대형구조물의 분산구조해석을 위한 PCG 알고리즘

Distributed Structural Analysis Algorithms for Large-Scale Structures based on PCG Algorithms

권 윤 한* 박 효 선**
Kwon, Yun-Han Park, Hyo-Seon

요 지

최근 공학분야에서 다루어지고 있는 문제의 규모가 대형화하고 있으며 이러한 대형구조물의 구조설계는 부재의 강도설계 및 절점의 변위조절을 위하여 많은 수의 구조해석을 요구한다. 한 대의 개인용 컴퓨터에 의한 대형구조물의 구조해석은 대용량의 기억장치와 많은 계산 시간이 요구되므로 반복적 해석이 필요한 대형구조물의 설계에 효율적으로 이용되기 어려운 실정이다. 따라서, 본 논문에서는 이러한 문제에 대한 대안으로 다수의 개인용 컴퓨터들을 네트워크로 연결하여 고성능 병렬연산시스템을 구성하고 이에 적합한 두 가지 형태의 분산구조 방정식해법들을 반복법인 PCG 알고리즘을 이용하여 개발하였다. 대형구조물을 위한 분산구조해석법은 구조해석 과정에 요구되는 각 컴퓨터 상호 간의 통신회수와 통신량을 최소화할 수 있도록 개발되었다. 분산구조해석법의 성능은 대규모 3차원 트러스 구조물 및 144층 가새 튜브구조물의 구조해석에 적용하여 분석하였다.

핵심용어 : PCG 알고리즘, 분산구조해석, 병렬구조해석, 대형구조물

Abstract

In the process of structural design for large-scale structures with several thousands of degrees of freedom, a plethora of structural calculations with large amount of data storage are required to obtain the forces and displacements of the members. However, current computational environment with single microprocessor such as a personal computer or a workstation is not capable of generating a high-level of efficiency in structural analysis and design process for large-scale structures. In this paper, a high-performance parallel computing system interconnected by a network of personal computers is proposed for an efficient structural analysis. Two distributed structural analysis algorithms are developed in the form of distributed or parallel preconditioned conjugate gradient (DPCG) method. To enhance the performance of the developed distributed structural analysis algorithms, the number of communications and the size of data to be communicated are minimized. These algorithms are applied to the structural analyses of three large space structures as well as a 144-story tube-in-tube framed structure.

Keywords *: PCG algorithm, Distributed structural analysis, Parallel structural analysis, Large-scale structure*

* 학생회원 · 영남대 건축공학과, 석사
** 정회원 · 영남대 건축공학과 조교수, 공학박사

• 이 논문에 대한 토론을 1999년 12월 31일까지 본 학회에 보내주시면 2000년 3월호에 그 결과를 게재하겠습니다.

1. 서 론

대형 구조물의 3차원해석, 유한 요소 해석법, 구조 최적화(structural optimization) 문제, 비탄성 해석법등과 같은 많은 계산시간과 대용량의 기억장치를 필요로 하는 구조해석법은 이론적으로 상당히 발전되었지만, 현실적으로는 개인용 컴퓨터나 워크스테이션과 같은 집중형 컴퓨터를 이용하는 경우 하드웨어와 소프트웨어의 한계성으로 인하여 효율적 해석 및 설계가 어려운 실정이다.

이러한 맥락에서 전산 공학 분야의 연구 결과 및 발달 중 공학 분야에 큰 영향을 미치고 있는 것 중의 하나가 여러 개의 프로세서 또는 컴퓨터를 동일한 구조해석 문제에 동시에 효율적으로 이용할 수 있는 병렬컴퓨터(parallel computer)의 개발 및 보급이다. 1980년대 후반부터 구조공학 분야에서는 한 개의 프로세서를 이용한 기존의 직렬구조해석법(sequential structural analysis)이 점차 다양한 형식의 효율적 병렬구조해석법(parallel structural analysis)으로 전환 개발되고 있다.[1),2)] 이러한 병렬구조해석법은 사용되는 하드웨어 측면에서 대형 고성능 전산기를 사용하는 경우와 다수의 기존 PC나 워크스테이션을 이용하는 경우로 나누어 생각할 수 있다.

