 0.271$ 의 지프 분포를 따르며,

비디오 서버로 단위시간당 요청수인 λ 는 2/second이고, 각 비디오들의 길이는 기 설정된 시간(예를 들면, 1시간~3시간) 사이에서 랜덤하게 설정하였다고 가정하자. 또한, 1000개의 비디오 파일에 대하여 비트율은 HD영상급의 비트율인 10.4Mb/s~20.8Mb/s로 랜덤하게 설정하고, 각 비디오 파일의 크기는 32MB의 세그먼트의 배수로 가정하자. 이때, 클라이언트의 한 비디오 내에서 요청을 종료하는 시간은 $\theta = 0.2$ 의 지프 분포를 따를 수 있다.

[0067] 도 5는 SSD 캐시의 크기와 디스크 대역폭 절약의 관계를 나타낸 것이다. 도 5의 그래프를 통하여, SSD 캐시의 크기가 커질수록 대역폭에 보다 더 큰 절약이 나타난 것을 판단할 수 있으며, SSD의 대역폭이 낮은 값으로 제한된다면 세그먼트들은 SSD에서 지원을 받는 것이 어려운 것으로 판단할 수 있다. 또한, SSD의 크기가 주어졌을 때, SSD 대역폭이 커짐에 따라 디스크 대역폭 절약율도 증가함을 알 수 있다. 또한, 세그먼트의 인기도가 캐시의 결정에 영향을 주도록 한 SPC 알고리즘이 SEVC보다 더 좋은 성능임을 유추할 수 있다.

[0068] 도 6은 S^{ssd} 가 2TB일 때, DRAM의 크기와 디스크 대역폭 절약의 관계를 나타낸 그래프이다. 도 6을 통하여 SSD만 사용한 SPC 방법과 SSD와 DRAM을 혼합하여 사용한 DRAM_combined 방법을 비교하여 DRAM 인터벌 캐싱의 효과를 평가할 수 있다. 도 6의 그래프를 통하여 DRAM의 크기가 커질수록 디스크 대역폭의 절약율이 증가함을 판단할 수 있다. 이때, SSD의 대역폭이 낮을 때, 더욱 효과적임을 알 수 있다. 이것은 SSD의 대역폭이 제한적인 상황에서 DRAM 캐시를 통하여 효과적인 성능 향상을 보일 수 있다.

[0069] 비디오의 인기도와 관련된 지프 분포의 θ 에 따른 변화를 나타낸 것으로, θ 가 감소할수록 인기도는 한쪽으로 집중되는 것을 의미한다. 도 7은 S^{ssd} 가 2TB일 때, 지프 파라미터 θ 에 따른 디스크 대역폭 절약율을 나타낸 그래프이다. 도 7의 그래프와 같이, $\theta = 0.8$ 의 경우, θ 가 증가할수록 SPC가 SEVC보다 더 좋은 성능을 보임을 판단할 수 있다. SEVC 방식에서는 SSD에 인기도가 낮은 세그먼트들도 저장하는 것을 허락하기 때문이다. 반면에, θ 가 낮을 경우, DRAM 캐싱은 더 효과적인 것으로 판단할 수 있다.

[0070] 이상에서 설명된 장치는 하드웨어 구성요소, 소프트웨어 구성요소, 및/또는 하드웨어 구성요소 및 소프트웨어 구성요소의 조합으로 구현될 수 있다. 예를 들어, 실시예들에서 설명된 장치 및 구성요소는, 예를 들어, 프로세서, 콘트롤러, ALU(arithmetic logic unit), 디지털 신호 프로세서(digital signal processor), 마이크로컴퓨터, FPGA(field programmable gate array), PLU(programmable logic unit), 마이크로프로세서, 또는 명령(instruction)을 실행하고 응답할 수 있는 다른 어떠한 장치와 같이, 하나 이상의 범용 컴퓨터 또는 특수 목적 컴퓨터를 이용하여 구현될 수 있다. 처리 장치는 운영 체제(OS) 및 상기 운영 체제 상에서 수행되는 하나 이상의 소프트웨어 애플리케이션을 수행할 수 있다. 또한, 처리 장치는 소프트웨어의 실행에 응답하여, 데이터를 접근, 저장, 조작, 처리 및 생성할 수도 있다. 이해의 편의를 위하여, 처리 장치는 하나가 사용되는 것으로 설명된 경우도 있지만, 해당 기술분야에서 통상의 지식을 가진 자는, 처리 장치가 복수 개의 처리 요소(processing element) 및/또는 복수 유형의 처리 요소를 포함할 수 있음을 알 수 있다. 예를 들어, 처리 장치는 복수 개의 프로세서 또는 하나의 프로세서 및 하나의 콘트롤러를 포함할 수 있다. 또한, 병렬 프로세서(parallel processor)와 같은, 다른 처리 구성(processing configuration)도 가능하다.

