다중 프로세서에서의 캐시접근 실패율을 위한 경험적 모델링

(Empirical Modeling for Cache Miss Rates in Multiprocessors)

이 강 우 † 양 기 주 †† 박 춘 식 †††
(Kangwoo Lee) (Gijoo Yang) (Choonshik Park)

요 약 본 논문에서는, 소규모 시뮬레이션을 통하여 수집된 표본에 통계적인 추정방법을 적용하여 모델을 구하는 경험적 모델링 기법을 제안한다. 이 기법을 이용하여 대칭형 구조를 갖는 다중프로세서 시스템에서의 캐시접근실패율을 위한 두 종류의 모델을 구하였다. 목표시스템의 사양이 고정되었을 때 입력데이타의 크기변화에 따르는 모델과, 입력데이타의 크기가 고정되었을 때 목표시스템의 프로세서 수의 변화에 따르는 모델이다. 모델의 정확성을 제고하기 위하여 한 프로그램에 존재하는 공유데이타들에 대하여 종류별 캐시접근실패에 대한 개별적인 모델들을 구한 후 이들을 종합함으로써 최종적인 모델을 구하였다. 또한 최소 제곱 추정법과 로버스트 추정법을 병용하여 이탈점으로 인한 왜곡을 최소화함으로써 모델의 정확도를 향상시켰다. 경험적 모델링은 표본에 대한 분석이 필요 없으면서도 모델의 정확도가 매우 높다. 또한 소규모의 시뮬레이션만 수행하면 되고, 실험을 통하여 일련의 표본을 수집할 수만 있으면 모든 분야의 연구에 적용할 수 있다. 경험적 모델을 이용한 24가지 경우의 예측시도 중 17번의 경우에는 1% 미만의 예측오차율을 보였으며, 나머지 경우에도 매우 높은 정확도를 보였다. 특히 프로그램의 실행양식이 불규칙하거나, 표본의 수가 충분하기에는 부족한 경우에도 좋은 결과를 보여준다.

키워드 : 경험적 모델링, 예측 모델, 통계적 추정기법, 캐시접근실패, 대칭형 다중프로세서 시스템

Abstract This paper introduces an empirical modeling technique. This technique uses a set of sample results which are collected from a few small scale simulations. Empirical models are developed by applying a couple of statistical estimation techniques to these samples. We built two types of models for cache miss rates in Symmetric Multiprocessor systems. One is for the changes of input data set size while the specification of target system is fixed. The other is for the changes of the number of processors in target system while the input data set size is fixed. To develop accurate models, we built individual model for every kind of cache misses for each shared data structure in a program. The final model is then obtained by integrating them. Besides, combined use of Least Mean Squares and Robust Estimations enhances the quality of models by minimizing the distortion due to outliers. Empirical modeling technique produces extremely accurate models without analysis on sample data. In addition, since only small scale simulations are necessary, once a set of samples can be collected, empirical method can be adopted in any research areas. In 17 cases among 24 trials, empirical models present extremely low prediction errors below 1%. In the remaining cases, the accuracy is excellent, as well. The models sustain high quality even when the behavioral characteristics of programs are irregular and the number of samples are barely enough.

Key words : Empirical Modeling, Prediction Model, Statistical Estimation Methods, Cache Misses, Symmetric Multiprocessors

† 종신회원 : 동국대학교 정보통신공학과 교수
klee@dgu.ac.kr
(Corresponding author임)
†† 정 회 원 : 동국대학교 정보통신공학과 교수
gjyang@dgu.ac.kr
††† 학생회원 : 동국대학교 정보통신공학과
chris@sheenbang.com
논문접수 : 2005년 4월 15일
심사완료 : 2005년 9월 28일

1. 서 론

소프트웨어 시뮬레이션은 시스템을 실제로 구현하지 않고도 목표시스템의 장단점을 쉽고 빠르게 파악할 수 있는 매우 유용한 방법이다. 그러나 일반적으로 속도가 느리고 많은 양의 메모리를 요구하므로 시뮬레이션 할 수 있는 목표시스템이나 작업부하 프로그램의 크기에

한계가 있다. 특히 실제로 사용될 응용프로그램을 작업부하로 사용함으로써 가장 높은 수준의 정확도를 제공하는 실행-구동 시뮬레이션[1,2]에서는 이러한 한계가 더욱 두드러진다. 즉, 컴파일된 작업부하와 작업부하 내에서 정의된 데이타, 그리고 시뮬레이터 자체의 대용량 자료구조를 저장하기 위하여 대단히 많은 메모리 공간이 필요하다. 또한, 프로세서 에뮬레이터가 컴파일된 작업부하의 명령어를 하나씩 읽어 들여 실행하는 과정은 실제 시스템에서 보다 수백 내지 수천 배 또는 그 이상의 시간을 요구한다[1,2].

이러한 한계를 극복하기 위한 방안 중 하나가 분석적 모델링이다. 분석적 모델링이란 임의의 사건의 발생에 영향을 미치는 요소들을 추출하여 이들이 독립적 또는 상호 의존적으로 해당 사건에 영향을 미치는 방식을 수학적 공식으로 표현하는 방법이다. 이를 이용하면 시뮬레이션보다 빠르고 쉽게 목표시스템의 성능을 알 수 있다. 그러나 일반적으로 수학적인 절차가 매우 까다로우며, 특히 영향을 미치는 요소들이 많아지면 모델이 더욱 복잡해지며 모델을 도출하는 과정에서 수많은 가정이 수반된다. 이러한 이유로 인하여 분석적 모델은 대체적으로 정확도가 낮으며, 일정한 수준의 정확도를 갖더라도 적용되는 범위가 제한적이다[3-8].

한편 경험적 모델링은 임의의 사건에 대하여 사전에 수집된 일련의 데이타에 통계적인 추정방법을 적용하여 데이타의 변화 추이를 최적으로 표현하는 수학적인 공식 찾아내는 방법이다. 이 방법은 사전에 수집된 데이타들에는 사건에 영향을 미치는 정적·동적 특성이 반영된 결과라는 사실에 이론적 근거를 둔다. 따라서 수집된 데이타들에 내재되어 있는 특성을 정확히 추정할 수 있다면 다른 모델링 방법보다 정확도가 높은 모델을 구축할 수 있다[9-11]. 경험적 모델링 방법을 이용하면 대용량의 메모리와 오랜 실행시간을 요구하는 시뮬레이션의 단점과, 구축과정이 어려우며 정확도가 떨어지는 분석적 모델의 단점을 모두 극복할 수 있다. 이 밖에도 경험적 모델링은 다음과 같은 장점을 가지고 있다. 첫째, 수집된 표본에 내재되어 있는 목표시스템과 작업부하의 동작특성을 수학적인 공식으로 도출한 결과이므로 정확도가 매우 높다. 둘째, 통계적 추정방법을 적용하므로 수집된 표본에 대한 별도의 분석이 필요 없다. 셋째, 소규모의 시뮬레이션만 수행하면 되므로 일반적인 실험실에서도 쉽게 활용할 수 있다. 넷째, 실험을 통하여 일련의 표본을 수집할 수 있기만 하다면 다른 분야의 연구에도 적용할 수 있다. 특히 최근에는 경험적 모델 구축에 필수적인 우수한 시뮬레이터들이 많이 소개되어 있고 통계적 추정기법을 적용하는 방법도 매우 간단하므로 경험적 모델링 기법을 쉽게 활용할 수 있다.

본 논문에서는, 대칭형 다중프로세서(Symmetric Multiprocessors, SMP) 시스템의 핵심적인 성능 요소인 캐시접근실패율에 대한 두 종류의 경험적 모델을 소개한다. 즉, 쉽게 수행할 수 있는 몇 가지 소규모 시뮬레이션을 통하여 성능 데이타 표본을 수집한 후 통계적인 추정방법을 적용하여 경험적 모델을 구축한다. 첫 번째 모델은 고정된 사양의 목표시스템에 있어서 입력 데이타의 크기가 변화함에 따르는 모델이며, 둘째는 입력 데이타의 크기가 고정되어 있을 때 목표시스템의 프로세서 수의 변화에 따르는 모델이다. 한편 통계적인 추정기법으로는 최소 제곱 추정법(Least-Square Estimation)과 로버스트 추정법(Robust Estimation)[12-14]을 함께 사용하였다.

본 논문에서는 경험적 모델의 정확성을 제고하기 위하여 다음과 같은 세 가지 방법을 채택하였다. 첫째, 일반적으로 하나의 프로그램에는 여러 개의 행렬 구조의 공유데이타가 존재하며, 이들은 상이한 실행특성을 가지므로 이들의 데이타 접근과 캐시접근실패도 상이한 유형을 갖는다. 데이타별로 고유한 실행특성을 반영하기 위하여 각 공유데이타에 대한 개별적인 모델을 구한 후 이들을 종합하여 전체적인 모델을 구축하였다. 둘째, 캐시접근실패는 발생원인에 따라 여러 가지 종류로 구분된다[15]. 본 논문에서는, 각각의 공유데이타들에 대하여, 각 종류의 캐시접근실패에 대한 개별적인 모델을 구한 후 이들을 종합하여 전체적인 모델을 구축하였다. 셋째, 일반적으로 사용되는 최소 제곱 추정법은 다른 표본점들과는 현저하게 격리된 특이점에 예민하여 추정식이 왜곡되는 경우가 많으므로 모델의 정확도가 낮아진다. 이에, 특이점으로 인한 왜곡을 최소화해주는 로버스트 추정법을 병용함으로써 모델의 정확도를 향상시켰다.

본 논문에서 제안하는 방법으로 구해진 경험적 모델을 이용하여 24가지의 상황에 대하여 예측을 시도하여 17번의 경우에는 1% 미만의 예측오차율을 보였으며, 나머지 경우에도 매우 높은 정확도를 보인다. 특히 프로그램의 실행양식이 불규칙하거나, 표본의 수가 충분하기에는 부족한 경우에도 좋은 결과를 보여준다. 이는 SMP의 캐시접근실패율을 위한 경험적 모델의 정확성과 더불어 경험적 모델링을 위하여 본 연구에서 제시한 방법들의 타당성과 유용성에 대한 명확한 증거가 된다.

본 논문은 다음과 같이 구성되었다. 2장에서는 본 연구와 관련된 연구 동향과 사례를 요약하고, 3장에서는 경험적 모델링 기법을 자세히 소개한다. 4장은 통계적인 추정기법인 최소 제곱 추정법과 로버스트 추정법을 다룬다. 5장에서는 실험 환경과 더불어 본 논문이 제안하는 모델링 기법을 적용한 예측 모델 및 예측결과를 제시하며, 6장에서 본 논문에 대한 토론과 함께 논문을 마친다.

2. 기술 동향 및 관련 연구

2.1 대칭형 다중프로세서 시스템의 시장 동향

다중 프로세서 시스템은 오랫동안 고성능 서버시장을 주도하여 왔다. Gartner 그룹의 연구에 따르면, 서버의 세계 시장규모는 2003년 2사분기 이후 연 평균 7.7%씩 성장하여 2004년 1, 2사분기에는 총 233억 달러에 이르렀으며, 2005년에는 2004년에 비하여 6.3% 성장한 500억 달러 수준에 이를 것으로 전망된다[16]. 한편, 1990년대 후반부터 프로세서의 속도가 급속히 빨라지면서 SMP 시스템이 서버시장을 주도하여 왔다. 최근 들어서는 그리드 등 네트워크에 기반한 다양한 구조의 컴퓨팅 환경이 소개되고 있는데, Gartner 그룹의 또 다른 보고에 의하면, 미래에 등장할 새로운 형태의 서버들도 SMP의 핵심 기술을 채용할 것이며, SMP는 초대형 시스템의 단말에 위치하는 서버로서 그 시장을 지속적으로 확장시켜 나갈 것이라고 한다[17].

SMP 시장의 지속적인 성장을 뒷받침하는 두 가지 단서가 있다. 첫째, 현재 서버들이 가장 많이 채택하고 있는 프로세서인 Xeon이나 Itanium을 생산하는 Intel의 기술이 서버의 미래를 결정하는 가장 중요한 요소이다[17]. 그리고 Itanium-2는 로직 추가 없이 4개의 프로세서를 결합하여 SMP를 구성할 수 있으며, SMP를 가능하게 하는 칩셋을 더하면 32개 또는 그 이상의 프로세서들을 이용하여 대형 SMP를 구성할 수 있다[18]. 예를 들어, SGI Altix 3000은 최대 512개의 Itanium으로 구성된 SMP 시스템이다. 둘째, 대표적인 운영체제 중 하나인 Windows 2000 서버가 SMP 시스템을 지원하는 것도 SMP 시장의 지속적인 성장에 대한 또 다른 단서이다. Windows 2000 서버는 4개의 프로세서로 구성된 SMP를, Advanced 서버와 Datacenter 서버는 각각 8개 및 32개의 프로세서로 구성된 SMP를 지원한다[19]. 현재 컴퓨터 산업을 주도하고 있는 양대 기업의 이상과 같은 기술 동향을 근거로 할 때 SMP 시장의 지속적인 성장을 쉽게 예견할 수 있다.

