데이타 그리드 컴퓨팅 환경에서 디스크 캐쉬 교체 연구

박홍진
상지대학교 컴퓨터정보공학부
hjpark1@sangji.ac.kr

A Study on Disk Cache Replacement for Data Grids Computing Environment

Hong-Jin Park
School of Computer, Information and Communications Computer
Sang-Ji University

요 약

웹 인프라와 P2P 기술 개념을 기반으로 연구되고 있는 데이터 그리드 컴퓨팅 환경에서 중요한 문제 중 하나는 거대한 데이터 자료 접근에 관한 이슈이다. 요청된 자원 공간을 확보하기위해 저장되어 있는 자원을 선택하는 "캐쉬 교체 정책" 연구는 많이 연구되어 왔으나, 상황을 미리 예측하거나 스택과 같은 추가적인 자원이 필요하다. 본 논문은 기존 연구의 문제점을 해결하고 요청된 파일에 대응되는 파일의 개수가 되도록 적게 하기위해 파일 크기를 고려한 캐쉬 교체 정책을 제안하고 있다.

1. 서론

생명공학, 천문학, 인공지능등 최첨단 분야의 기술이 고도로 발달됨에 따라 고가의 슈퍼 컴퓨팅 사용에 대해 저비용의 고효율적인 대안으로 그리드 컴퓨팅이 제안되고 오고 있다. 그리드 컴퓨팅은 지리적으로 분산된 고성능 컴퓨터, 대용량 저장 장치, 첨단 장비 등의 자원들을 고속 네트워크로 연결해 상호 공유 및 이용할 수 있도록 하는 차세대 디지털 신경망 서비스이다. 그리드(Grid)란 용어를 사전에서 찾아보면 격자, 지도의 모눈, 망상조직, 라디오/TV 등의 방송망, 네트워크 등의 의미를 가진다. 웹(Web)이라는 용어가 사전적으로는 거미집 모양의 네트워크를 뜻한다는 것을 비추어 볼 때, 그리드라는 용어는 웹보다 더 촘촘하게 연결되는 네트워크의 의미를 내포하고 있다. 즉 월드 와이드 웹(WWW)을 대체할 차세대 인터넷 기술로 그리드가 주목받고 있는 것이다. 그리드 컴퓨팅에서 데이터 그리드(Data Grid)는 고 성능의 이질적인 노드와 데이터 저장 자원이 지리적으로 분산되어 있는 플랫품의 네트워크를 의미한다. 저장 노드는 대용량의 저장 장치, 제3저장 장치, 고성능 저장 시스템(HPSS)등으로 사용자에게 지역적으로나 원격으로 데이타 생성, 삭제, 읽기, 쓰기 조작등이 가능한 장치이다[1].

데이터 그리드에서 중요한 문제는 상호 연결된 네트워크의 높은 지연 시간과 저장 장치에 거대한 자료에 대한 접근에 관련된 문제이다[2]. 이 문제를 해결하기 위한 방법으로 원격 접근에 따른 오버헤드를 피하고 자원에 대한 신뢰성을 향상하기 위해 원격 데이터를 지역(local)으로 데이터를 캐슁(caching)하는 것이다. 캐슁 기술은 컴퓨터 시스템, 데이터베이스, 웹 캐슁등 저장 시스템의 성능과 신뢰성을 향상시키기 위해 사용되어 왔다. 그러나 데이터 그리드에서 사용하는 캐슁 기술은 기존 캐슁 기술과 여러 가지 면에서 서로 다르다. 예를 들어 데이터 그리드에서는 사용되는 네트워크도 노드 거리가 먼 WAN을 기반하고 있으며, 파일 크기도 기가바이트

용량이며, 캐쉬 크기도 수백 기가바이트에서 수십 테라바이트를 수용할 수 있어야한다. 기존 웹에서 사용하는 캐슁 기술은 전송 지연 시간도 몇초에서 몇분 정도이고, 캐슁 기술이 선택적인 사항임에 반해, 데이터 그리드에 환경에서는 데이터 전송 지연 시간은 몇분에서 몇 시간까지 매우 긴 시간이 걸리며, 웹 기술 사용도 필수적이다. 웹 캐슁과 데이터 그리드에서 캐슁과의 보다 세부적인 차이점은 <표 1> 있다[3].

