웹 서버 클러스터에서 사용자 응답시간 개선을 위한 메모리 관리

(Memory Management for Improving User Response Time in Web Server Clusters)

정 지 영 † 김 성 수 ††
(Ji Yung Chung) (Sungsoo Kim)

요 약 클러스터 시스템의 각 노드에 존재하는 메모리들을 효율적으로 관리하기 위하여 네트워크 메모리의 개념이 등장하였으며 빈번하게 디스크를 접근하는 응용분야에서 속도 향상을 위해 사용될 수 있다. 이는 전통적인 메모리 계층(hierarchy) 구조인 메모리와 디스크 사이에 네트워크 메모리를 추가함으로써 얻어진다. 본 논문에서는 웹 서버 클러스터를 대상으로 문서의 접근 유형에 대한 사전 정보를 요구하지 않고 실제적으로 구현 가능하며 다양한 웹 문서 접근 확률 분포 값에 대하여 우수한 사용자 응답시간을 가지는 메모리 관리 기법을 제안하고 다양한 시뮬레이션을 통해 제안된 방식의 우수성을 검증하였다.

Abstract The concept of network memory was introduced for the efficient exploitation of main memory in a cluster. Network memory can be used to speed up applications that frequently access large amount of disk data. In this paper, we present a memory management algorithm that does not require prior knowledge of access patterns and that is practical to implement under the web server cluster. In addition, our scheme has a good user response time for various access distributions of web documents. Through a detailed simulation, we evaluate the performance of our memory management algorithms.

1. 서 론

최근 인터넷의 사용이 크게 증가하면서 웹을 이용한 상품 및 서비스가 크게 보급, 확산되고 있고 이러한 서비스의 예로 기업의 광고 및 상품의 검색, 전자상거래나 정보 관리 등을 들 수 있다. 현재 인터넷과 웹은 성공적으로 인식되고 있으며 앞으로도 많은 성장 가능성이 있지만 제한된 대역폭과 서버 측 용량의 한계로 인한 여러 문제를 야기하고 있다. 인터넷을 이용하는 사용자가 많아질수록 사용자 모두가 제한된 자원을 효율적으로 이용하기는 더욱 어려워지며 따라서 웹 서버의 성능에 관한 연구가 중요시된다.

웹 서버의 신뢰도와 가용도를 제공하기 위하여 많은 서비스 제공자들은 결함 허용 컴퓨터를 이용하여 그들의 서버를 구성해 왔다[1]. 이러한 결함 허용 컴퓨터들은 하드웨어를 중복시키거나 자기 검사(self-checking)를 수행함으로써 결함을 방지하기 때문에 시스템을 구축하기 위해 많은 비용을 필요로 한다. 이에 비해 저가의 PC나 워크스테이션을 이용하는 클러스터링 기법은 비용 측면에서 매우 유리하며 특히 웹 서비스나 정보 시스템과 같이 고성능과 고가용도를 요구하는 응용 분야에서 결함 허용 컴퓨터의 대안으로 등장하고 있다[2, 3].

현재 인기 있는 웹 사이트에서는 매일 수백만의 사용자 요구를 수용하기 위해 다수의 서버를 사용하고 있으며 하나의 가상 URL 인터페이스를 이용하여 사용자에게 투명성을 제공하는 방식을 일반적으로 채택하고 있다[4].

초기에 클러스터 시스템의 성능을 제한하는 지배적인 요소는 프로세서의 속도였으나 하드웨어가 향상됨에 따라 프로세서가 시스템 전체 성능에 미치는 영향이 감소

· This work is supported in part by the Ministry of Information & Communication in Republic of Korea ("University Support Program (2001)" supervised by IITA).
† 학생회원 : 아주대학교 정보통신전문대학원
abback@madang.ajou.co.kr
†† 종신회원 : 아주대학교 정보통신전문대학원 교수
sskim@madang.ajou.ac.kr
논문접수 : 2001년 3월 5일
심사완료 : 2001년 7월 5일

되었다. 현재는 메모리 대역폭이 성능의 병목으로서 프로세서의 역할을 대신하고 있다[5]. 네트워크 역시 항상 클러스터 시스템의 성능을 제한하는 요소였으나 100Mbps 이더넷과 같은 고속 네트워크의 등장으로 영향이 감소하는 추세이다. 이러한 견지에서 볼 때 효율적인 메모리 관리는 클러스터 시스템의 전체 성능을 향상시키는 중요한 요소라 할 수 있다.