2.2 모델링 관련 연구

다중 프로세서의 대표적인 구조인 SMP 시스템의 성능을 좌우하는 핵심요소인 캐시메모리에 초점을 둔 분석적 모델링을 다룬 다양한 연구가 소개되어 있다[3-5]. 이러한 모델들은 작업부하로 사용된 각 응용 프로그램들의 데이타 공유에 관한 구조적인 특성을 기초로 하였다. 예를 들어, [3]이 제시하는 구조적 모델은 수정된 공유 블록을 접근하는 확률에 기초하였으며, 이로부터 얻어진 파라미터들로부터 캐시접근실패율을 포함한 몇 가지 성능에 대한 모델을 소개하였다. [4]에서는 블록 무효화의 원인이 되는 쓰기접근의 연속성을 토대로 한 모델을 제시한다. 이 모델은 간결하지만 연속된 쓰기접근만을 기초로 하므로 정확도에 한계가 있다. [5]에서는 캐시접근실패율과 전체적인 성능에 관련한 모델을 각각 소개하지만, 모델을 구축하는 방법이나 모델 자체가 매우 복잡하여 정확도가 떨어지며 활용 범위에도 제한이 있다.

컴퓨터 시스템에 관련된 일반적인 분석적 모델에 대한 연구는 다음과 같다. [6]은 프로세서의 수보다 많은 수의 독립적인 작업들을 수행하는 병렬 시스템의 성능을 Markov 체인을 이용한 분석적 모델로 표현하였다. 한편 [7]에서는 분석적 모델링 방법이 광범위하게 사용되고 있음에도 불구하고 모델링 방법론에 대한 근본적인 고찰이 미흡함을 인식하여 분석적 모델링의 대표적인 방법인 평균값 분석 방법과 개방형 대기열 이론 방법의 차이점과 더불어 실질적인 적용 방안에 대한 연구를 보여준다. 그리고 [8]에서는 목표 시스템의 특성과 응용 프로그램의 특성을 각각 수집하여 데이타베이스를 구축하고 이들을 이용하여 단일 프로세서 모델과 통신 모델을 설정함으로써 다중 프로세서에서의 성능을 예측할 수 있는 방안을 제안하였다.

이상과 같이 분석적 모델에 대한 깊은 관심과 다양한 연구에도 불구하고, 구조적인 특성을 기초로 한 분석적 모델들은 다음과 같은 공통적인 문제점을 갖는다. 첫째, 특정한 작업부하의 특징에 기초하므로 모델을 구할 때 사용된 작업부하 이외에는 모델을 적용하기 어렵다. 둘째, 파라미터들의 값이 목표 시스템의 구조적인 사양에 따라 변하므로 새로운 사양을 갖는 목표시스템에 대해서는 모델을 적용할 수 없다. 셋째, 상이한 원인에 의하여 발생하는 사건들이 원인별로 구분되지 않고 동일한 방법으로 모델링되어 정확도가 낮다.

컴퓨터 시스템의 성능을 다룬 경험적 모델링에 관한 연구로는 [9-11] 등이 있다. [9]에서는 실제 시스템에서 실측된 프로그램의 실행시간을 사전 데이타로 사용하며, 그나마 프로그램의 실행 시간이 지수분포를 갖도록 임의로 조절하였다. 따라서 목표 시스템 내부에서 발생하는 다양한 사건들에 대한 면밀한 분석이 불가능하며, 측정된 시간 단위를 수십 millisecond로 하여 측정된 결과에 대한 정확도가 매우 낮다. 또한 모델과 실측된 값과의 오차에 대한 분석이 이루어지지 않았다. [10]에서는 사전에 수집된 데이타들을 이용하여 경험적 모델을 구하되, 모델과 실제 데이타들 사이의 오차가 임계치보다 클 경우 데이타를 추가로 수집하여 다시 모델을 구하는 과정을 반복한다. 그러나 일정한 수준의 정확도를 얻기 위하여 얼마나 많은 데이타를 필요로 하는 지에 대한 분석이 나타나 있지 않으므로 이 방법의 효용성에 대한 판단이 불가능하다. 또한 모델을 구하기 전에 데이타들

의 복잡도 특성을 구해야 하며 이를 위하여 특정한 두 가지 소프트웨어를 필요로 한다는 단점이 있다. 이와 같이 경험적 모델링 방법을 컴퓨터 시스템의 성능 평가에 적용한 최근의 사례가 있으나, 아직은 방법론이 정립되어 있지 않았으며 모델의 정확도에도 한계가 있는 것으로 관찰된다.

3. 경험적 모델링

3.1 프로그램의 동작특성

모든 프로그램은 실행 시의 동작특성을 결정하는 고유한 알고리즘을 갖는다. 즉, 프로세서가 데이타에 접근할 때 프로그램에 내재되어 있는 알고리즘에 따라 일정한 패턴을 유지하며, 데이타접근 패턴은 프로그램의 알고리즘에 의해서만 결정된다[4,5,22]. 이 사실은 프로그램이 SMP에서 실행될 때에도 적용된다. 각 프로세서는 자신에게 할당된 데이타 영역에 있어서 단일 프로세서 시스템에 있어서와 동일한 프로그램 이미지를 실행한다. 따라서 프로세서들은 단일 프로세서 시스템에서와 동일한 데이타 접근 패턴을 유지한다. 이 때 프로세서들이 원격 공유데이타에 접근함으로 인하여 캐시접근실패가 발생할 수도 있는데, 이러한 데이타 접근에 있어서도 알고리즘적인 특성에 따라 일정한 패턴이 유지된다.

3.2 입력 데이타의 크기

일반적으로 프로그램의 성능을 표현할 때 시간적인 복잡도를 사용한다. N이 프로그램에 입력되는 데이타의 크기를 나타낼 때, 시간적인 복잡도는 $O(f(N))$과 같이 표현된다. $f(N)$은 특별히 정의된 것은 아니지만, 이상적인 컴퓨터 환경에서의 명령어 수 또는 데이타 접근 수를 의미한다[22]. 즉 입력 데이타의 크기는 주어진 프로그램의 데이타 접근 수를 결정하는 유일한 요소이다.

이 사실은 SMP에서도 마찬가지로 적용된다. 단, 공유데이타는 일정한 방법에 의하여 균등한 크기로 분할되어 각 프로세서에게 할당된다. 그러나 P개의 프로세서로 구성된 SMP에서 각 프로세서에게 할당된 데이타의 크기는 N/P이므로 $O(N/P)=O(N)$이다. 즉, 한 프로그램이 프로세서의 수가 일정한 SMP에서 실행될 때, 그의 시간적인 복잡도는 단일 프로세서 시스템에서 실행될 때와 동일하다.

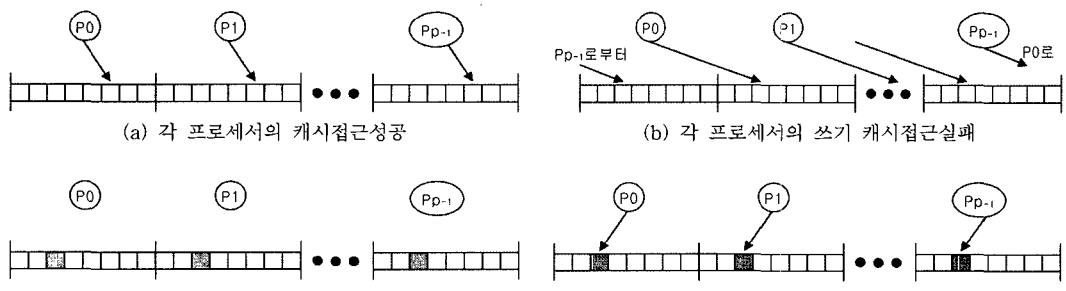
한편, 3.1절에서 본 바와 같이, 프로그램이 SMP에서 실행될 때, 프로세서들은 캐시접근실패를 경험하게 된다. 이 때, 캐시접근실패 수는 데이타 접근 수의 일부이며, 따라서 캐시접근실패 수도 입력 데이타의 크기에 따르는 함수로 표현된다.

3.3 프로세서의 수

프로그램이 SMP에서 실행될 때 프로세서들에게 데이타를 할당하는 방식은 프로그래머에 의하여 임의로 결정되지만, 프로그램을 작성할 때와 실행할 때의 효율성을 모두 고려하여 다음과 같은 방법들이 가장 많이 사용된다[4,5,23]. 다음의 논의에 있어서 공유데이타는 N개의 원소로 이루어진 1차원 행렬 구조이며, SMP는 $P(P\geq 2)$개의 프로세서로 구성되었다고 가정하자.

첫째, 블록 할당방법은 공유데이타를 N/P개의 연속적인 원소들로 구성된 블록들로 나누어 i-번째 블록을 i-번째 프로세서에게 할당한다. 둘째, 회전식 할당방법은 공유데이타의 i-번째 원소를 $i\%P$-번째 프로세서에게 할당한다. 마지막으로, 블록 회전식 할당방법은 공유데이타를 $n(n<N)$개의 연속적인 원소들로 구성된 블록들로 분할하여 i-번째 블록을 $i\%P$-번째 프로세서에게 할당한다.

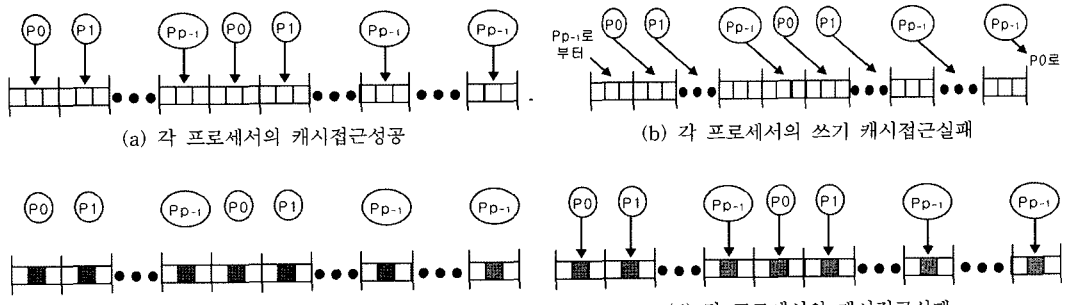
블록 할당방법의 경우 프로세서들의 동작은 그림 1과 같다. 즉, 한 프로세서 P_i가 자신에게 할당된 데이타에 접근할 때 다른 프로세서 $P_j(j\neq i)$들도 자신에게 할당된 데이타에 접근한다(그림 1(a)). P_i가 원격 캐시에 저장되어 있는 데이타에 쓰기동작을 수행하면 P_j도 또 다른 원격 캐시에 저장되어 있는 데이타에 쓰기동작을 수행한다(그림 1(b)). 원격 프로세서의 쓰기 동작으로 인하여 이 데이타들을 포함하는 블록들이 각각 무효화된다(그림 1(c)). 프로세서들이 무효화된 데이타를 접근할 때 캐시접근실패가 발생한다(그림 1(d)). 물론 프로세서들이 록-스텝(lock-step)방식으로 명령어를 수행하는 것



(a) 각 프로세서의 캐시접근성공 (b) 각 프로세서의 쓰기 캐시접근실패

(c) 각 프로세서의 무효화된 캐시블록 (d) 각 프로세서의 캐시접근실패

그림 1 블록 할당방법에서 프로세서들의 데이타 접근 양식



(a) 각 프로세서의 캐시접근성공 (b) 각 프로세서의 쓰기 캐시접근실패

(c) 각 프로세서의 무효화된 캐시블록 (d) 각 프로세서의 캐시접근실패

그림 2 회전식 블록 할당방법에서 프로세서들의 데이타 접근 양식

은 아니지만, 각 프로세서 마다 실행하는 명령어의 순서는 동일하다.[1)]

한편 블록 회전식의 경우 프로세서들의 동작은 그림 2와 같다. 회전식 할당방법은 블록 회전식 방법에서 블록의 크기가 1인 특별한 경우이다.

그림 1과 그림 2에서 보는 바와 같이, 캐시접근실패는 각 프로세서가 자신에게 할당된 영역 밖에 있는 데이타를 접근함으로써 비롯된다. 따라서 입력 데이타의 크기가 일정할 때, 캐시접근실패 수는 공유데이타의 분할 경계의 총합에 비례한다. 한편, 공유데이타의 크기가 일정할 때, 분할 경계의 크기는 블록 할당방법에서는 프로세서의 수 P에 비례한다. 회전식 할당방법과 블록 회전식 할당방법에서는 각각 입력데이타의 크기 및 묶음의 수에 영향을 받을 뿐, 프로세서의 수에는 영향을 받지 않는다. 2차원 행렬구조 데이타의 경우는 조금 다르다. 즉, 분할 경계는, 한 방향으로만 블록 할당방법을 적용한 경우에는 P에 비례하며, 양방향으로 블록 할당방법을 적용한 경우에는 $\sqrt{P}$에 비례한다. 또한, 한 방향이든 양방향이든 회전식 할당방법이나 회전식 블록할당방법의 경우에는 프로세서의 수에 영향을 받지 않는다.

그리고 프로세서가 접근하는 데이타의 메모리상의 위치가 동적으로 결정된다면, 접근되는 데이타의 메모리상의 위치를 정적인 분석을 통해서는 예측할 수 없다. 이 때에는 데이타의 위치가 랜덤하다고 보며, 이 데이타가 원격 프로세서에게 할당되었을 확률은 데이타의 할당방법에 무관하게 $(P-1)/P$이며 캐시접근실패 수는 $(P-1)/P$에 비례한다.

