추적지향 시뮬레이션에 의한 TLB 성능 분석

박 장 석†

요 약

본 논문에서는 가상기억장치의 가상주소 번역 성능을 개선시켜 주는 TLB(Translation Lookaside Buffer)의 성능 분석을 통하여, 향후의 추가적인 성능 개선 방향을 진단한다. ATUM 추적 입력을 사용한 추적지향 시뮬레이션에 의해, 연관성, 엔트리 수, 페이지 크기 증가에 따른 TLB 실패율의 변화 정도를 측정하고, 기존 성능 향상 방법의 한계와 문제점을 고찰한다. 그리고, TLB 성능 향상 방법의 향후 연구방향에 대해 논한다.

The TLB Performance Analysis by a Trace-driven Simulation

Jang Suk Park†

ABSTRACT

In this paper, we present the performance enhancement directions of TLB in terms of TLB performance analysis. We analyze the characteristics of TLB miss ratio according to the increase of associativity, the number of entries and page sizes by the trace-driven simulation to use the ATUM trace inputs. We also discuss with the problems and limits of the classic methodology for TLB performance enhancement. Finally, the research direction is presented.

1. 서 론

TLB는 최근에 사용된 가상주소에 대한 물리주소를 저장하는 일종의 캐시 기억장치이다. 가상 기억장치를 지원하는 대부분의 컴퓨터는 주소 번역 시간을 감소시키기 위하여 프로그램의 시간 구역성(temporal locality) 및 공간 구역성(spatial locality)을 이용한 TLB를 지원한다. 일반적으로 가상주소의 번역은 세그먼트 표와 페이지 표라고 하는 두 번의 주소 번역 표의 접근을 필요로 하는데, 이 오버헤드는 물리 캐시를 접근하기 전에 TLB 내에 저장된 엔트리들을 하드웨어적으로 조사하게 함으로써 감소될 수 있다. 이와 같은 이유로 TLB는 중앙처리장치의 중요한 경로에 위치하며, 가상 기억장치를 지원하는 시스템에서 필요한 자원의 하나로 간주되고 있다. TLB의 성능은 TLB 접근 시에 발생하는 실패 회수를 전체 접근 수로 나눈 값의 백분율을 의미하는 실패율(miss rate)로 표현한다. 평균적으로 TLB의 실패율은 약 0.1-2% 정도[1]이나, TLB 실패 서비스에 소요되는 시간을 의미하는 TLB 실패 패널티는 VAX 8800의 경우 모든 기계 사이클의 4.6% 정도를 소비한다[2]. 또, 대용량 데이터베이스 관련 응용프로그램에서는 5-18% 정도나 소비[3] 된다고 보고되고 있어, TLB 실패에 대한 오버헤드는 고성능 컴퓨팅의 주요한 장애가 되어 이를 감소시키기 위한 연구가 필요되고 있다.

또, 주소 공간 사용면에서 변화가 예상되는 최근의 컴퓨팅 추세에도 TLB의 실패 패널티를 감소시키는 연

† 정 회 원 : 정보통신연구관리단 정보기술평가1실
논문접수 : 1997년 9월 25일, 심사완료 : 1997년 11월 24일

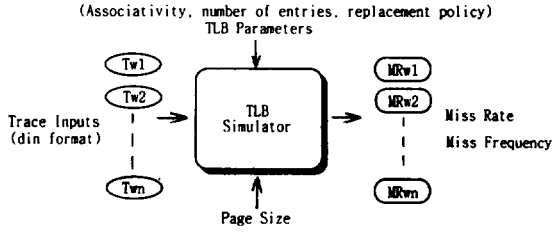
구가 필요한 요인이 되고 있다. 객체지향 시스템, 기억장치 사상 파일(memory mapped file), 멀티미디어 응용프로그램, 그리고 분산 응용프로그램 등은 프로세스에 의해 사용되는 주소 공간을 더 많이 요구[3]하고 있어서, 가상 기억 메모리 공간의 구역성을 감소시키고 있다. 이러한 환경 변화는 TLB의 사상 범위를 더 넓게 요구하게 되고, 이로 인하여 TLB 실패율은 상대적으로 증가하게 된다. 또한, 현재의 SMP(Symmetric Multiprocessor) 또는 MPP(Massively Parallel Processor)의 운영체제로 부각되고 있는 마이크로커널의 기본 방향은 기계에 독립적인 커널 기능을 마이크로커널 위에서 동작하는 서버로 전이시켜 TLB의 사용을 증가시키고 있다. 이러한 사실들은 하드웨어 관리 TLB 보다 하드웨어 설계를 단순화시키고, 페이지표의 구조에 융통성을 제공하나, 실패 패널티는 상대적으로 큰 소프트웨어 관리 TLB[4~8]를 사용하는 프로세서의 경우 심각한 성능 문제를 야기시킬 수 있다.

따라서, 본 논문에서는 TLB의 실패 패널티를 감소시키는 선행연구로 연구방향 도출을 위하여 추적지향 시뮬레이션에 의해 전통적인 TLB 파라미터 변경에 따른 상용 마이크로프로세서의 TLB 성능 분석에 의하여 기존의 성능 향상 방법의 한계와 문제점을 진단하고, 소프트웨어 관리 TLB에 적용할 수 있는 추가적인 성능 개선방안에 대해 논한다. 2장에서는 본 논문에서 사용한 시뮬레이션의 방법에 대해 살펴보고, 3장에서는 하드웨어 기반의 여러 가지 성능 향상 방법을 추적 지향 시뮬레이션에 의해 성능 향상 결과를 고찰한다. 마지막으로 4장에서는 결론으로 시뮬레이션 주요 결과의 종합적 의미를 고찰하고, 향후 연구방향에 대하여 논한다.