클러스터 시스템의 각 노드에 존재하는 메모리들을 효율적으로 관리하기 위하여 네트워크 메모리의 개념이 등장하였으며 빈번하게 디스크를 접근하는 응용분야에서 속도 향상을 위해 사용될 수 있다. 이는 전통적인 메모리 계층(hierarchy) 구조인 메모리와 디스크 사이에 네트워크 메모리를 추가함으로써 얻어진다. 즉 디스크에서 데이타를 가져오는 시간보다 다른 노드의 메모리에 존재하는 데이타를 가져오는 시간이 더욱 빠른 특성을 이용한다. 효율적인 메모리 관리는 유용한 페이지를 메모리 계층에서 사용자에게 더욱 가까이 유지하도록 하는 것이다. 따라서 메모리 페이지 교체 정책을 적용할 때 자신의 메모리만 참조하는 것이 아니라 클러스터 내에 존재하는 메모리를 총체적인 관점에서 고려하여야 한다.

본 논문에서는 기존에 제안된 클러스터 시스템의 메모리 관리들을 분석하고 웹 문서들의 접근 확률 분포가 변하더라도 항상 최적의 사용자 응답 시간을 가질 수 있는 메모리 관리 정책을 제안한다.

논문의 구성으로 2장에서는 관련 연구를 알아보고 3장에서는 본 논문에서 고려하는 웹 서버의 구조를 설명한다. 또한 4장에서는 제안된 클러스터 메모리 관리 정책에 대하여 설명하고 5장에서는 시뮬레이션을 통해 기존의 방법들과 성능을 비교하였다. 마지막으로 6장에서는 결론을 내린다.

2. 관련연구

클러스터 시스템의 메모리 관리에 관한 몇몇 연구들은 휴지 상태에 있는 메모리를 이용하는 방법을 제안하고 있다[6, 7, 8]. 즉 활동 중인 노드의 메모리에서 오버플로우가 발생하면 현재 사용되지 않는 다른 노드의 메모리를 이용함으로써 일시적으로 확장된 메모리를 소유하는 방식이다. 이러한 방법들은 구현이 용이하다는 장점을 가지고 있으나 사용자의 요구가 제한된 메모리 안에 존재하지 않으면 비록 다른 노드의 메모리에 존재한다 할지라도 디스크에서 데이타를 가져와야 하는 단점이 있다. 또한 부하 분배를 통해 모든 노드가 동일한 부하를 지니고 있는 클러스터 시스템에서는 활동 중인 노드와 휴지 상태에 있는 노드가 동시에 존재하지 않으므로 효과가 없다.

또한 수학적인 모델을 통하여 분산 메모리 시스템에서 중복된 페이지를 관리하는 연구가 존재하는데 이는 사용자의 접근 유형에 대한 사전의 정보를 요구한다[9]. 그러나 미래의 접근 유형에 대한 정보를 요구하는 알고리즘은 실제적으로 구현되기 어려운 단점을 지니고 있다. 이 외에도 클러스터 시스템의 공유 메모리 모델에 대한 연구[10]나 메모리 계층구조의 영향에 대한 연구도 존재한다[11].

가장 기본적인 클러스터 메모리 관리 정책인 Greedy Forwarding[12] 알고리즘은 클러스터 내에 존재하는 메모리를 전체적인 관점에서 다루기는 하지만 노드들 사이에서 메모리의 내용을 서로 참조하지 않는다. 이것은 각 노드가 메모리 내에 존재하는 페이지를 제거할 때 전통적인 LRU 방식을 사용함을 의미하며 따라서 동일한 데이타가 여러 노드의 메모리에 존재하게 된다. 이와 같은 불필요한 데이타의 중복은 이용 가능한 메모리의 크기를 줄일 수 있으므로 비효율적이다.

이와 대조적인 방법으로 Duplicate Elimination[13] 방법을 들 수 있다. 이 방법은 클러스터 전체 메모리에서 최대한 중복을 허용하지 않는 방식으로 각각의 노드는 단일 LRU 리스트와 중복 LRU 리스트를 유지하며 메모리 페이지 교체 시 중복 LRU 리스트의 페이지를 우선적으로 제거하는 방식이다. 이러한 방식을 사용할 경우 클러스터 시스템 전체 메모리의 적중률은 증가하나 상대적으로 로컬 적중률은 감소하게 되어 다른 메모리의 접근 횟수가 증가한다. 특히 자주 요구되는 웹 문서가 존재하여 이에 대한 빈번한 요구가 발생할 때는 주로 로컬 적중률의 성능이 중요한 영향을 미치므로 이 역시 효율적이지 않다.

이와 같이 대조적인 두 가지 방법을 절충하기 위하여 제안된 알고리즘들 중 N-chance Forwarding[12]은 전체 메모리에서 유일한 페이지를 제거할 때 랜덤하게 선택된 n개의 노드들에게 그 페이지를 전송함으로써 사용자 응답 시간을 향상시킬 수 있으나 파일 접근 확률의 영향을 고려하지 않았다.

