구조물의 동적 해석을 위한 병렬처리기법의 개발

Development of Parallel Processing Technique for Dynamic Analysis of Structures

심 재 수* 박 명 균**
Shim, Jae-Soo Park, Myoung Gyun
(논문접수일 : 2000년 5월 18일 ; 심사종료일 : 2001년 5월 29일)

요 지

구조물이 점점 더 커짐에 따라 그들을 분석하고 설계하는 것이 더 복잡해지고 더 많은 시간이 요구된다. 현재 사용되는 단일 프로세서를 가진 컴퓨터는 그와 같은 구조물을 해석하기에 효율적이지 못하다고 여겨진다. 이 논문에서는 거대규모의 구조물을 분석하기 위하여 컴포넌트 모우드법(CMM)과 메시지전달 시스템(MPI)을 이용함으로써 표준 병렬기법과 고도로 효율적이고 이식성 있는 프로그램을 새로 개발하였다. 이 연구에서는 구조물의 동적 해석을 위해 병렬처리기법을 지닌 컴퓨터 프로그램이 제시되고 새로 개발된 프로그램이 신뢰도를 갖고 있다는 것이 입증된다. 또한 이 프로그램은 상업용 프로그램보다 훨씬 처리속도가 빠르고 병렬처리 컴퓨터에서도 사용될 수 있다는 것을 보여준다.

핵심용어 : 병렬처리기법, 컴포넌트 모우드법(CMM), 메시지전달시스템(MPI)

Abstract

As structures are getting larger, analyzing and designing them become more complicated and require much more time. It is noted that the application of the currently used computers with uni-processor is not efficient in analyzing those structures. In this study, by using component mode method(CMM) and message passing system(MPI), the standard parallel paradigm, a highly efficient and portable parallel program suitable for analysis of large-scale structures has been newly developed. The computer program for this purpose with a parallel processing technique for dynamic analysis of structures is presented, and it is proved that this new program developed in this study has a high reliability. Also it is known that it can compute much faster than the commercial program and it is able to be used on the parallel processing computers.

Keywords : *paralled processing technique*

1. 서 론

산업이 발전하고 인구가 증가함에 따라 기존 구조물의 확장과 새로운 시설물에 대한 수요가 폭발적으로 늘어나고 있다. 그러나 많은 구조물의 건설은 시설물의 입지조건에 제한을 두게 되었고, 또한 구조물의 규모가 장대화, 거대화되고 형태가 복잡, 세밀해져 구조물 해석의 주요한 인자인 자유도(D.O.F)의 수가 증가되고 있다. 따라서 적정수준의 정확도를 갖는 해를 구하기 위하여 컴퓨터의 막대한 연산시간과 메모리 용량이 필

* 경희대학교 토목건축공학부 교수, 공학박사
** 건화엔지니어링 구조부 이사, 공학박사/토목구조기술사

• 이 논문에 대한 토론을 2001년 9월 29일까지 본 학회에 보내주시면 2001년 12월호에 그 결과를 게재하겠습니다.

요하게 되었다. 이러한 문제점들을 해결하기 위해 보다 빠른 성능의 컴퓨터와 프로그램이 필요하게 되었고 단일 프로세서가 내장된 컴퓨터보다 수십 배에서 수천 배 이상의 고성능 처리능력을 가지는 병렬처리 컴퓨터 시스템과 병렬처리 기법이 새로운 연구분야로 대두되었다.[1)~7)] 병렬처리 기법은 많은 수의 자유도를 가진 거대구조물을 해석할 때 동시에 처리할 수 없는 부분을 여러 개의 영역으로 블록화하고, 데이터를 다중 프로세서에서 나누어 해석함으로써 고속연산 능력을 향상시키는 기법을 의미한다.

본 연구에서는 부분 구조 합성법을 사용하여 각각의 부분구조계 해석에서 강성, 질량행렬을 블럭화하였고 동적해석을 독립적으로 수행될 수 있도록 하여 연산시간을 절약하였으며 여러 병렬 컴퓨터에서도 사용 가능하도록 표준 MPI(Message Passing Interface) 병렬처리 언어와 병렬처리 컴파일러를 적용하여 이식성(Portability)과 환산성(Scalability)을 갖도록 하였다.[3)~4)] 따라서 본 논문은 병렬처리 기법과 컴포넌트 모드법(Component Mode Method: CMM)을 결합하여 전체 구조물을 부분 구조물로 분리하고 이것을 다중 프로세서를 이용하여 병렬처리함으로써 구조물의 동적해석을 보다 효율적으로 처리할 수 있는 병렬처리 프로그램을 개발하고자 한다.

