분산 공유메모리 시스템 상에서 홈-이전 프로토콜

김 태 규, 홍 영 식
동국대학교 컴퓨터공학과

A Home Migrating Protocol for Distributed Shared Memory Systems

Tae Gyu Kim, Young Sik Hong
Dept. of Computer Engineering, Dongguk Univ.

요 약

분산 공유메모리 시스템(DSM)의 성능 향상을 위해 일관성 모델의 측면에서 많은 연구가 진행되었다. 분산 공유메모리 시스템의 성능을 저하시키는 가장 큰 요인은 거짓 공유 문제와 별도의 통신비용 문제를 들 수 있는데, 동기화 연산에 의한 일관성 유지 방법, 홈-기반 접근방법 등의 보다 완화된 메모리 모델로서 이러한 문제점을 해결하려는 연구가 진행되어 왔고, 어느 정도 타당한 결과를 보였다. 본 논문에서는 동기화 연산에 의한 일관성 모델을 기초로 동적 홈-기반 접근방법을 제안하며, 이것은 홈에서의 이점 및 부하를 여러 프로세서에게 분산시켜 시스템 전반의 성능향상을 가져온다.

1. 서론

분산 시스템에서는 공유메모리를 구성하는 것이 어렵기 때문에 일반적으로 메시지 전송을 통해 프로세서간 통신이 이루어지고 있다. 그러나 메시지 전송에 의한 통신은 기존 병렬 알고리즘의 수정이 필요하며, 프로그래밍의 복잡도를 증가시키는 등 여러 단점이 있기 때문에 분산시스템 상에서 가상의 공유 메모리를 구성하는 분산 공유메모리 시스템이 제시되었다

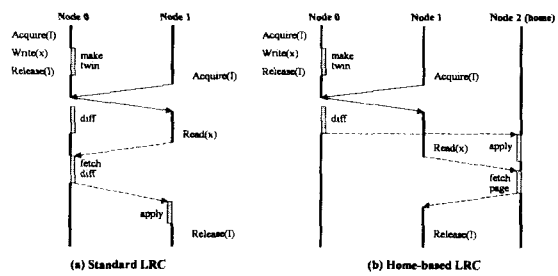
분산 공유메모리 시스템을 소프트웨어적으로 구성하는 경우, 프로토콜 상의 오버헤드와 프로그램 수행과는 관계없는 별도의 통신이 큰 문제점으로 지적되고 있는데, 특히 거짓 공유(false sharing)에 의한 핑퐁(ping-pong)현상은 많은 불필요한 메시지를 발생시켜 시스템 성능을 저하시킨다. 즉 두 개의 프로세서가 하나의 페이지를 공유하되 실제 참조하는 부분의 주소공간이 분리되어 있고, 각각의 프로세서에서 쓰기 연산을 동반한 메모리 참조 연산이 일어날 경우 실제 자신이 접근하는 주소공간이 변경되지 않았음에도 불구하고 메시지 전송을 통해 유효한 페이지를 유지해야 하는 문제점이 있다.

이러한 거짓공유 문제의 해결을 위해 TreadMarks 시스템[2]에서 다중 기록자 프로토콜을 제안하였다. 다중 기록자 프로토콜에서는 거짓 공유 상태에서 각각의 프로세서에게 동시에 별도의 동기화 과정 없이 페이지 내용에 대한 변경을 허용하는 것이다. 이러한 다중 기록자 프로토콜은 거짓 공유문제를 많은 부분 해결하고 있지만 유효한 페이지(clean-copy)를 얻기 위해서 각각의 다중 기록자들 모두에게 변경부분을 전송 받아야 하는 단점이 있어, 이로 인해 많은 프로세서가 참여한 DSM시스템 상에서 성능이 저하되는 것으로 밝혀졌다[7].

본 논문에서는 다중 기록자 프로토콜을 보완하기 위한 홈-기반 접근방법에 대해 살펴보고 이러한 방법에서 고정되어 있는 홈-프로세서의 역할을 수행 중 동적으로 변경시키는 알고리즘들을 제안하며 이에 따른 성능향상을 분석한다.

