슈퍼 컴퓨팅 환경에서 메타 컴퓨터를 위한 전산 유체 해석 프로그램의 수행 시간 예측 도구 개발

Predicting Execution Times of CFD Solvers
for the Metacomputing System on a Supercomputing Environment

강경우[1], °라선욱[2]
Kyung-Woo Kang, Seon-Uk Na

A metacomputing system is constructed by integrating the heterogeneous supercomputers at KORDIC supercomputer center. The implementation is directed toward building up an easy-to-use supercomputing environment for Computational Fluid Dynamics (CFD) simulations. The key feature is an automatic resource selection According to the predicted performance, the system assigns the request job to the most suitable resource which satisfies the user's demands. Verification tests are conducted by using example CFD applications. An overview of the system is given and derivation of performance prediction models is described in details. Discussion is made of the test results to illustrate applicability and usefulness of the proposed models.

1. 서론

다양한 고성능 컴퓨터의 보급이 증가하고 컴퓨터간 통신이 고속화됨에 따라 주어진 고성능 컴퓨터들을 활용하는 메타컴퓨터의 필요성이 제기되고 있다. 메타컴퓨터는 고속 네트워크로 연결된 이기종, 고성능 컴퓨팅 자원들을 하나의 컴퓨팅 자원과 같이 사용할 수 있게 하는 소프트웨어 시스템이다. 네트워크로 연결된 슈퍼컴퓨팅 자원들을 사용할 때 지금까지 일반적인 방법은 각 사용자가 원하는 컴퓨터 시스템에 접속하여 사용하는 것이었다. 만약 사용자가 접속한 시스템이 많은 작업으로 인해 과부하 상태에 있을 때, 사용자는 또 다른 시스템에 접속하여 현재 부하를 알아보고 과부하 상태가 아니라면 접속한 시스템에서 작업을 수행하게 된다. 이와 같은 과정은 사용자가 각 시스템을 직접 방문하여야 하고 이에 따라 수행하기 원하는 작업을 같이 옮겨야 하는 어려움이 있다. 현재까지 이와 같은 작업을 자동화하기 위한 노력들이 있어 왔다[1, 2, 3, 4, 7].

본 연구에서는 복수의 슈퍼컴퓨터를 운영하는 컴퓨팅 환경의 이용을 효율화하기 위한 기반 연구를 수행하였다. 컴퓨팅 환경의 이용을 효율화하기 위해서는 제출된 작업에 대해 빨리 결과를 줄 수 있는 적합한 슈퍼컴퓨터를 선정하는 것이 필요하다. 슈퍼컴퓨터의 자동 선정을 위해서는 두 가지 방법에 의해 결정될 수 있다. 첫 번째 방법은 사용자에게 사용 가능한 컴퓨터들의 현재 부하를 이용하여 예측하는 방법이다[3]. 두 번째 방법은 한 단계 더 나가서 수행할 작업을 각 컴퓨터 시스템에 할당했을 때 예상되는 성능을 이용하는 방법이다[7, 8]. 첫 번째 방법은 쉽고 간단하게 구현할 수 있으나, 이기종 컴퓨팅 환경에는 적합하지 않다. 기종이 서로 다르면 부하를 표기하는 방법도 다르고 값이 의미하는 바도 서로 다르므로 일관성 있게 비교할 수 없다. 두 번째 방법은 사용자 작업을 분석하고 컴퓨터의 하드웨어 특징을 고려하여 작업이 각 컴퓨터에서 현재 어느 정도의 성능정도의 성능이 나올지를 가늠하는 방법이다.

더 세부적으로 본 연구에서는 전산 유체 해석 문제를 분석하기 위한 프로그램의 성능예측 방법에 대한 연구를 수행하였다. 전산 유체 해석 문제를 해석하기 위한 프로그램들은 프로그램의 일부분을 가지고 전체를 예측하기 쉬운 형태로 작성되어 있다. 이에 대한 자세한 내용은 2절과 3절에서 다룬다. 본 문서의 내용은 다음과 같다. 2절에서 슈퍼컴퓨터의 많은 사용자들이 풀고 있는 전산 유체 해석 문제들의 형태를 알아본다. 문제의 형태는 본 연구의 범위를 정확하게 이해하는데 도움을 준다. 3절에서는 본 연구에서 제안하는 수행시간 예측 방법과 메타컴퓨터 구현을 위한 시

1) 연구 개발 정보 센타 슈퍼 컴퓨팅 사업단 (305-333, 대전시 유성구 어은동 51 번지)
2) 한국 과학 기술원 항공 우주 공학과 (305-701, 대전시 유성구 구성동 373-1 Tel: 042-869-3756)

스템 구조를 기술한다. 4절에서는 개발된 시스템을 이용하여 실험한 실험 결과를 보인다. 마지막으로 5절에서 결론을 말하고 마친다.