대형 고성능 전산기를 사용하는 경우는 효율성이 높아 외국에서 많은 연구[1),3),4)]가 진행되고 있고, 이러한 대형 고성능 전산기를 이용한 구조해석 알고리즘은 공유메모리형식 또는 분산메모리형식 등의 각 전산기의 특성에 맞는 병렬구조해석 알고리즘으로 개발되어 이용되고 있다. 그러나 대형전산기의 도입에 관련한 초기 비용 및 유지·관리비용이 많이 소요되고, 특정한 사양으로 도입된 전산기는 다양한 사용자들의 요구에 따라 적절하게 이용되기 어렵다.

그러나 후자의 경우는 여러 대의 개인용 컴퓨터나 워크스테이션을 PVM[5)](Parallel Virtual Machine)이나 MPI[6)](Message Passing Interface)와 같은 병렬라이브러리(library)를 사용하여 가상의 병렬연산시스템으로 구성하여 고성능연산에 이용한다. 이러한 방식은 대형전산기와는 달리 다양한 사용자의 요구에 적합하도록 가상 병렬연산시스템을 구성할 수 있으며 개인용 컴퓨터나 워크스테이션에 사용되는 프로세서의 기능이 많이 향상되고 있으므로 대형고성능 연산기에 대한 대안으로 인식되고 있다. 그러나 가상병렬연산시스템에 사용되는 개인용 컴퓨터나 워크스테이션들이 분산되어 있으므로 각 컴퓨터 상호간의 데이터 통신에 소요되는 시간이 비교적 길다는 문제점을 가지고 있다. 그러므로 이에 적절한 분산구조해석(distributed structural analysis) 알고리즘을 개발하면 고가의 고성능전산기의 도입 없이도 효율적 대형구조물의 해석 및 설계가 가능하게 된다.[7),8),9),10)]

해석에 사용되는 전산기의 성능에 무관하게 대형 구조물의 해석시간은 대부분 구조방정식의 해를 구하는 과정에서 소요되므로 효율적 분산구조 해석법의 개발은 효율적 구조방정식해법에 의해 좌우된다. 일반적으로 구조방정식해법은 직접법과 반복법으로 분류될 수 있으며, 직접법은 자유도수가 증가함에 따라 계산시간이 비선형으로 증가하는 반면, PCG (preconditioned conjugate gradient)[11)]와 같은 반복법은 적은 량의 기억량과 저장장치만을 필요로 하기 때문에 많은 수의 자유도를 가지는 대형 구조물의 해석에 있어서 직접법보다 더 적합하며, 또한 부재단위계산 기법을 도입함으로써 전체강성매트릭스를 조합하지 않고 부재 강성매트릭스만을 구성하여 계산하므로 효율적이다.[4)]

따라서 본 논문에서는 다수의 개인용 컴퓨터 또는 워크스테이션을 이용하여 고성능연산기능을 구현할 수 있는 분산연산 시스템을 개발하고 이에 적합한 분산구조해석법을 PCG 알고리즘을 이용하여 개발하고자 한다. 개발된 분산구조해석법의 성능은 3차원의 대형 트러스 및 프레임 구조물의 해석에 적용하여 해석에 소요되는 시간과 성능향상도(speed-up)[2)]로서 평가한다.

2. 분산연산시스템

본 논문에서 사용된 분산연산시스템 모델은 10 Mbits/sec급의 ethernet card로 연결된 네트워크 환경 하에서 Windows95 체제의 개인용 컴퓨터들을 PVM을 이용하여 구성하였다. 분산연산을 수행하기 위한 분산연산 프로그래밍 모델은 crowd 프로그래밍 패러다임[5)] 중의 하나인 Master-Slave 모델로서 본 연구실에서 보유하고 있는 개인용 컴퓨터

를 필요에 따라 최대 9대까지 연결하여 구성하였다.[7]

분산연산시스템에서 어느 한 대의 컴퓨터(일반적으로 성능이 가장 우수한 컴퓨터)가 Master가 되며 그 역할로는 구조물의 초기 데이터의 입력, 계산량의 균등 분할 및 각 Slave들로의 할당 그리고 최종 결과를 조합하여 출력하는 기능을 담당하는 것이다. Master를 제외한 나머지 컴퓨터들은 Slave가 되며 이들의 역할은 구조방정식을 계산하는 주연산을 담당하게 된다.