또한 Hybrid 알고리즘[13]은 웹 서버 클러스터 상에서 페이지 중복을 적절하게 유지하기 위하여 무조건 중복 LRU 리스트에서 제거하지 않고 단일 LRU 리스트와 중복 LRU 리스트의 기대값을 비교하여 작은 쪽을 제거하는 방식을 채택한다. 기대값은 해당 페이지를 메모리에 가져오기 위해 걸리는 시간과 페이지 접근 시간에 대한 값을 고려하여 결정된다. 특히 이 연구에서는 다양한 웹 문서 접근 확률 분포에 따라 평균적으로

Hybrid 알고리즘이 우수함을 보였으나 모든 경우에 대해 우수하지는 않았다.

일반적으로 웹 문서의 접근 확률 분포는 Zipfian 분포[14]를 따르는 것으로 알려져 있으며 분포에 영향을 주는 파라미터인 skew는 0부터 2사이의 값이 주로 사용된다. 이 분포에서 skew가 0인 경우 문서들 간의 접근 확률은 균일하며 2에 가까워질수록 문서들 간의 접근 확률 차이가 커지게 된다.

본 논문에서는 웹 서버 클러스터를 대상으로 문서의 접근 유형에 대한 사전 정보를 요구하지 않고 실제적으로 구현 가능하며 다양한 skew 값에 대하여 항상 우수한 사용자 응답시간을 가지는 메모리 관리 정책을 제안한다.

3. 웹 서버 클러스터 구조

본 논문에서 고려하는 웹 서버 클러스터는 사용자 요구 분배를 위한 부하 분배기와 요구된 문서를 서비스하는 다수의 노드가 고속의 네트워크로 연결되어 있는 시스템으로서 그림 1은 이와 같은 구조를 보여주고 있다. 이 구조에서 각각의 노드는 자신의 메모리와 디스크를 가지고 있고 다른 노드와 상호 통신할 수 있는 프로세스를 가지고 있다.

사용자의 요구는 부하 분배기에 전달되며 부하 분배기는 라운드 로빈 (Round-robin) 방식으로 각 서버에게 사용자의 요구를 전달한다. 사용자의 요구를 받은 웹 서버 노드는 요구된 데이타를 자신의 메모리나 디스크, 또는 다른 노드의 메모리나 디스크에서 가져오며 부하 분배기를 거치지 않고 바로 클라이언트에게 응답을 보내는 직접 라우팅 방식을 이용한다.

클러스터 시스템에서 서비스하는 웹 문서들은 각 노드의 디스크에 적절히 분산되어 저장되어 있으며 사용자의 요구를 받은 노드를 주(primary) 노드, 요구된 데이타를 디스크에 저장하고 있는 노드를 소유자(owner) 노드라 부른다. 이는 경우에 따라 각 노드가 주 노드 또는 소유자 노드가 될 수 있음을 의미한다. 이 때 소유자 노드는 전체 메모리에서 그들이 디스크에 저장하고 있는 데이타 페이지의 중복 정보를 유지하여야 한다[16].

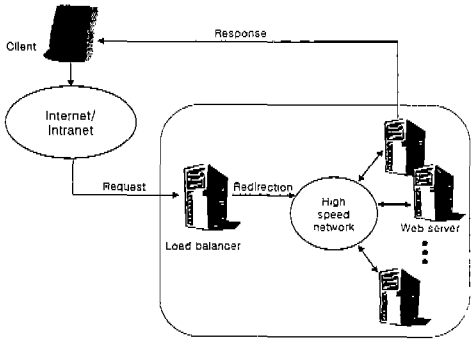

그림 1 웹 서버 클러스터 구조

각 노드가 페이지의 소유자 노드에 의해 페이지 상태 변화를 통보 받아야 할 시점은 클러스터 전체 메모리에서 유일하게 존재하던 페이지가 중복된 페이지로 변하는 경우와 중복되어 있던 페이지가 유일한 페이지로 변하는 경우이다. 이러한 메시지들은 즉시 보내지거나 다른 정보와 함께 피기 백(piggy back) 될 수 있는데 피기 백 될 경우 추가 적인 오버헤드는 미미한 것으로 알려져 있다[13].

4. 클러스터 메모리 관리

서론에서 언급했듯이, 효율적인 메모리 관리는 유용한 페이지를 메모리 계층에서 사용자에게 더욱 가까이 유지하도록 하는 것이다. 본 논문에서 고려하는 웹 서버 클러스터의 메모리 계층구조는 주 노드 메모리, 소유자 노드 메모리, 다른 노드의 메모리 그리고 디스크로 구성되며 그림 2는 요구를 받은 노드를 기준으로 메모리 구조를 보여주고 있다.