2. 전산 유체 해석 프로그램의 특성

전산 유체 해석이란 유체유동현상을 지배하는 Navier-Stokes 방정식을 컴퓨터를 이용하여 근사해를 구하는 방법이다[5]. Navier-Stokes방정식은 연립 편미분방정식으로 이에 대한 해석적인 해를 구하는 것은 거의 불가능하다. 이러한 이유로 컴퓨터의 발전과 더불어 이에 대한 수치적인 해를 근사적으로 구하고자 하는 노력이 계속되어 왔으며 현재는 고속의 컴퓨터를 이용하여 단시간에 이에 대한 해를 구하고 있다. 전산 유체 해석에서 연립 편미분방정식인 Navier-Stokes방정식을 적절한 가정을 도입하여 컴퓨터에서 해석 가능한 대수적인 연립방정식으로 변환하여 이에 대해서 반복적인 방법으로 해를 구하게 된다. 따라서 전산 유체 해석프로그램은 변환된 연립대수방정식에 대한 해를 구하는 프로그램이라고 생각하면 된다. 그러나 이러한 대수방정식은 선형연립방정식이 아니라 각각의 방정식이 서로 연관되어 있는 비선형 연립방정식이 된다. 따라서 이에 대한 해석은 시간항과 공간항이 밀접하게 관련이 되어 있어 각 시간단계에 대해서 공간적으로 모든 계산영역에 대해서 반복적으로 해를 구할 수밖에 없다. 그림 1에 이에 대한 개념이 나타나 있다.

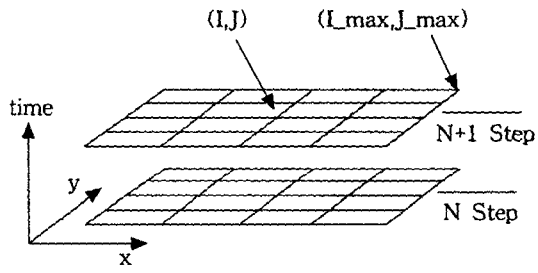

Fig. 1 Concept of the CFD analysis

```
....
do n = 1, Nstep_Max
    do I = 1, I_Max
    do J = 1, J_Max
        . . .
    enddo
    enddo
enddo
....
```

Fig. 2 Structure of a CFD analysis program

그림 2 에 나타난 바와 같이 해석프로그램은 각 N단계에서 x와 y방향(3차원의 경우 x, y, z)으로 각각의 격자점 (I, J)에서 연립방정식의 해를 구하게 된다. 그리고 이때 각각의 격자점 (I, J)에서의 연립방정식은 적어도 M(M≥4)개의 방정식으로 구성되고 전산 유체 해석 프로그램은 그림 2 와 같은 알고리즘으로 표현 할 수 있다 위와 같은 프로그램의 구조를 살펴보면 가장 바깥의 N 루프가 Nstep_Max 회 반복되는 형태로 되어 있다. 따라서 전체 계산수행시간은 (각 N 루프의 계산시간 × Nstep_Max)정도가 된다고 할 수 있다. 그리고 각각의 N루프에서의 계산시간은 프로그램 수행 중에 거의 일정하다고 생각해도 될 것이다.

3. 핵심 루프를 변형한 수행시간 예측

구조화된 전산 유체 해석 프로그램들은 전체 수행시간의 대부분을 소비하는 코드가 있다. 본 연구에서는 구조화된 전산 유체 해석 프로그램 내에서 수행시간의 대부분을 소비하는 부분을 조사하고 이를 변형함으로써 전체 수행시간을 예측한다.

[정의1] 핵심루프

전산 유체 해석 프로그램 내에는 많은 시간을 소비하는 한 개의 루프가 존재한다. 이 루프를 핵심루프라 한다.

핵심루프의 특징은 다음과 같다.