3. PCG 알고리즘

본 논문에서 분산구조해석법에 적용될 PCG 알고리즘은 제어매트릭스(preconditioning matrix)와 기본적인 공액경사도법(conjugate gradient method)의 조합으로 이루어졌으며, 부재단위로 해석을 수행하기 때문에 적은 저장량만을 필요로 하므로 대형 구조물의 구조해석 문제에 대해서 계산상 효율적이고, 알고리즘 내의 대부분의 계산들이 매트릭스와 벡터의 곱셈들로 이루어졌기 때문에 직접법보다 병렬화가 용이하다. 이러한 이유에 의해 대형 구조물의 구조 방정식을 풀기에 매우 적합하다는 것이 많은 연구에 의해 증명되었다.[3),12),13)]

$Kx = f$와 같은 구조방정식해법을 위한 PCG 알고리즘은 변위벡터 x_i를 반복적으로 가정하여 잔차벡터 $r_i = f - Kx_i$가 허용오차 이내로 수렴하게 될 때 종료된다. 변위벡터의 수렴에 소요되는 반복회수를 줄이기 위한 방안으로 제어매트릭스를 이용하며 일반적인 부재단위의 PCG 알고리즘의 흐름도는 다음과 같다.

step 1) 제어매트릭스 구성 $C = M^{-1}$

step 2) 초기값 x_0 선택

$r_0 = f - Kx_0$ 계산

$h_0 = C\ r_0$ 계산

$p_0 = h$

for $i = 0, 1, 2, \cdots$ (수렴할 때까지 반복)

step 3) $\alpha_i = \dfrac{r_i \cdot h_i}{Kp_i \cdot p_i}$

step 4) $r_{i+1} = r_i - \alpha_i Kp_i$

$x_{i+1} = x_i + \alpha_i p_i$

$h_{i+1} = C\ r_{i+1}$

step 5) $\dfrac{r_{i+1} \cdot h_{i+1}}{r_0 \cdot h_0} \le \eta$ then stop, else continue

step 6) $\beta_i = \dfrac{r_{i-1} \cdot h_{i-1}}{r_i \cdot h_i}$

step 7) $p_{i-1} = h_{i-1} + \beta_i p_i$

continue iterative cycle

여기서 K는 부재단위 강성매트릭스이며, p_i는 공액 탐색 방향, r_i는 잔차벡터, x는 변위벡터, f는 하중 벡터, h_i는 임시벡터이고, α_i 와 β_i는 p_i방향으로 이동 거리를 제어하는 스칼라 값이며, η는 수렴오차이다.

$Kx = f$와 같은 구조방정식은 제어 매트릭스 M을 양쪽 항에 미리 곱함으로써 식 (1)과 같은 PCG 알고리즘의 방정식이 구성된다.

$$M^{-1}Kx = M^{-1}f \qquad (1)$$

제어매트릭스 M은 대칭 · 양정(positive symmetric) 매트릭스이고 강성매트릭스 K로부터 구해지며, 본 논문에서는 여러 가지 제어매트릭스 중 수렴성과 계산량의 관점에서 적절하며 구성이 쉬운 Jacobi 제어매트릭스(대각 제어매트릭스)를 사용하였다.[3)] PCG 알고리즘 내에서 제어매트릭스 M과 잔차 벡터 r_0의 부가적인 계산이 필요하지만 제어매트릭스를 도입함으로써 전체 알고리즘의 수렴성을 향상시키는데 도움이 된다. 제어 매트릭스 M이 강성매트릭스 K와 비슷할수록 더 빠른 수렴성을 얻게 되나, 그만큼 더 많은 계산량이 요구된다.[13]

또한 부재단위기법[4)]을 적용하게 되면 전체 강성매트릭스를 조합할 필요없이 부재 강성매트릭스만을 사용하여 부재단위로 매트릭스-벡터(Kp)의 내적 계산을 수행하도록 하였다.

분산 PCG 알고리즘과 구별하기 위해 일반적인 PCG 알고리즘을 순차 PCG(Sequential PCG; SPCG) 알고리즘이라 칭한다.

4. 분산 PCG 알고리즘

부재 단위의 SPCG 알고리즘에서 1회 반복시의 계산량은 크게 하나의 매트릭스-벡터 계산(Kp_k)과 2개의 내적 계산($p^T{}_k \cdot (Kp_k)$과 $r^T{}_{k+1} \cdot z_{k+1}$), 그리고 세 개의 벡터계산($x_{k+1} = x_k + a_k p_k$, $r_{k+1} = r_k + a_k[K] p_k$, $p_{k+1} = -z_{k+1} + \beta_k p_k$)으로 구성된다. PCG 알고리즘의 계산량 중에서 매트릭스-벡터 계산이 대부분을 차지하며 이것이 알고리즘의 판단 기준이 될 수 있으며, 분산연산에 있어서 이 부분을 어떻게 분할하여 계산하느냐에 따라 분산해석법의 성능이 좌우된다. 부재단위 PCG 알고리즘에서는 전체 강성 매트릭스를 조합할 필요가 없으므로, 대부분의 계산 과정은 절점과 부재 단위로 수행되며 계산 시간과 필요 기억량은 부재와 절점의 수에 따라 선형적으로 증가하게 된다.