2. 시뮬레이션 방법

2.1 시뮬레이션 도구

TLB 성능을 평가하기 위하여는 TLB 시뮬레이터가 필요하다. 본 연구에서는 추적 지향 캐시 시뮬레이터인 Dinero III [1, 21]를 수정하여 TLB 시뮬레이션을 수행하였다. 수정된 시뮬레이터는 Dinero III과 마찬가지로 추적 주소를 입력으로 받아서 정해진 TLB 파라미터와 페이지 크기에 대해 TLB 실패율과 실패 빈도수를 출력으로 생성한다. Dinero III와의 차이점은 Dinero III가 가상주소의 블록 번호를 비교하는데 반하여, 수정된 TLB 시뮬레이터는 가상주소의 세그먼트 번호 및 페이지 번호를 비교하는 점이 다르다.



(그림 1) TLB 시뮬레이터의 입력과 출력
(Fig. 1) The inputs and outputs of TLB simulator

(그림 1)은 TLB 시뮬레이터의 입출력 파라미터의 관계를 도시한 것이다. TLB 시뮬레이터는 시스템 페이지 크기, 연관성 정도, 엔트리 수, 페이지 대체 알고리즘 등과 같은 TLB 파라미터가 정해졌을 때, 입력으로 주어지는 din 포맷[21]의 Tw1, Tw2, ..., Twn과 같은 작업 부하에 대한 추적 입력에 대해 각각 MRw1, MRw2, ..., MRwn과 같이 TLB을 접근할 시의 실패율과 실패 빈도수 등의 결과를 출력한다.

2.2 추적 입력

추적 입력으로는 ATUM(Address Tracing Using Microcode) 추적 데이터 [1, 21, 22]을 사용하였다. ATUM의 기본적 방법은 운영체제보다 낮은 수준에서 주소 정보를 추적 기록하는 것이다. ATUM 추적 데이터는 컴퓨터의 마이크로코드를 변경하여 프로세서가 주기억장치를 접근하는 모든 주소의 추적 정보를 주기억장치의 지정된 장소에 저장하게 하고, 이를 주기적으로 디스크 또는 테이프 장치로 쓰기를 수행한 것이다[22]. 이 방법은 운영체제를 포함한 컴퓨터에서 동작하는 모든 프로그램의 완전한 추적을 기록할 수 있다. 본 논문에서 사용한 ATUM 데이터는 VAX 8200의 마이크로코드를 수정하여 여러 가지의 다양한 부하를 수행한 결과를 기록한 것으로, Patterson[1]이 제공하는 추적 결과를 사용하였다. 왜냐하면, 이것은 운영체제의 동작을 포함하는 유일한 공개된 정보이기 때문이다. ATUM 추적에 있는 주소는

VAX의 가상주소이며, ATUM 데이터는 과학 응용프로그램, VAX/VMS 운영체제 하에서 수행된 시뮬레이션, SPICE 회로 분석 프로그램, 파스칼 컴파일러 등의 시스템 프로그램을 수행 시에 발생한 기억 장치의 참조 자료를 모은 것이다.

시뮬레이션에 사용한 ATUM 추적 데이터는 〈표 1〉에서와 같다. 추적 데이터는 각각 30-40 만번 정도의 기억 장치 참조를 가진다. 〈표 1〉에서 첫 번째 행은 각 추적 데이터의 이름이고, 두 번째 행은 추적 데이터 중 명령어 참조의 백분율을 나타내고, 세 번째 행은 추적 입력의 기억장치 참조 수를 나타낸다.

〈표 1〉 트레이스 설명
〈Table 1〉 Trace description

Trace Name	Instruction References(%)	Number of References
dec0.000	50	361,982
fora.000	52	387,934
forf.000	52	368,212
fsxzz.000	51	239,334
ivex.000	60	341,968
lisp.000	58	291,390
macr.000	55	342,828
memxx.000	49	444,849
mul2.000	56	372,104
mul8.003	46	429,432
pasc.000	46	422,090
spic.000	50	446,701
ue02.000	56	357,810

2.3 TLB 성능 측정 방법

시뮬레이션 과정에서는 〈표 2〉의 마이크로프로세서 TLB 파라미터를 기본으로 TLB 성능을 측정하였다. 〈표 2〉에서 2-way, 4-way 등은 부분 연관의 정도를 나타내는 것이고[9], 'unified'는 명령어 TLB와 데이터 TLB가 분리되지 않은 통합 TLB을 말한다. TLB 엔트리의 대체 알고리즘으로는 가장 많이 사용하는 LRU 방식을 사용하였다. 연관성 증가와 엔트리 수 증가에 따른 TLB 성능 측정에서 페이지 크기는 현재 가장 많이 사용하고 있는 4 K 바이트를 사용하였다.