2. 동적해석의 병렬처리

2.1 병렬 프로그램 모델

동적해석을 위한 병렬처리 프로그램의 개발은 병렬컴퓨터의 구조와 병렬처리 언어에 따라 프로그램의 결과가 다르게 나타나며 병렬 프로그램의 모델은 크게 세가지로 나뉘어진다. 즉, 병렬컴퓨터의 구조와 병렬처리 언어의 이해에 따라 크게 메시지 전달(Message Passing), 데이터 병렬(Data Parallel), 그리고 공유 메모리 병렬(Shared Memory Parallel)이 있다.

메시지 전달 방법은 이식성과 환산성이 있는 분산 메모리 시스템에 적합한 방법으로 각 프로세서가 독립적으로 프로그램을 실행하며 포트란(Fortran), C, C^{++} 등의 재래언어를 사용하므로 편리하나 프로세서간의 통신이 필수적이다. 필요한 데이터를 이동하거나 프로세서간에 업무를 효율적으로 분산시키기 위한 제어 메시지의 전달이 프로그래머에 의해 제어되므로 프로그래머의 능력에 따라 프로그램의 병렬처리 능력이 좌우된다. 메시지 전달은 LAN, CHIMP, MPICH, UNIFY, Chameleon, MPL, NX등 많은 메시지 전달 시스템이 있으나 대표적으로 PVM(Parallel Virtual Machine)과 MPI(Message Passing Interface)을 사용하고 있다. 특히 MPI는 프로그래머가 프로그램을 자유로이 조절할 수 있도록 하여 프로그램의 동작을 제어하여 각 프로세서간의 통신은 명시적인 언어로써 처리하여 실용성(Practicability), 풍부한 기능성(Functionality), 이식성(Portability), 효율성(Efficiency) 등이 뛰어나 메시지 전달 라이브러리로 표준화되어 널리 사용되고 있다.[4)]

본 연구에서는 분산 메모리 시스템이나 초병렬 컴퓨터(MPP : Massively Parallel Processor)시스템에 병렬 프로그램 모델로서 국제적으로 표준화되고 기존의 재래언어를 사용할 수 있는 MPI와 Fortran을 사용하였으며 병렬컴퓨터는 분산메모리 시스템으로 최근에 도입된 총 프로세서 수가 136개(시스템 프로세서 8개, 사용자 프로세서 128개)인 초병렬 컴퓨터인 Cray T3E를 사용하였다.

2.2 병렬처리 프로그래밍 기법

병렬처리는 계산 과정에서의 여러 가지 작업 중 병렬로 처리될 수 있는 부분들을 동시, 혹은 정해진 시간 내에 처리하는 방식으로 병렬처리 기법의 세분화된 형태는 통신최소화(Communication Minimization), 영역분할(Domain Decomposition), 작업균형(Load Balancing), 행렬분할(Matrix Decomposition) 등이 있으며 네가지 병렬처리 기법을 결합하여 구조물의 동적해석을 위한 병렬처리에 적합한 프로그램을 구성하였다.[5)~6)]

컴포넌트모드법은 전체 구조계를 몇 개의 부분 구조계로 분할하여 각 부분구조계에서 경계요소를 포함한 소수의 D.O.F만을 남기고 축소한 후 각 부분구조계를 결합하여 축소된 전체 구조계를 구성함으로서 주어진 정밀도에서 연산시간을 줄이는 동적해석 방법이다.

부분구조계의 연산과 D.O.F의 축소는 동시에 수행할 수 있는 과정이며 부분 구조계의 자동 생성(Generation)에 필요한 최소의 데이터와 각 프로세서에서의 연산후 축소된 D.O.F는 통신을 최소화시키므로 병렬처리에 가장 적합한 동적 해석법이다.

병렬처리 기법 중 가장 중요한 기법은 첫째로 통신의 최소화이다. 병렬처리 프로그램의 성능 개선은 직렬처리 프로그램에 비해 고속 연산을 할 수 있다는 것인데 이 고속 연산에 제한을 주는 주요인은 프로세서간의 통신이다. 연산과정에서 동시에 작업할 수 있는 부분을 각 프로세서에서 연산할 경우 그 연산 시간에 비하여 프로세서간의 통신 시간이 더 길다면 병렬처리 프로그램보다 직렬처리 프로그램이 더 효율적일 것이다. 그러므로 병렬 프로그램의 최대 수행 효율을 얻기 위해서는 통신을 최소화하여야 한다. 본 연구에 사용된 표준 MPI 통신으로 집단적 통신(Collective Communication)인 MPI_Scatter/MPI_Gather를 사용하였고[7] 각 프로세서의 부분구조계 연산을 위하여 MPI_Scatter/MPI_Gather 메시지 전달함수를 한번만 사용하여 통신의 최소화를 기하였다. 또한 부분 구조계의 요소, 절점 등을 자동 생성하기 위한 절점좌표와 부재의 특성등의 최소 데이터를 전송한 후 각 프로세서에서 부분구조계의 강성, 질량행렬을 구성하고 Ritz- Lanczos 벡터를 사용하여 부분 구조계를 축소하여 전송데이터를 최소화하였다. 이러한 데이터는 메시지 전달함수에 적합한 1차원 배열로 압축 저장하여 MPI_Scatter/MPI_Gather 메시지 전달함수에 의해 각 프로세서에 송수신하였다.