이와 같이 공유데이타의 구조와 할당 방법에 따라 분할 경계는 매우 다양한 방식으로 정의되며, 결과적으로 캐시접근실패에 미치는 영향도 다양하다. 그러나 본 논문에서 사용한 SPLASH[20]와 SPLASH-2[21] 프로그램에서의 공유데이타들은 모두 위의 방식 중 하나를 채택하였다. 물론 더욱 복잡한 데이타 구조와 분할 방식도 가능하지만, 프로그램을 작성하기가 매우 어려우며 실행시에도 데이타 접근에 있어서 공간적·시간적 지역성이 모두 결여되어 심각한 성능 저하의 원인이 된다. 특히, SPLASH와 SPLASH-2는 매우 다양한 방식을 사용하는 프로그램들을 포함하고 있으며, 많은 연구 논문에서 사용되고 있으므로 이 정도의 분석이면 충분하다고 할 수 있다. 그럼에도 불구하고 공유데이타의 구조와 할당 방법이 성능에 미치는 영향이 지대하므로, 현재 본 저자의 연구실에서는 이 분야에 대한 심도 있는 연구를 진행하고 있다.

3.4 모델링 과정

이상의 논의로부터 다음과 같은 결론을 이끌어 낼 수 있다. 첫째, 한 프로그램이 일정한 사양의 SMP에서 실행될 때, 데이타 접근 수와 공유데이타로 인하여 발생하는 캐시접근실패 수는 입력 데이타의 크기(N)에만 의존한다. 둘째, 한 프로그램에서 입력 데이타의 크기가 일정할 때, 데이타 접근 수와 캐시접근실패의 수는 프로세서의 수(P)에만 의존한다. 셋째, 이와 같은 독립변수 x (N 또는 P)와 종속변수 y(데이타 접근 수 또는 캐시접근실패 수)는 프로그램에 내재되어 있는 알고리즘적 특성에 따라 연계되어 진다. 즉, y와 x의 관계는 다음과 같다.

$$y(x) = y(x\,;\,\mathbb{A}) \qquad (1)$$

식 (1)에서 $\mathbb{A} = (a_1,\ldots,a_M)$는 프로그램의 알고리즘적인 특성에 따라 그 값이 결정되는 매개변수들이다. 제4장에서 소개할 최소 제곱 추정법과 로버스트 추정법을 사용

1) 물론 프로그램 내에 변수들의 값에 따라 분기가 결정되는 조건 명령어들이 존재하기도 한다. 그러나 일반적으로 분기가 발생할 확률은 모든 프로세서에서 동일하며, 이는 SMP의 구조적 특징을 최대한 활용하기 위하여 프로그램을 작성할 때에 의도된 것이다. 최악의 경우에는 조건문의 분기가 랜덤 변수에 의하여 결정될 수도 있지만 이 또한 모든 프로세서들에 있어서 동일한 조건이 된다. 실제로 본 연구에서 다룬 6개의 작업부하 프로그램 중 4개가 이러한 경우에 속하는데, 이들 프로그램에서 각 프로세서의 데이타 접근에 있어서도 공통적인 패턴이 관찰되었다. 본 연구에서 다루는 주제의 범위를 넘어서므로 더 이상 자세히 다루지는 않겠지만, 본 저자는 이를 "규칙적으로 불규칙적인(regularly irregular)" 현상이라고 정의한다.

하여 $\mathbb{A} = (a_1,...,a_M)$의 값을 알아내면 식 (1)이 곧 모델이 된다. 그리고 이 모델의 x에 임의의 값을 대입하면 그에 따르는 성능을 쉽게 예측할 수 있다.

이상의 모델링 과정은 다음과 같이 요약된다.

① 고정된 사양의 SMP에 대해, 서로 다른 크기의 작은 입력 데이타들$(N_1, ..., N_n)$을 부과하여 각 경우마다 데이타접근 수와 캐시접근실패 수$(y_1, ..., y_n)$를 측정함으로써 n개의 표본점$((N_1, y_1),...,(N_k, y_n))$을 수집한다.

② 제4장에서 소개하는 최소 제곱 추정법과 로버스트 추정법을 사용하여, $y(N) = y(N;a_1,...,a_M)$을 최적화하는 $\mathbb{A} = (a_1,...,a_M)$을 구한다.

③ 위의 식에 고정된 사양의 SMP 시스템에 대하여, 알고자하는 입력 데이타의 크기(N_{large})를 대입하여 데이타 접근 수와 캐시접근실패 수(y_{large})를 예측한다.

또한

① 고정된 크기의 입력데이타에 대해, 서로 다른 작은 프로세서 수$(P_1, ..., P_n)$를 부과하여 각 경우마다 데이타 접근 수와 캐시접근실패 수$(y_1, ..., y_n)$를 측정함으로써 n개의 표본점$((P_1, y_1),...,(P_k, y_n))$을 수집한다.

② 제4장에서 소개하는 최소 제곱 추정법과 로버스트 추정법을 사용하여, $y(P) = y(P;b_1,...,b_M)$을 최적화하는 $\mathbb{B} = (b_1,...,b_M)$을 구한다.

③ 위의 식에 고정된 크기의 입력데이타에 대하여 알고자하는 프로세서의 수(P_{large})를 대입하여 데이타 접근 수와 캐시접근실패 수(y_{large})를 예측한다.

4. 통계적 추정

4.1 곡선 적합(Curve Fitting)

곡선 적합이란, 실험을 통하여 측정된 데이타들을 일정한 좌표공간에 위치시켰을 때, 이 점들을 포함하는 최적으로 알맞은 곡선의 함수를 찾아내는 통계적인 방법이다. 여기에는 여러 가지 방법이 있는데, 본 논문에서 제안하는 캐시접근실패율에 대한 경험적 모델들은 각각 측정 데이타에 영향을 미치는 독립변수가 단 하나씩이므로 단일 변수를 위한 곡선 적합법을 사용한다[12-14].

일반적으로, 추정식 $y(x) = y(x;a_1,...,a_M)$은 (식 2)와 같이 독립 변수 x와 미지의 계수 $a_j (j = 1,...,M)$를 포함하는 선형 조합이다.

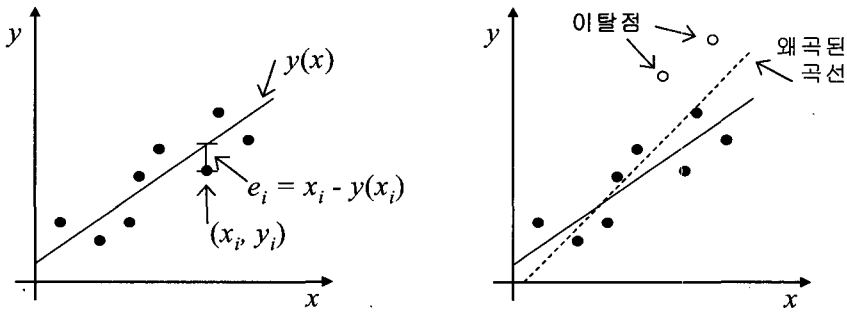
$$y(x) = a_1 + a_2x + ... + a_Mx^{M-1} = \sum_{k=1}^{M} a_kX_k(x) \qquad (2)$$

이 때 $X_k(x)$는 기저 함수로써 선형함수는 물론 삼각함수나 지수함수 등과 같은 비선형 함수일 수도 있다. 식 (2)의 계수들을 찾아내는 방법을 모수 추정이라고 하며, 이는 N개의 표본점에 대하여 추정식 $y(x)$가 표본점들의 위치를 최적으로 표현할 수 있도록 M개의 계수 또는 모수의 값들을 조정하는 과정을 말한다. 이론적으로 추정식 $y(x)$가 나타내는 곡선이 모든 표본점들과 정확하게 일치할 확률은 매우 낮다. 본 논문에서는 이 때 발생하는 오차를 최소화하기 위한 방안으로써 최소 제곱 추정법과 로버스트 추정법을 사용한다.

4.2 최소 제곱 추정

그림 3(a))는 곡선 적합의 예를 보여준다. 이 그래프에서 x-축은 입력 데이타의 크기 또는 프로세서의 수를 나타내며, 각 점들은 x-축의 특정한 값에 대하여 시뮬레이션을 통하여 수집된 데이타 접근의 수 또는 캐시접근실패의 수를 나타낸다. 이 때, 주어진 표본점들과 추정식 사이에는 오차(e_i)가 발생하는데, 최소 제곱 추정법은 이러한 오차들의 제곱의 합을 최소화하는 곡선 $y(x)$를 찾아내는 방법이다.

측정된 N개의 표본점 (x_i,y_i)에 대한 측정오류가 $y(x)$ 주변에 형성되는 독립적이고 동일하게 무작위로 분포된 가우시안 정규분포를 갖는다고 가정하자. 식 (3)에서와 같이, N과 Δy가 상수일 때, 모든 표본점들의 발생확률은 각 표본점의 발생 확률의 곱과 같다.



(a) 최소 제곱 추정법 (b) 이탈점에 의한 왜곡

그림 3 곡선 적합

$$P=\prod_{i=1}^{N}\left\{\exp\left[-\frac{1}{2}\left(\frac{y_i-y(x_i)}{\sigma^2}\right)^2\right]\Delta y\right\} \quad (3)$$

그러므로 N개의 표본점들에 가장 근접한 추정식을 구하는 것은 다음의 식 (4)를 최소화 하는 $a_j(j=1,...,M)$의 값을 찾는 것과 동일하다.

$$\sum_{i=1}^{N}\left[\frac{y_i-y(x;a_1,...,a_m)}{\sigma_i}\right]^2 \quad (4)$$

이에 대한 해결방법은 다음과 같다[13,14].

$$a_j=\sum_{k=1}^{M}[\alpha]_{jk}^{-1}\beta_k \quad (5)$$

이 때, $\alpha_{kj}=\sum_{i=1}^{N}\frac{X_j(x_i)X_k(x_i)}{{\sigma_i}^2}$, 또는 $A_{ij}=\frac{X_j(x_i)}{\sigma_i}$일 때, $[\alpha]=A^T\cdot A$ (6)

그리고 $\beta_k=\sum_{i=1}^{N}\frac{y_iX_k(x_i)}{{\sigma_i}^2}$, 또는 $b_i=\frac{y_i}{\sigma_i}$일 때, $[\beta]=A^T\cdot b$ (7)

σ_i는 i번째 표본점의 오차인데, 그 값을 알 수 없다면 일반적으로 $\sigma_i=1$과 같이 상수로 정해질 수 있다[14].

4.3 로버스트 추정

표본점들 중에서 때로는 일정한 패턴을 따르는 대부분의 표본점과는 달리 그들과 현격하게 격리되어 있는 이탈점(outlier)이 관찰되는 경우가 있다. 이러한 경우, (그림 3(b))에서 보는 바와 같이 이탈점으로 인하여 추정식이 왜곡된다. 로버스트 추정식이란 이러한 이탈점의 영향에 의한 추정식의 왜곡률을 최소화하는 추정기법이다. 따라서 로버스트 추정식은 최소 제곱 추정식 보다는 이탈점에 덜 민감하다. 식 (3)과는 달리, 로버스트 추정식의 유사함수는 식 (8)과 같다[13].

$$P=\prod_{i=1}^{N}\{\exp[-\rho(y_i,y(x_i;a_1,...,a_M))]\Delta y\} \quad (8)$$

식 (8)에서 ρ는 가중함수로써, 지역적인 추정식의 경우, 실제로 측정된 y_i와 각 점이 어떤 가중 인자 σ_i가 적용될 때 예측되는 $y(x_i)$값 사이의 오차에 의존한다. 널리 사용되는 가중함수에는 여러 가지가 있으나 본 논문에서는 $\rho(z)=\ln(1+z^2)$ 그리고 $\psi(z)=2z/(1+z^2)$로 하는 Caushy의 함수를 사용하였다. σ_i를 알 수 없을 경우, 이는 반드시 추정 되어야 하는데 본 논문에서는 [13]에서 소개된 방법 중에서 $\sigma_i=\sqrt{(y_i-y(x_i;a))^2}/N$을 사용하였다. 식 (8)을 최대화 하는 것은 $a=a_1,...,a_M$일 때, a에 대하여 다음의 식 (9)를 최소화하는 것과 같다.

$$\sum_{i=1}^{N}\rho\left(\frac{y_i-y(x_i;a)}{\sigma_i}\right) \quad (9)$$

식 (9)를 최소화하는 것은 식 (10)의 M차 연립방정식의 해법과 같다[13].

$$\psi=\frac{d\rho}{dz} \text{ 일 때, } \sum_{i=1}^{N}\psi\left(\frac{y_i-y(x_i;a)}{\sigma_i}\right)X_j(x_i)=0,\quad j=1,2,...,M \quad (10)$$

가중 최소 제곱법이라고 불리는 이 해법은 반복적이며 최소 제곱법으로 얻어지는 a의 초기 측정치로부터 시작한다. [13]에서, A가 $A_{ij}=\frac{X_j(x_i)}{\sigma_i}$인 $N\times N$행렬이고, $Y=[y_i]$, W는 식 (11)과 같은 w_i의 대각행렬일 때, $a=(A^TWA)^{-1}A^TWY$ 이다.

$$w_i=\frac{(\psi(y_i-y(x_i;a)))/\sigma_i}{(y_i-y(x_i;a))/\sigma_i},\ i=1,2,...,N \quad (11)$$

5. 실험 및 예측 모델

5.1 실험 환경 및 시뮬레이션 방법

본 논문에서 사용한 CacheMire[1]는 컴파일된 병렬 프로그램의 실행 이미지를 목표시스템의 각 프로세서에 할당하여 실행 이미지에 있는 명령어를 하나씩 순차적으로 실행하는 프로그램-구동 시뮬레이터이다. 이를 이용하여 명령어, 데이타 접근 및 접근되는 데이타의 가상주소 및 캐시를 포함한 메모리 시스템에서 발생하는 사건들을 실시간으로 수집하였다.