〈표 2〉 상용 프로세서의 TLB 특성
〈Table 2〉 Number of TLB slots for commercial processors

Processor	Associativity	Instruction TLB Size	Data TLB Size	Page Size
IBM RS/6000	2-way	32	128	4K
MIPS R2000	full	64 unified	–	4K
MIPS R4000	full	48 unified	–	4K-16M
HP 9000/700	full	100	100	4K-32K
INTEL 486	4-way	32 unified	–	4K

TLB 성능 척도로는 〈표 1〉에 있는 13 개의 작업 부하에 대해 각각 실패 빈도수, 실패율, 3C 빈도수를 측정하여, 각각의 산술 평균 값을 기준으로 고찰하였다. 실패 빈도수는 작업부하의 수행 과정에서 발생한 TLB 실패 회수를 말하며, 실패율은 작업 부하의 실패 회수를 TLB 전체 접근 회수로 나눈 값의 백분율을 의미한다. 3C 빈도수로는 TLB의 특성 변화에 따른 초기접근실패의 영향을 관찰하기 위하여 충돌 및 용량 접근실패는 합산하여 표현하고, 초기접근실패만 분리하여 나타내었다.

그리고, 시뮬레이션 과정에서 사용되는 TLB 파라미터는 표현의 간략화를 위하여 영문 명칭의 머리 글자를 인용하여 아래와 같이 정하였다. 'u'는 통합 TLB(unified TLB)을 의미하고, 'i'와 'd'는 분리 TLB (split TLB)에서 각각 명령어 TLB(instruction TLB)와 데이터 TLB(data TLB)를 의미한다. 'a'는 연관성을 의미하고, 'i', 'd', 'a'에 같이 쓰여진 숫자는 〈표 2〉에 있는 TLB 파라미터의 수치 값으로 크기의 정도를 나타낸다. 예를 들면, a8은 연관성의 정도가 8인 것을 의미하며, i100-d100-a100은 명령어 TLB의 엔트리 수가 100이고, 데이터 TLB 엔트리 수가 100이며, 연관성 정도가 100인 경우, 즉, 완전 연관인 것을 의미한다. 또, 수치 값 대신에 a*, u*와 같이 표현한 것은 각각 연관성의 변화, 통합 TLB 크기의 변화 등과 같이 시뮬레이션에 사용된 모든 경우를 지칭하는 것이다.

3. 시뮬레이션 결과 및 고찰

3.1 개 요

TLB의 특성은 연관성 정도, 엔트리 수, 시스템 페이지 크기 및 대체 알고리즘(replacement algorithm)과

같은 파라미터에 의해 영향을 받는다. 이와 같은 파라미터의 성능 척도는 일반적으로 실패율에 의해 측정되며, 실패율은 TLB가 사상하는 주기억장치의 영역 정도에 많은 영향을 받는다[18]. TLB의 사상 영역은 시스템의 페이지 크기와 TLB 엔트리 수의 곱으로 표현하며, TLB의 사상 영역이 증가하면 상대적으로 실패율은 감소하고, 사상 영역이 감소하면 실패율은 높아진다.

TLB의 성능 향상 요인으로는 연관성 증가, 엔트리 수 증가, 페이지 크기 증가 등이 있다. 엔트리 수가 증가하거나 또는 시스템의 페이지 크기가 증가하면 TLB의 사상 영역이 확대되므로, 상대적으로 성능 향상이 이루어진다. 또, 연관성이 증가하면 TLB 실패 시에 가상주소에 대한 새로운 번역 값을 저장하기 위해서 대체하는 TLB 엔트리의 범위가 증가하기 때문에, 연관성이 증가할 수록 엔트리 중 가장 오랜 것을 대체할 수 있게 되고, 이에 따라 프로그램 구역성에 의해 성능이 증가하게 된다.

그런데, 일반적인 계층적 기억장치의 성능 평가에서 연관성 또는 엔트리 수를 증가하는 것은 실패율은 감소시키나, 하드웨어 비용을 증가시키고, 상대적으로 접근 시간을 증가시켜서 전체적인 관점에서 보면 상대적으로 속도가 감소될 수도 있다고 알려져 있다[1, 13, 19]. 따라서, 전체적인 성능 개선은 이와 같은 상황을 고려하여 총체적으로 고려하여야 하나, 본 연구에서 실패율을 성능 비교의 기본 척도로 고려한다.

3.2 연관성을 증가하는 방법

〈표 3〉은 연관성 증가에 따른 TLB 실패율 측정 결과이다. 시뮬레이션에서는 TLB 엔트리 수가 일정한 경우에 연관성의 변화에 따른 TLB 실패율의 변화를 관찰하기 위하여, 〈표 2〉에서 TLB 엔트리 수가 최대인 경우와 최소인 경우를 선택하여, 연관성 증가에 따른 여러 가지의 TLB 성능 척도를 측정하였다.

〈표 3〉에서 a1인 경우는 직접 연관을 의미하고, a 값의 크기가 TLB 크기와 같은 경우, 즉, u32-a32인 경우와 i100-d100-a100인 경우는 완전 연관을 의미한다. 첫째 행은 TLB 파라미터로 연관성과 TLB 크기를 나타내며, 두 번째 행은 평균 기억장치 참조 수에 대한 실패 빈도수이고, 세 번째와 네 번째 행은 전체 실패 빈도수 중에서 용량접근실패와 충돌접근실패의 합과 초기접근실패의 빈도수를 나타낸다. 다섯 번째 행은 TLB 실패율이며, 마지막 행은 초기접근실패로 인한 실패 빈도수가 전체 실패 수에 대한 비율을 백분율로 표현한 것이다.