2. 기존연구

Shrimp 시스템[7] 에서는 자원의 변경부분을 생성하기 위한 연산을 줄이고 다중 기록자 프로토콜을 사용하지만 하나의 프로세서(home)에서는 항상 모든 변경부분이 적용되어 유효한 복사본을 유지하는 홈-기반 접근방법이 연구되었다. 자동-갱신 연성일관성(AURC) 과 홈-기반 지연 연성 일관성(HLRC) 모델이 그것이다. HLRC 모델의 경우 홈(home)은 자원의 변경부분을 모두 전송 받아서 항상 유효한 복사본을 유지하고, 동기획득을 원하는 프로세서는 홈으로부터 공유페이지 전체를 전송 받는 방법을 사용하고 있다. 이러한 방법은 변경부분 연산 오버헤드가 많이 줄어들고, 홈 프로세서에서는 페이지 부재가 일어나지 않으며, 동기획득을 원하는 프로세서는 한번의 메시지 전송으로 유효한 복사본을 전송 받을 수 있는 이점을 갖는다. 따라서 응용 프로그램에 따라 홈을 적절히 지정하면 홈의 이점에 의한 성능향상을 가져올 수 있는데, 예를 들어 읽기 연산 빈도가 높은 다수의 프로세서와 쓰기 연산 빈도가 높은 하나의 프로세서가 있는 경우 쓰기 연산 빈도가 높은 프로세서를 홈으로 지정하면 홈에 의한 성능향상을 기대할 수 있다.



[그림 1] Home-Based Lazy Release Consistency

홈-기반 접근방법에서는 어떤 프로세서가 페이지의 홈으로 지정되는지에 따라 시스템의 성능 차이가 클 수 있으며 응용 프로그램 수행 초기에 홈-프로세서의 위치가 정적으로 고정되기 때문에 여러 응용프로그램에 따른 메모리 접근 패턴에 적절히 적용시키기가 힘들다. 또한 이러한 문제점으로 인하여 많은 변경부분 메시지를 발생시킨다. 따라서 최근 홈-기반 프로토콜 상에서 고정되어 있는 홈을 프로그램 수행 도중 동적으로 이전시키는 여러 방법들이 제안되었다.

JUMP 시스템[1] 에서는 고정되어 있는 홈으로 인하여 다른 프로세서들이 많은 수의 변경부분 메시지를 발생시키고 이로 인하여 시스템 전반적인 성능 저하가 있다고 지적하고 이러한 변경부분 메시지를 줄이기 위해 홈-이전을 제안하고 있다. 즉 변경부분은 프로세서에서 공유메모리에 대한 쓰기 연산이 수행되어 메모리 내용을 변경시키고 이것을 홈-프로세서에게 전송하기 위해 생성되는데 이러한 메시지의 생성을 막기 위하여 페이지의 홈을 쓰기 연산이 발생한 프로세서로 이전시키는 방법을 사용하고 있다. 이와 같이 쓰기 연산이 수행된 프로세서로 홈이 이전되면 자신이 이미 유효한 페이지를 갖고 있기 때문에 별도의 변경부분을 생성 및 전송할 필요가 없게 된다.

그러나 이러한 방법은 다른 프로세서에서 쓰기 연산이 수행되기만 하면 홈이 이전되므로 매우 빈번한 홈-이전이 발생하게 되며, 이러한 홈이 이전한 사실을 다른 프로세서에게 알리기 위해 비용이 큰 방송통신을 자주 발생시키게 된다.

JIAJIA 시스템[6]에서는 홈-이전이 결정되는 시점을 장막(barrier)과 장막 사이에서 결정하고 있다. 즉 이 구간에서 공유페이지가 오직 하나의 프로세서에 의해서만 변경이 된 경우 장막 관리자가 페이지의 홈을 페이지를 변경한 프로세서로 이전시키는 방법을 사용한다. 또한 이전된 홈의 위치를 다른 프로세서에게 알리기 위하여 방송통신 등의 별도의 메시지를 발생시키지 않고 장막 관리자가 장막-이탈 메시지에 홈-이전 정보를 포함하여 전송하는 방법을 사용한다.

동적 홈-기반 지연 연성 일관성 모델(DHLRC)[9] 에서는 홈-기반 프로토콜 상에서 홈-프로세서가 갖는 장점을 최대한 모든 프로세서에게 적용시키기 위해 홈을 이전시킨다. 별도의 메시지 없이, 즉 기존의 동기 획득연산과 페이지 요구 연산 메시지를 이용하여 홈을 페이지 부재가 자주 일어나는 프로세서로 이전하는 것이다. 이를 위해 각 프로세서는 자신의 페이지 부재 회수를 기록하고 일정 임계값을 넘으면 기존의 홈 프로세서에게 이전을 요청한다. 후에 다른 프로세서에서 페이지 부재가 자주 일어나면 다시 홈이 해당 프로세서로 이전될 수 있다.