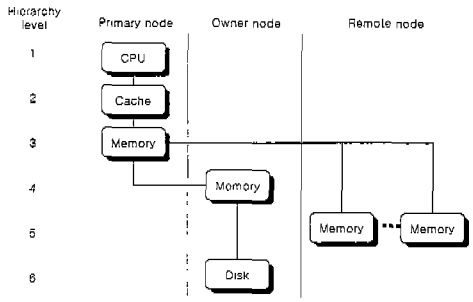

그림 2 메모리 계층구조

사용자의 요구는 라운드 로빈 방식에 의해 주 서버에 전달되고 페이지 단위로 나누어진다. 만일 요구된 페이지가 주 서버에 존재하면 즉시 서비스를 시작하고 존재하지 않으면 주 노드는 소유자 노드에게 페이지를 요구한다. 소유자 노드는 자신의 메모리에 요구된 페이지가 존재하면 이를 주 노드에게 전달한다. 그러나 요구된 페이지가 소유자 노드의 메모리에도 존재하지 않을 경우, 소유자 노드는 해당 페이지를 가지고 있는 다른 노드에 요구를 전달한다. 이 요구를 받은 노드는 해당 페이지를 소유자

노드를 거치지 않고 바로 주 노드에게 전달하게 된다.

그러나 만일 요구된 페이지가 다른 노드들에도 존재하지 않으면 소유자 노드는 자신의 디스크에서 요구된 페이지를 가져와 주 노드에 전달하며 이와 같은 방식은 클러스터 전체의 메모리를 효율적으로 관리하는 방법으로 알려져 있다. 특히 소유자 노드가 디스크로부터 메모리로 페이지를 읽어올 때 이 페이지는 FIFO 리스트에서 유지되며 페이지 교체 시 우선적으로 제거하여야 한다.

페이지 교체 정책은 미래에 사용될 확률이 가장 낮은 페이지를 제거하는 것이며 이를 위한 기준으로 단일 서버에서는 시간 정보를 이용한다. 즉 가장 오래 전에 사용되었던 페이지를 우선적으로 제거하는 LRU 방식이 일반적으로 우수한 것으로 알려져 있다.

LRU 방식은 기본적으로 메모리에서 적중이 실패했을 때 이를 가져오기 위한 비용이 동일하다는 가정을 전제로 한다. 그러나 클러스터 시스템에서는 이러한 가정이 유효하지 않다. 적중되지 않은 페이지를 디스크에서 가져오는 것과 다른 노드의 메모리에서 가져오는 시간이 같지 않기 때문이다. 따라서 다른 노드에서 가져올 수 있는 페이지는 디스크에서 가져와야 하는 페이지와 비교하여 제거될 확률을 높이는 것이 타당하다. 한편 제거될 확률을 어느 정도 높여야 하는가는 사용자의 웹 문서 접근 확률 분포에 따라 결정된다.

제안된 페이지 교체 정책인 DREAM(Duplication RatE Adjustment Method)은 이러한 특성을 고려하여 페이지 교체 시 적용하며 동시에 웹 문서 접근 확률 분포에 따라 전체 메모리를 통해 중복되어 있는 페이지의 제거 확률을 조절하는 파라미터 α를 사용한다. 구체적으로 표현하면 메모리 페이지 교체 시 아래와 같은 방법을 이용하는데 이를 위해서는 관련 연구에서 언급된 Duplicate Elimination이나 Hybrid 방식과 마찬가지로 단일 LRU 리스트와 중복 LRU 리스트를 유지하여야 한다.

if ($W_s \times C_s < W_d \times C_d \times \alpha$)
  Replace the LRU single page
else
  Replace the LRU duplicate page

여기서 W_s와 W_d는 각각 단일 페이지와 중복 페이지에 대한 weighting factor로 현재 시간에서 마지막으로 참조된 시간을 뺀 값에 반비례한다. 또한 C_s는 단일 페이지 제거 후 다시 메모리로 가져오기 위해 필요한 시간이며 C_d는 중복 페이지 제거 후 다시 메모리로 가져오기 위해 필요한 시간으로 정의된다.

W_s, W_d, C_s, C_d는 모두 양수이며 C_s는 C_d에 비하여 항상 큰 특성을 지니고 있다. C_s는 전체 메모리 내에서 유일한 페이지를 제거한 후 다시 가져와야 하는 시간이므로 디스크에서 가져와야 하지만 C_d는 다른 메모리에 중복되어 있는 페이지를 가져오면 되므로 상대적으로 작기 때문이다. 따라서 C_s/C_d는 항상 1보다 크다고 할 수 있다.

특정한 α값에 대하여 제안된 방식의 행위를 살펴보기 위하여 먼저 α가 0인 경우를 고려해 보자. 이 경우 W_s와 C_s는 임의의 양수이기 때문에 $W_s \times C_s$ 역시 양수가 되고 따라서 교체 정책에서 조건문은 거짓이 되어 항상 else 문이 수행된다. 이는 무조건 중복 LRU 리스트에서 페이지를 제거하는 Duplicate Elimination 방법과 동일한 수행임을 알 수 있다. 또한 α가 1인 경우에는 단일 LRU 리스트와 중복 LRU 리스트의 기대값 E(p)를 비교하여 작은 쪽을 제거하는 방식을 채택하는 Hybrid 알고리즘이 된다.