〈표 3〉 연관성 증가에 의한 평균 TLB 성능 척도 변화
〈Table 3〉 Average TLB performance metrics variation versus k-way set-associative TLB

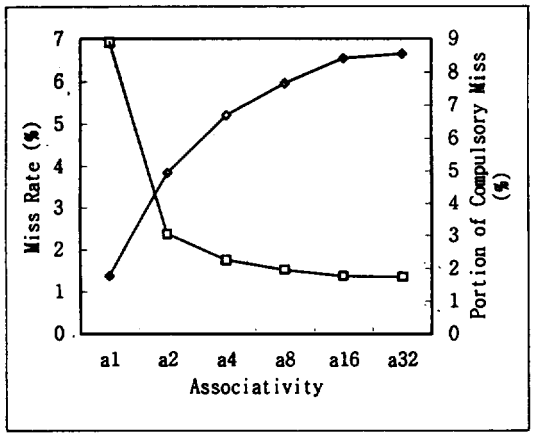
TLB Parameter	Miss Frequency	Capacity Miss & Conflict Miss	Compulsory Miss	Miss Rate (%)	Portion of Compulsory Miss(%)
u32, a1	24,406	23,973	433	6.92	1.77
u32, a2	8,761	8,328	433	2.38	4.94
u32, a4	6,464	6,031	433	1.76	6.69
u32, a8	5,644	5,211	433	1.53	7.66
u32, a16	5,134	4,701	433	1.38	8.42
u32, a32	5,057	4,624	433	1.36	8.55
i100, d100, a1	3,962	3,529	433	1.09	10.9
i100, d100, a2	2,098	1,665	433	0.57	20.6
i100, d100, a4	1,624	1,191	433	0.44	26.6
i100, d100, a25	1,325	892	433	0.36	32.6
i100, d100, a50	1,316	883	433	0.36	32.9
i100, d100, a100	1,312	879	433	0.36	33.0

주) 평균 기억장치 참조 수 : 369,741

(그림 2)와 (그림 3)은 각각 u32-a*와 i100-d100-a* 인 경우의 실패율과 초기접근실패 비율의 관계를 도시한 것이다. (그림 2)에 의하면, u32-*인 경우에 연관성이 증가하는 모든 경우의 실패율은 감소하지만, 상대적으로 초기접근실패의 비율은 증가함을 알 수 있다. 그러나, 전체적인 비율은 10% 미만으로 적은 부분을 차지한다. (그림 3)의 i100-d100-a*인 경우에도 전체적인 결과는 u32-a*인 경우와 유사하다. 차이점으로는 전체적으로 실패율이 낮다는 것과, 이에 따라 상대적으로 초기접근실패의 비율이 높다는 것이다. u32-a*에 비하여 실패율은 더 낮으며, 각 경우에 있어서 초기접근실패 비율이 더 큰 것을 알 수 있다. 이것은 연관성의 증가로 인하여 실패율은 낮아지나, 상대적으로 초기접근실패의 비중은 증가함을 의미한다.

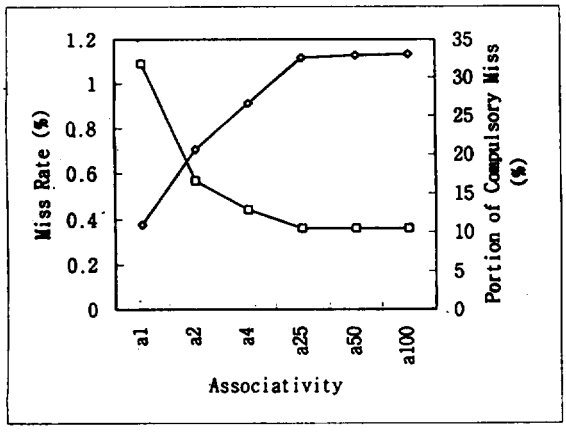
그리고, (그림 2)의 u32-a*일때, 연관성이 1인 경우 즉 직접 사상인 경우와 연관성이 2인 경우에 실패 발생 빈도는 급격히 감소하나, 연관성이 TLB 용량의 반 이상인 경우에는 추가적으로 연관성이 증가하여

완전 연관이 되는 경우에도 성능 개선 효과는 크지 않은 것을 알 수 있다. 또, (그림 3)의 i100-d100-a*인 경우에도 연관성 정도가 25, 50, 100 인 경우에 연관성의 증가에 따른 실패 발생 빈도의 감소는 크지 않은 것으로 관찰되었다. 따라서, 이 경우 부가되는 하드웨어 비용에 비하여 상대적으로 성능 향상의 효과는 작은 것을 의미한다.



(그림 2) u32-a*인 경우의 실패율과 초기접근실패 비율과의 관계

(Fig. 2) The relationship between miss rate and the portion of compulsory miss in u32-a*



(그림 3) i100-d100-a*인 경우의 실패율과 초기접근실패 비율과의 관계

(Fig. 3) The relationship between miss rate and the portion of compulsory miss in I100-d100-a*

연관성 증가에 따른 시뮬레이션 결과를 종합하면 다음과 같다. 연관성 증가에 의한 실패율 감소는 충돌 및 용량접근실패의 감소에 의한 것이다. 초기접근실패 빈도수는 연관성의 변화에 관계없이 일정하므로, 연관성이 증가함에 따라 초기접근실패의 상대적인 비율은 증가한다. 또, 연관성이 TLB 크기의 1/2 정도의 연관성을 갖는 경우에 추가적인 연관성 증가로 인한 실패 빈도수 감소 효과는 얻기 어려운 것으로 판단된다.