(1) 핵심루프는 수 백번 또는 수 천번 반복한다.
(2) 한번 수행하는데 수 십초의 시간을 필요로 한다.
(3) 핵심루프내의 수행시간은 몇 번째 반복인지에 관계없이 거의 일정하다. 이 특징은 반복 첨자의 값에 상관 없이 데이터만 바꾼 채 같은 과정을 이용하기 때문이다.

이와 같은 특징을 고려한다면 핵심루프를 한번 수행했을 때 속도는 핵심루프의 전체 수행 속도를 예측하는 지표가 될 수 있다. 즉, 반복횟수가 100번이고 한번 루프를 수행하는데 필요한 시간이 30초라면 핵심루프를 수행하는데 필요한 시간은 약 3000초이다.

본 연구에서는 CFD 프로그램을 읽고 핵심루프의 수행 횟수를 변형한다. 그리고, 슈퍼컴퓨터 군에 있는 사용가능 컴퓨터에서 수행하여 수행시간 을 측정한다. 수행 후 제일 먼저 결과를 내는 슈퍼컴퓨터가 원본 프로그램을 수행시켰을 때도 제일 빨리 결과를 낼 수 있다.

프로그램작성자는 프로그램 내에서 핵심루프가 어디인지 표기해 주어야 한다. 프로그램 내에는 수많은 루프가 있다. 이들 중 시간을 대부분 소모하는 핵심루프는 바르게 판단 될 수 없기 때문에 프로그래머는 핵심루프가 시작하는 지점을 다음과 같이 지정해 주어야 한다.

```
      . . . .
      i = 0
cdir! transform
      do 10 j = 1, jmax
      . . . .
10    continue
      . . . .
```

Fig. 3 Source program with the directive identifying kernel loop

```
      . . . .
      i = 0
      Start_time = second()
c     do 10 j = 1, jmax
      . . . .
10    continue
      end_time = second()
      elapsed_time = end_time   start_time
      call cs_send(myparent, IREALX,
     +            elapsed_time, 1, 1, 9998, ierr)
      . . . .
```

Fig. 4 The target source program after transformation

그림 3 과 같이 표기된 프로그램은 그림 4 와 같이 변형한다. 변형 후 성능예측 시스템은 대상이 되는 슈퍼컴퓨터로 프로그램과 자료를 옮기고 수행하게 된다. 이 과정은 다음 알고리즘으로 요약할 수 있다(그림 5).

알고리즘 3.1: 핵심루프를 이용한 성능예측
입력 :
1. 핵심루프를 지정한 구조화된 CFD 프로그램
2. 프로그램 수행에 필요한 자료
3. 프로그램을 컴파일하기 위한 명령과 컴파일 후 수행하기 위한 명령

출력 : 프로그램을 수행시킬 때 가장 빨리 결과를 얻을 수 있는 슈퍼컴퓨터
수행 과정 :
(1) 프로그램 파일들 중 핵심루프를 포함하는 모듈을 찾음
(2) 찾은 모듈을 AST(Abstract Syntax Tree)로 변형하여 핵심루프의 시작과 끝을 찾음
(3) 루프삭제하고 루프를 한번 수행하는데 걸린 시간을 측정하기 위한 수행시간측정 코드 삽입
(4) 슈퍼컴퓨터로 프로그램과 자료를 전송
(5) 각 슈퍼컴퓨터에서 지정된 방법에 따라 컴파일
(6) 각 실행화일을 실행
(7) 각 실행화일이 수행 후 실행 시간 리턴
(8) 리턴된 값들을 이용하여 가장 빠른 컴퓨터를 결정.