본 논문에서는 개인용 컴퓨터들로 구성된 분산연산 시스템을 이용하여 대형구조물 해석을 수행하기 위해, PCG 알고리즘을 문제분할 방법 및 데이터 전송방법에 따라 분산 PCG 알고리즘 모델 I, II를 개발하였다. 개발된 분산 구조해석 알고리즘은 기존의 수치해석(PCG)에 대해 고유 알고리즘의 특성을 수정하지 않으면서 주 계산을 담당하는 Slave 연결대수와 상관없이 항상 동일한 계산 결과를 해석하도록 개발되었으며 각 알고리즘의 특성은 다음과 같다.

4.1 분산 PCG 알고리즘 모델-I

구조방정식을 구성하는 전체 부재와 자유도 수를 가상병렬시스템에 이용되는 개인용 컴퓨터들에 균등하게 분배하여 변위벡터를 계산할 수 있도록 기존 PCG 알고리즘을 수정하여 분산 PCG 알고리즘 모델-I (이하 DPCG-I)을 개발하였다. DPCG-I의 문제 분할과 할당방법은 부재수(m)와 전체 자유도 수(nd)를 Slave 수(ns)로 나눈 정수만큼의 각종 데이터(강성 매트릭스와 방향 벡터를 구성하기 위한 데이터들)들을 분배하여 각 Slave들이 거의 동일한 데이터량을 가지도록 분할한다. 즉, 각 Slave들의 계산량은 〔m/ns〕과 〔nd/ns〕로 분할되므로 전체 계산량을 Slave 수로 나눈 만큼의 양이 되며 부재번호와 자유도번호가 부여되는 순서대로 분할되기 때문에 분할되는 부재번호와 자유도번호는 서로 무관하게 분할된다.

DPCG-I 알고리즘의 단계별 흐름도는 아래와 같으며 여기서 M은 Master가 수행하는 부분을, S는 Slave가 수행하는 과정을 말하며, 첨자 G는 전체 자유도 크기의 데이터를, s는 nd/ns의 분할된 데이터를, G(s)는 각 Slave에서 자체적으로 계산된 전체 자유도 크기의 데이터를 나타내며, $K^s{}_e$는 각 Slave들에게 할당된 각 부재들의 강성매트릭스를 나타낸다.

step 1) 데이터의 분할 및 할당

M: 데이터를 균등하게 분할하여 각 Slave로 전송한다.

S : Master로부터 데이터들을 할당받는다.

step 2) 제어매트릭스 구성

S: 할당된 데이터를 이용하여 제어매트릭스 $C^{G(s)}$ 를 구성한 후 Master로 송신한다.

M: 각 Slave의 $C^{G(s)}$를 전송 받아 전체 제어매트릭스 $C^G = \sum_{i=1}^{ns} C^{G(s)}$ 를 계산한 후 각 Slave에 필요한 부분만을 전송한다.

S: Master로부터 각 Slave에 계산상 필요한 만큼의 제어매트릭스 요소들을 전송 받아 C^s에 저장한다.

step 3) 벡터들의 초기화

S: 다음과 같이 각 Slave에 할당된 만큼의 각종 데이터들을 초기화하고 δ_0^s를 계산한 후, Master로 전송한다.

$x_0^s = 0$, $r_0^s = f^s - K^s{}_e x_0^s$, $h_0^s = C^s r_0^s$, $p_0^s = h_0^s$, $\delta_0^s = r_0 \cdot h_0$

M: 각 Slave들로부터 δ_0^s를 송신 받아 $\delta_0 = \sum_{s=1}^{ns} \delta_0^s$ 를 계산한다.

step 4) 방향 벡터 구성

S: 계산된 방향 벡터 p^s를 Master에 전송한다.