3.3 엔트리 수를 증가하는 방법

〈표 4〉는 시뮬레이션 환경에서 페이지 크기가 4 K 바이트인 경우에 엔트리 수 증가에 따른 TLB 실패율의 변화를 측정한 결과이다. 시뮬레이션에서는 TLB 연관성을 a4로 고정하고, 엔트리 수 변화에 따른 TLB 실패율의 변화를 관찰하였다. 그리고, (그림 4)와 (그림 5)는 각각 u*-a4와 i*-d*-a4인 경우의 실패율과 초기접근실패 비율의 관계를 도시한 것이다. 〈표 4〉에 의하면 엔트리 수가 변화하는 모든 경우에 초기접근 실패 빈도수가 일정한 것을 알 수 있으며, 이로부터 초기접근실패 빈도수는 TLB 엔트리 수와 무관함을 알 수 있다. 또, (그림 4)에서 엔트리 수가 증가함에 따라 용량접근실패가 감소하므로 전체 실패 발생 빈도수는 감소함을 알 수 있다.

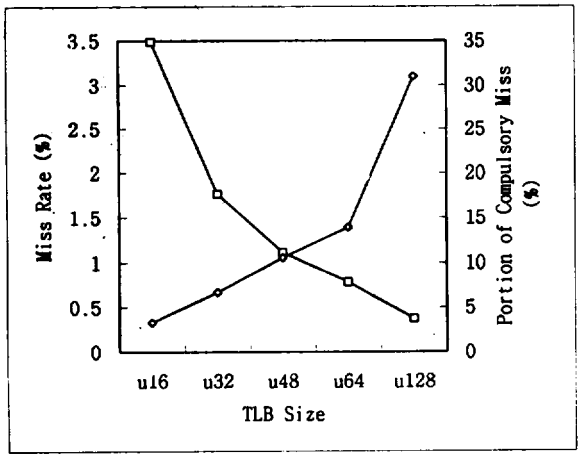
〈표 4〉 엔트리 수 증가에 의한 평균 TLB 성능 척도 변화

〈Table 4〉 Average TLB performance metrics variation versus number of TLB entries

TLB Parameter	Miss Frequency	Capacity Miss & Conflict Miss	Compulsory Miss	Miss Rate (%)	Portion of Compulsory Miss(%)
u16, a4	12,677	12,234	433	3.49	3.4
u32, a4	6,464	6,031	433	1.76	6.7
u48, a4	4,112	3,679	433	1.11	10.5
u64, a4	3,128	2,696	433	0.78	13.8
u128, a4	1,399	966	433	0.38	30.9
i16, d16, a4	10,172	9,739	433	2.79	4.3
i32, d32, a4	5,126	4,693	433	1.39	8.4
i48, d48, a4	3,285	2,852	433	0.89	13.2
i64, d64, a4	2,581	2,148	433	0.69	16.8
i128, d128, a4	1,624	1,191	433	0.44	26.6

주) 평균 기억장치 참조 수: 369,741

그리고, i16-d16-a4의 경우와 u32-a4의 경우를 비교하면 TLB 내의 엔트리 수는 같지만, 데이터와 명령어 부분을 분리한 경우와 데이터와 명령어 간에 전체 엔트리를 공유하는 경우의 관계를 유추할 수 있다. 동일한 크기로 엔트리를 분리한 경우가 공유한 경우보다 실패율이 증가함을 알 수 있다. 이 현상은 응용 프로그램 수행 시에 요구되는 데이터 및 명령어 엔트리의 요구사항 정도가 다르다는 것을 말하며, Clark [17]의 연구 결과인 데이터 엔트리가 명령어 엔트리보다 더 많이 필요로 한다는 연구 결과가 타당성이 있다는 것을 유추할 수 있다.

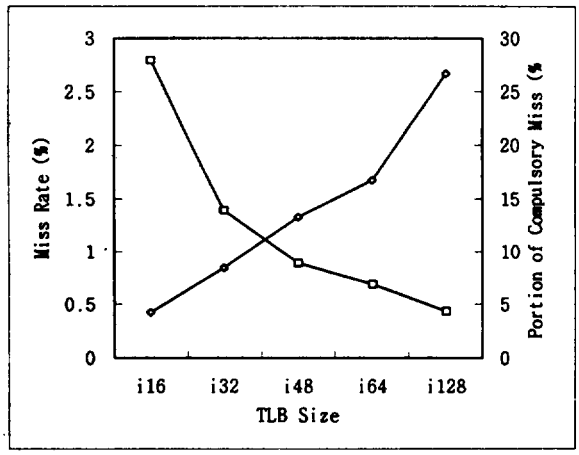


(그림 4) 통합 TLB이고 a4인 경우의 실패율과 초기접근실패 비율과의 관계

(Fig. 4) The relationship between miss rate and the portion of compulsory miss in unified TLB and a4

또, (그림 4)에서 엔트리 수가 64에서 128로 증가함에 따라 초기접근실패는 14%에서 31%로 급격히 증가함을 알 수 있다. 이 사실은 향후의 TLB가 대부분 대용량의 엔트리 수를 가질 것을 고려할 때, TLB 실패에서 초기접근실패의 비중이 높아진다는 것을 암시한다.

엔트리 수에 따른 시뮬레이션 결과를 종합하면 다음과 같다. 초기접근실패 빈도수는 엔트리 수의 변화에 관계없이 일정하다. 따라서, 엔트리 수 증가에 따른 전체 실패 빈도수는 감소하나, 이는 용량접근실패의 감소로 인한 것이기 때문에 상대적으로 초기접근실패의 비중은 증가함을 알 수 있다. 그리고, 앞에서 고찰한 연관성 증가에 따른 실패율의 관계는 일정 수준에 근접하면 연관성 증가에 따른 추가적인 개선 효과는 얻기 어려우나, (그림 4)와 (그림 5)와 같이 엔트리 수를 증가하는 경우에 본 시뮬레이션에서 수행한 128 엔트리 수준 이내에서는 엔트리 수가 증가함에 따라 용량접근실패는 계속적으로 어느 정도는 감소함을 알 수 있었다.