목표시스템은 무한 크기의 프로세서 캐시, 블록크기 32Bytes, 그리고 Illinois 일관성 프로토콜[2]을 사용하도록 정의하였다. 캐시의 크기를 무한대로 설정한 이유는 주어진 프로그램의 알고리즘적인 특성에 따라 변하는 공유데이타의 성능을 관찰하는 데 있어서 블록교체로 인하여 공유데이타의 동작파악을 저해하는 원인을 제거하기 위함이다. 이는 자주 취해지는 가정[4,15]으로써 본 논문에서 제안하는 경험적 모델링 기법 및 모델의 정확도 등 연구의 타당성에 영향을 주지 않는다.

입력 데이타의 크기에 따르는 캐시접근실패의 수를 모델링하기 위하여 목표시스템의 사양을 고정시킨 경우에는 목표시스템에서의 프로세서의 수를 8개로 하였다. 또한, 입력 데이타의 크기가 일정할 때, 프로세서의 수에 따르는 캐시접근실패를 모델링할 때에는 프로세서의 수를 2~64개까지 변화시킨다.

시뮬레이션을 통하여 작업부하에 있는 공유데이타 별로 데이타 접근 수(REF)와 각 종류의 캐시접근실패 수를 수집하였다. 이 때 캐시접근실패는 [15]에서 제안하는 방식대로 최초접근실패(CM, cold miss), 진성공유데이타접근실패(TSM, true sharing miss), 가성공유데이타접근실패(FSM, false sharing miss)로 구분하였다.

표 1은 SPLASH와 SPLASH-2의 프로그램들 중에서 본 연구에서 사용한 6개의 벤치마크 프로그램들과

표 1 고정된 사양의 목표시스템에서의 입력 데이타의 크기

작업부하 프로그램	프로세서의 수	표본 수집을 위한 입력 데이타의 크기	캐시접근실패 예측을 위한 입력 데이타의 크기	
LU	8	48, 64, 80, 96, 112, 128, 144	288	512
MP3D		2K, 4K, 8K, 16K, 32K, 64K, 128K	256K	512K
OCEAN		34, 66, 130	258	514
FFT		26, 28, 210, 212, 214	216	218
BARNES		512, 768, 1K, 1260, 1536, 1792, 2K	4K	8K
RADIX		2K, 4K, 6K, 8K, 10K, 12K, 14K	512K	1M

표 2 고정된 크기의 입력데이타에 대한 SMP의 프로세서의 수

작업부하 프로그램	고정된 입력 데이타 크기	표본 수집을 위한 프로세서의 수	캐시접근실패 예측을 위한 프로세서의 수
LU	256	2, 4, 8, 16	32, 64
MP3D	32K		
OCEAN	130		
FFT	216		
BARNES	1024		
RADIX	128K		

고정된 사양의 SMP에서 경험적 모델을 구축하는데 필요한 표본점을 수집하기 위한 입력 데이타의 크기들, 캐시접근실패율을 예측하고자 하는 입력 데이타의 크기를 보여준다. 프로그램들 중에서 OCEAN은 3개, FFT는 5개 및 나머지 프로그램들에서는 7개의 표본점을 수집하였다. 또한 표 2는 고정된 크기의 입력 데이타에 대하여 표본을 수집하기 위한 프로세서 수들과 경험적 모델을 이용하여 캐시접근실패율을 예측하고자하는 프로세서의 수를 보여준다.

다음 절부터는 각 벤치마크들에 대한 간단한 설명과 더불어 실험결과, 경험적 모델 및 모델을 이용한 예측결과를 제시한다. 입력데이타의 크기나 프로세서의 수에 따르는 데이타접근 및 캐시접근실패의 수에 대하여 자세한 설명이 가능하지만, 공간적인 제약에 따라서, 최대한 간략하게 설명하기로 한다. 그림 4부터 그림 9까지는 입력 데이타의 크기(x-축)에 따르는 데이타접근 수와 각 종류의 캐시접근실패 수(y-축)를 보여준다. 표 3부터 표 14에서는 곡선 적합에 의한 예측모델과 함께, 고정된 시스템에서 큰 크기의 데이타를 입력했을 때에 대한 예측 결과와, 고정된 입력 데이타에 대해서 많은 수의 프로세서에 대한 예측 결과를 보여준다. 예측 오차율은 다음과 같이 계산되었다.

$$\text{예측 오차율 (\%)} = \frac{\text{시뮬레이션 결과} - \text{예측된 결과}}{\text{시뮬레이션 결과}} \times 100 \quad (12)$$

5.2 LU

LU는 밀집정방행렬의 LU-분해를 수행하는 프로그램이다. N은 행렬의 행 또는 열의 수이며 프로그램내의

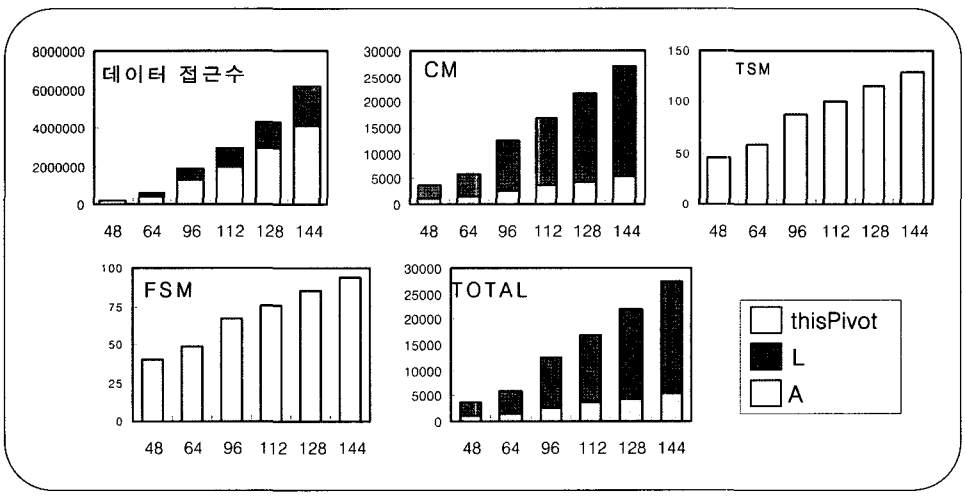


그림 4 LU의 시뮬레이션 결과

*for-loop*가 반복되는 횟수이기도 하다. 공유데이타는 두 개의 $N \times N$행렬인 A, L과 하나의 N개의 원소를 가진 1차원 행렬 *thisPivot*이 있다. 그러므로 $O(N^2)$의 최초접근실패가 예측된다. *for-loop*에서 $i=0,1,...,N-1$으로 변함에 따라, i번째 반복시기에서 A는 i번째 열부터 우측의 $N-i$번째 열까지 한번씩 접근되며, L의 i번째 열은 $N-i$번 접근되어진다. 그러므로 이들에 대한 데이타 접근 수는 $O(N^3)$이다. 두 개의 큰 배열 A와 L에 대한 접근은 최초접근실패만을 야기하고 *thisPivot*에 대한 접근에서 $O(N)$개의 공유데이타접근실패가 발생한다.

표 3과 표 4에서 보듯이, 경험적 모델을 이용한 예측 오차는 극히 작다. 이것은 LU가 가지고 있는 데이타 접

표 3 고정된 사양의 SMP에 대한 입력데이타의 크기에 따르는 경험적 모델과 예측 결과

L U			입력 데이타 크기 = 288			입력 데이타 크기 = 512		
배 열	항 목	예 측 모 델	실 험 결 과	예 측 결 과	오차 (%)	실 험 결 과	예 측 결 과	오차 (%)
A	REF	$1.332N^3+6.358N^2+27.840N+891.126$	32,349,490	32,341,280	0.025	180,532,208	180,444,752	0.048
	CM	$0.250N^2-0.036N+4.100$	20,739	20,745	-0.029	65,538	65,570	-0.050
L	REF	$0.667N^3-0.032N^2+2.275N-85.238$	15,925,054	15,925,757	-0.004	89,478,144	89,485,728	-0.008
	CM	$1.000N^2+6.044N-42.954$	84,632	84,626	0.007	265,183	265,146	0.014
this Pivot	REF	$1.000N^2+2.000N+2.968$	83,517	83,517	0.000	263,165	263,165	0.000
	CM	$0.125N+1.000$	37	37	0.000	65	65	0.000
	TSM	$0.875N+3.000$	1,793	1,793	0.000	3,193	3,196	-0.093
	FSM	$0.558N-12.932$	148	148	0.000	276	274	0.072
Total	REF	$1.999N^3+7.327N^2-23.566N+803.06$	48,358,061	48,350,554	0.016	270,273,517	270,193,645	0.030
	CM	$0.375N^2+0.861N+2.269$	31,359	31,362	-0.010	98,754	98,771	-0.017
	TSM	$0.875N^2+11.543N-51.624$	75,842	75,839	0.004	235,225	235,208	0.007
	FSM	$0.562N-12.932$	148	148	0.000	276	274	0.725
	Total Miss	$1.250N^2+12.967N-61.928$	107,349	107,349	0.000	334,255	334,253	0.001
	Miss rate(%)	$\frac{1.125N^2+12.967N-61.928}{1.999N^3+7.327N^2-23.566N+803.06}\times 100$	0.2220	0.2220	0.000	0.1237	0.1237	0.000

표 4 고정된 크기의 입력데이타에 대한 프로세서의 수에 따르는 경험적 모델과 예측 결과

L U			프로세서의 수 = 32			프로세서의 수 = 64		
배 열	항 목	예 측 모 델	실 험 결 과	예 측 결 과	오차 (%)	실 험 결 과	예 측 결 과	오차 (%)
A	REF	22,764,012	22,764,012	22,764,012	0.000	22,764,012	2,276,4012	0.000
	CM	16,384	16,384	16,384	0.000	16,384	16,384	0.000
L	REF	11,184,638	11,184,638	11,184,638	0.000	11,184,638	11,184,638	0.000
	CM	8,256	8,256	8256	0.000	8,256	8,256	0.000
	TSM	$8,057.018P-6,689.17$	254,511	251,135	1.326	508,959	508,960	-0.000
this Pivot	REF	66,045	66,045	66,045	0.000	66,045	66,045	0.000
	CM	32	32	32	0.000	32	32	0.000
	TSM	$221.650P-74.557$	7,325	7,018	3.476	14,111	14,111	0.000
	FSM	$10.032P-3.019$	318	318	0.000	318	318	0.000
Total	REF	34,014,695	34,014,695	34,014,695	0.000	34,014,695	34,014,695	0.000
	CM	24,672	24,672	24,672	0.000	24,672	24,672	0.000
	TSM	$8,278.668P-6,763.727$	261,836	258,153	1.406	523,070	523,071	0.000
	FSM	$10.032P-3.019$	318	318	0.000	639	639	0.000
	Total Miss	$8,288.700P-6,766.746$	286,828	283,143	1.284	548,381	548,382	0.000
	Miss rate(%)	$\frac{8,288.700P-6,766.746}{34,014,695}\times 100$	0.8432	0.8324	1.280	1.6121	1.6121	0.000

근패턴과 프로세서간의 통신이 매우 규칙적인 알고리즘적인 특성으로 인하여 예측이 쉽기 때문이다.

5.3 MP3D

MP3D는 우주공간에서의 입자들의 운동에 대하여 1차원 배열인 *Particles*안에 있는 분자들의 위치와 속도를 반복적으로 시뮬레이션하는 프로그램이다. 입력 데이타 크기 N은 *Particles*안에 있는 분자의 수를 나타낸다. 또 다른 1차원 배열 *Ares*는 시간이 경과함에 따라 추가되는 분자들을 나타내며 이 배열의 크기는 N에 비례한다. 3차원 공간을 이산적으로 나타내기 위한 배열 *Cells*의 크기는 N에 무관하다.

이 프로그램의 시뮬레이션 반복회수는 5회($O(1)$)이며, 각 반복시기마다 모든 *Particles* 배열 안의 분자들이 접근되고 *Cells* 배열 공간 안에서 이동 한다($O(N)$ 접근). 분자가 움직일 때 마다 3차원 좌표상의 입자들의 위치 변화를 나타내기 위하여 *Cells* 요소들이 갱신되므로, 매 반복시기마다 *Cells*의 각 요소들은 $O(N)$번 접근된다. *Ares* 배열의 각 분자들 또한 *Cells* 배열 공간 안에서 이동하여 단 한번의 충돌검사를 수행 한다($O(N)$회 접근). 그러므로 MP3D에서 전체적인 데이타 참조회수는 $O(N)$이고 진성공유데이타접근실패 수는 또한 $O(N)$이다.

