비점성 압축성 코드의 병렬화 기법에 의한 슈퍼컴퓨터 CRAY T3E의 성능 분석

고 덕 곤*
시스템공학연구소

초 록

The performances of the CRAYT3E and CRAYC90 were compared in the point of aerodynamics. The CRAYC90 with and without the highest vector option was run, respectively. The CRAYT3E was run with various processors (from 1pe to 32pes). The communication utilities of MPI and SHMEM were used to inform the boundary data to the other processors. The DADI Euler solver, which is implicit scheme and use central difference method, was used. The domain decomposition method was also used. As the result, the CRAYC90 with the highest vector option is 5.7 times faster than the CRAYT3E with 1 processor . However, because of the scalability of the CRAYT3E, the CRAYT3E with more than 6 processors is faster than CRAYC90. In case that 32 processors used, the CRAYT3E is 6 times faster than CRAYC90 with the highest vector option.

1. 서 론

컴퓨터의 계산 속도는 급속도로 발전해 왔으나 순차처리 벡터형 슈퍼컴퓨터의 계산 속도는 이제 한계점에 다다르고 있다. 뿐만 아니라 주기억장치의 용량도 물리적 현상이 주는 한계에 따라 용이하게 확장할 수 없는 면이 있다. 그러나 지구상의 많은 현안 문제를 해결하기 위해서는 TByte(10^{12}bytes)급의 주 기억장치와 TFlops(10^{12}Flops:Floating P.O.P.S.)의 계산 속도를 필요로 한다. 이는 순차처리용의 기존 슈퍼컴퓨터로서는 그 구현이 현실적으로 불가능한 목표이다. 이 목표에 근접하기 위하여는 von Neumann 방식의 현 컴퓨터와는 전혀 다른 방법이 필요함을 알 수 있다. 현재 이 목표에 접근할 수 있는 방법으로 제안되고 발전되고 있는 것이 병렬형 슈퍼컴퓨터이다. 병렬형 슈퍼컴퓨터는 CPU를 계속 추가할 수 있는 확장성이 있어 기억용량이나 계산 속도의 한계를 극복할 수 있는 장점이 있다.

현재 국내에서 운용되고 있는 벡터형 슈퍼컴퓨터와 병렬형 슈퍼컴퓨터의 성능을 공기역학 문제 해석의 관점에서 비교 분석하고자 한다. 먼저 벡터형 슈퍼컴퓨터의 경우 peak-performance가 1 GFlops인 CPU가 16개 장착이 되어있다. 그러나 일반적으로 사용하는 프로그램은 이상적으로 100% 벡터화가 불가능하여 1GFlops를 얻는 것은 불가능하다. 병렬형 슈퍼컴퓨터는 peak-performance가 0.9GFlops인 DEC Alpha EV5 cpu가 128개 장착되어 있다. [본 연구에 CRAY C90와 CRAY T3E의 한 CPU당 계산 속도를 서로 비교 분석하고, CRAY T3E에서 병렬화 하였을 경우 또한 비교 분석 하고자 한다.

계산 방법에 있어 벡터형 슈퍼컴퓨터인 경우 계산 영역을 한 프로세서에 할당하여 계산

하지만 병렬형 슈퍼컴퓨터의 경우는 사용되는 프로세서 수로 계산 영역을 나누어 계산한다. 그러므로 물리적 경계가 아니고 단지 프로세서의 경계인 경우는 서로 데이터를 주고 받는 통신이 필요하게 된다. 프로세서간에 통신하는 방법에는 PVM과 MPI 등의 범용의 라이브러리와 CRAY사 고유의 SHMEM등을 사용할 수 있다. 통신 시간별로 구분하면 SHMEM, MPI, PVM 순으로 효율적이다. 본 연구에서는 데이터 통신을 위하여 MPI와 SHMEM을 사용하였다. 병렬의 경우 계신 영역이 분리된 부분은 서로 연속되지 못하므로 explicit한 계산이 되고 프로세서내부의 계산 영역은 implicit한 방법을 사용하여 계산하였다.