α가 C_s/C_d인 경우를 생각해 보자. 이 경우는 if 문에서 W_s와 W_d를 비교하는 것과 동일한데 이는 단순히 시간 정보만을 비교하는 것으로 각 노드가 메모리 내에 존재하는 페이지를 제거할 때 LRU 방식을 사용함을 의미하며 Greedy Forwarding 방식이 됨을 알 수 있다.

나머지 다른 구간에서 α의 영향을 살펴보면 다음과 같다. α가 0보다 작은 경우에는 조건문의 우변이 음수가 되어 α가 0인 경우와 동일한 수행을 하므로 0인 경우와 차이가 없다. 또한 α가 0과 1 사이인 경우에는 Duplicate Elimination 방법과 Hybrid 방법의 사이에서 중복된 페이지의 제거율이 결정되며 α가 1과 C_s/C_d 사이인 경우에는 Hybrid 방법과 Greedy Forwarding 방법의 사이에서 중복된 페이지의 제거율을 가지게 된다. 마지막으로 α가 C_s/C_d 보다 큰 경우는 단일 페이지의 제거율을 높이는 것을 의미하나 이는 타당하지 않으므로 고려 대상이 되지 않는다.

그러므로 α의 유효한 범위는 아래 식과 같으며 이 중 α가 0인 경우는 Duplicate Elimination, 1인 경우는 Hybrid, C_s/C_d인 경우는 Greedy Forwarding 알고리즘이 됨을 알 수 있다. 그림 3은 유일한 페이지에 비해 중복된 페이지가 제거될 확률과 α 파라미터의 관계를 보여주고 있다.

$$0 \le \alpha \le C_s / C_d$$

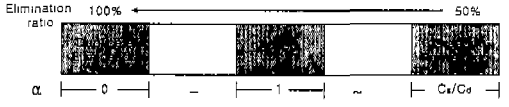

그림 3 중복 페이지 제거율과 α의 관계

기존에 제안되었던 방식들은 웹 문서 접근 확률 분포로 가정되는 Zipfian distribution의 skew 파라미터에 따라 우수한 성능을 나타내는 방식이 결정된다. 예를 들어, 서비스되는 웹 문서의 크기와 클러스터 전체 메모리의 크기가 유사할 경우 skew 파라미터가 0 근처에서는 Duplicate Elimination이, 2 근처에서는 Greedy Forwarding이, 그 사이에서는 Hybrid가 우수하다. 이러한 특성은 5장의 시뮬레이션에서 보여진다.

DREAM은 α 파라미터를 0, 1 또는 C_s / C_d로 설정함으로써 기존 알고리즘들 간의 전이가 가능하므로 다양한 skew 값에서 우수한 사용자 응답시간을 가질 수 있으며 α의 유효 범위 중 다른 값을 이용하면 기존 방식에서 표현되지 못한 중복 페이지의 제거율을 설정하는 것이 가능하다. 또한 기존의 Hybrid 방식에 비해 단지 α를 곱하는 시간만이 추가되므로 응답시간 향상을 위한 오버헤드가 거의 없다는 장점을 지니고 있다.

5. 성능 평가

이 장에서는 4장에서 제안된 DREAM 정책의 성능을 평가하기 위하여 다양한 시뮬레이션을 수행하였으며 그 결과를 보여준다. 시뮬레이션 프로그램은 프로세스 기반의 시뮬레이션 언어인 Simscript II.5를 통해 구현되었으며 실험을 위해 사용된 파라미터 값은 표 1과 같다.

서비스되는 파일들은 각 노드의 디스크마다 동일한 크기로 나누어져 배치되어 있으며 사용자의 요구는 부하 분배기에서 라운드 로빈 방식에 의해 각각의 웹 서버에 전달되는 것으로 가정하였다. 0에서 2 사이의 Zipfian skew는 대부분의 웹 사이트에서 발생하는 범위로 일반적으로 기존의 연구들에서 흔히 사용되는 가정이다.