M: 각 Slave에서 계산된 방향벡터를 전

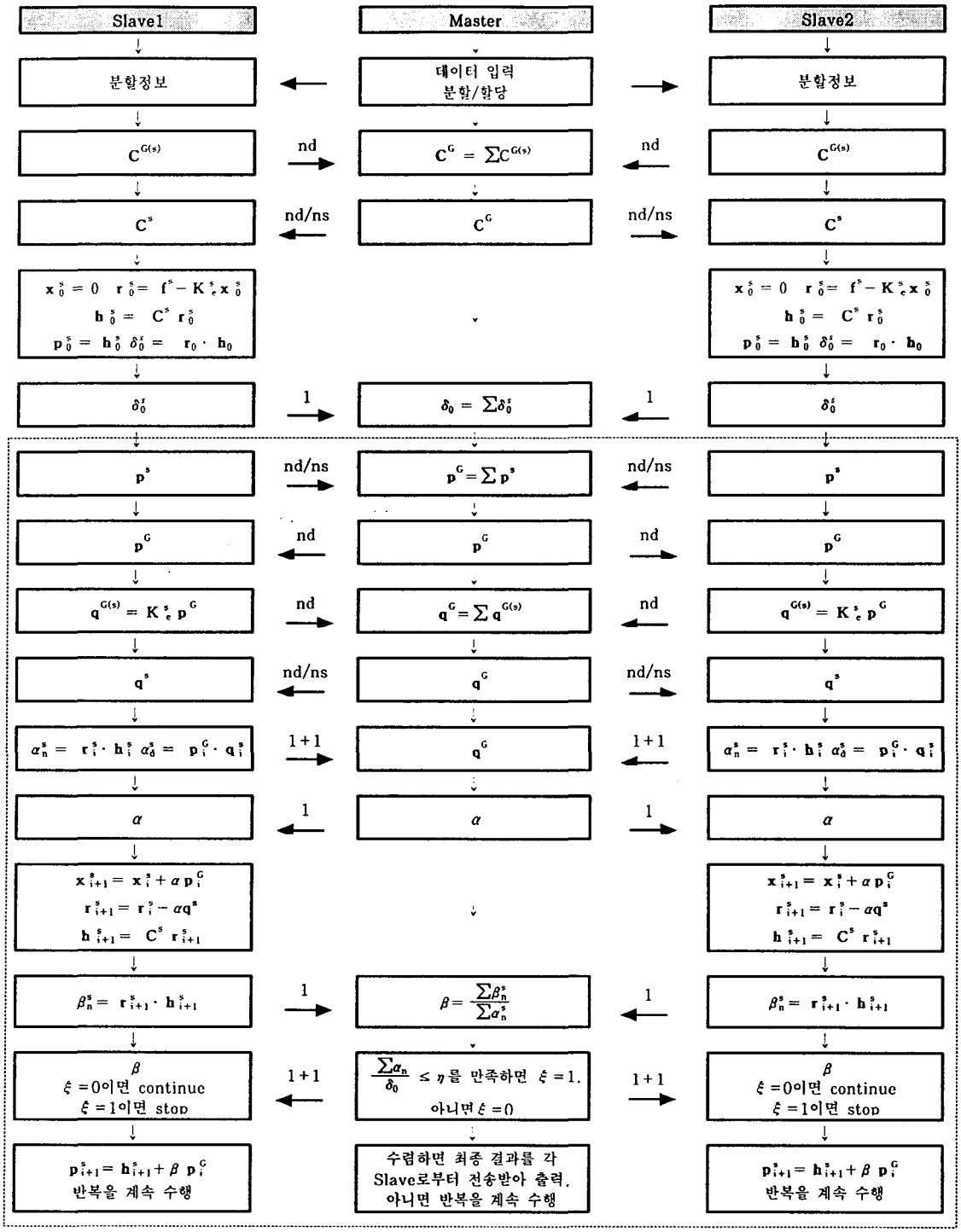


그림 1 DPCG-I 알고리즘 흐름도 (점선상자 부분: 알고리즘의 반복구간)

송 받아 전체의 방향 벡터 $p^G = \sum_{i=1}^{ns} p^s$를 계산하고 이를 전 Slave로 복사한다.

S: Master로부터 전체 방향 벡터 p^G를 전송 받는다.

step 5) 임시 벡터 계산

S: 각 Slave에서 임시 벡터 $q^{G(s)} = K^s_e p^G$를 계산하고 이를 Master로 전송한다.

M: 각 Slave에서 계산된 임시 벡터를 전송 받아 전체의 임시 벡터 $q^G = \sum_{i=1}^{ns} q^{G(s)}$를 계산한 후 다시 각 Slave로 분할된 크기만큼 전송한다.