본 연구에서 사용되는 문제는 공기역학에서 프로그램의 정확도를 확인하기 위하여 널리 풀리고 있는 3차원 ONERA M6 날개를 고려하였다. 이 날개에 대한 유동실험의 결과가 받음각 3.06, 5.06 및 6.06도에 대하여 상세히 보고되어 있다[1]. 이 날개는 받음각이 커지면 유동의 박리가 발생하게 되는데 이 박리를 예측할 수 있는 난류 모델은 아직 개발되어 있지 않다. 받음각 5.06도에 한하여 Johnson-King의 난류 모델이 근접하게 예측을 하였지만 아직 개선할 여지가 많이 있다[2]. 그러므로 계산으로 예측할 수 있는 범위는 3.06도의 받음각까지로 유동의 박리가 발생하지 않는 범위이다. 이 받음각에서는 Euler 방정식을 이용하여 계산하여도 실험치에 근접한 압력 계수를 얻을 수 있다.

2. 지배 방정식

3차원 압축성 비점성 유동의 지배 방정식은 질량, 운동량 그리고 에너지의 보존을 나타내는 Euler 방정식이다. 3 차원 압축성 Euler 방정식을 보존형으로 나타내면 지배 방정식은 다음과 같다.

$$\frac{\partial \mathbf{Q}}{\partial t}+\frac{\partial \mathbf{E}}{\partial x}+\frac{\partial \mathbf{F}}{\partial y}+\frac{\partial \mathbf{G}}{\partial z}=0 \qquad (1)$$

지면을 효율적으로 사용하기 위하여 각 변수들과 적분방법은 기술하지 않았다. 본 연구에서 사용한 방법은 Jameson의 중심차분법과 Pulliam Chausse의 DADI방법을 사용하였다. 이는 참고문헌에 자세히 설명되어 있으므로 참고하기 바란다.

3. 병렬 처리와 영역분할

병렬처리는 연산을 수행할 프로세서들과 프로세서간의 필요한 데이터를 주고 받을 통신 라이브러리를 갖추고 있어야 한다. 각 프로세서의 연산은 순차처리 컴퓨터와 다를 것이 없다. 그러나 통신 라이브러리는 병렬 처리 컴퓨터에서만 존재하는 기법이다. 본 프로그램은 시간 전진법에 의해 정상화된 해를 구하게 된다. 그러므로 각 시간 스텝

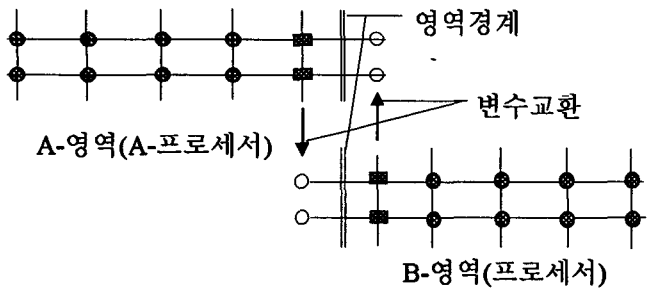

Fig. 1 Overlap 기법

사이에서 서로의 데이터를 주고 받아야 한다. 아래의 그림에서 영역 내의 격자점인 검은 원은 이웃하는 격자점이 모두 존재하므로 계산에 필요한 유동의 유량을 모두 계산할 수 있지만 경계하는 점인 검은 네모의 계산은 이웃하는 격자점이 없어 계산 할 수가 없다. 그래서 가상 격자점을 하나 만들고 가상 격자점에 해당하는 값은 통신을 이용하여 다른 프로세서에 할당된 영역의 값을 이용한다. 그러므로 통신 라이브러리를 이용하여 흰 원에 해당하는 값을 다른 영역에서 가져온다. 이 기법을 overlap기법이라 한다.