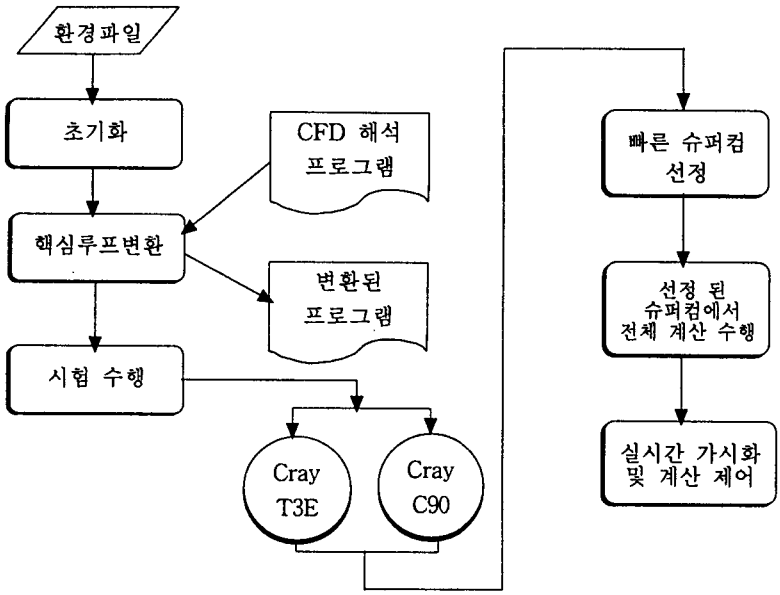

Fig. 5 Resource selection flow based on the kernel loop model

4. 실험 및 제안

본 연구는 연구개발정보센터 슈퍼컴퓨터센터에서 보유하고 있는 슈퍼컴퓨터 상에서 PVM[9]을 이용하여 구현되었다. 대상 기종은 MPP(Massively Parallel Processing)방식인 CRAY T3E를 대상으로 하였다. CRAY T3E는 128개의 CPU를 가진 초고속 병렬 컴퓨터로써 사용자는 복수개의 CPU를 독점하여 사용할 수 있다. 그렇기 때문에 같은 프로그램이라면 언제 수행시켜도 동일한 수의 CPU를 할당받는다면 동일한 시간 안에 결과를 얻을 수 있다. CRAY T3E에서 프로그램 수행 시간에 영향을 줄 수 있는 주 된 요인은 현재 사용 가능한 CPU들의 개수이다

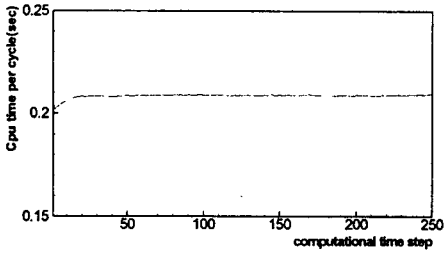

Fig. 6 Computing time per cycle of the 2-D Navier-Stokes analysis with 4 PEs on CRAY T3E

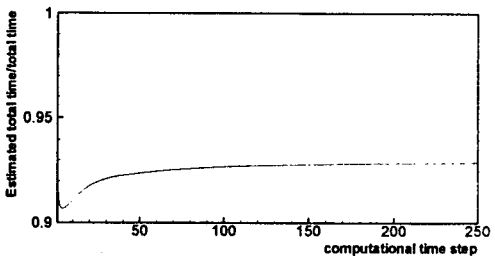

Fig. 7 Predicted total computing time of the 2-D Navier-Stokes analysis with 4 PEs on CRAY T3E

그림 6은 CRAY T3E, 4 개의 cpu에서 PVM을 이용한 2차원 Navier-Stokes 프로그램의 각 계산 단계에서 핵심루프의 cpu 경과 시간을 나타내었다. 각 계산 단계마다 핵심루프의 계산 시간이 거의 일정하게 유지되는 것을 알 수 있다. 그림 7은 2차원 Navier-Stokes 프로그램의 각 계산 단계까지의 핵심루프의 평균 계산 시간으로 예측한 전체 계산 시간과 프로그램의 실제 계산 시간의

비를 나타내는 것으로 계산 시작 후 10초 정도 지나면 전체 계산 시간의 92% 정도를 예측하는 것을 알 수 있다.

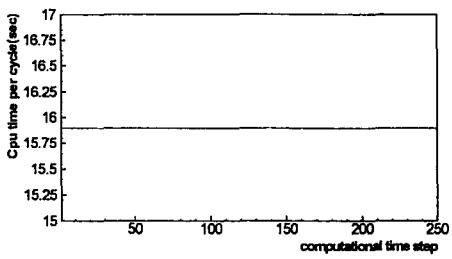

Fig. 8 Computing time per cycle of the NAS LU Benchmark solver with 8 PEs on CRAY T3E

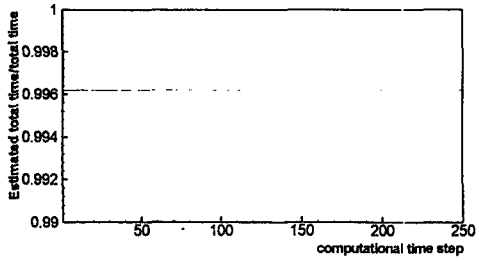

Fig. 9 Predicted total computing time of the NAS LU Benchmark solver with 8 PEs on CRAY T3E