MP3D에서는 프로세서가 접근하는 공유데이타의 메모리에서의 위치는 대부분 예측이 불가능하다. 하나의 프로세서가 하나의 분자를 *Cells* 공간 안에서 이동시킬

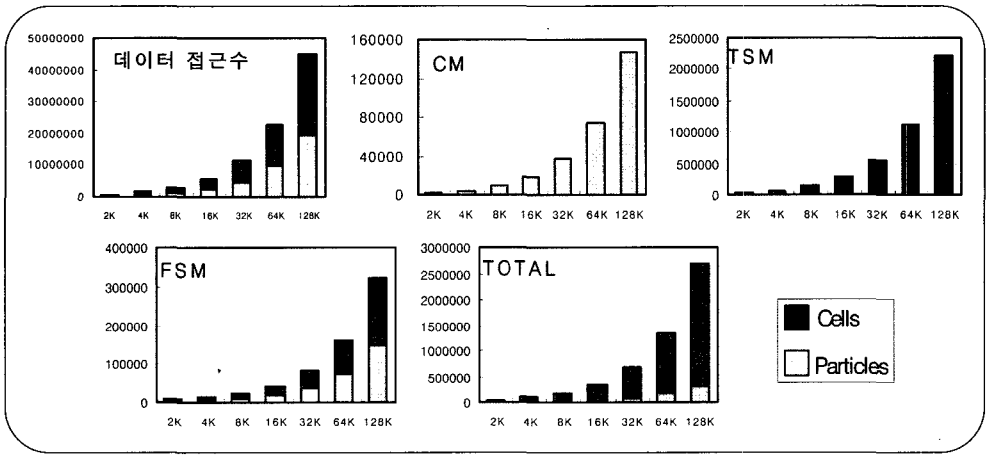


그림 5 MP3D의 시뮬레이션 결과

표 5 고정된 사양의 SMP에 대한 입력데이타의 크기에 따르는 경험적 모델과 예측 결과

MP3D			입력 데이타 크기 = 256K			입력 데이타 크기 = 512K		
배 열	항 목	예 측 모 델	실 험 결 과	예 측 결 과	오차 (%)	실 험 결 과	예 측 결 과	오차 (%)
Parti-cles	REF	$146.780N+686.656$	38,475,880	38,476,804	-0.002	76,961,752	76,954,296	0.009
	CM	$1.127N+1.314$	295,420	295,410	0.003	590,648	590,818	-0.029
	TSM	$0.170N-1.703$	44,370	44,365	0.011	89,578	88,732	0.944
	FSM	$1.101N+1608.389$	292,524	290,424	1.064	543,500	579,251	-6.578
Cell	REF	$195.085N+42710.3$	51,182,884	51,183,032	-0.000	102,322,920	102,323,352	0.000
	TSM	$16.821N-1034.9$	4,411,209	4,408,433	0.062	8,840,015	8,817,901	0.250
	FSM	$1.319N+2071.14$	345,947	347,954	-0.580	686,345	693,838	-1.091
Total	REF	$341.865N+43396.956$	89,658,764	89,659,836	-0.001	179,284,672	179,277,648	0.004
	CM	$1.127N+1.314$	295,420	295,410	0.003	590,648	590,818	-0.028
	TSM	$16.991N-1036.603$	4,455,579	4,452,798	0.062	8,929,593	8,906,633	0.257
	FSM	$2.420N+3679.529$	638,471	638,378	0.015	1,229,845	1,273,089	-3.516
	Total Miss	$20.538N+2644.240$	5,389,470	5,386,586	0.054	10,750,086	10,770,540	-0.190
	Miss rate(%)	$\frac{20.538N+2644.240}{341.865N+43396.956}\times 100$	6.9761	6.0078	0.055	5.9961	6.0077	-0.193

표 6 고정된 크기의 입력데이타에 대한 프로세서의 수에 따르는 경험적 모델과 예측 결과

MP3D			프로세서의 수 = 32			프로세서의 수 = 64		
배 열	항 목	예 측 모 델	실 험 결 과	예 측 결 과	오차 (%)	실 험 결 과	예 측 결 과	오차 (%)
Parti-cles	REF	4,808,082	4,808,082	4,808,082	0.009	4,808,082	4,808,082	0.000
	CM	$-63.942(P-1)/P+36,971.992$	36,907	36,910	-0.008	36,909	36,909	0.000
	TSM	$7,030.883(P-1)/P-544.440$	5,972	6,266	-4.935	6,394	6,376	0.272
	FSM	$26,709.281(P-1)/P+17,392.337$	43,267	43,266	0.000	43,977	43,684	0.665
Cell	REF	6,434,519	6,434,519	6,434,519	0.000	6,434,519	6,434,519	0.000
	TSM	$790,231.812(P-1)/P-114,912.906$	654,388	650,624	0.575	695,470	662,971	4.672
	FSM	$18,123.068(P-1)/P+34,385.582$	43,762	51,942	-18.692	686,345	693,838	-1.092
Total	REF	11,242,601	11,242,601	11,242,601	0.000	11,242,601	11,242,601	0.000
	CM	$-63.942(P-1)/P+36,971.992$	36,907	36,910	-0.008	36,909	36,909	0.000
	TSM	$797,262.688(P-1)/P-115,457.344$	660,360	656,890	0.525	701,864	669,347	4.633
	FSM	$26,711.701(P-1)/P+51,777.922$	87,029	95,208	-9.398	730,322	737,522	-0.986
	Total Miss	$823,910.438(P-1)/P+797,203.062$	784,296	789,008	0.054	1,469,095	1,443,778	1.723
	Miss rate(%)	$\frac{823,910.438(P-1)/P+797,203.062}{11,242,601}\times 100$	6.9761	7.0180	-0.601	13.0672	12.8420	1.723

때, 임의로 생성된 값에 따라 프로세서가 충돌을 시험해야 하는지 여부가 결정되기 때문이다. 이러한 충돌들은 *Particles* 배열에서 공유데이타접근실패를 발생시키는 유일한 원인이다. 더욱이 *Cells* 배열의 각 원소의 메모리 상의 위치는 분자의 공간좌표에 의해 선택되어지고 프로세서나 분자의 메모리 상의 위치와는 무관하다. 이러한 불규칙적인 알고리즘적 특성 때문에 심각한 예측 오류가 발생할 수 있다.

그러나 위의 표 5와 표 6에서 볼 수 있듯이 극히 정확한 경험적 모델을 구할 수 있었던 이유는 다음과 같다. 첫째, 본 논문의 제 3.3절에서 논의한 바와 같이, 프로세서들의 데이타 접근 패턴이 매우 규칙적으로 불규칙적이기 때문이다. 둘째는 로버스트 추정식이 표본 중의 이탈점들에 의한 왜곡을 잘 극복하였기 때문이다.

5.4 OCEAN

OCEAN은 바다의 물분자 움직임을 일정한 횟수만큼 반복적으로 시뮬레이션한다. 각 반복시기마다 크기가 고정된 여러 개의 2차원 격자 모양으로 이산적으로 모델

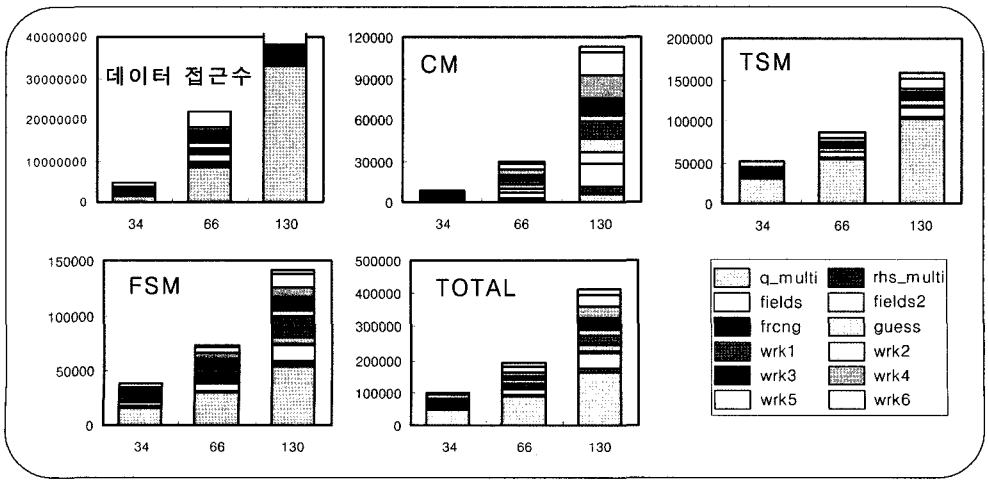


그림 6 OCEAN의 시뮬레이션 결과

표 7 고정된 사양의 SMP에 대한 입력데이타의 크기에 따르는 경험적 모델과 예측 결과

OCEAN			입력 데이타 크기 = 258			입력 데이타 크기 = 514		
배 열	항 목	예 측 모 델	실 험 결 과	예 측 결 과	오차 (%)	실 험 결 과	예 측 결 과	오차 (%)
q_ multi	REF	$1601.66N^2+66699N-2884388$	122,349,648	120,936,976	1.155	454,333,088	454,551,456	-0.048
	CM	$0.332N^2+0.547N+16.560$	22,259	22,258	0.004	88,019	88,018	0.001
	TSM	$743.669N+5466.656$	202,766	197,331	2.680	375,802	387,709	-3.168
	FSM	$394.174N+2809.42$	107,470	104,506	2.758	200,657	205,415	-2.037
rhs_ multi _	REF	$259.86N^2+10586.8N-463576$	19,787,814	19,565,040	1.126	73,597,080	73,631,600	-0.047
	CM	$0.332N^2+0.109N+0.436$	22,130	22,129	0.005	87,778	87,777	0.001
	TSM	$23.236N+400.526$	6,400	6,399	0.016	12,234	12,344	-0.899
	FSM	$42.816N+351.367$	11,676	11,398	2.381	22,847	22,359	2.136
fields	REF	$444.00N^2-960.24N+992.28$	29,307,692	29,307,710	0.000	116,810,424	116,810,624	0.000
	CM	$1.000N^2+0.001N+0.959$	66,565	66,564	0.002	264,197	264,195	0.001
	TSM	$88.451N+53.098$	22,812	22,873	-0.267	45,658	45,516	0.311
	FSM	$103.547N+19.069$	26,712	26,735	-0.086	52,964	53,245	-0.531
fields2	REF	$12.001N^2-2.427N+17.222$	798,199	798,226	-0.003	3,169,046	3,169,388	-0.011
	CM	$0.500N^2+0.000N+0.000$	33,282	33,282	0.000	132,098	132,098	0.000
	FSM	$11.924N+3.152$	3,095	3,079	0.517	6,123	6,132	-0.147
frcng	REF	$26.000N^2-0.019N+2.780$	1,730,666	1,730,669	0.000	6,869,098	6,869,118	0.000
	CM	$0.250N^2+0.000N-0.019$	16,641	16,640	0.006	66,049	66,048	0.002
	FSM	$3.984N-4.468$	1,019	1,022	-0.294	2,052	2,042	0.487
guess	REF	$48.000N^2-96.061N+98.078$	3,170,440	3,170,408	0.001	12,632,160	12,632,220	0.000
	CM	$0.5000N^2+3.000N-5.989$	34,050	34,050	0.000	133,634	133,633	0.001
	FSM	$33.000N$	8,514	8,514	0.000	16,962	16,962	0.000
wrk1	REF	$210.00N^2+64.069N+36.962$	13,994,992	13,994,972	0.000	55,514,096	55,513,992	0.000
	CM	$0.750N^2+0.003N-0.022$	49,923	49,922	0.002	198,147	198,145	0.001
	TSM	$3.500N+71.000$	974	974	0.000	1,868	1,868	0.000
	FSM	$155.090N-85.05$	39,861	39,927	-0.166	79,123	79,631	-0.642
wrk2	REF	$348.00N^2-150.1N+1099.1$	22,868,724	22,868,836	0.000	91,350,256	91,350,920	-0.001
	CM	$0.250N^2+0.250N+0.471$	16,706	16,705	0.006	66,178	66,177	0.002
	TSM	$66195.5N+72.75$	5,781	5,780	0.017	11,445	11,444	0.009
	FSM	$35.824N-57.136$	9,180	9,185	-0.054	18,400	18,356	0.239
wrk3	REF	$684.00N^2-2304.0N+2310.6$	44,937,648	44,937,652	0.000	179,528,112	179,528,112	0.000
	CM	$0.750N^2+0.003N-0.022$	49,923	49,922	0.002	1,989,147	1,989,145	0.000
	TSM	$63.000N+198.000$	16,452	16,452	0.000	32,580	32,580	0.000
	FSM	$96.969N+77.062$	25,226	25,094	0.523	50,896	49,918	1.922
wrk4	REF	$216.00N^2-479.95N+383.62$	14,254,368	14,254,325	0.000	56,820,000	56,819,804	0.000
	CM	$1.000N^2+0.000N+0.000$	66,564	66,564	0.000	264,196	264,196	0.000
	TSM	$42.000N+65.992$	10,902	10,901	0.009	21,654	21,653	0.005
	FSM	$690146N+24000$	17,791	17,862	-0.399	43,368	43,367	0.002
wrk5	REF	$839.99N^2-2783.8N+2655.8$	55,198,176	55,197,784	0.001	220,496,352	220,494,848	0.001
	CM	$1.000N^2+0.000N+0.000$	66,564	66,564	0.000	264,196	264,196	0.000
	TSM	$83.999N+191.999$	21,864	21,863	0.005	43,368	43,367	0.002
	FSM	$90.27N-60.587$	24,051	24,189	-0.574	47,269	47,634	-0.772
wrk6	REF	$24.000N^2+02016N+427.26$	1,597,964	1,597,962	0.000	6,341,132	6,341,118	0.000
	CM	$0.250N^2+0.000N+1.014$	16,642	16,642	0.000	66,050	66,050	0.000
	TSM	$0.000N+7090$	7,090	7,090	0.000	7,090	7,090	0.000
	FSM	$23.645N-10.790$	6,172	6,089	1.345	12,315	12,142	1.405
Total	REF	$4713.52N^2+69573.5N-3339941$	329,996,291	328,360,560	0.496	1,277,460,884	1,277,713,200	-0.020
	CM	$6.914N^2+3.913N+13.389$	461,249	461,241	0.002	1,828,689	1,828,678	0.001
	TSM	$1069.981N+6616.99$	282,645	282,664	-0.007	544,708	556,578	-2.179
	FSM	$1062.879N+3086.639$	280,767	277,299	1.235	545,655	549,399	-0.686
	Total Miss	$6.914N^2+2136.77N+9719.01$	1,030,064	1,021,204	0.860	2,919,052	2,934,655	-0.535
	Miss rate(%)	$\frac{6.914N^2+2136.77N+9719.01}{4713.52N^2+69573.5N-3339941}\times 100$	0.3121	0.3110	0.052	0.2285	0.2297	-0.525