[0071] 소프트웨어는 컴퓨터 프로그램(computer program), 코드(code), 명령(instruction), 또는 이들 중 하나 이상의 조합을 포함할 수 있으며, 원하는 대로 동작하도록 처리 장치를 구성하거나 독립적으로 또는 결합적으로(collectively) 처리 장치를 명령할 수 있다. 소프트웨어 및/또는 데이터는, 처리 장치에 의하여 해석되거나 처리 장치에 명령 또는 데이터를 제공하기 위하여, 어떤 유형의 기계, 구성요소(component), 물리적 장치, 가상장치(virtual equipment), 컴퓨터 저장 매체 또는 장치에 영구적으로, 또는 일시적으로 구체화(embodiment)될 수 있다. 소프트웨어는 네트워크로 연결된 컴퓨터 시스템 상에 분산되어서, 분산된 방법으로 저장되거나 실행될 수도 있다. 소프트웨어 및 데이터는 하나 이상의 컴퓨터 판독 가능 기록 매체에 저장될 수 있다.

[0072] 실시예에 따른 방법은 다양한 컴퓨터 수단을 통하여 수행될 수 있는 프로그램 명령 형태로 구현되어 컴퓨터 판독 가능 매체에 기록될 수 있다. 상기 컴퓨터 판독 가능 매체는 프로그램 명령, 데이터 파일, 데이터 구조 등을 단독으로 또는 조합하여 포함할 수 있다. 상기 매체에 기록되는 프로그램 명령은 실시예를 위하여 특별히

설계되고 구성된 것들이거나 컴퓨터 소프트웨어 당업자에게 공지되어 사용 가능한 것일 수도 있다. 컴퓨터 판독 가능 기록 매체의 예에는 하드 디스크, 플로피 디스크 및 자기 테이프와 같은 자기 매체(magnetic media), CD-ROM, DVD와 같은 광기록 매체(optical media), 플롭티컬 디스크(floptical disk)와 같은 자기-광 매체(magneto-optical media), 및 롬(ROM), 램(RAM), 플래시 메모리 등과 같은 프로그램 명령을 저장하고 수행하도록 특별히 구성된 하드웨어 장치가 포함된다. 프로그램 명령의 예에는 컴파일러에 의해 만들어지는 것과 같은 기계어 코드뿐만 아니라 인터프리터 등을 사용해서 컴퓨터에 의해서 실행될 수 있는 고급 언어 코드를 포함한다.

[0073] 이상과 같이 실시예들이 비록 한정된 실시예와 도면에 의해 설명되었으나, 해당 기술분야에서 통상의 지식을 가진 자라면 상기의 기재로부터 다양한 수정 및 변형이 가능하다. 예를 들어, 설명된 기술들이 설명된 방법과 다른 순서로 수행되거나, 및/또는 설명된 시스템, 구조, 장치, 회로 등의 구성요소들이 설명된 방법과 다른 형태로 결합 또는 조합되거나, 다른 구성요소 또는 균등물에 의하여 대치되거나 치환되더라도 적절한 결과가 달성될 수 있다.

[0074] 그러므로, 다른 구현들, 다른 실시예들 및 특허청구범위와 균등한 것들도 후술하는 특허청구범위의 범위에 속한다.

도면

도면1

```

1: Temporary variables: i, m and k;
2: for i = 1 to Nv do
3:   for m = 1 to Sssd do
4:     Ci, mgain ← -∞;
5:     Ci, mprefix ← 0;
6:   end for
7: end for
8: for i = 1 to Sssd do
9:   if m ≥ Si(1) then
10:    Ci, mgain ← maxk ∈ {k | S1(k) ≤ m} G1(k);
11:    Ci, mprefix ← arg maxk ∈ {k | S1(k) ≤ m} G1(k);
12:   end if
13: end for
14: for i = 2 to Nv do
15:   for m = 1 to Sssd do
16:     Ci, mgain ← Ci, m-1gain
17:     for k = 1 to Nis do
18:       if m - Si(k) > 0 then
19:         if Ci, mgain > Ci-1, m-Si(k)}gain + Gi(k) then
20:           Ci, mgain ← Ci-1, m-Si(k)}gain + Gi(k);
21:           Ci, mprefix ← k;
22:         end if
23:       end if
24:     end for
25:   end for
26: end for
27: i ← Nv, m ← Sssd;
28: while i > 0 do
29:   Fi ← Ci, mprefix;
30:   m ← m - Si(Fi);
31:   i ← i - 1;
32: end while