< 표 1 > 웹 캐슁과 데이터 그리드 캐슁 비교

특성	웹 캐싱	데이터 그리드 캐슁
파일/객체 크기	메가바이트(MB)	기가바이트(GB)
캐쉬 크기	수십에서 수백 메가 바이트	수백 기가에서 수십 테라바이트
전송 시간	몇초에서 몇분	몇초에서 몇시간
캐쉬 요구	선택적인 사항	강제적인 사항
객체 참조 시간	거의 순간적	몇초에서 몇분
배치 요구	전형적으로는 하나의 요구가 하나의 배치 참조	전형적으로 하나의 요구는 수백 파일과 연관
네트워크 대역폭	표준 인터넷 용량	초고속 기가 네트워크 용량

본 논문은 데이터 그리드환경에서 캐슁 교체 전략을 기술한다. 대용량의 데이타 저장 장치에 요청은 동시에 수십에서 수백, 수천이 될 수 있다. 각각의 요청은 일반적으로 큐잉되며 어느 파일이 검색되어야 할지 먼저 결정하게 된다. 이 결정은 “파일 허가 전략”이라고 한다. 파일 허가 전략후에 파일은 캐쉬 교체 정책을 수행하게된다. 캐쉬 교체 정책은 요청하는 파일에 공간을 확보하기 위해 저장되어 있는 파일을 선택하는 작업이다.

본 논문의 구성은 다음과 같다. 2장에서는 현제 데이터 그리드환경에서의 디스크 캐쉬 교체 정책에 대해서 비교 평가한다. 3장에서는 본 논문에서 제안하고 있는 알고리즘을 설명하며, 4장에서는 결론을 설명한다.

2. 기존 연구

가장 널리 알려져 있고, 운영체제에서 많이 사용되는 페이지 교체 정책이 LRU (Least Recently Used)와 LFU (Least Frequently Used)가 있다[4]. 최근 참조된 시점(LRU)이나 참조되어진 빈도수(LFU)를 가지고 캐쉬될 파일을 선정하는 알고리즘이다.

[3] 논문에서 제안한 LCB-K (Least Cost Beneficial based on K backward reference)는 데이터 그리드의 파일 교체를 위한 정책이다. 이 알고리즘은 파일 선택을 위해 유틸리티 함수를 사용하며 유틸리티 함수는 최대 K 까지 최근의 참조회수*검색 값(cost)/파일의 크기로 나타낸다. 캐쉬되어야 할 각각의 파일들을 유틸리티 함수로 계산하여 상대적인 순위를 매겨지며 가장 낮은 유틸리티 값을 가진 첫 번째 파일이 선택되어 교체되며, 요청하는 파일의 공간 확보를 위해 계속해서 유틸리티 함수에서 낮은 값을 가진 파일이 선택되어 지는 알고리즘이다.

EBR (Economic-Based cache Replacement) 정책은 [5]에서 제안하였다. 이 정책은 파일의 값(cost)을 최소화 시키는 반면 저장 공간을 최대화 시키는 경제적인 모델을 기반하고 있으며, 확률 기반 유틸리티 함수 기법을 사용한다. 파일을 저장하기 위해 충분한 공간이 있으면 새롭게 도착된 파일을 디스크 상에 자동으로 저장한다. 만약 디스크가 충분한 공간이 없으면, EBR은 디스크 상에서 최소 값을 가진 파일을 선택한다. 선택을 하기위해 중복 최적기(Replica Optimiser)를 이용한다. 중복 최적기는 이익(profit)을 최대화시키기 위해 저장되어 있는 각각의 파일의 값을 로그에 유지하면서, 이 값이 미래 소득 예측 함수의 입력 값으로 사용된다. 이 예측 함수는 과거의 윈도우 W 시간에 요청을 기반하여 앞으로 W 시간에 요청을 미리 예측하여 값을 산출한다.