표 1 시뮬레이션 파라미터

Parameter	Value
Number of nodes	3
Memory per node	64MB
Number of files	384
File size	512KB
Page size	8K
Message cost	1ms / page
Link bandwidth	15MB / sec
Disk bandwidth	10MB / sec

클러스터 전체 메모리의 크기와 서비스되는 파일들 크기의 비율은 특정한 skew 값에 대하여 메모리 갱신 정책을 결정하는 중요한 요소이다. 즉 노드의 수나 메모리의 크기가 증가하더라도 서비스되는 파일들의 크기가 동일하게 증가하면 최소의 응답시간을 갖는 메모리 갱신 정책의 결정에는 거의 영향을 미치지 않는다. 시뮬레이션의 간소화를 위해 노드의 수를 3으로 하였고 클러스터 전체 메모리의 크기와 서비스되는 파일들 크기의 비율을 동일하게 놓고 실험한 결과 그림 4와 같은 결과를 얻었는데 이는 노드의 수를 8개로 실험한 [13]의 결과와 유사하였다. 이 그림에서 Greedy Forwarding은 skew가 낮을 때 가장 좋지 않은 성능을 나타내지만 skew가 증가함에 따라 응답시간이 급격히 감소하는 특징을 지니고 있고 Duplicate Elimination은 skew가 낮을 때 좋은 성능을 나타내나 skew가 증가하더라도 응답시간 향상의 폭이 크지 않으므로 skew가 클 경우 좋지 않은 성능을 나타내는 것을 볼 수 있다.

Hybrid는 평균적으로 전 구간에 걸쳐 좋은 성능을 나타내지만 skew가 아주 낮을 때나 높을 때는 좋은 대안이 될 수 없다. 이에 비해 DREAM은 skew에 따라 α를 조절함으로써 모든 구간에서 좋은 응답시간을 가진다. 그래프에서 보여지고 있는 DREAM의 응답시간은 각각의 skew 마다 유효 범위 내에서 α를 0.1씩 증가시켜 가며 최소 값을 갖는 것을 선택하였다.

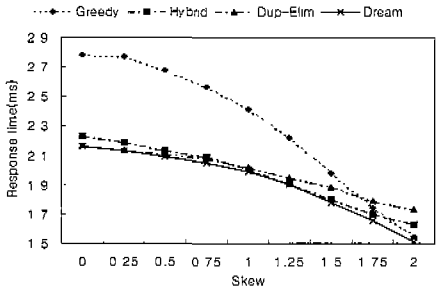

그림 4 Skew에 따른 응답시간 (파일 수: 384)

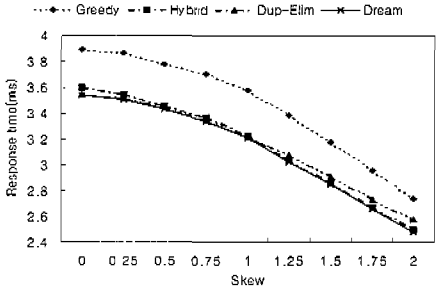

그림 5 Skew에 따른 응답시간 (파일 수: 768)

그림 4는 웹 서버 클러스터에서 서비스되는 파일들의 크기를 클러스터 전체 메모리 크기와 동일하게 하여 시뮬레이션을 수행한 결과이다. 이에 비해 그림 5와 그림 6은 서비스되는 파일들의 크기를 각각 200%와 50%로 설정하여 실험하였다.

그림 5에서 Greedy Forwarding은 skew가 높은 경우에도 최악의 성능을 가지는데 이는 서비스되는 데이타의 크기가 메모리 크기 보다 충분히 클 경우 중복을 최소화해야 디스크 접근 횟수를 줄일 수 있기 때문이다. 이와 같은 견지에서 볼 때 Duplicate Elimination은 효과적이고 따라서 skew가 높은 경우에도 Greedy Forwarding 보다 좋은 성능을 나타낸다.

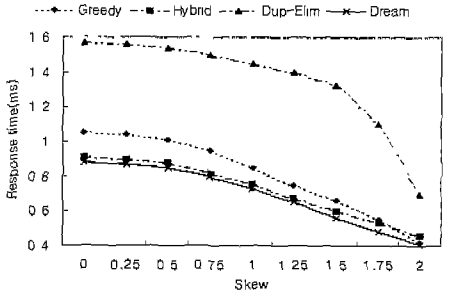

그림 6 Skew에 따른 응답시간 (파일 수: 192)

그림 6은 서비스되는 데이타 크기를 메모리 크기의 절반으로 줄였기 때문에 그림 5와 정 반대의 효과가 발생한다. 즉 Duplicate Elimination이 항상 최악의 응답시간을 가진다. 이는 메모리가 충분할 경우 디스크 접근 횟수가 줄어들게 되며 따라서 자주 요구되는 페이지를 다른 노드의 메모리보다는 자신의 메모리에서 직접 서비스하는 것이 효율적이기 때문이다.