둘째로는 컴퓨터 프로그래밍에서 영역의 분할로써 이것은 계산 영역을 공간적으로 분할하는 것을 말한다. 병렬처리 기법으로 사용되는 영역 분할 방식으로는 Schwarz 방식, 다중 격자(Multigrid) 방식, 직교 순환 이등분법(Orthogonal Recursive Bisection), 분광 순환 이등분법(Spectral Recursive Bisection)이 있는데 본 연구에서는 직교 순환 이등분법(ORB)을 사용하여 정보를 프로세서의 배열 좌측과 우측의 절반으로 송신하고 이 프로세서들은 자기 영역의 상부와 하부의 절반인 프로세서에 데이터를 보내 순환적 이등분 방식으로 처리함으로써 프로세서의 수가 늘어남에 따라 연산 시간은 대수적으로 줄이도록 하였다.

셋째로 작업을 동시에 시행하여야 하므로 각 프로세서의 연산 양이 균일하게 분배되어야 한다. 작업의 균형이란 병렬처리 프로그램은 각 프로세서의 연산 과정에서 마지막 프로세서에서의 연산 과정이 끝날 때까지 작업을 멈추고 다음 과정으로 진행하지 못하는데 각 프로세서에서 소요되는 연산 시간이 비슷한 경우 전체 프로그램의 연산 시간이 단축되어 효율성을 향상시킬 수 있다는 것을 뜻한다. 본 연구에서 사용된 MPI_Scatter/Gather는 한 프로세서에서 다중 프로세서로 데이터를 동시에 전송하거나 다른 모든 프로세서에서 정보를 얻기 때문에 각 프로세서에서 가장 긴 연산 시간을 가진 프로세서의 작업에 의해 전체 병렬처리 프로그램의 연산 과정이 영향을 받는다. 컴포넌트 모드법의 경우 각 프로세서에 할당된 부분 구조계의 연산 과정은 강성이나 질량행렬이 자유도에 관련이 되므로 각 부분 구조계의 자유도를 비슷하게 함으로써 각 프로세서에서의 지연 시간을 줄였다. 부분 구조계가 2개인 경우 전체 구조계를 대칭으로 2등분하고 Ritz-Lanczos 벡터를 사용하여 축소구조계도 같은 D.O.F만 남김으로서 각 프로세서에서 같은 연산시간이 소요되도록 하였으며 부분 구조계가 4개의 경우도 각 프로세서의 연산시간이 비슷하도록 축소된 구조계의 D.O.F는 같도록 하여 작업의 균형을 이루도록 하였다. 여기서 축소된 부분구조계는 경계영역의 자유도와 내부영역의 정규화된 자유도의 합으로 구성되며 Ritz-Lanczos 벡터를 사용하여 같은 수의 자유도만 남기고 축소하므로서 작업균형을 이루었다.

넷째로 행렬분할 기법으로 컴포넌트 모드법에서는 전체 구조계의 행렬을 몇 개의 부분 구조계의 행렬로 할당하는 과정[8]으로 각 프로세서에 메시지 전송함수에 의해 전달된 최소한의 데이터에서 부분 구조계의 강성, 질량행렬을 계산한 후 Ritz-Lanczos 벡터에 의해 축소된 강성, 질량행렬이 형성되는데 그 과정은 다음과 같다.[9]

① r번째 부분 구조의 운동방정식 형성

주 프로세서에 입력된 데이터는 MPI_Scatter 메시지 전송함수에 의해 각 프로세서에 1차원 배열의 압축 저장된 데이터로 전송되어 각 프로세서에서 부분 구조계의 요소, 절점을 자동 생성한 후 질량과 강성을 계산하여 r번째 부분 구조의 운동방정식을 형성한다. 또한 식(1)에서 생성된 행렬과 벡터는 동적 구조 시스템을 줄이기 위해 a개의 경계 영역의 자유도(Boundary D.O.F)와 b개의 내부 영역의 자유도(Interior D.O.F)로 나누어 구분한다.

$$[M]^{(r)}\{\ddot{U}\}^{(r)}+[K]^{(r)}\{U\}^{(r)}=\{F\}^{(r)} \qquad (1)$$

여기서,

$$[M]^{(r)}=\begin{bmatrix}[M_{BB}] & [M_{BI}]\\ [M_{IB}] & [M_{II}]\end{bmatrix}^{(r)}$$

: $(a+b)\times(a+b)$의 질량행렬

$$[K]^{(r)}=\begin{bmatrix}[K_{BB}] & [K_{BI}]\\ [K_{IB}] & [K_{II}]\end{bmatrix}^{(r)}$$