LVCT (Least Value-based on Caching Time) 정책은 [6]에서 제안하였다. 이 정책은 얼마나 곧 파일이 재 접근되어지는가(caching time)을 고려하여 파일을 교체한다. 유틸리티 함수 값에서 가장 작은 값을 지닌 파일(들)이 선택되어 지며, 유틸리티 함수는 (1/캐슁 시간)* cost/파일의 크기이다. 이 정책은 각 파일의 캐슁 시간을 유지하기 위해 캐싱 시간

과 파일의 크기를 지닌 캐슁 시간 스택(caching time stack)이 사용된다. 즉, 각 파일 마다 접근 되어지는 시간을 고려한 캐슁 시간과 파일의 크기가 캐슁 시간 스택에 저장하게 된다. 예를 들어 파일 f가 교체되기 위해 접근되어지면 파일 f을 스택의 꼭대기(top)로 옮김 후에 파일f의 캐슁 시간을 0으로 한다. 계속해서 재접근 되어진 파일을 스택의 꼭대기에 옮김으로써 스택의 바닥(bottom)에 있는 파일은 가장 재접근이 안 된 파일이고, 이 파일(들)을 교체한다는 것이다. 즉, 가장 큰 캐싱 시간을 지닌 파일이 가장 접근이 되지 않은 파일이고 이 파일(들)을 교체한다.

위에서 기술한 기존 정책의 문제점은 다음과 같다. LRU와 LFU 정책은 널리 알려진 정책이나 너무나 적은(혹은 단순한) 정보를 기반으로 파일 선정하는 문제점이 있다. [3]에서 제안한 LCB-K 정책의 주요 문제점은 향후 파일 검색 값(retrieval cost) 미리 예측해야 된다는 점이다. 추가적인 문제점은 선택 후보가 되기 위해서는 과거에 적어도 한번은 검색되어야 한다는 단점이 있다. [4]에서 제안한 정책은 경제적인 모델을 위해 과거의 정보를 기반으로 앞으로의 요청을 미리 예측해야한다는 점이다. LVCT 정책은 캐싱 시간을 유지하기 위해 캐싱 시간과 파일 크기를 지닌 스택을 유지해야한다는 점이다. 접근되어지는 파일은 스택의 꼭대기로 옮겨지는 추가적인 작업도 필요하다. 전체적은 보면 공간 확보를 위해 앞으로 상황을 예측해야하며, 필요한 공간을 확보하기위해 요청된 파일 보다 상대적으로 작은 파일들이 모든 교체될 가능성이 있다. 또한, 교체를 위해 스택과 같은 불필요한 자원도 유지 하고 있음을 알 수 있다.

3. 제안 정책

본 논문에서 제안한 정책은 기존 알고리즘에 비해 불확실한 미래의 상황을 예측하여 교체하기 보다, 요청된 파일에 대응되는 파일의 개수가 되도록 적게 하기 위해 파일 크기를 고려한 교체 정책을 제안한다.

데이터 그리드 환경에서 사용자들이 지역적인 저장 자원 관리자에 대해 파일 요청은 동시에 수백 내지 수천 건이 될 수 있으며, 저장 자원 관리자는 이를 먼저 큐잉 시킨다. 큐잉된 요청 메세지는 저장 자원 관리자내에 파일 허가 모듈이 어는 파일을 먼저 검색해야 될지를 결정한 후에 파일 교체 모듈에 요청 메시지를 전달한다.

파일 교체 모듈은 먼저 요청한 파일이 지역적인 디스크에 있는 지 확인한다. 만약, 지역적인 디스크에 있으면 요청한 사용자에게 요청한 파일을 전달해 준다. 만약 지역적인 디스크에 요청 파일이 없으면 원격지의 디스크에 사용자 요청파일을 요구하여 지역적인 디스크에 캐슁을 하는데 이때 지역적인 디스크에 새로운 파일에 대한 공간이 없으면 파일 교체 알고리즘에 의해서 교체될 파일이 선택된다. 현재 지역적으로 저장되어 있는 파일들이 선택 후부가 되는데 이를 “unpinned″이라고 하며, 선택 후보 파일 중 교체하기 위해 선택된 파일은 ”pinned″이라한다.

본 논문에서 제안한 파일 교체 알고리즘은 <표 2>과 같다. 사용자가 요청한 파일의 크기를 r이라고 할때, 알고리즘은 먼저, 지역적으로 현재 저장되어 있는 파일 중 크기가 r인 파일을 찾는다. 만약 새로 요청한 파일 크기와 저장 되어있는 파일 크기 r이 있으면 파일 크기가 같은 파일이 교체하기 위해 선정이 되면 이는 알고리즘에서 최적 교체이다.