그러나 이러한 경우라 할지라도 Greedy Forwarding

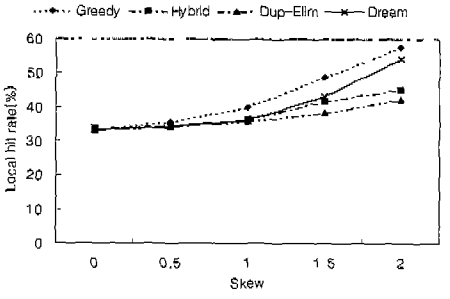

그림 7 로컬 메모리 적중률 비교

이나 Duplicate Elimination과 비교하여 Hybrid가 항상 좋은 응답시간을 갖지는 않는다. 이에 비해 DREAM은 웹 문서 접근 확률 분포가 다양한 웹 사이트들을 대상으로 좋은 응답시간을 유지할 수 있기 때문에 유리하다.

그림 7, 그림 8, 그림 9는 각각 skew의 변화에 따라 주 노드 메모리의 적중률, 다른 노드의 메모리 적중률, 그리고 디스크 접근률을 보여주고 있다.

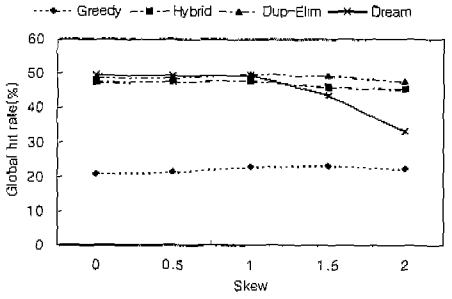

그림 8 글로벌 메모리 적중률 비교

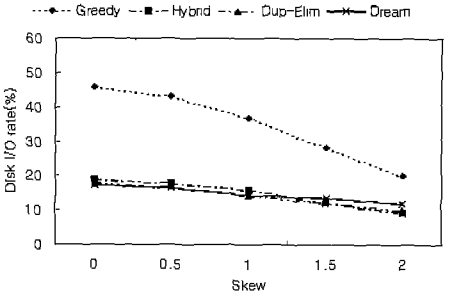

그림 9 디스크 접근률 비교

Greedy Forwarding의 경우 skew가 0일 경우에는 디스크 접근 횟수가 많으므로 좋지 않은 성능을 나타내나 skew가 2일 경우에는 로컬 메모리 적중률이 매우 높기 때문에 좋은 성능을 나타냄을 알 수 있다. 특히 그림 8에서 Greedy Forwarding의 클러스터 전체 메모리 적중률은 다른 알고리즘에 비해 매우 낮은데 이는 skew가 낮을 때는 디스크 접근률이 높기 때문이고 skew가 높을 때는 로컬 적중률이 높기 때문이다. 또한 Duplicate Elimination은 skew가 증가하더라도 로컬 메모리와 글로벌 메모리 적중률의 변화가 거의 없으며 디스크 접근률에 있어서도 일정한 수준의 적중률을 나타낸다.

Hybrid는 skew가 낮은 경우 Duplicate Elimination을, skew가 높은 경우 Greedy Forwarding에 가까운

적중률을 보이나 그 정도가 미미하다. 즉 skew가 높은 경우 로컬 적중률이 높은 것이 유리한데 Duplicate Elimination 보다는 좋으나 Greedy Forwarding에는 훨씬 못 미친다.

이에 비해 DREAM은 각 skew에서 로컬 메모리 적중률과 글로벌 메모리 적중률의 비율이 최적화 되어 있다. Skew가 높은 경우에도 Greedy Forwarding보다 로컬 적중률이 조금 낮기는 하나 그림 9에서 볼 수 있듯이 디스크 접근률이 낮으므로 향상된 응답시간을 가질 수 있다. 즉 그림 8에서 skew가 증가할수록 글로벌 메모리 적중률이 감소하는 것은 로컬 메모리의 적중률이 높아지기 때문이다.

6. 결 론

저가의 PC나 워크스테이션을 이용하는 클러스터링 기법은 비용 측면에서 매우 유리하며 특히 웹 서비스나 정보 시스템과 같이 고성능과 고가용도를 요구하는 응용 분야에서의 사용이 증가하고 있다. 초기에 클러스터 시스템의 성능을 제한하는 지배적인 요소는 프로세서의 속도였으나 현재는 그 영향이 감소하고 있으며 메모리 대역폭이 성능의 병목이 되고 있다. 따라서 효율적인 메모리 관리는 클러스터 시스템의 전체 성능을 향상시키는 중요한 요소라 할 수 있다.

본 논문에서는 웹 문서들의 접근 확률 분포가 변하더라도 우수한 사용자 응답 시간을 가질 수 있는 메모리 관리 정책을 제안하고 시뮬레이션을 통해 성능평가를 수행하였다.