```

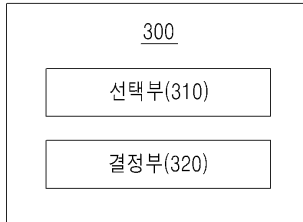
도면2

```

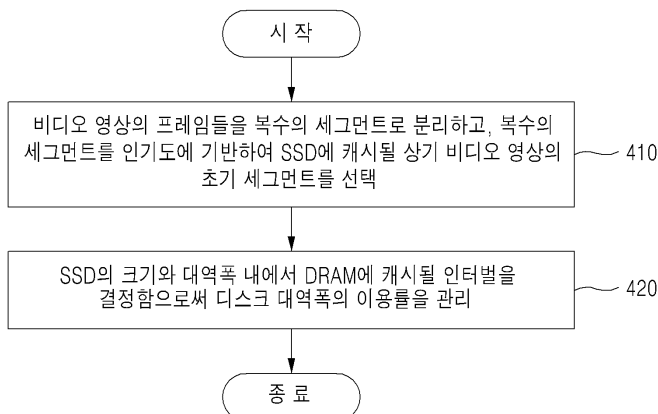
1: Temporary variable for storing bandwidth:  $t^{bw} \leftarrow 0$ ;
2: Temporary variable for storing bandwidth:  $t^{dram} \leftarrow 0$ ;
3: Array of requests for videos stored on SSD:  $A^{ssd}$ ;
4: Array of requests for videos stored only on disk:  $A^{disk}$ ;
5: while  $A^{ssd} \neq \emptyset$  do
6:   Find a request, R which has the largest interval;
7:    $A^{ssd} \leftarrow A^{ssd} - \{R\}$ ;
8:   if  $t^{bw} + r(R) \leq B^{ssd}$  then
9:      $t^{bw} \leftarrow t^{bw} + r(R)$ ;
10:  else
11:     $A^{disk} \leftarrow A^{disk} \cup \{R\}$ ;
12:  end if
13: end while
14: while  $A^{disk} \neq \emptyset$  do
15:   Find the request R which has the smallest interval;
16:    $A^{disk} \leftarrow A^{disk} - \{R\}$ ;
17:   if  $t^{dram} + l(R) \leq S^{dram}$  then
18:      $t^{dram} \leftarrow t^{dram} + l(R)$ ;
19:   end if
20: end while

```

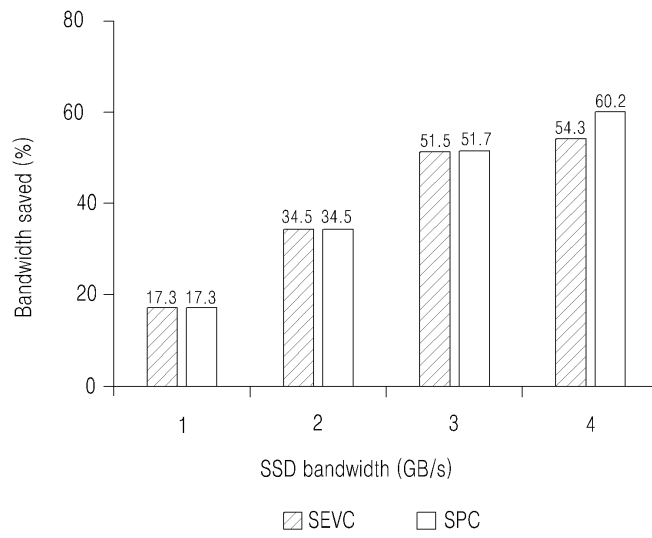
도면3



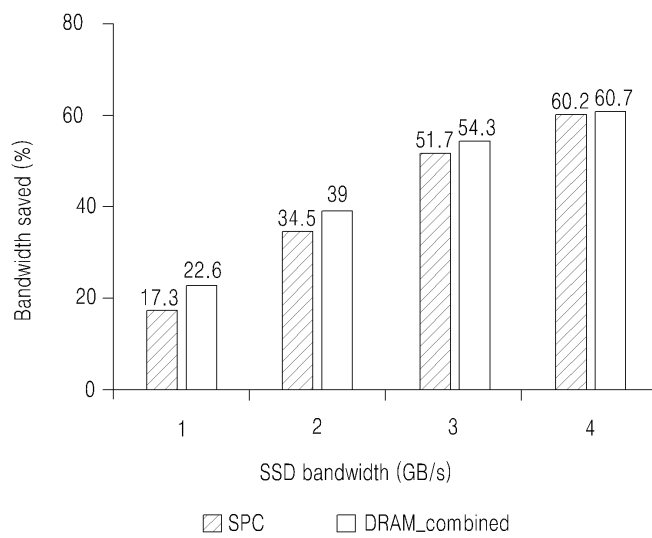
도면4



도면5



도면6



도면7

