# 워크스테이션 네트워크 기반 Serverless 네트워크 가상 메모리

강 현 수, 허 신
한양대학교 전자계산학과

# Serverless Network Virtual Memory on a Network of Workstations

Hyun Soo Kang, Shin Heu
Dept. of Computer Science and Engineering, Hanyang University

## 요 약

분산 시스템이 고성능의 네트워크로 연결되면서 네트워크 메모리(network memory)라는 새로운 메모리 계층이 등장하였다. 기존 운영 체제가 가상 메모리를 위해 로컬 하드디스크를 사용하는 반면, 네트워크 메모리는 네트워크로 연결된 각 노드들 중에서 유휴 상태에 있는 노드의 메모리를 가상 메모리로 사용한다.

네트워크 메모리를 활용하는 기존 연구의 대부분은 하나나 그 이상의 관리 서버 노드를 두어 관리 서버가 페이징 디바이스의 역할을 하는 원격 노드들을 관리하게 한다. 관리 서버 노드는 각 노드의 메모리 활용 상태를 점검하여 로컬 노드에게 페이지를 제공할 수 있는 원격 노드와의 중재 역할을 담당한다. 그러나 만약 관리 서버에 문제가 발생할 경우 관리 서버와 연결된 모든 노드들에게도 그 영향이 파급될 수가 있다.

본 논문에서는 serverless하게 노드들의 관계를 설정함으로 관리 서버 노드의 문제로 야기되는 다른 노드들의 다운 현상을 최소화 할 수 있는 serverless 네트워크 가상 메모리를 제시한다.

## 1. 서 론

기존의 운영체제는 가상 메모리를 사용하기 위해 물리적인 하드디스크를 사용한다. 그러나 분산 시스템(distributed system)이 고성능의 네트워크로 연결되면서 네트워크 메모리(network memory)라는 새로운 메모리 계층이 등장하였다.

네트워크 메모리는 네트워크로 연결된 각 노드들 중에서 유휴 상태에 있는 노드의 메모리를 작업 중인 다른 노드의 페이징 디바이스(paging device)로 제공하는 것을 의미한다[1][2][3]. 즉, 다른 노드의 메모리를 사용하여 가상 메모리의 역할을 수행할 수 있게 되는 것이다.

충분히 빠른 속도를 제공하는 네트워크가 설정이 되어 있을 경우에 네트워크 메모리를 사용하면 기존 하드디스크를 사용하는 것보다 빠른 속도로 페이지 교체를 할 수 있다. 물리적인 탐색 시간(seek time)을 제거할 수 있을 뿐만 아니라 로컬 디스크를 통한 데이터 처리 속도보다 네트워크 통신을 통한 데이터 처리 속도가 더 빠르고, 또한 급속하게 발달되고 있기 때문이다[3].

기존의 네트워크 메모리를 사용하여 가상 메모리를 구현하는 경우를 살펴보면 대부분이 중앙에 전역적인 메모리 관리 서버(global memory management server)를 두거나, 여러 개의 서버를 두어 노드를 관리하는 방법으로 이루어진다. 관리 서버는 각 노드의 메모리 활용 상태를 주기적으로 점검하여 로컬 노드(local node)가 페이징 디바이스를 요구할 경우, 가용 메모리(available memory)를 가진 원격 노드(remote node)를 로컬 노드의 페이징 디바이스로 연결시킨다. 페이징 서버가 된 원격 노드는 로컬 노드가 페이지인(pagein)이나 페이지아웃(pageout) 등의 서비스를 요구할 경우, 그에 대하여 적절한 응답을 하게 된다. 그러나 만약 관리 서버에 문제가 발생할 경우 로컬 노드와 페이징 서버 노드 사이에 정보 전달이 없어지게 되므로 작업을 중단하게 된다.

본 논문에서는 serverless하게 노드들의 관계를 설정함으로 관리 서버 노드의 문제로 야기되는 다른 노드들의 다운 현상을 최소화 할 수 있는 serverless 네트워크 가상 메모리를 제시한다.

## 2. 관련 연구와의 비교

현재 원격 메모리(remote memory)를 포함하여 네트워크 메모리의 개념을 가진 여러 가지 연구들이 진행되고 있다.

Comer와 Griffioen이 제시한 원격 메모리 모델(remote memory model)은 여러 개의 원격 메모리 서버(remote

memory server)를 두어 페이징과 관련된 작업을 수행한다[4].

각 노드들이 자원을 공유하려고 하면 반드시 원격 메모리 서버를 통해서 해야하고, 노드간에 직접적인 자원 공유는 불가능하다.

최근에 버클리 대학에서 제안된 Network RAM은 전역적인 자원 관리자(global resource manager)를 통해 가용 메모리를 가진 노드를 구분 짓게 한다[5].

본 논문에서는 이러한 기존 연구들과 비슷한 기능을 제공하는 네트워크 가상 메모리를 제안한다.