S: Master로부터 할당된 만큼의 임시 벡터를 전송 받아 $\mathbf{q}^s$로 저장한다.

step 6) 탐색 크기 결정

S: 각 Slave에서 탐색 크기를 결정하는 $\alpha_n^s = \mathbf{r}_i^s \cdot \mathbf{h}_i^s$과 $\alpha_d^s = \mathbf{p}_i^G \cdot \mathbf{q}_i^s$를 계산한 뒤 Master로 전송한다.

M: 각 Slave에서 전송 받은 데이터를 이용하여 탐색크기 $\alpha = \sum\alpha_n^s / \sum\alpha_d^s$ 를 계산한 뒤 이를 각 Slave들에게 전송한다.

S: Master로부터 탐색 크기 α를 전송 받는다.

step 7) 각종 벡터 update

S: 다음과 같이 각종 벡터들을 update한다.

$$\mathbf{x}_{i+1}^s = \mathbf{x}_i^s + \alpha \mathbf{p}_i^G,$$

$$\mathbf{r}_{i+1}^s = \mathbf{r}_i^s - \alpha \mathbf{q}^s,$$

$$\mathbf{h}_{i+1}^s = \mathbf{C}^s \mathbf{r}_{i+1}^s,$$

$$\beta_n^s = \mathbf{r}_{i+1}^s \cdot \mathbf{h}_{i+1}^s$$

M: 각 Slave들로부터 β_n^s을 전송 받아 $\beta = \sum\beta_n^s / \sum\alpha_n^s$를 계산한다.

step 8) 수렴성 점검 및 방향 벡터 update

M: 수렴여부($\sum\alpha_n / \delta_0 \le \eta$)를 계산하여 수렴하면 반복여부 변수 $\xi = 1$, 그렇지 않으면 $\xi = 0$을 저장한다. 각 Slave로 β와 ξ를 전송한다. 수렴하면 최종결과를 각 Slave들로부터 전송 받아 출력하고 프로그램을 종료하며 수렴하지 않는다면 step 4로 이동하여 반복을 계속 진행한다.

S: Master로부터 β와 ξ를 전송 받는다. ξ가 1이면 프로그램을 종료하며 ξ가 0이면 아래와 같이 방향벡터를 수정한 뒤, step 4로 이동하여 반복을 계속 진행한다.

$$\mathbf{p}_{i+1}^s = \mathbf{h}_{i+1}^s + \beta \mathbf{p}_i^G$$

2대의 Slave 컴퓨터들로 구성된 분산연산 시스템에서의 DPCG-I 알고리즘의 흐름도는 그림 1과 같다. 그림에서 수평 화살표는 Master-Slave 간의 통신을 의미하며 화살표 상부의 문자는 통신량을 의미한다.

4.2 분산 PCG 알고리즘 모델-II

분산 PCG 알고리즘 모델-II (이하 DPCG-II)는 가상 병렬시스템에 분산되어 있는 각 컴퓨터 상호간의 데이터 통신에 소요되는 시간이 길다는 문제점을 극복할 수 있도록 구조물을 구성하는 부재와 자유도수의 분할 방식을 변경하여 알고리즘에 소요되는 통신량과 통신회수를 절감한 분산해석 모델이다. DPCG-II의 문제분할 방법은 먼저 구조물 전체 절점 수를 Slave 수로 나눈 뒤, 분할된 절점들을 구성하고 있는 부재들에 대한 데이터를 각 Slave들에게 할당하는 방법을 사용하였다. 이로 인해 각 Slave들이 제어매트릭스 C와 벡터 q를 Message Passing에 의해 구성하지 않고, 할당된 정보들만 가지고 이들을 구성함으로써 계산상 불필요한 통신을 제거하였다. 또한 DPCG-I 알고리즘에서 반복구간의 통신은 Master와 Slave간의 통신인데 비해, DPCG-II에서는 데이터를 공유하고 있는 각 인접 Slave 간에 필요한 데이터들(분할되는 구조물의 경계부분의 자유도에 대한 데이터)만 통신을 하도록 함으로써 전체 통신량과 통신 집중현상을 감소시키고자 하였다.

DPCG-II 알고리즘의 단계별 흐름도는 아래와 같다.