데이터 통신은 MPI와 SHMEM을 사용하였다. MPI라이브러리는 블럭킹인 MPI_SEND와 넌블럭킹인 MPI_IRECV를 사용하였다. 블럭킹 MPI_SEND와 MPI_RECV를 사용할 경우, 이는 블럭킹 SEND나 RECV가 서로 끝나지 않으면 RETURN을 해주지 않아 다른 일을 할 수가 없다. 그러므로 프로세서들이 쌍으로 물려 일단 자기들의 통신이 완전히 끝나야 다른 프로세서들과 통신을 할 수 있기 때문에 프로세서간 통신을 병렬로 수행하지 못하고 순차적인 통신을 해야 한다. 블럭킹 MPI_SEND와 MPI_RECV도 프로세서가 4개 이상이면 two step으로 분리하여 병렬로 처리 할 수가 있다. 넌블럭킹 MPI_IRECV를 이용하면 데이터를 다른 프로세서로부터 받아들이면서 다른 일도 할 수 있는 특성을 이용하여, 데이터를 보내는 MPI_SEND작업을 동시에 수행시켜 모든 프로세서가 병렬로 통신을 이룩할 수가 있다.

프로그램에서 사용되는 인덱스나 상수들 그리고 격자계의 입출력은 한 프로세서에서 읽어 다른 프로세서들에 보내는 방법을 사용하였다. 상수들은 MPI_BCAST를 사용하였고 격자계는 MPI_SEND와 MPI_RECV를 이용하여 다른 프로세서에 보냈다.

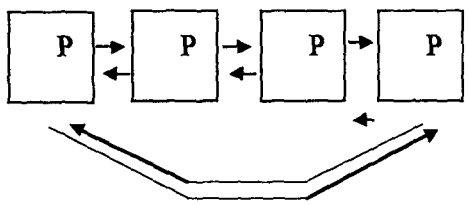

Fig. 2 Parallel communication

SHMEM의 경우는 MPI와 달리 한방향 통신이다. 통신을 할 때 두 프로세서 모두에 명령어가 필요했던 MPI와 달리 한 프로세서만 명령어를 사용하면 된다. 이 때 주의할 점은 A프로세서가 B프로세서의 메모리에서 어떤 데이터를 GET 또는 PUT할 때 B 프로세서가 미리 준비되어 있는지 확인 할 필요가 있다. 즉 GET 명령어의 경우 B 프로세서의 연산이 완전히 끝나지 않아 메모리에는 연산되지 않은 값을 저장하고 있을 수 있다. 그러나 SHMEM은 한방향 통신이기 때문에 B 프로세서의 상태에 상관없이 데이터를 가져올 수 있다. 이를 막기 위하여 통신하기 전에는 반드시 BARRIER함수를 불러 아래와 같이 동기시켜야 한다.

```
CALL BARRIER
CALL GET 또는 PUT
CALL BARRIER
```

SHMEM의 GET과 PUT은 같은 기능을 하나 PUT이 다소 통신시간이짧은 것으로 알려져 있어 본 연구에서는 PUT을 사용하였다.