표 8 고정된 크기의 입력데이타에 대한 프로세서의 수에 따르는 경험적 모델과 예측 결과
(공간적인 제약에 따라 *q_multi*와 *rhs_multi*를 제외한 나머지 행렬을 통합하였음)

OCEAN			프로세서의 수 = 32			프로세서의 수 = 64		
배 열	항 목	예 측 모 델	실 험 결 과	예 측 결 과	오차율 (%)	실 험 결 과	예 측 결 과	오차율 (%)
q_ multi	REF	30609328	30,609,328	30,609,328	0.000	30,609,328	30,609,328	0.000
	CM	$9.000\sqrt{P}+9$	5,717	5,717	0.000	5,753	5,753	0.000
	TSM	$68471.312\sqrt{P}-72730.601$	196,336	201,181	-2.468	475,067	475,066	0.000
	FSM	$28492.214\sqrt{P}-12682.999$	98,030	101,285	-3.320	216,340	215,254	0.502
rhs_ multi	REF	4952992	4,952,922	4,952,922	0.000	4,952,922	4,952,922	0.000
	CM	5626	5,626	5,626	0.000	5,626	5,626	0.000
	TSM	$2234.143\sqrt{P}-2652.001$	6,084	6,284	-3.287	15,288	15,221	0.438
	FSM	$2865.714\sqrt{P}+87.996$	11,367	11,550	-1.610	23,075	23,013	0.269
All other arrays	REF	47204070	47,204,070	47,204,070	0.000	47,204,070	47,204,070	0.000
	CM	$255.638\sqrt{P}+105532.976$	106,556	106,555	0.001	107,578	107,578	0.000
	TSM	$28838.513\sqrt{P}-27126.130$	84,883	86,227	-1.583	201,582	201,581	0.000
	FSM	$30071.044\sqrt{P}+21702.458$	142,532	141,986	0.383	262,089	262,270	-0.069
Total	REF	82766390	82,766,390	82,766,390	0.000	82,766,390	82,766,390	0.000
	CM	$264.638\sqrt{P}+116839.977$	117,899	117,898	0.001	117,989	118,957	-0.820
	TSM	$99543.969\sqrt{P}-99204.734$	287,303	293,692	-2.224	691,937	691,868	0.010
	FSM	$61428.977\sqrt{P}+9107.454$	251,929	254,821	-1.148	501,504	500,537	0.193
	Total Miss	$161237.594\sqrt{P}+26742.695$	657,131	666,411	-1.412	1,311,339	1,311,362	-0.002
	Miss rate(%)	$\frac{161237.594\sqrt{P}+26742.695}{82766390}\times 100$	0.7940	0.8052	-1.411	1.5844	1.5844	0.000

링된 공간 배열을 이용하여 다양한 미분 방정식이 계산되며, 각 격자는 해양분지를 수평으로 나눈 구역을 나타낸다. 이 벤치마크에 사용된 알고리즘은 다중 격자 Successive Over Relaxation 방법이며 OCEAN에서 사용된 주요 데이타 구조는 물분자의 운동을 모델링하기 위한 방정식과 연관된 이산 데이타를 저장하는 25개의 2차원 배열이다. 그 중에서 *q_multi*와 *rhs_multi*는 여러 층으로 이루어진 격자를 나타내는 배열이며 각각의 크기는 $O(N^2)$이다. 각 반복시기마다 $O(N^2)$의 두 격자배열이 접근된다. 프로세서간의 데이타 공유는 오직 각 프로세서에게 할당된 데이타의 구획 경계에서만 나타나므로 진성공유데이타접근실패의 수는 $O(N)$이다.

OCEAN은 두 가지 측면에서 분석적으로 모델링하기 가장 까다로운 프로그램이다. 우선, 프로그램에서 사용하는 데이타가 많아 매우 많은 양의 메모리를 요구한다. 이로 인하여 본 연구에서는 단지 세 개의 작은 입력 데이타에 대한 표본점만을 수집할 수 있었다. 이것은 2차 다항식에 있는 세 개의 미지의 계수를 추정하기위한 최소한의 요구량이다. 뿐만 아니라, OCEAN의 수행시간은 입력 데이타의 값에 크게 의존하는데 이는 계산되어 갱신된 수치들의 변화량의 총합이 일정한 임계치보다 작을 때 까지 반복적으로 수행되기 때문이다. 더욱이 임계치 이하에 다다를 때까지 일련의 계산이 반복되는 횟수도 입력 데이타의 값에 따라 일정하지 않으므로 표본점들의 측정오차가 상당히 클 것이다.

이러한 이유로 인하여 경험적 모델의 정확도가 매우 낮을 것으로 예상된다. 만일 우리가 예측하고자 하는 큰 입력 데이타에 대하여 OCEAN 프로그램이 불규칙적인 반응을 보인다면 예측 결과는 분명히 나쁘게 된다. 그럼에도 불구하고, 캐시접근실패율에 대하여 극히 우수한 예측 오차율을 보이는 모델을 도출할 수 있었다. 결론적으로 말하자면, 앞의 MP3D의 경우에서 본 바와 같이, OCEAN을 실행하는데 있어서 프로세서들의 데이타 접근 패턴이 매우 규칙적으로 불규칙적이었으며 로버스트 추정식을 이용하여 표본에 존재할 수 도 있는 이탈점들에 의한 왜곡 효과를 최소화할 수 있었기 때문이다.

5.5 FFT

FFT는 radix-$\sqrt{N}$ 여섯 단계 FFT 알고리즘의 1차원 버전이다. 중요한 입력 데이타는 두 개의 $\sqrt{N}\times\sqrt{N}$ 배열로써, *X*는 데이타 포인트를, *trans*는 roots-of-

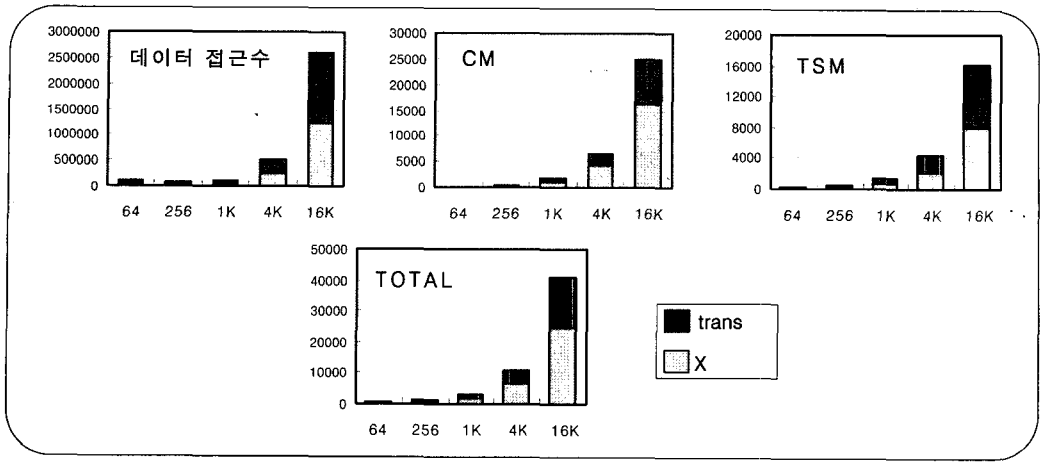


그림 7 FFT의 시뮬레이션 결과

unity를 담고 있다. 작은 크기의 1차원 배열인 *umain*과 *umain2*는 공유되지 않으므로 캐시접근실패의 수는 무시한다. $\sqrt{N}\times\sqrt{N}$의 배열을 반영하기 위하여 입력 데이타의 크기를 $n=\sqrt{N}$으로 한다. FFT는 반복적이지 않으며 X행렬과 *trans*행렬의 각 원소들은 $O(1)$번 접근된다. 그러므로 전체 데이타 접근 수와 최초접근실패 수는 $O(n^2)$이고, 진성공유데이타접근실패 수 역시 $O(n^2)$이다.

표 9와 표 10에서 보듯이, 예측된 데이타접근실패율과 실측된 데이타접근실패율에 대한 오차는 1% 미만이다. 이는 LU에서와 같이 FFT의 알고리즘적인 특성으로 인한 프로그램의 동작특성이 매우 규칙적이기 때문이다.

5.6 BARNES

BARNES는 Barnes-Hut의 계층적인 N-body알고리즘을 사용하여 3차원 공간에서 입자들의 이동을 일정한 횟수만큼 시뮬레이션 한다. 크기 N의 1차원 배열 *bodytab*은 시뮬레이션 될 입자들의 정보를 담고 있다. $O(N)$개의 원소로 구성된 octree구조의 *ltab*은 트리의 터미널 노드에 있는 입자들에 대한 배열이고, $O(\log N)$개 요소로 구성된 octree구조의 *ctab*은 트리의 비 터미널 노드들에 대한 배열이다.

세 가지 배열의 크기에 따라 최초접근실패 수는 $O(N+\log N)$이다. 각 반복시기마다 *bodytab*의 모든 $O(N)$개의 입자들이 octree에 있는 $O(\log N)$개의 비 터

표 9 고정된 사양의 SMP에 대한 입력데이타의 크기에 따르는 경험적 모델과 예측 결과

FFT			입력 데이타 크기 = $n=\sqrt{2^{16}}$			입력 데이타 크기 = $n=\sqrt{2^{18}}$		
배 열	항 목	예 측 모 델	실 험 결 과	예 측 결 과	오차 (%)	실 험 결 과	예 측 결 과	오차 (%)
X	REF	$103.10n^2-4058.9n+74832.6$	5,734,400	5,792,624	-1.015	25,034,752	25,024,166	0.042
	CM	$0.9373n^2+8.0950n-10.322$	63,488	63,492	-0.006	249,852	249,855	0.000
	TSM	$0.4373n^2+7.0953n-10.335$	30,464	30,468	-0.013	118,272	118,271	0.000
trans	REF	$111.10n^2-4058.9n+74832.6$	6,258,688	6,316,912	-0.930	27,131,904	27,121,318	0.039
	CM	$0.5040n^2+0.9460n-1.6187$	33,272	33,269	0.006	132,600	132,600	0.000
	TSM	$0.4409n^2+6.1946n+12.0528$	30,468	30,494	-0.087	118,776	118,771	0.004
Total	REF	$214.20n^2-8117.8n+149665$	11,993,088	12,109,536	-0.971	52,166,656	52,145,484	0.041
	CM	$1.441n^2+9.041n-11.941$	96,760	96,761	-0.001	382,456	382,455	0.000
	TSM	$0.8782n2+13.2899n+1.7178$	60,932	60,962	-0.049	237,048	237,042	0.003
	Total Miss	$2.320n^2+22.330n-10.223$	157,692	157,723	-0.0020	619,504	619,470	0.001
	Miss rate(%)	$\frac{2.320n^2+22.330n-10.223}{214.20n^2-8117.8n+149665}\times 100$	1.3149	1.3025	0.943	1.1875	1.1880	-0.042

표 10 고정된 크기의 입력데이타에 대한 프로세서의 수에 따르는 경험적 모델과 예측 결과

FFT			프로세서의 수 = 32			프로세서의 수 = 64		
배 열	항 목	예 측 모 델	실 험 결 과	예 측 결 과	오차 (%)	실 험 결 과	예 측 결 과	오차 (%)
X	REF	5734400	5,734,400	5,734,400	0.000	5,734,400	5,734,400	0.000
	CM	$54636.625(P-1)/P+19863.195$	72,704	72,792	-0.121	73,815	73,646	0.229
	TSM	$54636.625(P-1)/P-13160.803$	39,680	39,768	-0.222	40,837	40,622	0.526
trans	REF	625868	625,868	625,868	0.000	625,868	625,868	0.000
	CM	33278	33,278	33,278	0.000	33,278	33,278	0.000
	TSM	$54372.164(P-1)/P-12896.730$	39,680	39,776	-0.242	40,882	40,625	0.629
Total	REF	6360268	6,360,268	6,360,268	0.000	6,360,268	6,360,268	0.000
	CM	$54636.625(P-1)/P+53141.195$	105,982	106,070	-0.083	107,093	106,924	0.158
	TSM	$109008.789(P-1)/P-26057.533$	79,360	79,544	-0.232	81,719	81,247	0.578
	Total Miss	$163645.406(P-1)/P+27083.662$	185,342	185,614	-0.147	188,812	188,171	0.339
	Miss rate(%)	$\frac{1636452406(P-1)/P+27083.662}{6360268}\times 100$	2.9141	2.9183	-0.144	2.9686	2.9585	0.340