여기서, 첨자 Gs는 할당된 자유도 개수와 경계부분의 자유도 개수의 합에 해당하는 데이터를, s는 각 Slave에 할당된 데이터를, ssb는 다른 Slave로 송신하는 부분을, srb는 수신 받은 부분을 나타낸다.

step 1) 데이터의 분할 및 할당

M: 데이터를 균등하게 분할하여 각 Slave로 전송한다.

S: Master로부터 데이터들을 할당받는다.

step 2) 제어매트릭스 구성

S: 할당된 데이터만를 이용하여 제어매트릭스 $\mathbf{C}^s$를 구성한다.

step 3) 벡터들의 초기화

S: 다음과 같이 각 Slave에 할당된 만큼의 각종 데이터들을 초기화하고 δ_0^s를 계산한 후, Master로 전송한다.

$$\mathbf{x}_0^s = 0,\ \mathbf{r}_0^s = \mathbf{f}^s - \mathbf{K}_c^s \mathbf{x}_0^s,$$
$$\mathbf{h}_0^s = \mathbf{C}^s \mathbf{r}_0^s,\quad \mathbf{p}_0^s = \mathbf{h}_0^s,$$
$$\delta_0^s = \mathbf{r}_0 \cdot \mathbf{h}_0$$

M: 각 Slave들로부터 δ_0^s를 송신 받아 $\delta_0 = \sum_{s=1}^{ms} \delta_0^s$를 계산한다.

step 4) 방향 벡터 구성

S: 계산된 방향 벡터 $\mathbf{p}^s$중 인접한 Slave에 계산상 필요한 요소만($\mathbf{p}^{ssb}$: Slave로 분할된 자유도 데이터 중 경계부의 자유도)을 모아 해당되는 인접 Slave에 전송함과 동시에 자신에게 필요한 경계부의 방향벡터 요소($\mathbf{p}^{srb}$)들을 인접 Slave들에서 전송 받아 새로운 방향 벡터를 구성한다($\mathbf{p}^{Gs} = \mathbf{p}^s + \mathbf{p}^{srb}$).

step 5) 임시 벡터 계산

S: 각 Slave에서 임시 벡터 $\mathbf{q}^s = \mathbf{K}_c^s \mathbf{p}^{Gs}$를 계산한다.

step 6) 탐색 크기 결정

S: 각 Slave에서 탐색 크기를 결정하는 $\alpha_n^s = \mathbf{r}^s \cdot \mathbf{h}^s$과 $\alpha_d^s = \mathbf{p}^G \cdot \mathbf{q}^s$를 계산한 뒤 Master로 전송한다.

M: 각 Slave에서 전송 받은 데이터를 이용하여 탐색크기 $\alpha = \sum\alpha_n^s / \sum\alpha_d^s$ 를 계산 한 뒤 이를 각 Slave들에게 전송한다.

S: Master로부터 탐색 크기 α를 전송 받는다.

step 7) 각종 벡터 update

S: 다음과 같이 각종 벡터들을 update한다.

$$\mathbf{x}_{i+1}^s = \mathbf{x}_i^s + \alpha\, \mathbf{p}_i^{Gs},$$
$$\mathbf{r}_{i+1}^s = \mathbf{r}_i^s - \alpha \mathbf{q}^s,$$
$$\mathbf{h}_{i+1}^s = \mathbf{C}^s\, \mathbf{r}_{i+1}^s,$$
$$\beta_n^s = \mathbf{r}_{i+1}^s \cdot \mathbf{h}_{i+1}^s$$

M: 각 Slave들로부터 β_n^s을 전송받아 $\beta = \sum\beta_n^s / \sum\alpha_n^s$를 계산한다.

step 8) 수렴성 점검 및 방향 벡터 update

M: 수렴여부($\sum\alpha_n / \delta_0 \le \eta$)를 계산하여 수렴하면 반복여부 변수 $\xi = 1$, 그렇지 않으면 $\xi = 0$을 저장한다. 각 Slave로 β와 ξ를 전송한다. 수렴하면 최종결과를 각 Slave들로부터 전송 받아 출력하고 프로그램을 종료하며 수렴하지 않는다면 step 4로 이동하여 반복을 계속 진행한다.

S: Master로부터 β와 ξ를 전송 받는다. ξ가 1이면 프로그램을 종료하며 ξ가 0이면 아래와 같이 방향벡터를 수정한 뒤, step 4로 이동하여 반복을 계속 진행한다.

$$\mathbf{p}_{i+1}^s = \mathbf{h}_{i+1}^s + \beta\, \mathbf{p}_i^{Gs}$$

DPCG-II의 흐름도는 그림 2와 같다.