4. 계산 결과 및 토의

먼저 각 컴퓨터의 단일 프로세서 성능을 파악하기 위하여 본 연구의 프로그램을 각 컴퓨터의 단일 프로세서에서 실행시켰다. 이 계산에서 사용된 격자계는 유동 방향으로 129개, 날개면에 수직한 방향으로 33개, 그리고 날개의 길이 방향으로 30개의 격자를 설정하였다. 본 연구에서 사용된 프로그램은 모든 do loop이 벡터화가 가능하다. 그리고 유동방향의 do loop의 항상 제일 안쪽에 있어 벡터 길이가 거의 유동방향의 격자 수와 비슷해 진다. 이 프로그램을 벡터 최적화한 경우와 벡터화를 억제하여 스칼라 처리한 경우에 대하여 CRAY C90에서 계산하였다. CRAY T3E에서는 프로세서가 data cashe와 secondary cashe에서 각각 4개, 8개씩의 데이터를 한 번에 메모리에서 읽는다. 그러므로 모든 data가 연속하여 있으면 메모리에서 data를 읽는데 걸리는 시간을 최소화 할 수 있어 프로그램의 최적화를 이룰 수 있다. CRAY T3E는 stream 버퍼를 사용할 수도 있고 하지 않을 수도 있다. [이 버퍼도 cashe와 같은 역할을 수행하는 데 MPI나 PVM을 사용할 때는 기본적으로 stream 버퍼가 on되고 SHMEM을 사용할 때는 off 되도록 되어 있다. Stream 버퍼를 사용하지 않을 경우와 사용할 경우의 계산 속도 차이는 10000개의 데이터를 곱하기 연산하였을 경우 메모리 접근 속도의 차이 때문에 계산속도가 3배 가량 차이가 난다. 본 연구에서는 stream 버퍼를 모두 on한 상태에서 계산하였다.] 본 연구의 프로그램은 사용하는 모든 배열들이 유동방향을 Do 루프의 제일 안쪽의 것으로 연속하여 있으므로 최적화가 상당한 진전이 되어 있다고 할 수 있다. 이 계산에서는 밀도에 대한 L_2 norm의 residual이 10^{-9}까지 수렴하기 위하여 약 2000번의 반복 계산이 필요하였다. 병렬 계산의 경우 CRAY T3E의 확장성(scalability)을 알아보기 위하여 프로세서를 1개, 2개, 4개, 8개, 16개, 그리고 32개 사용하였다. 전체적인 격자계의 숫자는 $129 \times 33 \times 30$이고 각 프로세서에 할당되는 격자는 총 격자 수를 프로세서 수로 나눈 양과 같다. Fig. 3a는 계산 시간을 wall clock으로 본 그림이다. 프로세서 수가 증가 하면서 계산 시간이 *1/CPU #)* 곡선에 가깝게 줄고 있는 것을 볼 수 있다. Fig. 3b는 계산 속도의 성능 향상(speedup) 을 본 그림이다. 성능 향상을 구하는 식은 다음과 같다.

$$S = T_1 / T_n \qquad (2)$$

여기서 T_1 : 단일 프로세서를 사용하였을 경우의 계산 시간(Elapsed Time)

T_n : n개의 프로세서를 사용하였을 경우의 계산 시간(Elapsed Time)

SHMEM의 통신 속도가 MPI의 통신 속도보다 빠르므로 성능 향상이 더 현저하다. 두 경우 모두 이상적인 경우에 근접하다가 통신 양이 많은 32개 프로세서에서 overhead 많아 기울기가 작아지고 있다. 이는 각 프로세서에서 수행하는 단위 연산에 대하여 통신에 소요되는 시간이 사용하는 프로세서의 수가 증가하면서 그 비중이 커지기 때문이다. 모든 프로세서가 사용한 cpu 시간을 합한 경우는 Fig. 3c이다. 이 그림에서 병렬화에 따른 overhead를 바로 볼 수 있다. 매우 큰 양은 아니지만 약간씩 증가하는 것을 확인할 수 있다. 프로세서32개와 MPI를 사용하였을 경우 총 CPU시간은 6117초에 이르러 병렬화에 따른 overhead가 21%에 이른다. 같은 조건에서 SHMEM을 사용하였을 경우 5976초가 걸려 overhead가 18%정도이다. MPI와 SHMEM의 차이는 약 3%정도이다.

벡터계산의 CRAY C90와 병렬계산의 CRAY T3E를 비교할 때 프로세서를 6개 이상만 사용하면 병렬 계산의 CRAY T3E가 더 빠른 결과를 보여줄 것이라고 판단할 수 있다.

Fig. 4a는 ONERA M6 날개의 격자계를 보여주는 그림이다. 이 그림은 프로세서16개를 사

용한 경우이다. 유동 방향으로 4개, 날개표면에 수직한 방향으로 2개 날개의 스팬 방향으로 2개의 프로세서를 할당하였다. Fig. 4b는 날개 표면과 대칭면에서 등압력선을 보여준다. ONERA M6 날개의 특징인 λ-충격파가 매우 잘 포착되었다. Fig. 4c는 이 때의 수렴성이다. DADI의 일반적인 특성처럼 잘 수렴하고 있는 것을 알 수 있다. 수렴성은 CRAY C90의 단일 프로세서나 CRAY T3E 8개의 프로세서를 사용한 경우나 크게 다르지 않다.