(그림 5) 분리 TLB이고 a4인 경우의 실패율과 초기접근실패 비율과의 관계

(Fig. 5) The relationship between miss rate and the portion of compulsory miss in split TLB and a4

3.4 페이지 크기를 증가하는 방법

운영체제의 시스템 페이지 크기가 증가하면, TLB의 사상 영역이 증가하므로 TLB의 실패율은 개선된다. 그러나, 페이지 크기가 증가하면, 내부 단편화와 외부 단편화가 증가하여 기억장치 공간의 낭비를 초래하게 되므로, 시스템 페이지 크기를 무한정 크게 할 수가 없다. 과거에 기억장치의 가격이 비쌀 경우는 이와 같은 기억장치의 낭비를 줄이기 위하여 페이지 크기를 작게하였으나, 최근에 와서는 기억장치의 가격 하락으로 시스템 페이지 크기를 크게 하는 추세이다. 여기서는 페이지 크기 변화에 따른 TLB 실패율의 변화를 살펴본다.

〈표 5〉는 페이지 크기 증가에 의한 평균 TLB 성능의 측정 결과이다. 성능 측정 대상 파라미터로는 엔트리

수가 제일 적은 u32-a4 경우와 제일 많은 i100-d100-a100 인 경우를 선택하여 성능 척도를 측정하였다. 그리고, (그림 6)과 (그림 7)은 각각 u32-a4와 i100-d100-a100 인 경우의 실패율과 초기접근실패 비율의 관계를 도시한 것이다.

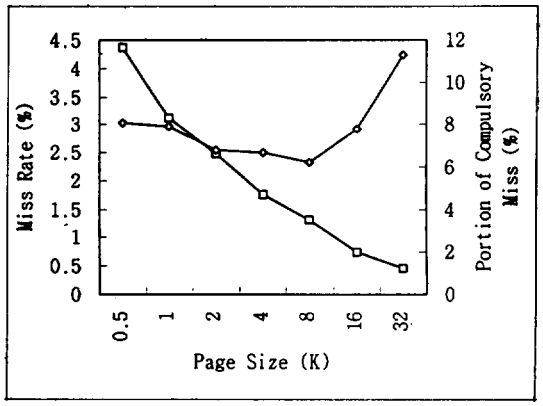
〈표 5〉 페이지 크기 증가에 의한 평균 TLB 성능 척도 변화
〈Table 5〉 Average TLB performance metrics variation versus page size

TLB Parameter	Page Size (K)	Miss Frequency	Capacity Miss & Conflict Miss	Compulsory Miss	Miss Rate (%)	Portion of Compulsory Miss(%)
u32, a4	0.5	16,131	14,825	1,307	4.36	8.1
	1	11,481	10,573	908	3.11	7.9
	2	9,199	6,570	629	2.49	6.8
	4	6,464	6,031	433	1.75	6.7
	8	4,888	4,585	303	1.32	6.2
	16	2,746	2,533	213	0.74	7.8
	32	1,712	1,519	193	0.46	11.3
i100, d100, a100	0.5	5,170	3,863	1,307	1.4	25.3
	1	3,325	2,417	908	0.9	27.3
	2	2,160	1,531	629	0.58	29.1
	4	1,312	879	433	0.35	33
	8	751	448	303	0.2	40.3
	16	431	218	213	0.12	49.4
	32	249	56	193	0.07	77.5

주)평균 기억장치 참조 수:369,741

〈표 5〉에 의하면, 페이지 크기가 0.5 KB 부터 32 KB 까지 페이지 크기가 2의 배수로 증가할 때 실패 빈도수는 상대적으로 감소함을 알 수 있다. 연관성 증가나 엔트리 수를 증가하는 경우와는 다르게, 페이지 크기가 증가함에 따라 충돌 및 용량접근실패와 초기접근실패 모두가 감소함을 알 수 있다. 또, 전체적으로 충돌 및 용량접근실패의 빈도수가 초기접근실패보다 높은 것을 알 수 있다.

i100-d100-a100인 경우에서 페이지 크기가 작을 때는 대부분의 실패는 충돌 및 용량접근실패이나, 페이지 크기가 증가할 수록 초기접근실패의 빈도수가 높아지는 것을 알 수 있다. 페이지 크기가 16 KB일 때 용량접근실패와 초기접근실패의 발생 빈도는 균형을 이루며, 32 KB일 때는 오히려 초기접근실패가 용량접근실패보다 높은 부분을 차지함을 볼 수 있다. 또, (그림 6)에서와 같이 페이지 크기가 증가함에 따라 실패율은 감소를 나타내나, 초기접근실패의 비중이 적은 폭이지만 감소하는 경우도 발생하였다. 이는 페이지 크기가 증가함에 따라 충돌 및 용량 접근 실패의 비율이 초기접근실패 보다 더 많이 증가하였기 때문이다. 그러나, 이 경우도 페이지 크기가 32 K로 증가할 때는 초기접근실패 비율이 증가하므로, 초기접근실패를 제거하는 것은 전체적으로 실패 패널티를 감소시키는데 유효한 방법임을 알 수 있다.