이때 얻을 수 있는 이점으로는 홈에서 페이지 부재가 일어나지 않는 점과 변경부분 생성을 위한 연산 부하를 줄일 수 있는 장점을 모든 프로세서가 공유할 수 있다는 것인데, 이것은 페이지 부재가 자주 일어나는 프로세서에게 페이지의 홈 권한을 이전하여 페이지 부재를 줄일 수 있게 한다. 또한 이러한 홈의 이전은 만약 프로그램 수행 초기에 지정되었던 홈이 더 이상 홈과 관련된 페이지에 접근을 하지 않는 경우, 홈의 권한을 페이지 부재가 자주 일어나는 프로세서로 이전함으로써 계속해서 변경부분을 전송 받아야 하는 오버헤드를 줄일 수 있다.

3. 홈의 특성을 고려한 홈-이전 프로토콜

홈-기반 프로토콜 상에서 홈은 항상 유효한 페이지를 유지하고 있기 때문에 이에 따른 몇 가지 특성을 갖는다. 먼저 홈에서는 메시지 전송을 필요로 하는 페이지 부재가 발생하지 않으며, 쓰기 연산을 수행 후에 변경부분 생성 및 전송이 필요하지 않다. 즉 홈-프로세서에서는 별도의 비용 없이 항상 유효한 페이지에 접근이 가능하다. 하지만 이를 위해 다른 모든 프로세서로부터 변경부분을 전송 받고 이것을 현재 페이지에 적용시켜 유효한 페이지를 유지해야만 한다. 또한 다른 프로세서들의 페이지 요청에 대해 전체 페이지를 전송해 주어야 하는 책임을 지고 있다. 따라서 홈-프로세서에서는 그 특성에 따라 페이지 부재 및 변경부분과 관련된 메시지의 수가 줄어들기도 하고 프로그램 수행과는 별도의 부하가 발생하기도 한다. 본 논문에서는 이러한 홈의 특성에 따라 홈이 적절히 이전될 수 있는 새로운 홈-이전 프로토콜을 제안한다.

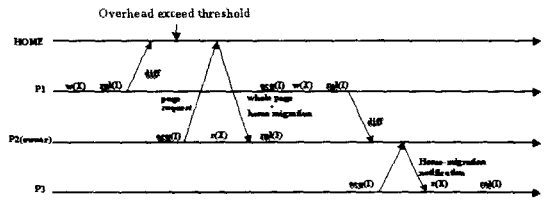
앞서 설명한 바와 같이 페이지의 홈은 유효한 페이지를 유지하기 위해 별도의 부하가 발생하게 된다. 이러한 부하는 다른 프로세서로부터 전송되는 변경부분 수신 및 페이지에 변경부분을 적용시키는 비용, 페이지 요청메시지의 수신 및 전체 페이지의 전송에 따른 비용이다. 즉
변경부분과 관련된 비용은
$C_{diff} = N_{RecvDiff} \times C_{ProcDiff}$ 이고,
페이지 전송과 관련된 비용은
$C_{ServPage} = N_{RecvPageReq} \times C_{TransPage}$ 로 생각해 볼 수 있다.
이러한 별도의 비용을 앞서 설명한 홈에서의 이점과 연관지어 홈에서의 전체 부하는

$$O_{Home} = (C_{Diff} + C_{ServPage}) - (N_{PageFault} \times C_{PageFault} + N_{Write} \times C_{CreateDiff})$$

즉, 페이지와 변경부분에 따른 비용에서 페이지 부재 및 변경부분 생성과 관련된 비용을 뺀 나머지를 전체 홈에서의 부하로 볼 수 있다. 이렇게 계산된 전체 홈에서의 부하가 양의 값을 갖는다는 것은 홈에서의 부하가 이점보다 크기 때문에 현재 홈으로 지정된 프로세서는 이러한 오버헤드에 의하여 프로그램의 수행시간이 길어지게 되고 이것은 전체 프로그램 수행시간을 증가시킨다.

따라서 여기서는 홈-이전의 기준을 홈에서의 전체 부하로 보고 이 값이 일정 임계값을 초과하는 경우 홈의 이전을 수행한다. 이러한 홈의 이전과 홈-이전에 대한 이전알림은 다음과 같은 절차를 통해 이루어진다.