그림 8은 NAS의 병렬 컴퓨터 벤치 마킹 프로그램[10] 중에서 LU 프로그램(C class,162X162X162 격자수)을 CRAY T3E 8개 cpu를 이용하여 계산했을 때 각 계산 단계에서 핵심루프의 cpu 경과 시간을 나타내었다. 핵심루프가 프로그램의 거의 대부분을 차지하고 각 단계마다 계산 량이 일정하므로 핵심루프의 계산 시간이 거의 일정하게 유지되는 것을 알 수 있고 그림 9는 LU프로그램의 각 계산 단계까지의 핵심루프의 평균 계산 시간으로 예측한 전체 계산 시간과 프로그램의 실제 계산 시간의 비를 나타낸다. 핵심루프의 한번 계산에 15.8초 정도의 시간이 걸리고 거의 일정하게 유지되므로 핵심루프의 처음 몇 번의 시험 수행만으로 전체 계산 시간의 99.6%를 예측 할 수 있었다. 그러므로 핵심루프의 계산을 약 10~20초 정도까지 계산하여 전체 계산 시간을 예측하는 것이 가능하다.

5. 결론

복수의 이 기종 슈퍼컴퓨터를 사용할 때 사용자가 더 빠른 결과를 얻어보기를 원한다면 부가적인 노력이 필요하다. 사용자가 각 컴퓨터를 방문하여 사용 가능한 CPU가 얼마나 있는지를 알아내야 하고 자신의 프로그램이 그와 같은 환경에서 어느 정도의 수행시간이 필요한지 추측해야 한다. 본 연구에서는 복수의 이 기종 슈퍼컴퓨터를 효율적으로 사용할 수 있게 하는 도구를 개발하였고 전산 유체 해석 프로그램에 적용하여 유용성을 확인하였다. 제안된 방법은 대상이 되는 응용분야의 해석 프로그램의 특성을 이용하여 프로그램을 변형하고 각 슈퍼컴퓨터에서 핵심코드를 시험 수행시키는 방법이다. 본 연구를 통해 전산 유체 해석 프로그램에서 핵심루프의 시험 수행 시간은 차후 전체 수행시간을 예측하는 지표로 사용할 수 있음을 보였다.

참 고 문 헌

[1] A. Reinefeld et al, The MOL Project: An Open, Extensible Metacomputer, In proc. 6th Heterogeneous Computing Workshop (HCW 97), Geneva, pp. 17-31 1998.

[2] A. S. Grimshaw, J. B. Weissman, E. A. West, and E. C. Loyot, Jr, "Metasystems: An Approach Combining Parallel Processing and Heterogeneous Distributed Computing Systems", Journal of Parallel and Distributed Computing 21, 257-270, 1994

[3] Jinhwen Wang, Songnian Zhou, Khalid Ahmed, and Weihong Long, "LSBATCH: A Distributed Load Sharing Batch System", CSTR-286, CSRI, University of Toronto, April 1993

[4] K. Czajkowski, I. Foster, C. Kesselman, S. Martin, W. Smith and S. Tuecke, "A Resource Management Architecture for Metacomputing Systems", Preprint, mathematics and Computer Science Division, ANL, Argonne, Ill., 1997

[5] Klaus A. Hoffmann, "Computational Fluid Dynamics for Engineers," 1993

[6] Lee D.G., "A New Integrated Software Development Environment Based on SDL, MSC, and CHILL for Large-scale Switching Systems", ETRI journal, v.18, n.4, 265-286, Jan. 1997

[7] N. B. MacDonald, "Predicting Execution Times of Sequential Scientific Kernels", Proceedings International Workshop on Automatic Distributed Memory Parallelisation, Automatic Data Distribution and Automatic Parallel Performance Prediction, Saarbrucken, Germany, March 1-3, 1993.

[8] R. Wolski, N. Spring and C. Peterson, "Implementing a Performance Forecasting System for Metacomputing: The Network Weather Service", SuperComputing '97, UCSD Technical Report TR-CS97-540, May 20, 1997

[9] Al Geist et el., "PVM : Parallel Virtual Machine, A Users' Guide and Tutorial for Networked Parallel Computing," The MIT Press, 1994

[10] http://science.nas.nasa.gov/Software/NPB