그러나 다음의 몇 가지 차이가 있다

기존 연구는 관리 서버를 다른 노드들과 구분하였다. 만약 관리 서버에 문제가 발생할 경우 그와 연결되어 있는 모든 노드들도 그에 영향을 받아 실행 중인 작업을 중지해야 하는 등의 문제가 발생하게 된다. 그래서 이러한 점을 해결하기 위하여 부수적인 알고리즘의 개발을 요구한다.

그러나 본 논문에서는 초기부터 관리 서버를 다른 노드들과 명백히 구분하지 않으므로 이와 같은 문제 발생의 여지를 최소화시킨다 단지 현재 연결된 로컬 노드와 페이징 서버 노드만이 영향을 받게 될 뿐이다.

또한 본 논문에서 각 노드는 자율적인 에이전트(autonomous agent)의 역할을 수행하게 된다 즉, 각 노드는 스스로 자기 자신의 메모리 사용을 점검하게 되며, 다른 노드를 위한 페이징 디바이스의 역할을 할 수 있는지를 판단할 수 있다. 이렇게 함으로 네트워크 트래픽(network traffic)이나 그 외 여러 가지 문제들로 인한 노드의 다운 등을 배제하게 되어 유동적으로 효과적인 페이징 디바이스 역할을 하는 노드를 설정할 수 있고, 이러한 복합적인 이유로 serverless 네트워크 가상 메모리는 새로운 노드의 확장성이 용이하다는 장점을 지닌다.

## 3. Serverless 네트워크 가상 메모리

### 3.1 구조 및 실행 방식

본 논문에서 제안하는 serverless 네트워크 가상 메모리는 관리 서버를 명확하게 구분하지 않는다. 즉, 모든 노드가 관리 서버의 역할을 수행하며, 이 중 페이지 서버로 설정된 노드만이 실제적인 페이징과 관련된 서버로서의 역할을 수행하게 된다.

이러한 페이징 서버를 설정하기 위하여 브로드캐스팅(broadcasting)을 사용한다.

로컬 노드는 추가적인 메모리가 필요할 경우, 다른 원격 노드들에게 브로드캐스팅 메시지를 보낸다.

메시지를 받은 원격 노드들 중에서 가용 메모리가 있을 경우, 응답 메시지를 로컬 노드에게 전송한다 만약 가용 메모리가 없을 경우에는 아무 응답을 하지 않는다

로컬 노드는 필요한 메모리 크기가 충당이 될 때까지 응답을 받고 응답을 해 온 원격 노드들을 페이징 서버로 설정한다.

그리고 브로드캐스팅을 할 때 대기 시간(waiting time)을 설정하여 시간이 종료될 때까지 아무런 응답 메시지를 받지 못하면 자신의 로컬 디스크를 페이징 디바이스로 사용한다. 대기 시간을 설정하는 이유는 네트워크 메모리의 성능 향상과 관련이 있다 즉, 네트워크 메모리를 사용할 때 하드디스크를 사용하는 것보다 오랜 시간이 걸린다면 네트워크 메모리를 사용하는 의미가 없다. 그러므로 대기 시간은 일반 디스크를 사용할 때의 시간을 고려하여 설정한다.

### 3.2 페이징 서버 설정

모든 노드에는 자기 자신의 메모리를 점검할 수 있는 check_status 데몬이 존재하여 주기적으로 메모리 활용 상태를 점검하게 된다.

- check_status 데몬

  자신의 메모리 활용 상태를 검검하며, 다른 노드로부터 브로드캐스팅 메시지를 받았을 때 요청된 메모리 크기만큼을 제공할 수 있을 경우 응답 메시지를 전송한다.

페이징 서버로 설정이 되면 page_manager 데몬이 로컬 노드와 통신을 하며 페이지를 관리한다.

- page_manager 데몬

  로컬 노드가 페이지인 요청을 할 경우 요청된 페이지 데이터를 전송하며, 페이지아웃 요청을 할 경우 데이터를 읽어 와서 메모리에 다시 저장한다.

### 3.3 로컬 노드

#### 3.3.1 메시지 구조

페이징 서버와 로컬 노드의 연결이 이루어지기 하기 위하여 (그림 1)과 같은 메시지를 로컬 노드가 페이징 서버에게 전달한다.

| Message type | Page number |
|---|---|

(그림 1) 메시지 구조

메시지는 메시지 타입과 사용되고 있는 페이지 번호로 이루어진다

메시지 타입으로는 newPage, pageIn, pageOut, freePage의 네 가지가 있다. newPage는 로컬 노드에서 페이징 서버에게 페이지 사용을 처음 요구하기 위해 사용되며. pageIn, pageOut, freePage는 현재 사용되고 있는 페이지와 관계한 요청 메시지 타입이다.

#### 3.3.2 구성 요소

로컬 노드에는 server_scanner와 paging의 2가지 데몬 프로

세스가 존재한다.