: $(a+b)\times(a+b)$의 강성행렬

$$[\ddot{U}]^{(r)},[U]^{(r)}=\begin{Bmatrix}\{\ddot{u}_B\}\\ \{\ddot{u}_I\}\end{Bmatrix}^{(r)},\quad \begin{Bmatrix}\{u_B\}\\ \{u_I\}\end{Bmatrix}^{(r)}$$

: $(a+b)\times1$의 가속도, 변위 벡터

$$\{F\}^{(r)}=\begin{Bmatrix}\{F_B\}\\ \{F_I\}\end{Bmatrix}(r)$$: $(a+b)\times1$의 하중벡터

하첨자 B, I는 경계 영역의 자유도와 내부 영역의 자유도를 나타내고 상첨자 $^{(r)}$는 각 부분 구조계를 나타낸다.

② 동적 구조 시스템을 축소시키기 위해 Ritz-Lanczos 벡터 도입

$i=2,3,\cdots\cdots,b$ $(b>n)$(b : 전체자유도, n : 원하는 자유도)

$$\circ\ [K]\{\overline{X}_i\}=[M]\{X_i\} \tag{2}$$

$$\circ\ \{\overline{X}_i\}=[K^{-1}][M]\{X_i\} \tag{3}$$

$$\circ\ \alpha_i=\{\overline{X}_i{}^T\}[M]\{X_i\} \tag{4}$$

$$\circ\ \{\hat{X}_i\}=\{\overline{X}_i\}-\alpha_i\{X_i\}-\beta_{i-1}\{X_{i-1}\} \tag{5}$$

$$\circ\ \{\hat{X}_i\}=\{\hat{X}_i'\}-\sum_{K=1}^{i}(\{\hat{X}_i'{}^T\}[M]\{X_K\})\{X_K\} \tag{6}$$

$$\circ\ \beta_i=\left(\{\hat{X}_i{}^T\}[M]\{\hat{X}_i\}\right)^{\frac{1}{2}} \tag{7}$$

$$\circ\ \{X_{i+1}\}=\frac{\{\hat{X}_i\}}{\beta_i} \tag{8}$$

③ 생성된 Ritz-Lanczos 계수로부터 고유치 해석 수행

생성된 Ritz-Lanczos계수 α_i, β_i, $(i=1,2.\cdots\cdots.n)$을 사용하여 삼중대각(Tridiagonal)연산에 유리한 QL-알고리듬[10]으로 고유치 해석을 수행한다.

$$\{T_n\}\{\Phi_N\}=\frac{1}{\lambda}\{\Phi_N\} \tag{9}$$

여기서, $T_n=\begin{bmatrix}\alpha_1 & \beta_1 & & & & \\ \beta_1 & \alpha_2 & \beta_2 & & & \\ & \ddots & \ddots & \ddots & & \\ & & & \alpha_{n-1} & \beta_{n-1} & \\ & & & \beta_{n-1} & \alpha_n & \end{bmatrix}$

④ 축소 변환된 운동방정식 형성

각 프로세서에서 식(10)을 식(1)에 대입하고 변환행렬 $[T]^{(r)T}$을 각 항에 곱하면 식(11)과 같은 r번째의 부분 구조물의 축소 변환된 운동 방정식이 형성되는데 이때의 질량, 강성행렬과 하중 벡터는 MPI_Gather 메시지 전송함수 통신을 최소화하기 위해 1차원 배열로 압축 저장한다.

$$\{U\}^{(r)}=\begin{Bmatrix}\{U_B\}\\ \{U_I\}\end{Bmatrix}^{(r)}=\begin{bmatrix}[I_{BB}]^{(r)} & [0]^{(r)}\\ [T_{BI}] & [\Phi_N]^{(r)}\end{bmatrix}\begin{Bmatrix}\{Z_B\}\\ \{Z_N\}\end{Bmatrix}^{(r)}=[T]^{(r)}\{\overline{Z}\}^{(r)} \tag{10}$$

여기서, $[T]^{(r)}$: $(a+b)\times(a+n)$의 변환 행렬

$$[\overline{M}]^{(r)}\{\ddot{\overline{Z}}\}^{(r)}+[\overline{K}]^{(r)}\{\overline{Z}\}^{(r)}=\{\overline{F}\}^{(r)} \tag{11}$$

여기서,

$[\overline{M}]^{(r)}=[T]^{(r)T}[M]^{(r)}[T]^{(r)}$

: $(a+n)\times(a+n)$의 질량행렬

$[\overline{K}]^{(r)}=[T]^{(r)T}[K]^{(r)}[T]^{(r)}$

: $(a+n)\times(a+n)$의 강성행렬

$\{\overline{F}\}^{(r)}=[T]^{(r)T}\{F\}^{(r)}$: $(a+n)\times1$의 하중벡터

$\{\ddot{\overline{Z}}\}^{(r)}$, $\{\overline{Z}\}^{(r)}=\begin{Bmatrix}\ddot{\overline{Z}}_m\\ \ddot{\overline{Z}}_N\end{Bmatrix},\begin{Bmatrix}\overline{Z}_m\\ \overline{Z}_N\end{Bmatrix}$