향후 연구되어야 할 사항으로 최적의 α값을 제공하는 시뮬레이터의 제작을 들 수 있다. 이는 시뮬레이터를 통해 유효범위 내의 α를 순차적으로 적용하여 가장 좋은 값을 얻는 방식으로 구현될 수 있다. 또한 메모리의 결함 발생에 따른 중복 페이지 제거율의 변화가 고려되어야 한다. 메모리 데이타에서 결함이 발생할 경우 중복된 페이지는 손실비용이 적으나 유일한 페이지는 디스크를 접근하여야 하므로 중복 페이지의 제거율은 이를 고려하여 결정되어야 하기 때문이다.

참 고 문 헌

[1] R. Friedman and D. Mosse, "Load Balancing Schemes for High-Throughput Distributed Fault-Tolerant Servers," Journal of Parallel and Distributed Computing, pp. 475-488, Dec. 1999.

[2] V. Cardellini, M. Colajanni and P.S. Yu, "Dynamic Load Balancing on Web-server Systems," IEEE Internet Computing, pp. 28-39, May 1999.

[3] H. Zhu, T. Yang, Q. Zheng, D. Watson, O.H. Ibarra and T. Smith, "Adaptive Load Sharing for Clustered Digital Library Servers," Proceedings of the Seventh IEEE International Symposium on High Performance Distributed Computing, pp. 28-31, July 1998.

[4] V. Carellini, M. Colajanni and P. Yu, "Redirection Algorithms for Load Sharing in Distributed Web-server Systems," In Proceedings of the 19th IEEE International Conference on Distributed Computing Systems, pp. 528-535, May 1999.

[5] R. Buyya, High Performance Cluster Computing: Architectures and Systems, Prentice-Hall, p. 849, 1999.

[6] M. Feeley, et al., "Implementing Global Memory Management in a Workstation Cluster," In Proceedings of the 15th ACM SOSP, Dec. 1995.

[7] S. Venkataraman, M. Livny and J. Naughton, "Impact of Data Placement on Memory Management for Multi-Server OODBMS," In Proceedings of the 11th IEEE ICDE, Mar. 1995.

[8] S. Koussih, A. Acharya and S. Setia, "Dodo: A User-Level System for Exploiting Idle Memory in Workstation Clusters," 8th IEEE International Symposium on High Performance Distributed Computing, Aug. 1999.

[9] A. Leff, J. Wolf and P. Yu, "Replication Algorithms in a Remote Caching Architecture," IEEE Transactions on Parallel and Distributed Information Systems, Aug. 1993.

[10] D. Houzet, "A Shared Memory Model on a Cluster of PCs," 32th Annual Hawaii International Conference on System Sciences, Jan. 1999.

[11] X. Du and X Zhang, "Memory Hierarchy Considerations for Cost-effective Cluster Computing," IEEE Transactions on Computer, pp 915-933, Sep. 2000.

[12] M. Dahlin, R. Wang, T. Anderson and D. Patterson. "Cooperative Caching: Using Remote Client Memory to Improve File System Performance," In Proceedings of the First Symposium on Operating Systems Design and Implememtation, Nov. 1994.

[13] S. Venkataraman, M. Livny and J. Naughton, "Memory Management for Scalable Web Data Servers," 13th International Conference on Data Engineering, Apr. 1997.

[14] G. Zipf, Human Behavior and the Principle of Least Effort. Addison-Wesley, 1949.

[15] M. Oguchi and M. Kitsuregawa, "Using Available Remote Memory Dynamically for Parallel Data

Mining Application on ATM-Connected PC Cluster," 14th International Parallel and Distributed Processing Symposium, May 2000.
[16] 정지영, 김성수, "웹서버 클러스터를 위한 메모리 페이지 교체 정책", 2000년 한국정보과학회 추계학술발표 논문집, 제28권, 제1호, pp. 538-540, 2001. 4.

정 지 영

1998년 아주대학교 정보 및 컴퓨터공학부 (공학사). 2000년 아주대학교 정보통신전문대학원 정보통신공학과(공학석사). 현재 아주대학교 정보통신전문대학원 정보통신공학과 박사과정. 관심분야는 클러스터 시스템, 고가용성 시스템, 시스템 성능분석, 이동 컴퓨팅

김 성 수

1982년 서강대학교 전자공학과(공학사). 1984년 서강대학교 전자공학과(공학석사). 1995년 Texas A&M University, 전산학과(공학박사). 1983년 ~ 1986년 삼성전자(주) 종합연구소 컴퓨터연구실 (주임연구원). 1986년 ~ 1996년 삼성종합기술원 수석연구원. 1991년 ~ 1992년 Texas Transportation Institute 연구원. 1993년 ~ 1995년 Texas A&M University, 전산학과, T.A. 1997년 ~ 1998년 한국정보과학회, 한국정보처리학회 논문지 편집위원. 1996년 ~ 현재 아주대학교 정보통신전문대학원 부교수. 관심분야는 클러스터 시스템, 시스템 성능 분석, 고가용성 시스템, 이동 컴퓨팅.