1. 홈은 각각의 프로세서에 대한 페이지 요청 회수를 기록한다.
2. 주기적으로 홈의 부하를 계산하여 홈의 부하가 일정 임계값을 초과한 경우 홈 이전이 결정된다.
3. 홈 이전이 결정된 후에 일정 수 이상의 페이지 부재가 일어난 프로세서로부터 페이지 요청을 받은 경우 홈을 이것으로 이전한다. 홈-이전 정보는 전체 페이지에 포함되어 새로 전송된다.
4. 홈이 이전된 후에 변경부분 혹은 페이지 요청 메시지를 전송 받는 경우는 새로운 홈에게 전달한다. 메시지를 전송한 프로세서는 한번 전달 후에 새로운 홈을 인지하게 된다.
5. 다른 프로세서의 홈-이전 알림은 페이지의 소유자가 다른 프로세서로부터 동기획득 요청을 받은 경우 이에 대한 허가 메시지에 포함해서 전파한다.
6. 홈이 가장 최근에 페이지에 접근한 프로세서(소유자)에게 이전되었다면 별다른 홈-이전 알림이 필요하지 않다.
7. 홈이 소유자가 아닌 다른 프로세서로 이전된 경우, 새로운 홈이 소유자에게 동기 획득을 요청할 경우 홈-이전 정보를 포함하여 보내어 소유자에게 홈-이전을 알린다.



[그림2] 홈의 특성을 고려한 홈-이전의 예

[그림 2]는 페이지의 소유자에게 홈이 이전되고 다른 프로세서들이 홈-이전을 인지하게 되는 과정을 설명하고 있다. 홈 이전이 결정되면 페이지 요청한 프로세서 P2에게 페이지 전체와 함께 홈-이전 정보를 전송하여 홈을 이전한다. P1과 P3은 페이지의 소유자인 P2에게 동기획득을 하는 과정에서 홈-

이전을 인지하고 새로운 홈, P2에게 변경부분을 전송하게 된다.

4. 실험 결과 및 분석

본 논문에서 제안한 일관성 모델의 성능평가를 위해 Mint 시뮬레이터[4]를 사용하여 시뮬레이터의 후반부 부분에 일관성 프로토콜을 구현하여 실험하였다. 실험에 대한 기본적 연산의 인자는 [표 1.]과 같다. 성능을 측정하기 위해서는 SPLASH-2 벤치마크 프로그램[5]을 사용하였고 응용 프로그램으로는 Water Nsquared, Barnes-Hut, MP3D를 사용하였으며 문제 크기는 [표 2.] 와 같다

[표 1.] Basic Operation Cost

Operation	microsecond
Message Latency	50
Page Transfer	92
Receive Interrupt	430
Twin Copy	120
Diff Creation	560
Diff Application	215
Page Fault	290
Page Invalidation	200

[표 2.] Problem Size

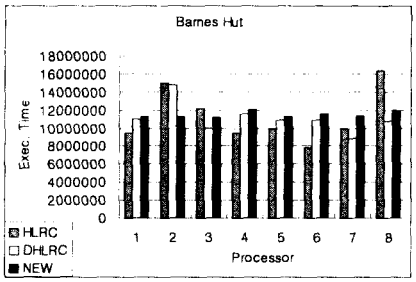
Application	ProblemSize
Water-Nsquared	343 mole
Barnes-Hut	tk14.O
MP3D	20000 mols

[표 3.] Simulation Result on Prcessor 8

App.	Alg.	Exec.Time	SpeedUp	Message
Water Nsquared	HLRC	3025380	4.10	9912
	DHLRC	2948347	4.15	9635
	NEW	2813042	4.30	9424
Barens Hut	HLRC	26234231	3.19	89653
	DHLRC	24352893	3.30	93363
	NEW	22600328	3.70	85361
MP3D	HLRC	1694321	4.1	4347
	DHLRC	1636231	3.98	4352
	NEW	1401611	4.6	3975

[표 3.]은 각 일관성 모델에 대해 세 가지 벤치마크 프로그램에 대한 실험 결과이다. 8개의 프로세서를 사용하였을 때 결과이며, 프로세서 수에 따른 성능향상(Speed Up), 수행 중 발생한 메시지 수로 비교하였다.