표 11 고정된 사양의 SMP에 대한 입력데이타의 크기에 따르는 경험적 모델과 예측 결과

BARNES			입력 데이타 크기 = 4K			입력 데이타 크기 = 8K		
배 열	항 목	예 측 모 델	실 험 결 과	예 측 결 과	오차 (%)	실 험 결 과	예 측 결 과	오차 (%)
body-tab	REF	$364.774N+30855.0$	1,525,892	1,524,970	0.060	3,016,576	3,019,086	-0.083
	CM	$364.774N-73.709$	19,251	19,607	-1.852	39,438	39,289	0.377
	TSM	$365.064N+483.034$	11,025	10,828	1.787	21,393	21,173	1.028
	FSM	$1.375N+398.981$	5,997	6,050	-0.884	11,670	11,687	-0.146
ctab	REF	$1937.16N\log N-4315971\log N+11821739$	25,855,492	26,296,320	-1.704	62,030,980	61,885,876	0.233
	CM	$0.3677N\log N+908.10\log N-2290.24$	7,378	7,366	0.160	14,690	14,696	-0.046
	TSM	$2.163N\log N+13366.05\log N-24127$	55,599	56,266	-1.128	102,165	102,026	0.136
	FSM	$0.076N\log N+838.346\log N-2089.8$	2,591	2,514	2.969	4,229	4,252	-0.552
ltab	REF	$1180.61N+541097$	6,351,969	5,376,861	15.351	9,678,841	10,212,625	-5.514
	CM	$3.575N+4.963$	14,443	14,649	-1.426	29,373	29,293	0.271
	TSM	$10.525N+5067.44$	46,717	48,177	-3.125	91,429	91,288	0.154
	FSM	$0.071N+108.698$	491	399	18.667	653	689	-5.663
Total	REF	$1937.16N\log N+1545.381N-4315971\log N+12393692$	33,733,353	33,198,151	1.587	74,726,397	75,117,587	-0.523
	CM	$0.368N\log N+8.380N+908.10\log N-2358.9$	41,072	41,622	-1.339	83,501	83,278	0.267
	TSM	$2.163N\log N+13.051N+13366.05\log N-27123.9$	112,512	115,231	-2.417	214,987	214,487	0.233
	FSM	$0.076N\log N+1.446N+838.346\log N-17019.699$	9,079	8,963	1.278	16,552	16,628	0.459
	Total Miss	$2.607N\log N+22.877N+15112.5\log N-31065$	652,663	655,816	-1.938	315,040	314,393	0.205
	Miss rate(%)	$\frac{2.607N\log N+22.877N+15112.5\log N-31065}{1937.16N\log N+1545.381N-4315971\log N+12393692}\times 100$	0.4822	0.4995	-3.581	0.4216	0.4185	0.725

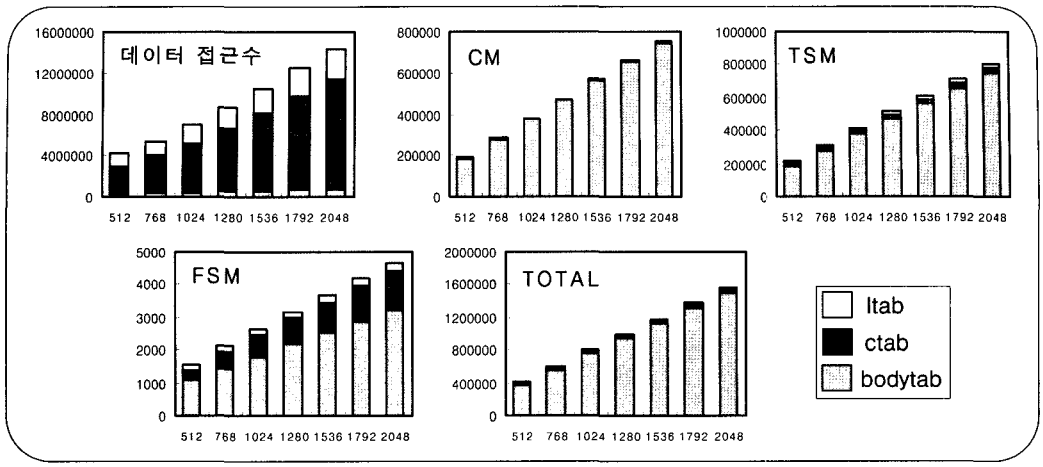


그림 8 BARNES의 시뮬레이션 결과

미널 *ctab*의 원소 중 일부를 부분적으로 순회하고 나서 *ltab*의 원소들을 접근한다. 그러므로 *ctab*과 *bodytab*에 대한 참조는 각각 $O(N\log N)$과 $O(N)$이다.

BARNES는 MP3D나 OCEAN과 같이 프로세서들의 동작특성이 매우 불규칙하다. 특히 octree 구조의 데이타의 터미널 노드 인근을 순회할 때에는 접근되는 데이타의 지역성이 매우 낮아 일반적인 행렬구조의 데이타로 이루어진 MP3D나 OCEAN보다 예측오차가 다소 크게 관측되었다. 따라서 보다 많은 수의 표본점을 수집하여 경험적 모델을 구축하는 데 활용함으로써 모델의 정확도를 높일 수 있다.

5.7 RADIX

RADIX는 N개의 정수형 데이타에 대하여 radix 정렬을 수행하는 프로그램이다. 각 반복시기마다 키 값들의

표 12 고정된 크기의 입력데이타에 대한 프로세서의 수에 따르는 경험적 모델과 예측 결과

BARNES			프로세서의 수 = 32			프로세서의 수 = 64		
배 열	항 목	예 측 모 델	실 험 결 과	예 측 결 과	오차 (%)	실 험 결 과	예 측 결 과	오차 (%)
bodytab	REF	405830	405,830	405,830	0.000	405,830	405,830	0.000
	CM	$2330.514(P-1)/P+2962.669$	5,256	5,220	0.678	5,401	5,256	2.670
	TSM	$10888.731(P-1)/P-5020.632$	5,749	5,527	3.847	5,960	5,697	4.413
	FSM	$4974.394(P-1)/P-2039.115$	2,858	2,779	2.735	3,014	2,857	5.209
ctab	REF	4687310	4,687,310	4,687,310	0.000	4,687,310	4,687,310	0.000
	CM	$-685.457(P-1)/P+2999.743$	2,244	2,335	-4.086	2,325	2,324	0.000
	TSM	$134459.781(P-1)/P-75390.804$	55,727	54,867	1.543	58,162	56,986	2.022
	FSM	$6522.676(P-1)/P-3533.383$	2,802	2,785	0.590	2,953	2,887	2.235
ltab	REF	1517418	1,517,418	1,517,418	0.000	1,517,418	1,517,418	0.000
	CM	$-4875.343(P-1)/P+8071.645$	3,362	3,348	0.396	3,319	3,272	1.401
	TSM	$75196.531(P-1)/P-39864.765$	34,249	32,981	3.699	36,754	34,156	7.069
	FSM	$101.390(P-1)/P+134.744$	177	232	-31.619	276	234	15.081
Total	REF	6610558	6,610,558	6,610,558	0.000	6,610,558	6610,558	0.000
	CM	$-3230.286(P-1)/P+14034.057$	10,862	10,903	-0.377	11,045	10,852	1.747
	TSM	$220545.047(P-1)/P-120276.203$	95,725	93,375	2.455	100,876	96,839	4.002
	FSM	$11598.460(P-1)/P-5437.754$	5,837	5,796	0.702	6,243	5,978	4.245
	Total Miss	$228913.219(P-1)/P-111679.906$	112,424	110,074	2.090	118,164	113,669	3.804
	Miss rate(%)	$\frac{229813.219(P-1)/P-111679.906}{6610558}\times 100$	1.7007	1.6651	2.093	1.7875	1.7195	3.804

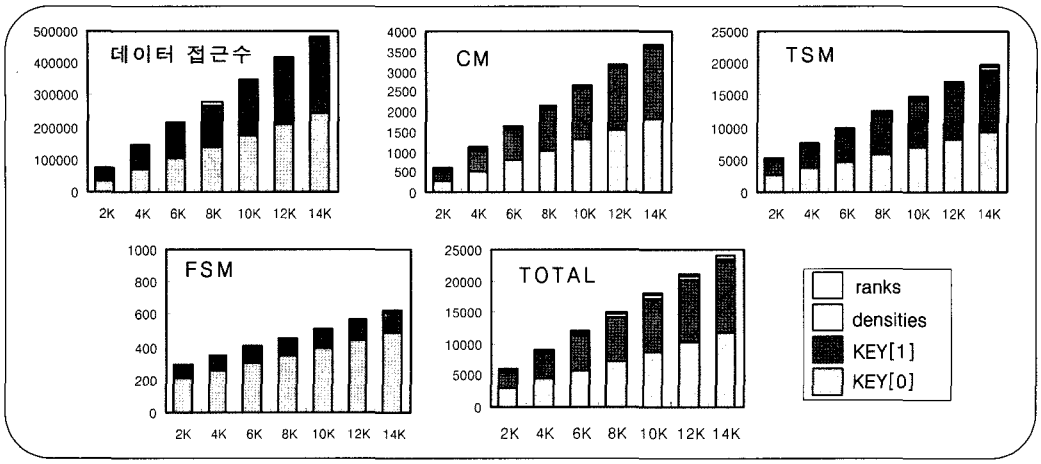


그림 9 RADIX의 시뮬레이션 결과

r-번째 비트를 근으로 하여 반복적인 정렬이 수행된다. 대부분의 데이타 접근은 1차원 배열인 *key[0]*와 *key[1]*의 배열에서 이루어지며 최초접근실패의 수는 $O(N)$이다. 정렬을 위한 반복의 회수는 가장 긴 비트의 길이의 키와 그 근에 의존하는데, 두 인자 모두 프로그램 실행 중에 결정되므로 반복회수는 일정하지 않다. 매 반복시기마다 접근되는 *key[0]*과 *key[1]*배열은 지역적인 히스토그램을 계산하기 위해 접근되고 부분적으로 정렬되어 다른 배열에 저장된다. 그러므로 데이타 접근수와 진성공유데이타접근실패 수는 모두 $O(N)$이다.

가성공유데이타접근실패의 경우는 진성공유데이타접근실패보다 예측이 훨씬 어렵다. 실제로 가성공유데이타

표 13 고정된 사양의 SMP에 대한 입력데이타의 크기에 따르는 경험적 모델과 예측 결과

RADIX			입력 데이타 크기 = 512K			입력 데이타 크기 = 1M		
배 열	항 목	예 측 모 델	실 험 결 과	예 측 결 과	오차 (%)	실 험 결 과	예 측 결 과	오차 (%)
KEY[0]	REF	$17.000N+0.004$	8,912,896	8,912,895	0.000	17,825,792	17,825,792	0.000
	CM	$0.125N+1.000$	65,537	65,537	0.000	131,073	131,073	0.000
	TSM	$0.554N+1379.67$	291,118	292,255	-0.391	583,680	583,132	0.094
	FSM	$0.003N+444.242$	2,235	1,815	18.792	2,974	3,186	-7.128
KEY[1]	REF	$16.000N-0.034$	8,388,608	8,388,609	0.000	16,777,216	16,777,216	0.000
	CM	$0.125N+3.013$	65,539	65,538	0.000	131,075	131,075	0.000
	TSM	$0.626N+742.67$	329,710	329,262	0.136	657,556	657,893	-0.051
	FSM	$0.002N+190.083$	1,691	1,682	0.532	3,173	3,177	-0.126
densities	REF	64,646,464	64,646,464	64,646,464	0.000	64,646,464	64,646,464	0.000
	CM	45	45	45	0.000	45	45	0.000
	TSM	336	336	336	0.000	336	336	0.000
ranks	REF	4,096	4,096	4,096	0.000	4,096	4,096	0.000
	CM	36	36	36	0.000	36	36	0.000
	TSM	288	288	288	0.000	288	288	0.000
Total	REF	$33.000N+10560.0$	17,301,504	1,7301,504	0.000	34,603,008	34,603,008	0.000
	CM	$0.250N+85.013$	131,076	131,075	0.001	262,148	262,148	0.000
	TSM	$1.181N+2746.3$	620,828	621,517	-0.111	1,241,236	1,240,914	0.026
	FSM	$0.006N+634.326$	3,926	3,497	10.927	6,147	6,363	-3.514
	Total Miss	$1.437N+3465.7$	755,830	756,089	-0.089	1,509,531	1,509,425	0.007
	Miss rate(%)	$\frac{1.437N+2913.7}{33.000N+10560.0}\times 100$	4.3686	4.3701	-0.034	4.3624	4.3621	0.006

표 14 고정된 크기의 입력데이타에 대한 프로세서의 수에 따르는 경험적 모델과 예측 결과

RADIX			프로세서의 수 = 32			프로세서의 수 = 64		
배 열	항 목	예 측 모 델	실 험 결 과	예 측 결 과	오차 (%)	실 험 결 과	예 측 결 과	오차 (%)
KEY[0]	REF	2228240	2,228,240	2,228,240	0.000	2,228,240	2,228,240	0.000
	CM	16385	16,385	16,385	0.000	16,385	16,385	0.000
	TSM	$139000.375(P-1)/P-34093.289$	101,839	100,563	1.252	110,313	102,735	6.870
	FSM	$18156.074(P-1)/P-11356.676$	6,903	6,232	9.720	8,124	6,515	19.806
KEY[1]	REF	2097137	2,097,137	2,097,137	0.000	2,097,137	2,097,137	0.000
	CM	16384	16,384	16,384	0.000	16,384	16,384	0.000
	TSM	$141441.875(P-1)/P-34635.703$	104,618	102,386	2.1333	102,515	104,596	2.029
	FSM	$2348.941(P-1)/P-1419.567$	949	855	9.803	967	892	7.756
Total	REF	4325377	4,325,377	4,325,377	0.000	4,325,377	4,325,377	0.000
	CM	32769	32,769	32,769	0.000	32,769	32,769	0.000
	TSM	$280442.250(P-1)/P-68728.992$	206,457	202,949	1.699	212,828	207,331	2.582
	FSM	$20505.016(P-1)/P-12776.243$	7,852	7,087	9.743	9,091	7,407	18.524
	Total Miss	$300947.250(P-1)/P-81505.234$	247,078	242,805	1.729	254,688	247,507	2.819
	Miss rate(%)	$\frac{300947.250(P-1)/P-81505.234}{4325377}\times 100$	5.7123	5.6135	1.730	5.8882	5.7222	2.819

접근실패의 예측은 어렵지만 발생수가 매우 작으므로 전체적인 예측결과에 큰 영향을 미치지 않는다. 그리고 입력 데이타의 크기를 고정시킨 채 구한 프로세서의 수에 따르는 경험적 모델을 이용하여 프로세서의 수에 대한 예측 오차율은, 고정된 사양에 대한 입력데이타의 크기에 따르는 경험적 모델의 정확성으로 미루어 보아, 표본의 수를 증가시킴으로써 다소 향상시킬 수 있을 것으로 기대된다.