5. 결론

본 연구의 비교 계산에서 CRAY T3E의 단일 프로세서의 성능은 벡터 계산의 CRAY C90에 비해 약 5.7배 느리지만 스칼라 계산의 CRAY C90보다는 2배 빠르다. 그리고 이 프로세서를 병렬 계산에 사용하였을 경우 CRAY T3E의 프로세서를 6개 이상 사용하면 CRAY C90보다 더 빠른 결과를 얻을 수 있다. 프로세서를 32개 사용하였을 경우 벡터 계산의 CRAY C90보다 5배 이상 빠르다. 그러므로 다수의 프로세서 사용을 기본으로 하는 병렬 계산에서는 CRAY T3E가 효율적으로 활용될 수 있음을 알 수 있다.

참고 문헌

[1] Maex, Y.P. "A practical implementation of turbulence medels for the computation of three-dimensional separated flows", Int. J. Num. Meth. Fluids. Vol. 13. 775-796 (1991)

[2] Abid, R. Vatsa, V.N. Johnson, D.A. and Wedan B.W.,"Prediction of seperated transonic wing flows with a non-equilibrium algebraic model", AIAA paper 89-0558, 1989

[3] Pulliam, T.H., "Artificial Dissipation Models for the Euler Equations", AIAA Journal, Vol. 24, No. 12, December 1984, pp. 955-984.

[4] Beam, R.M., and Warming, R.F., "An Implicit Finite-Difference Algorithm for Hyperbolic Systems in Conservation-Law Form", Journal of Computational Physics, Vol. 22, No. 1, Sept. 1976, PP.87-110.

[5] Pulliam, T.H. and Chausee D.S., " A Diagonal Form of an Implicit Approximate-Factorization Algorith" Journal of Computational Physics, vol 39, 1981

[6] Obayashi, S. and Kuwahara, K., "An Approximate LU Factorization Method for the Compressible Navier-Stokes Equations", Journal of Computational Physics, Vol 63, 1986, pp. 157-167

[7] Yoon, S. and Jameson, A.,"Lower-Upper Symmetric-Gauss-Seidel Method for the Euler and Navier-Stokes Equations", AIAA Journal, Vol. 26, Sep. 1988, pp. 1025-1027.

[8] A. Jameson, W. Schmidt & E. Turkel, "Numerical Solutions of the Euler Equations by Finite Volume Methods Using Runge Kutte Time Stepping Schemes", AIAA Paper 81-1259