Water Nsquared 의 경우 제안된(NEW) 모델은 HLRC, DHLRC 모델에 대해 각각 8.1%, 3.5%의 수행시간 감소와, 5.9%, 5%의 메시지 수 감소를 보이고 있다. 홈을 고정하고 수행하는 HLRC 모델에 비해 메시지 수가 줄어들고 이것은 전체적인 성능향상으로 나타나고 있다. Barnes-hut 의 경우에도 HLRC, DHLRC 모델에 비해 14.1%, 10.3%정도의 성능향상을 보이고 있다. 이것은 Water 의 경우보다 모든 프로세서들이 보다 적은 균일한 수행시간을 유지하고 있기 때문이다. 또한 MP3D의 경우 HLRC, DHLRC 모델에 비해 약 10.3%, 12.1%위 수행시간 감소를 보이고 있다. 제안된 모델에서 얻을 수 있는 성능 향상은 각각의 프로세서가 홈-프로세서의 부하를 줄여 균등한 수행시간을 유지하는 것이다. 예를 들어, [그림 3.]의 HLRC 모델에서 P8의 경우 페이지 부재 수는 가장 적지만 통신시간은 매우 길게 나타났다. 이것은 응용프로그램 수행과 별개의 홈-유지와 관련된 부하가 크기 때문이 이로 인하여 전반적인 응용프로그램의 수행시간이 증가하게 된다. 하지만 제안된 모델의 경우는 각각의 프로세서가 그 부하를 분담하고 있기 때문에 통신시간을 모든 프로세서가 비슷한 수준으로 유지함으로써 성능향상을 얻을 수 있다.



[그림3] 프로세서당 통신비용
(Barnes-Hut, 프로세서 개수 = 8)

5. 결론

본 논문에서는 홈-기반 접근방법에서 홈의 특성에 따라 홈을 동적으로 유지하는 새로운 홈-이전 프로토콜을 제안하였다. 홈-기반 접근방법에서 홈은 페이지 부재를 갖지 않는 장점이 있지만 이러한 상태를 유지하기 위해 공유메모리의 변경이 일어난 모든 프로세서에게 변경부분을 전송 받아야 하는 부하가 있다. 반면, 본 논문에서 제안한 홈-이전 정책에서는 이러한 홈의 이점 및 부하를 고려해서 홈을 이전하며 홈이 고정되어 발생하는 불필요한 오버헤드를 어느 정도 보완하여 시스템의 성능을 향상시킨다. 또한 홈 이전을 위해 별도의 메시지가 필요하지 않는 점도 장점이 될 수 있다.

향후 과제로는 응용 프로그램의 메모리 접근 형태와 제안된 홈-이전 방법과의 관계를 분석하고 보다 효율적인 이전정책에 따른 실험을 해보아야 할 것이다.

참고문헌

[1] Benny Wang-Leung Cheung, Cho-Li Wang and Kai Hwang, A Migrating-Home Protocol for Implementing Scope Consistency on a Cluster of Workstations, The 1999 International Conference on Parallel and Distributed Techniques and Applications.

[2] Pete Keleher, Alan L. Cox, Sandhya Dwarkadas, and Willy Zwaenepoel, TreadMarks: Shared Memory Computing on Networks of Workstations. IEEE Computer, Vol 29, No. 2, pp. 18-28, 1996

[3] Kai Li and Paul Hudak, Memory Coherence in Shared Virtual Memory Systems. ACM Transection on Computer Systems, Vol 7, No.4, November 1989.

[4] Jack E. Veenstra and Robert J. Fowler, MINT Tutorial and User Manual. Technical Report 452. The University of Rochester Computer Science Department, September 1993.

[5] Steven Cameron Woo, Moriyoshi Ohara, Evan Torrie, Jaswinder Pal Singh, and Anoop Gupta, The SPLASH-2 Programs: Characterization and Methodological Considerations. Proceedings of the 22nd Annual International Symposium on Computer Architecture, pages 24-36, June 1995.

[6] Weiwu Hu, Weisong Shi and Zhimin Tang, Home Migration in Home-Based Software DSMs, Proceedings of the 1st ACM workshop on Software DSM system, 1999.

[7] Yuanyuan Zhou, Liviu Iftode and Kai Li, Performance Evaluation of Two Home-Based Lazy Release Consistecny Protocols for Shared Virtual Memory Systems. Proceedings of the 2nd Symposium on Operating Systems Design and Implementation, October 1996.

[8] 김태규, 홍영식, 소프트웨어 분산 공유메모리 시스템 상에서 효율적인 일관성 모델, 1998년도 한국 정보과학회 학술발표논문집 Vol. 25, No. 2.