표 13에서 보듯이 *dentities*와 *ranks*의 데이타 접근수와 캐시접근실패 수는 모두 상수이다. 따라서 이값들을 표 14에는 포함시키지 않았다.

5.8 데이타들을 구분하지 않은 통합형 모델과의 비교

본 논문에서는 경험적 모델의 정확성을 제고하기 위한 다양한 노력의 일환으로, 한 프로그램에 여러 개의 공유 데이타가 있는 경우 각각의 공유데이타에 대한 경험적 모델을 개별적으로 구한 후 통합하는 방법을 사용하였다. 표 15는, 이러한 방법의 효용성을 보여주는 것으로써, 고정된 사양의 목표시스템에 대하여, 상이한 데이타를 구분하지 않는 방법으로 구한 경험적 모델과 본 논문에서와 같은 방법으로 구한 경험적 모델의 예측오차율을 비교하여 보여준다.

우선 MP3D와 BARNES의 경우에 있어서는 경험적 모델을 이용한 예측오차율이 다소 증가하였으나, 그 차이는 미미하다. 또한 모델의 예측오차율이 이미 극히 낮았으므로 이 정도의 차이는 큰 의미를 갖지 않는다. 반면에 오차율이 3% 내외로 다소 높았던 OCEAN, FFT와 RADIX의 경우에 있어서는 본 논문에서 제시한 방법으로 구한 모델들의 예측오차율이 1% 미만으로 현격하게 낮아지는 효과를 보았다. 결론적으로 말하자면, 하나의 프로그램에 여러 개의 데이타가 존재하는 경우 그들의 동작특성은 매우 상이하며, 따라서 데이타의 동작특성에 기반한 모델을 구축할 때는 본 논문에서와 같이 그들을 개별적으로 다루어야 함을 알 수 있다.

5.9 다른 벤치마크 프로그램

본 논문에서 선택한 벤치마크 프로그램들은 SPLASH와 SPLASH-2에 포함된 것들이다. SPLASH 프로그램 중에서 Pthor와 LocusRoute는 시뮬레이션되는 회로의 입력 값에 대한 명세가 없어서 충분한 크기의 입력 데이타를 구할 수 없었으므로 실험할 수 없었다.

한편 SPLASH의 Cholesky, Radiosity와 기타

표 15 데이타별로 구분하지 않은 모델과의 비교

Benchmarks		LU	MP3D	OCEAN	FFT	BARNES	RADIX
예 측 오차율(%)	데이타 별로 구분하지 않음	-0.080	-0.280	-3.938	-3.023	0.071	2.924
	각 데이타 별로 모델을 구함	0.000	-0.285	-0.525	-0.042	0.725	0.006

SPLASH-2 프로그램들에게 본 논문에서 제안하는 방법을 적용한 결과 매우 높은 예측 오차율을 보였다. 그 이유를 분석한 결과 이 프로그램들은 많은 수의 프로세스를 동적으로 생성하며, 이들 프로세스가 프로세서에게 할당되어 실행되는 과정 역시 동적으로 스케줄되는 공통적인 특성을 갖는다. 따라서 본 논문에서 제안하는 경험적 모델링 기법은 이와 같이 프로세스의 생성과 실행이 동적으로 결정되는 프로그램에는 적용될 수 없음을 밝힌다.

6. 결 론

본 논문에서는, 쉽게 수행할 수 있는 소규모 시뮬레이션을 통하여 표본들을 수집한 후, 최소 제곱 추정법과 로버스트 추정법을 적용하여 경험적 모델을 찾아내는 기법을 소개하였다. 이 방법을 이용하여 SMP에서의 캐시접근실패율에 대한 두 종류의 모델을 구하였다. 첫째는 고정된 사양의 목표시스템에 있어서 입력 데이타의 크기가 변화함에 따르는 모델이며, 둘째는 입력 데이타의 크기가 고정되어 있을 때 목표시스템의 프로세서 수의 변화에 따르는 모델이다.

모델의 정확성을 제고하기 위하여 하나의 프로그램에 존재하는 각각의 공유데이타들에 대하여 각 종류의 캐시접근실패에 대한 개별적인 모델들을 구축한 후 이들을 종합하여 최종적인 모델을 구하였다. 또한 최소 제곱 추정법과 로버스트 추정법을 병용하여 특이점으로 인한 왜곡을 최소화함으로써 모델의 정확도를 향상시켰다. 경험적 모델링의 장점은 다음과 같다. 첫째, 수집된 표본에 내재되어 있는 목표시스템과 작업부하의 동적 특성을 수학적인 공식으로 도출한 결과이므로 정확도가 매우 높다. 둘째, 통계적인 추정방법을 적용하므로 수집된 표본에 대한 별도의 분석이 필요 없다. 셋째, 소규모의 시뮬레이션만 수행하면 되므로 일반적인 실험실에서도 쉽게 활용할 수 있다. 넷째, 실험을 통하여 일련의 표본을 수집할 수 있기만 하다면 다른 분야의 연구에도 적용할 수 있다.

경험적 모델을 이용한 24가지 경우의 예측시도 중 17번의 경우에는 1% 미만의 예측오차율을 보였으며, 나머지 7번의 경우에도 매우 높은 정확도를 보였다. 특히 MP3D, OCEAN 및 BARNES와 같이 프로그램의 실행양식이 불규칙하거나, OCEAN와 FFT와 같이 표본점의 수가 충분히 많지 않은 경우에도 좋은 결과를 보여준다. 그러나 프로세스의 생성과 실행이 동적으로 결정되는 프로그램에는 경험적 모델링 방법이 적용되지 않는다.

SMP에서는 캐시접근실패 이외에도 데이타 무효화 신호나 블록을 메모리에 되쓰는 경우가 발생하지만 본 연구에서는 다루지 않았다. 이들의 발생원인이 캐시접근실패가 발생하는 원인과 밀접하게 관련되므로 본 논문에서 제안하는 방법을 적용할 경우 동일한 수준의 정확도를 보일 것이다. 또한 시스템의 성능에 관한 최종적인 척도는 실행시간이다. 따라서 위의 모델들과 함께 상호연결망에서의 데이타 전송에 대한 모델을 포함한다면 실행시간에 대한 모델을 구할 수 있을 것이다.

그리고 본 논문에서는 입력데이타의 크기에 따르는 모델과 프로세서의 수에 따르는 모델을 개별적으로 구하였다. 입력데이타의 크기와 프로세서의 수에 대한 모든 가능한 조합들에 대한 시뮬레이션 결과들을 수집하고, 적절한 표면 적합(surface fitting) 방법을 개발한다면 보다 완벽한 모델링 기법을 완성할 수 있을 것이다.

끝으로, 본 논문의 제 3.3절에서 밝힌 바와 같이 공유데이타의 구조와 할당 방법 등이 프로그램의 성능에 미치는 영향이 지대하므로, 현재 본 저자의 연구실에서 진행되고 있는 연구의 결과에 따라 보다 체계적이고 일반화할 수 있는 경험적 모델링 방법을 개발할 수 있을 것이다.

참 고 문 헌

[1] M. Brorsson, F. Dahlgren, H. Nilsson, and P. Stenstrom, "The CacheMire Test Bench--A Flexible and Effective Approach for Simulation of Multiprocessors," Proc. of 26th Annual IEEE International Simulation Symposium, pp. 41-49, Apr. 1993.

[2] D. Culler, J. P. Singh, A. Gupta, "Parallel Computer Architecture: A Hardware/Software Approach," Morgan Kaufmann, 1998.

[3] M. Dubois, and J. C. Wang, "Shared Block Contention in a Cache Coherence protocol," IEEE Transactions on Computers, Vol. 40 No. 5, pp. 317-328, May 1991.

[4] A. Gupta, W. Weber, "Cache Invalidation Patterns in Shared Memory Multiprocessors," IEEE Trans. on Computers 41(7): pp. 794-810, Jul. 1992.

[5] E. Rothberg, J. P. Singh, A. Gupta, "Working Sets, Cache Sizes, and Node Granularity Issues for Large-Scale Multiprocessors," Proc. of 20th Ann. Int'l. Symp. on Computer Architecture, pp. 14-25, May 1993.

[6] G. Weerasinghe, I. Antonios, L. Lipsky, "An Analytic Performance Model of Parallel Systems that Perform N Tasks Using P Processors that can Fail," IEEE Int'l. Symp. on Network Computing and Applications, pp. 310-319, 2001.

[7] R. E. Matick, "Comparison of Analytic Performance Models using Closed Mean-value Analysis versus Open-Queueing Theory for Estimating Cycles per Instruction of Memory Hierarchies," IBM Journal of Research and Development, Vol. 47, Issue 4, pp. 495-517, Jul. 2003.

[8] A. Snavely, L. Carrington, N. Wolter, J. Labarta, R. Badia, A. Purkastha, "A Framework for Performance Modeling and Prediction," Conference on High Performance Networking and Computing, Proceedings of the 2002 ACM/IEEE Conference on Supercomputing, pp. 1-17, 2002.
[9] D. Kerbyson, A. Hoisie, H. Wasserman, "Modeling the Performance of Large-Scale Systems," IEEE Proc. on Software, 150(4), pp. 214-221, Aug. 2003.
[10] I. Gluhovsky, B. O'Krafta, "Comprehensive Multiprocessor Cache Miss Rate Generation using Multivariate Models," ACM Trans. on Computer Systems, Vol. 23, No. 2, pp. 111-145, May 2005.
[11] G. Simsion, G. Witt, "Data Modeling Essentials (The Morgan Kaufmann Series in Data Management Systems) 3rd Edition," Morgan Kaufmann, 2004.
[12] R. Hogg, "Adaptive Robust Estimation," Journal of American Statistics Association, 69, pp. 909-927, 1974.
[13] R. L. Launer, G. N. Wilkinson, "Robustness in Statistics," Academy Press, 1978.
[14] W. Press, B. Flannery, S. Teukolsky, W. Vetterling, "Numerical Recipes," Cambridge University Press, 1986.
[15] M. Dubois, J. Skeppstedt, L. Ricciulli, K. Ramamurthy, P. Stenstrom, "Detection and Elimination of Useless Misses in Multiprocessors," Proc. of 20th Annual International Symp. on Computer Architecture, pp. 88-97, May 1993.
[16] M. McLaughlin, "Market Share: Servers, Worldwide, 2Q04," Gartner Research, 2004.
[17] G. Weiss, M. Chuba, "The Future of the Server: A Five-year Outlook," Gartner Research, Jul. 2003
[18] www.intel.com, "Transitioning to the Intel Itanium Architecture," Nov. 2003.
[19] M. I. Hubley, M. Ricjardson, "Windows 2000 Symmetric Multiprocessing(SMP): Perspective," Gartner Research, Oct, 2002.
[20] J. P. Singh, W-D. Weber, A. Gupta, "SPLASH: Stanford Parallel Applications for Shared-Memory," Comp. Arch. News, 20(1):5-44 Mar. 1992.
[21] S. C. Woo, M. Ohara, E. Torrie, J. P. Singh, A. Gupta, "The SPLASH-2 Programs: Characterization and Methodological Consideration," Proc. of 22nd Ann. Int. Symp. on Computer Architecture, pp. 24-36, May 1995.
[22] V. A. Aho, J. E. Hopcroft, J. D. Ullman, "Data Structures and Algorithms," Addison-Wesley Publishing Company, 1983.
[23] M. Parashar, S. Hariri, "Compile-Time Performance Prediction of HPF/Fortran 90D," IEEE Performance Evaluation, pp. 57-73, Spring 1996.

이 강 우

1985년 연세대학교 전자공학과 학사. 1991년 University of Southern California 컴퓨터 공학석사. 1997년 University of Southern California 컴퓨터 공학박사. 1998년~현재 동국대학교 정보통신공학과 부교수. 관심분야는 컴퓨터 구조, 임베디드 시스템, 네트워크 컴퓨팅

양 기 주

1982년 한양대학교 기계공학과. 1984년 University of Wisconsin, 컴퓨터 과학 1986년 University of Michigan, 컴퓨터 과학석사. 1991년 University of Delaware, 컴퓨터 과학박사. 1992년~1995년 한국통신 소프트웨어 연구소 선임연구원. 1995년~현재 동국대학교 부교수. 관심분야는 자연어 처리

박 춘 식

2003년 동국대학교 정보통신공학과. 2005년 동국대학교정보통신공학과 석사. 2005년~현재 동국대학교정보통신공학과 박사과정. 관심분야는 컴퓨터 시스템, 임베디드 시스템