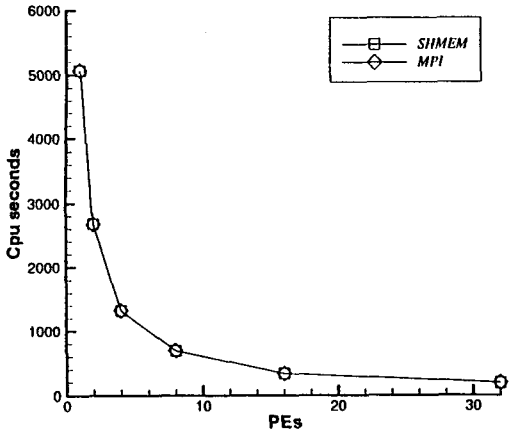

Fig. 3a Comparisons of elapsed cpu time

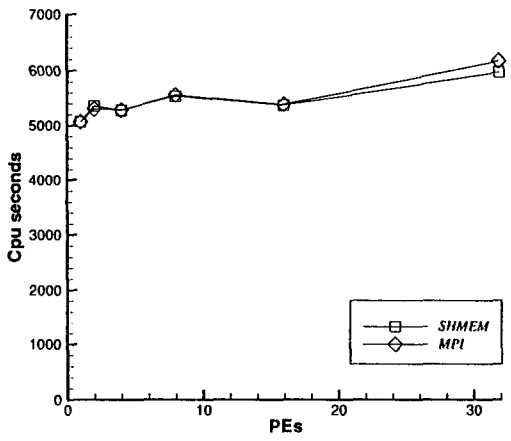

Fig. 3b Comparisons of speedup

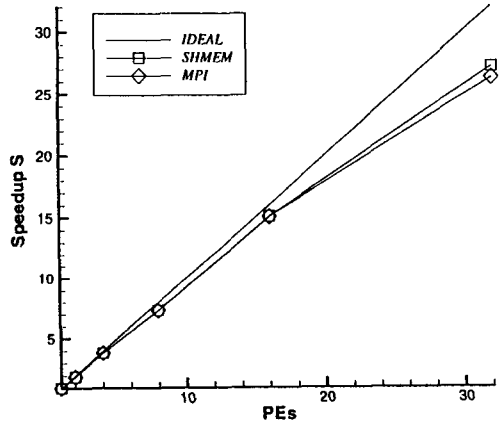

Fig. 3c Comparisons of total cpu time

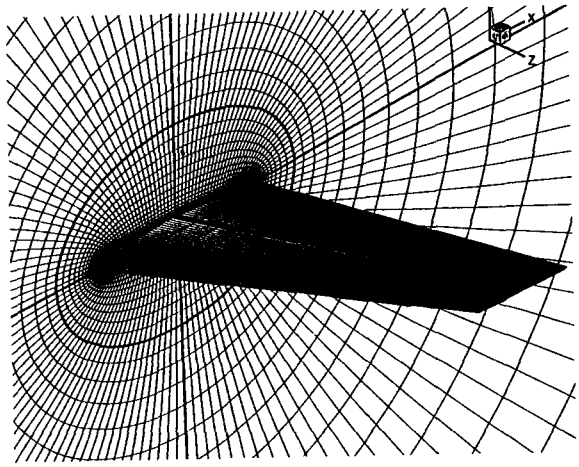

Fig. 4a Grid system of ONERA M6 wing

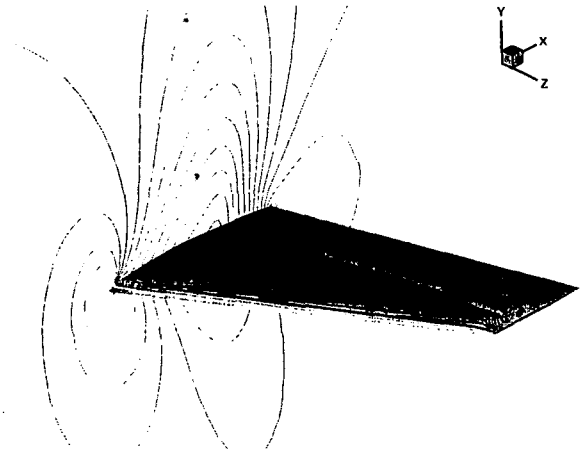

Fig. 4b Iso-pressure line

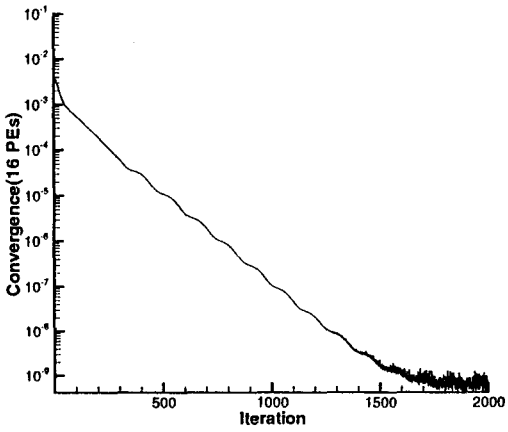

Fig. 4c Convergence history with 16 PEs.