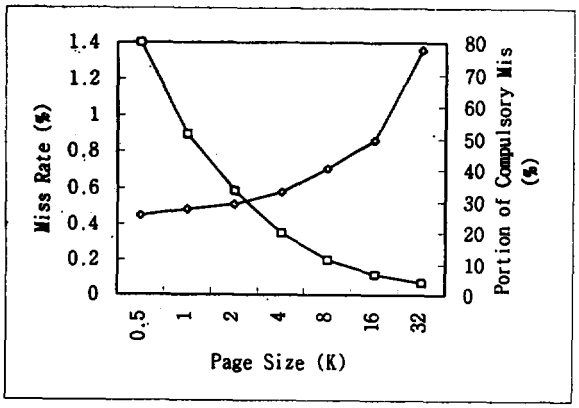


(그림 6) u32-a4인 경우의 실패율과 초기접근실패 비율과의 관계
(Fig. 6) The relationship between miss rate and the portion of compulsory miss in u32 and a4

페이지 크기 증가에 따른 시뮬레이션 결과를 종합하면 다음과 같다. 연관성 증가나 엔트리 수가 증가하는 경우와는 다르게, 페이지 크기가 증가함에 따라 3 C 모두가 감소한다. 그러나, 페이지 크기를 증가하는 경우도 연관성 또는 엔트리 수를 증가하는 것과 마찬가지로, 페이지 크기가 증가함에 따라 전체적으로 실패율은 감소하며, 이에 따라 초기접근실패의 비중은 상대적으로 증가함을 알 수 있었다.

또, (그림 7)의 i100-d100-a100 경우에서 페이지 크기가 32 KB 일 때와 같이 낮은 실패율을 갖는 파라미터 조건에서는, 초기접근실패의 비중이 78%나 차지하여 3 C에서 가장 높은 비중을 차지함을 알 수 있었다. 이 사실은 향후의 TLB가 많은 엔트리 수와 높은 연관성을 가지며, 페이지 크기도 증가될 것으로 전망

되기 때문에, 초기접근실패의 감소가 TLB의 성능 향상에서 중요한 요인이 될 수 있음을 유추할 수 있다.



(그림 7) i100-d100-a100인 경우의 실패율과 초기접근실패 비율과의 관계

(Fig. 7) The relationship between miss rate and the portion of compulsory miss in i100, d100 and a100

4. 결 론

본 연구에서는 ATUM 추적 입력과 수정된 dineroIII TLB 시뮬레이터를 사용하여 추적 지향 시뮬레이션 방법에 의해 TLB 성능 분석을 수행하고, 기존의 성능 향상 방법에 대한 주요 의미를 고찰하였다. 본 연구에서 수행한 시뮬레이션 결과의 종합적 의미와 그 이유는 다음과 같다.

1. TLB 실패율이 높을수록 용량접근실패와 충돌접근실패가 높은 비율을 차지하고, 실패율이 낮을수록 초기접근실패 비율이 높아진다.: 연관성 증가, 엔트리 수 증가 및 페이지 크기 증가에 의한 시뮬레이션의 종합 결과이다.

2. 하드웨어적인 TLB 파라미터 변경에 의해서는 초기접근실패 빈도수를 감소시킬 수 없다.: 하드웨어 적인 TLB 파라미터인 연관성 정도와 엔트리 수의 변경은 연관성 증가는 충돌접근실패에만, 엔트리 수 증가는 용량접근실패에만 영향을 주므로, 최초의 TLB 접근 시에 발생하는 초기접근실패는 방지할 수가 없다. 따라서, TLB가 완전 연관성을 갖고 엔트리 수가 상당하게 많다고 할 지라도 초기접근실패는 방지할 수가 없다.

3. 연관성 증가와 엔트리 수 증가는 TLB의 실패율을 감소시키는 유효한 방법이나, 전체적으로 성능 저하를 초래할 수도 있다.: 연관성의 증가와 엔트리 수 증가는 프로세서 다이에서 더 넓은 부분을 차지하게 되므로, 하드웨어 비용을 증가시킬 뿐만 아니라 TLB의 평균 접근 시간을 증가 시키므로, 전체적으로 성능이 감소될 수도 있다[1, 19].

4. 향후 기술적인 한계에 의하여 완전 연관은 아니나, 엔트리 수가 많은 TLB를 개발하는 것이 유리하다.: 현재 대부분의 프로세서의 TLB는 완전 연관성을 갖는 32개에서 100 개 이상의 엔트리 수를 가지나, 기술적인 한계로 인하여 완전 연관성을 갖지 않는 대용량의 TLB 설계에 관심이 집중되고 있다. 따라서, 고성능 TLB을 만들기 위해서는 TLB의 연관성은 완전 연관보다 낮게 하고, 대용량의 TLB을 구성하는 것이 유리하다[10]. 이 사실은 시뮬레이션 결과에 의해, 연관성이 TLB 크기의 1/2 정도의 연관성을 갖는 경우에 추가적인 성능 개선을 위하여는 연관성을 증가시키는 것보다 엔트리 수를 증가하는 것이 더 좋은 결과를 얻을 수 있음을 유추 할 수 있다.

5. 초기접근실패의 제거는 유망한 방법이다.: 연관성 증가 또는 엔트리 수를 증가하는 것은 TLB 실패율은 감소시킬 수 있으나, 프로세서 내에서 많은 부분을 차지하게 되어 지연 시간을 증가시켜서, 결과적으로 전체적인 성능을 저하시킬 수 있기 때문에, 추가적인 성능 개선을 위해서는 지연시간을 증가시키지 않는 초기접근실패의 제거가 중요하다.