■ server_scanner 데몬

브로드캐스팅 메시지를 보내어 페이징 서버를 찾는 일을 담당한다. 필요한 메모리 크기가 충당이 될 때까지 원격 노드로부터의 응답을 받고 응답을 해 온 원격 노드들을 페이징 서버로 설정한다

전송되는 메시지는 다음의 구조로 이루어진다.

| newPage | -1 |
|---|---|

페이지 사용 가능 여부를 아직 모르기 때문에 페이지 번호는 초기 값 -1로 설정한다.

■ paging 데몬

페이지 부재(page fault)시 페이징 서버 노드의 페이지 사용을 위하여 메시지를 전송하며 paging 관리를 한다

전송되는 메시지는 다음과 같이 구분 지을 수 있다

페이지 번호 #1에 대하여 페이지인을 요청할 경우 다음과 같은 메시지를 보낸다.

| pageIn | #1 |
|---|---|

페이지 번호 #2에 대하여 페이지아웃을 요청할 경우 다음과 같은 메시지를 보낸다

| pageOut | #2 |
|---|---|

페이지 번호 #3을 더 이상 사용할 필요가 없을 경우 page free를 요청한다.

| freePage | #3 |
|---|---|

(그림 2)에서는 위에서 나열된 로컬 노드와 페이징 서버간의 메시지 교환의 흐름을 나타내고 있다. 로컬 노드에 페이지 부재가 발생했을 때, 필요한 페이지 번호를 페이징 서버에 보내 요청을 한다 그리고 도착할 페이지 데이터를 위해 사용하지 않고 있는 다른 페이지 데이터를 페이징 서버로 보낸다.

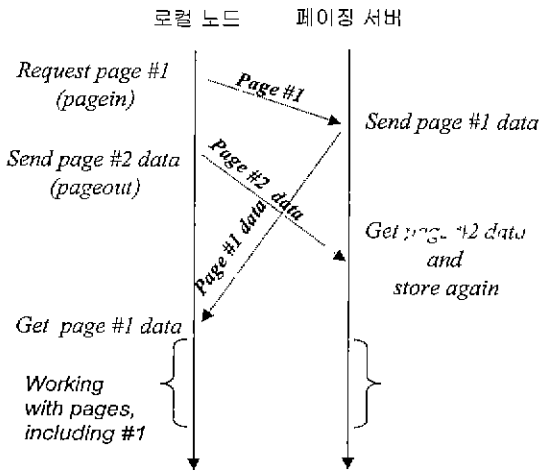
여기서 실제 주소(physical address)와 가상 주소(virtual address)와의 맵핑(mapping) 흐름은 생략하여 나타내지 않았다

## 4. 결 론

본 논문에서 제시한 serverless 네트워크 가상 메모리는 유휴 상태에 있는 노드의 메모리가 작업 중인 노드의 가상 메모리 역할을 수행하게끔 하여 일반적인 로컬 하드디스크를 사용할 때 보다 더 빠른 속도로 페이지를 제공한다. 또한 노드의 관계를 serverless하게 설정함으로 관리 서버 노드의 문제 발생으로 인하여 야기되는 데이터 손실 문제를 최소화 할 수 있다.

기존 연구들의 성능 평가에 의하면 약 8페이지를 교체(page swap)하기 위하여 하드디스크를 사용할 경우 약 9ms의 시간이 소요되었으나, 네트워크 메모리를 사용하여 약 2~3ms의 시간이 소요되었음을 보이고 있다 본 연구에서도 이와 비슷하거나 더욱 향상된 성능을 기대하고 있으며, 본 연구로 인하여 빠른 속도를 요구하는 공학 분야 및 여러 응용 분야에 속한 작업 실행의 효율성을 극대화 할 수 있을 것으로 기대한다

현재 본 연구는 Linux를 운영체제로 하는 펜티엄 PC를 대상으로 하여 구현 작업 중에 있으며, 최소한의 데이터 손실이라 할지라도 복구하여 신뢰성을 증진할 수 있는 알고리즘을 개발 중에 있다.




(그림 2) 메시지 교환 흐름도

## 참 고 문 헌

[1] Evangelos P Markatos, "Issues in Reliable Network Memory Paging," in Proc. of MASCOTS 96, San Jose, CA, Feb. 1995

[2] Thomas E Anderson, David E. Culler, David A Patterson, and the NOW team, "A Case for NOW," http://now.cs.berkeley edu/Case/case ps.

[3] Eric A Anderson and Jeanna M. Neefe, "An Exploration of Network RAM," http://www cs berkeley.edu/~eanders/projects/netram/cs252.ps

[4] D. Comer and J Griffioen, "A new design for distributed systems'The remote model," in Proc of the Summer 1990 USENIX conference, pp.127~135, June 1990

[5] Eric A Anderson, Jeanna M Neefe, Thomas E. Anderson, and David A Patterson, "Experience with Two Implementations of Network RAM," http://www cs.berkely edu/~eanders/projects/netram/usenix-netram.ps.