: $(a+n)\times1$의 가속도, 변위벡터

⑤ 축소된 전체 구조계의 동적 해석

각 프로세서에서 압축 저장된 1차원 배열의 질량, 강성행렬과 하중 벡터를 병렬처리 메시지 전달함수 MPI_Gather에 의해 전송하여 주 프로세서에서 각 부분구조계의 경계영역과 내부영역에 해당되는 부분을 결합함으로써 식(12)와 같은 축소된 전체 구조계의 운동방정식을 형성하고 Newmark의 직접 적분법에 의해 변위를 계산하거나 고유치 해석등의 동적 해석을 수행한다.

$$[\hat{M}]\{\ddot{\hat{Z}}\}+[\hat{K}]\{\hat{Z}\}=\{\hat{F}\} \tag{12}$$

여기서,

$[\hat{M}]=\sum_{r=1}^{P}[\overline{M}]^{(r)}$: 축소된 구조계의 질량행렬

$[\hat{K}]=\sum_{r=1}^{P}[\overline{K}]^{(r)}$: 축소된 구조계의 강성행렬

$\{\hat{F}\}=\sum_{r=1}^{P}\{\overline{F}\}^{(r)}$: 축소된 구조계의 하중벡터

$\{\ddot{\hat{Z}}\},\{\hat{Z}\}=\sum_{r=1}^{P}\{\ddot{\overline{Z}}\}^{(r)},\sum_{r=1}^{P}\{\overline{Z}\}^{(r)}$

: 축소된 구조계의 가속도,변위벡터

$\sum_{r=1}^{P}$은 1~P개의 부분 구조계에서의 행렬 및 벡터를 결합하여 축소된 전체 구조계를 형성하는 것을 나타내고 상첨자 $^{(r)}$는 각 부분 구조계를 나타낸다.

2.3 병렬처리 기법을 도입한 구조물 동적 해석 프로그램의 흐름도

2.2의 병렬처리 기법과 메시지 전달 함수를 도입한 구조물의 동적 해석의 흐름도는 그림 1과 같다.

3. 적용 예 및 고찰

3.1 병렬처리 프로그램의 적용

본 연구에서는 거대 구조물의 동적 해석을 효율적으로 하기 위하여 컴포넌트 모드법과 병렬처리 기법이 결합된 동적 해석 프로그램을 제시하였다. 구조물의 규모가 거대해지고 형태가 복잡하고 세밀해짐에 따라 동적 구조 해석에 막대한 연산시간과 컴퓨터 메모리의 용량이 요구되는데 본 연구에서는 이런 문제점을 극복할 수 있는 병렬처리 기법을 도입한 구조물의 동적 해석 프로