위의 결과를 종합할 때, 향후 초기접근실패에 대한 실패 패널티 감소 방법의 연구가 소프트웨어 제어 TLB의 실패 패널티 감소에서 궁극적으로 가장 중요하게 판단된다. 예전에는 TLB 실패 원인이 대부분

낮은 연관성, 적은 엔트리 수 등으로 인한 높은 충돌 접근실패와 용량접근실패 등에 기인하였으나, 최근의 프로세서의 TLB는 대부분이 완전 연관성을 갖고, 많은 엔트리 수를 가지거나 또는 완전연관은 아니나 많은 엔트리 수를 가질 것으로 전망됨에 따라, 충돌 접근실패와 용량접근실패보다 초기접근실패의 비중이 증가할 것으로 전망된다. 따라서, 실패 패널티가 큰 소프트웨어 관리 TLB가 하드웨어 관리 TLB 수준의 TLB 성능을 얻기 위해서는 하드웨어 관리 TLB에서는 감소하기 어려운 초기접근실패의 감소 기법에 대한 연구가 필요하다고 판단된다.

참 고 문 헌

[1] D.A. Patterson and J.L. Hennessy, Computer Architecture:A Quantitative Approach, Morgan Kaufmann Publishers, Inc., 1990.

[2] D.W. Clark, P.J. Bannon, and J.B. Keller, Measuring VAX 8800 Performance with a Histogram Hardware Monitor, The 15th Annual International Symposium on Computer Architecture, pp. 176-185, 1988.

[3] J. Huck and J. Hays, Architecture Support for Translation Table Management in Large Address Space Machines, The 20th Annual International Symposium on Computer Architecture, 1993.

[4] J.E. Smith, G.E. Dermer, and M.A. Goldsmith, Computer system employing virtual memory, American Patent No. 4,774,659, 1988.

[5] American Micro Devices, Am 29050 Microprocessor User's Manual, American Micro Devices, Inc., 1991.

[6] Hewlett-Packard, PA-RISC 1.1 Architecture and Instruction Set Reference Manual, Hewlett-Packard, Inc., 1992.

[7] G. Kane and J. Heinrich, MIPS RISC Architecture, MIPS Computer Systems, Inc., 1992.

[8] DEC, Alpha Architecture Handbook, Digital Equipment Corporation, 1992.

[9] D. Nagle, R. Uhlig and T. Stanley, Design Tradeoffs for Software-Managed TLBs, The 20th Annual International Symposium on Computer Architecture, 1993.

[10] R. Uhlig, D. Nagle, T. Stanley, T. Mudge, S. Sechrest, and R. Brown, Design Tradeoffs for Software-Managed TLBs, ACM Transactions on Computer Systems, vol. 12, no. 3, pp. 175-205, 1994.

[11] T.E. Anderson, H.M. Levy, B.N. Bershad, and E. D. Lazowska, The interaction of architecture and operating system design, The Fourth International Conference on Architecture Support for Programming and Operating Systems, 1991.

[12] M. DeMoney, J. Moore, and J. Mashey, Operating System Support on a RISC, COMPCON, pp. 138-143, 1986.

[13] M.D. Hill and A.J. Smith, Evaluating Associativity in CPU Caches, IEEE Transactions on Computer, vol. 38, pp. 1612-1630, 1989.

[14] M. Talluri, S. Kong, M.D. Hill, and D.A. Patterson, Tradeoffs in supporting two page sizes, The 19th Annual International Symposium on Computer Architecture, 1992.

[15] W. Stallings, Operating Systems, Macmillan Publishing Company, 1992.

[16] J.S. Park and G.S. Ahn, A Software-Controlled Prefetching Mechanism for Software-Managed TLBs, Microprocessing and Microprogramming Journal, vol. 41, no. 2, pp. 121-136, 1995.

[17] D.W. Clark and J.S. Emer, Performance of the VAX-11/780 Translation Buffer:Simulation and Measurement, ACM Transactions on Computer Systems, vol. 3, no. 1, pp. 31-62, 1985.

[18] J.B. Chen, A. Borg, and N.P. Jouppi, A simulation based study of TLB performance, The 19th Annual International Symposium on Computer Architecture, 1992.

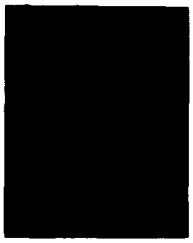
[19] D.A. Patterson and J.L. Hennessy, Computer Organization & Design:The Hardware/Software Interface, Morgan Kaufmann Publishers, Inc., 1994.

[20] Raj Jain, The art of computer systems perfor-

mance analysis, Wiley, 1991.

[21] M.D. Hill, Dinero III Documentation, Unpublished Unix-style Man. Page, Univ. of California Berkeley, 1985.

[22] A. Agarwal, R.L. Sites, and M. Horowitz, ATUM :A new technique or capturing address traces using microcode, The 13th Annual International Symposium on Computer Architecture, pp. 119-129, 1986.



박 장 석

1982년 경북대학교 전자공학과(학사)

1985년 경북대학교 전자공학과(석사)

1995년 경북대학교 컴퓨터공학과(박사)

1983년~1995년 한국전자통신연구원 컴퓨터연구단 선임연구원

1990년 정보처리기술사

1995년~현재 정보통신연구관리단 정보기술평가 1실장(책임연구원)

관심분야:컴퓨터 구조, 멀티미디어 시스템, 소프트웨어 공학