만약, 크기가 같은 파일이 없을 경우는 알고리즘에서는 k 값을 고려하여 파일 교체 알고리즘이 수행하며, k 값은 사용자가 요청한 파일 크기의 비례되는 크기이다. 예를 들어 사용자가 요청한 파일의 크기가 1000MB 이고, k값이 0.1(즉, 10%)이면, k의 크기는 요청한 파일 크기의 10% 비율인 100MB이다. 지역적인 디스크에서 같은 크기의 파일이 없고, k 값이 0.1일 경우 알고리즘은 요청 파일 크기에 k 값을 더한 크기(1100MB)를 q라고 하자. 알고리즘은 지역적인 디스크에서 r보다 크고 q보다 작은 파일(들)을 찾으며 이를 p라고 하자. 알고리즘에서는 찾아진 p 파일(들)을 LRU와 LFU을 고려하여 최종 교체 파일을 선정하여 교체한다.

만약 지역적 디스크에서 p 파일이 없을 경우는, 알고리즘은 요청 파일 크기의 k값을 뺀 k 값을 뺀 파일 크기를 갖는 파일(900MB)을 찾게되면 이를 q′라고하자. 알고리즘은 지역적인 디스크에서 r보다 작고 q′보다 큰 파일(들)을 찾으며 이를 p′라고 하자. 알고리즘에서는 찾아진 p′ 파일(들)을 LRU와 LFU을 고려하여 최종 교체 파일을 선정하여 교체한다.

만약 지역적 디스크에서 p′ 파일이 없을 경우에는 교체될 파일이 있을 때까지 k값을 증가하여 알고리즘을 계속 수행한다.

<표 2> 제안한 캐쉬 교체 정책

```
if(new_requested_file_size == unpinned_file_size)
        than {
              replace file;
              exit;
        }
/* K 값을 교려 */
 k=0;
 while( 1 ){
      /* k의 초기값은 0.1(즉, 10%)이며 10%씩 증가 */
      k += 0.1
        k_size = new_file_size * k
       i) new_file_size 보다 크고
              new_file_size + k_size 보다 작은 file들을 search
         선택된 파일들에 대해서
                    LRU와 LFU 알고리즘을 고려하여,
                    적합한 파일을 선택
         replace file;
         exit;
       ii)  new_file_size 보다 작고 new_file_size - k_size 보다 작은
              unpinned file들을 search
           선택된 파일들에 대해서
              LRU와 LFU 알고리즘을 고려하여, 적합한 파일을 선택
           replace files;
           exit;
 }
```

4. 결론

슈퍼컴퓨터 대안으로 최근의 데이터 그리드의 관련 연구는 생명공학, 천문학등을 중심으로 연구해 오고 있다. 기존윈 웹 기반에 컴퓨팅 환경보다 대용량이고 전송 시간이 최대 몇시간이 걸리는 데이터 그리딩 환경에서는 원격에 있는 자원에 대한 효율적인 접근 연구 즉, 캐쉬 교체 연구가 필요하다.

요청한 파일에 공간을 확보하기 위해 저장되어 있는파일을 선택하는 “캐쉬 교체“ 정책에 대한 기존 연구는 과거의 정보를 기반으로 향우 접근을 예측하거나, 스택등을 추가적으로 이용하는 기법등이 이용되고 있다.

본문은 요청된 파일에 대응되는 파일의 개수가 되도록 적게 하기 위해 파일 크기를 고려한 캐쉬 교체 정책을 제안하였다.

참고문헌

[1] A. Chervenak, I. Foster, C. Kesselman, C. Salisbury and S. Tueckes, "The data grid: Towards an architecture for the distributed management and analysis of large scientific data-sets", Journal of Network and Computer Applications, 2002

[2] J.H Abawajy, "File Replacement Algorithm for Storage Resource Managers in Data Grids", LNCS 3038, 2004

[3] E. J. Otoo, F. Olken and A. Shoshani, " Disk Cache replecement algorithm for storage resource managers in data grids", In The 15th Annual Super Computer Conference, Nov., 2002

[4] S. Aberham, G. Peter Baer, G. Greg, Operating system principles, 7th, Wiley, 2006

[5] M. Carman, F. Serafini, L. Stokinger, K. Stockinger, "Towards an Economy-Based Optimisation of File Access and Replication on a Data Grid", In Procedings of 2nd CCGRID 2002

[6] S. Jiang, X. Zhang, "Efficient Distributed Disk Caching in Data Grid Managemnet", In Proceedings of Cluster Computing, 2003