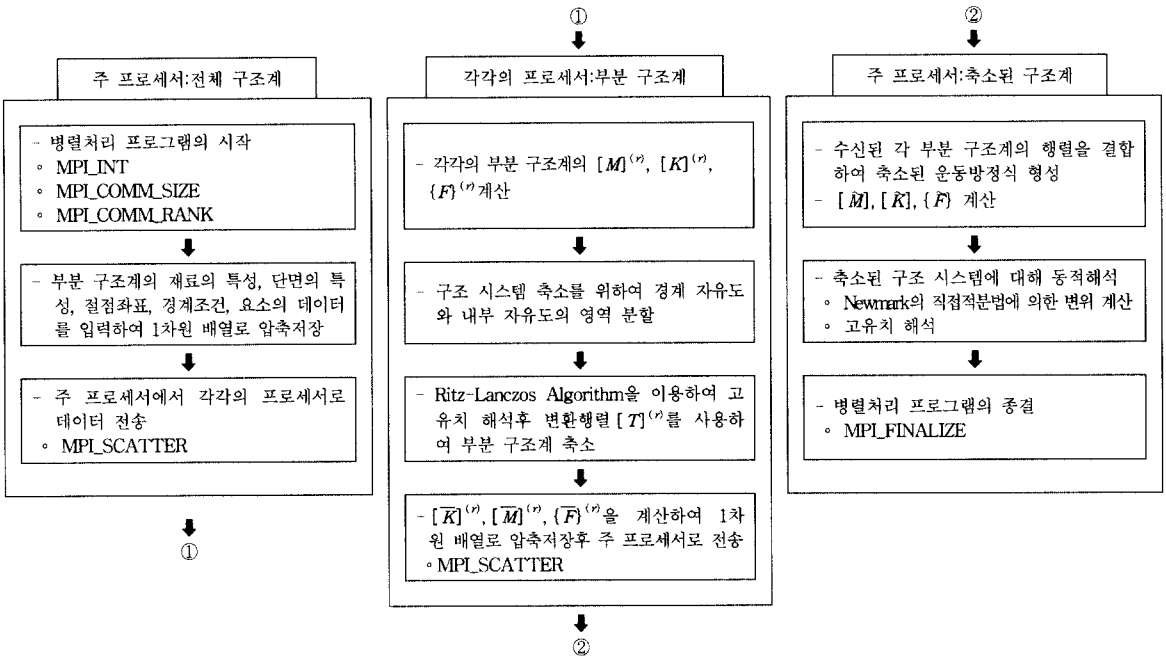

그림 1 병렬처리 기법을 도입한 구조물 동적해석의 흐름도

그램을 개발하였다. 거대 구조물의 경우 전체 구조 연산시간에서 강성행렬과 질량행렬의 연산시간이 대부분을 차지하는 것을 알 수 있으며 이와 같은 프로그램의 실행속도를 저해하는 병목인 행렬 계산은 부분 구조물의 해석을 부분으로 나누어 병렬처리함으로써 수행 능력을 향상시킴을 알 수 있었다. 본 연구에서 개발된 프로그램의 타당성을 입증하고자 몇 개의 트러스를 대상 구조물로 선정하여 적용하였다. 해의 정밀도를 검증하고자 E=1,000psi, A=1.0in^2, υ=0.25, ρ=0.1의 단면특성을 가지는 몇 개의 트러스 구조(12D.O.F, 18D.O.F, 244D.O.F)를 부분 구조계 2~4개로 변화시키면서 상업용 프로그램인 SAP2000과 결과를 비교 분석하였다.

- 12 D.O.F 2차원 단경간 트러스(그림 2)
  - 검토 D.O.F : 5th 절점 (+) Z 방향 변위
  - 하중재하절점 : 5th 절점에 (+) Z 방향으로 조화하중 P=100sin0.1t를 재하
- 18 D.O.F 2차원 2경간 연속 트러스(그림 3)
  - 검토D.O.F : 1st 절점 (-) Z 방향 변위
  - 하중재하절점 : 1st 절점에 (-) Z 방향으로 조화하중 P(t)=100sin20t를 재하
- 244 D.O.F 3차원 3경간 연속 입체 트러스(그림 4)
  - 검토 D.O.F : 44th 절점 (+) Z 방향 변위, 44th 절점은 중앙경간의 중간 하현재의 중앙점

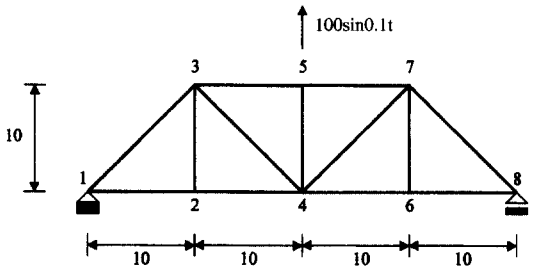

그림 2 12 D.O.F 2차원 단경간 트러스

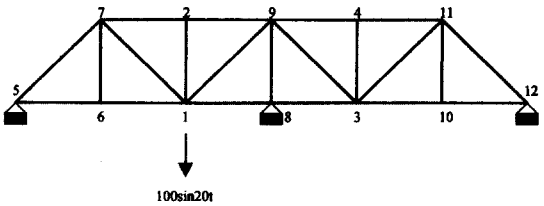

그림 3 18 D.O.F 2차원 2경간 연속 트러스

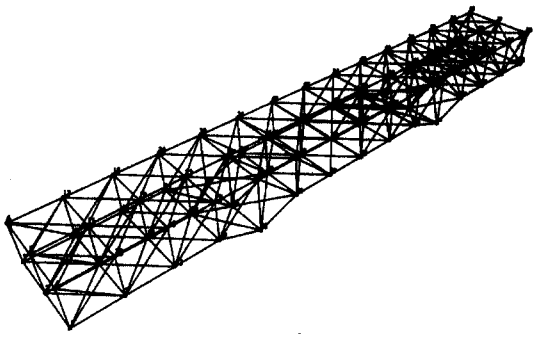
그림 4 244 D.O.F 3차원 3경간 연속 입체 트러스

- 하중재하절점 : 44th 절점에 (+)Z방향으로 조화하중 P(t)=100sin20t를 재하

3.2 병렬처리 프로그램의 해석결과 및 고찰

상기 구조물의 해석결과 및 고찰의 초점은 병렬처리 프로그램의 신뢰성과 연산시간의 단축이므로 이미 신뢰성이 입증된 SAP2000과 비교 분석하여 병렬처리 프로그램의 신뢰성과 성능을 입증하였다.

병렬처리 프로그램의 신뢰성은 244 D.O.F 3차원 3경간 연속입체 트러스의 해석으로 비교검토하였는데 SAP2000의 경우는 Subspace Iteration 방법으로 144개(전체 자유도의 50%정도)이상의 벡터를 사용하여야만 전체 자유도의 해석결과와 근사함을 알 수 있었다. 그러므로 전체 자유도의 50%이상인 168 D.O.F로 SAP 2000과 본 연구 프로그램의 해석치를 전체 구조물 해석치와 비교하여 분석하였다.

0.025초 간격의 44th 절점 Z방향 변위해석에서 표 1과 같이 본 연구와 SAP2000의 해석결과가 전체 구조물의 해석 결과와 근사하므로 본 프로그램은 충분한 신뢰성을 가짐을 알 수 있었다. 동적해석의 연산시간은 표 2와 같다. Cray T3E의 슈퍼 병렬 컴퓨터에서 작은 D.O.F의 구조물은 Cray T3E의 능력에 비하여 간단한 구조물로서 적은 자유도를 가지나 부분구조계의 구성요소를 자동 생성하기 위한 데이터는 구조물의 규모에 관계없이 일정수(경계요소와 정밀도를 주기위한 일부의 내부요소)가 필요하기 때문에 통신시간이 연산시간보다 길어져 직렬처리보다 계산시간이 늦어진다. (12D.O.F 7배, 18D.O.F 4배) 그러나 244D.O.F의 경우는 12 D.O.F, 18 D.O.F에 비하여 구조물의 규모가 커짐에 따라 병렬처리 효율이 좋아짐을 알 수 있

었다. 또한 각 부분구조계의 구성요소를 자동 생성하기 위한 기본데이터(경계D.O.F는 반드시 고려해야 하고 내부D.O.F는 일부를 고려해야 한다.)는 부분구조계별로 거의 비슷하기 때문에 전송에 따른 작업시간이 부분구조계의 수가 증가 할수록 커져 2개의 부분구조계가 4개의 부분 구조계보다 더 효율적으로 나타나나 구조물이

표 1 SAP2000과 병렬처리 프로그램 변위 해석결과 비교

시간간격(초)	전체 자유도(244 자유도) 직접적분법 해석	sap2000 144자유도 subspace iteration 해석	sap2000 168자유도 subspace iteration 해석	본 연구 프로그램 168자유도 컴포넌트 모드법 해석
0.000	0.0	0.0	0.0	0.0
0.025	-0.00146595980	-0.00143755000	-0.00144377000	-0.0014481107
0.050	-0.01127812600	-0.01104590000	-0.01109310000	-0.0111441320
0.075	-0.03520458000	-0.03449320000	-0.03463730000	-0.0348077560
0.100	-0.07454206300	-0.07310020000	-0.07339530000	-0.0737706780
0.125	-0.12541893000	-0.12315300000	-0.12362600000	-0.1242847300
0.150	-0.17950143000	-0.17657400000	-0.17720300000	-0.1782014700
0.175	-0.22595175000	-0.22280000000	-0.22350800000	-0.2248775500
0.200	-0.25416264000	-0.25143200000	-0.25209600000	-0.2538444300
0.225	-0.25658589000	-0.25498200000	-0.25545200000	-0.2575785600
0.250	-0.23093373000	-0.23102600000	-0.23116500000	-0.2336688100
0.275	-0.18117357000	-0.18321200000	-0.18293000000	-0.1858147900
0.300	-0.11702518000	-0.12081800000	-0.12010200000	-0.1233683300
0.325	-0.05203394000	-0.05693300000	-0.05586120000	-0.0594846000
0.350	-0.00064604444	-0.00565159000	-0.00438985000	-0.0082918399
0.375	0.02503604500	0.02105810000	0.02228560000	0.0182646400
0.400	0.01804972600	0.01609490000	0.01704800000	0.0131591170
0.425	-0.02175222000	-0.02109560000	-0.02062250000	-0.0240519620
0.450	-0.08755879300	-0.08429860000	-0.08442990000	-0.0870463290
0.475	-0.16704092000	-0.16183700000	-0.16258400000	-0.1640869700
0.500	-0.24504854000	-0.23910700000	-0.24036000000	-0.2405994300
0.525	-0.30693906000	-0.30176300000	-0.30330600000	-0.3023756600
0.550	-0.34176144000	-0.33880100000	-0.34035400000	-0.3386570900
0.575	-0.34455610000	-0.34484400000	-0.34611700000	-0.3443813700
0.600	-0.31724434000	-0.32110400000	-0.32185500000	-0.3210802800

표 2 직렬처리와 병렬처리시 연산시간의 비교

자유도 수	부분 구조계 수	알고리즘	CPU 시간(초)	비율(%)	프로세서 수
244	2	병렬(parallel)	0.062681	87.8	2
		직렬(serial)	0.071388	100	1
	4	병렬(parallel)	0.108742	92.9	4
		직렬(serial)	0.117040	100	1
18	3	병렬(parallel)	0.031476	377.1	3
		직렬(serial)	0.008347	100	1
12	2	병렬(parallel)	0.030950	752.3	2
		직렬(serial)	0.004114	100	1

수천~수만의 D.O.F가 되어 부분구조계에서 기본데이터보다 자동생성되는 테이터가 월등히 많은 경우에는 부분 구조계가 많아질수록 병렬처리의 효율성이 좋아진다.

244D.O.F의 구조물은 병렬처리 프로그램이 직렬처리 프로그램 보다 87.8~92.9% 정도로 연산 시간이 단축됨을 알 수 있다.

4. 결 론

많은 수의 자유도를 갖는 거대 구조물의 동적 해를 구하는 것은 구조물의 규모가 거대해지고 형태가 복잡하고 세밀해짐에 따라 단일 프로세서의 직렬처리로 프로그램을 실행할 경우 막대한 컴퓨터의 연산시간과 메모리의 용량을 필요로 하게 되었는데 이러한 직렬처리 프로그램의 단점을 극복하기 위해 병렬처리 기법이 요구되었다. 따라서 본 연구에서는 병렬처리 컴퓨터의 기능과 특성에 가장 적합한 컴포넌트 모드법과 병렬처리 기법을 결합하여 구조물의 동적 해석을 보다 효율적으로 처리할 수 있는 프로그램을 개발하였으며 몇 개의 수치 예제를 적용하여 비교 분석함으로써 다음과 같은 결과를 얻을 수 있었다.

1. 컴포넌트 모드법과 병렬처리 기법의 결합으로 거대 구조물 해석에 효율적인 동적 해석용 프로그램을 개발하였으며 표준 MPI로 프로그래밍하므로써 병렬 컴퓨터에 사용 가능한 호환성있는 프로그램을 개발하였다.
2. 전체 자유도를 사용한 동적 해석결과와 신뢰성이 검증된 상업용 프로그램인 SAP2000과 본 논문의 프로그램 해석 결과를 비교하면 SAP2000과 본 논문의 해석 결과 모두 전체 자유도 해석결과와 근사하므로 본 논문의 프로그램의 신뢰도는 충분함을 알 수 있었다.
3. Cray T3E의 슈퍼 병렬 컴퓨터에서 직렬처리와 비교하여 병렬처리시 연산처리 시간이 단축됨을 알 수 있었고 구조물의 크기가 커짐에 따라 연산시간이 비례하여 빨라짐을 알 수 있었다.

이상과 같이 본 연구에서 제시한 프로그램은 병렬처리 기법을 동적 해석에 도입함으로써 해의 정밀도를 확보하고 연산과정을 단축시키기 때문에 거대 구조물 시스템에 적용할 경우 효율적임을 보여 주었다.

참 고 문 헌

1. Heller, D., "A Survey of Parallel Algorithms in Numerical Linear Algebra", SIAM Review, 26(4), pp.740~777, 1978
2. J. J. Dongarra, J. Du Croz, S. Hammarling, and I. Duff, "A Set of Level3 Basic Linear Algebra Subprogram TOMS", Vol. 16. No. 1, 1990 pp.1~16
3. Walker, D. W., "The Design of a Standard Message Passing Interface for Distributed Memory Concurrent Computers", Parallel Computing, 1994, pp.20, pp.657~673
4. Still, C. H., "Portable Parallel Computing via the MPI1 Message Passing Standard", Computers in Physics, 8(5), 1994, 9/10 pp.533~539
5. Lou Baker, Bradley J. Smith, *Parallel Programming*, McGraw-Hill, 1997
6. *Message Passing and Distributed Computing*, Pittsburgh Supercomputing Center, Nov., 1996, pp.11~14
7. Nick Nevin, "The Performance of LAM 6.0 and MPICH 1.0.12 on a Workstation Cluster", Ohio Super-Computer Center Technical Report OSC-TR-1996. 4.
8. 심재수, 박명균, "개선된 컴포넌트 모드법을 이용한 거대 구조물의 동적 해석", 한국전산구조공학회 논문집 제6권 제1집, 1993, pp.37~44
9. 박태현, "Lanczos 알고리듬과 Ritz Vector를 이용한 Component Mode Method에 의한 거대 구조물의 동적 해석", 경희 대학교 대학원 석사학위 논문, 1995. 2
10. William H., *Numerical Recipes in Fortran : The Art of Scientific Computing*, Cambridge University Press, 1992
11. 박명균, "구조물의 동적 해석을 위한 병렬처리기법을 도입한 프로그램 개발", 경희대학교 대학원 박사학위 논문, 1998, 8