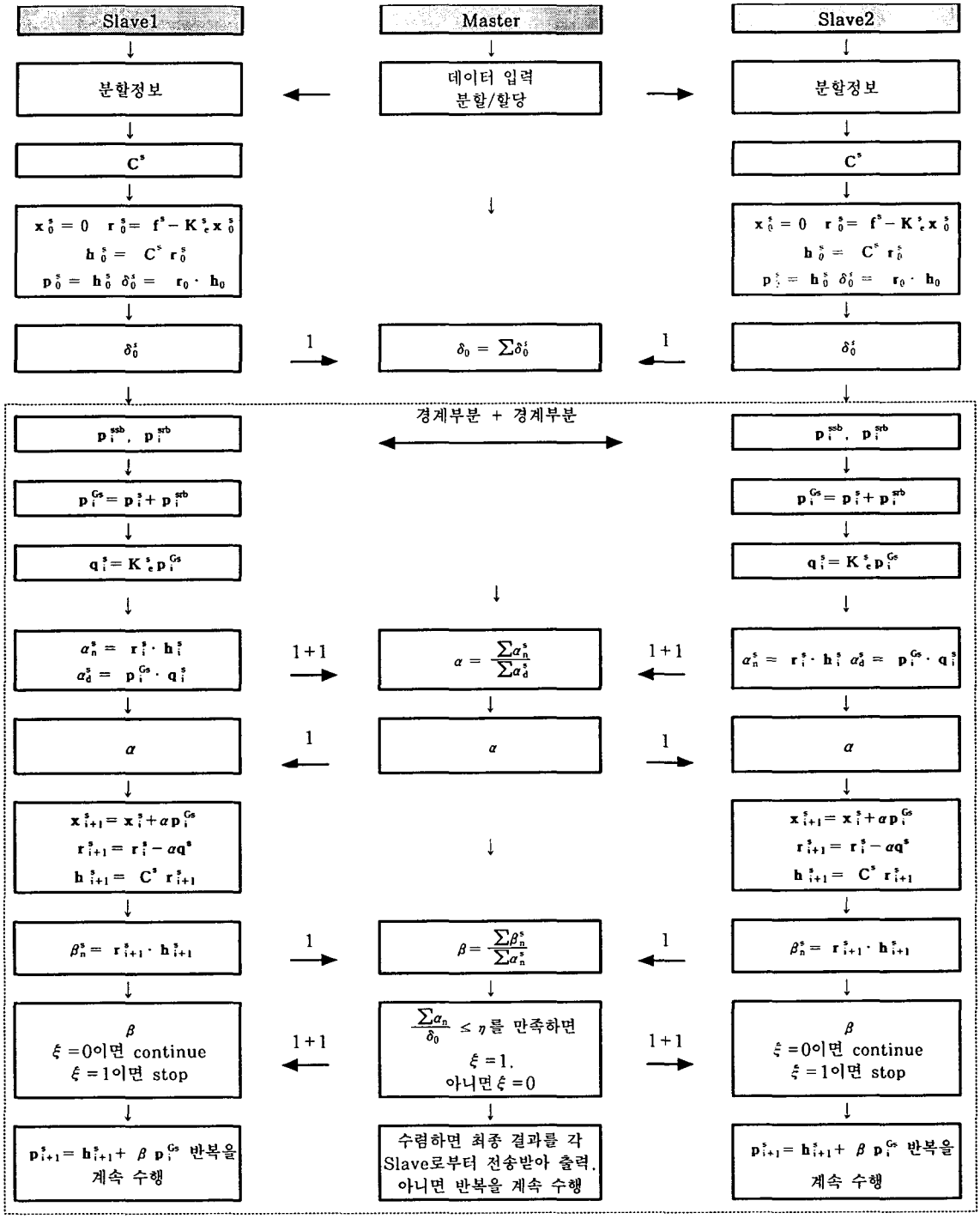


그림 2 DPCG-II 알고리즘 흐름도 (점선상자 부분: 알고리즘 반복구간)

5. 예제 적용 및 분석

개발된 두 개의 분산 알고리즘과 분산시스템의 효율성 및 분산 성능을 분석하기 위해 대형의 입체 트러스 구조물과 프레임 구조물을 적용하였다.

5.1 3차원 트러스 구조물 (30층, 60층, 90층)

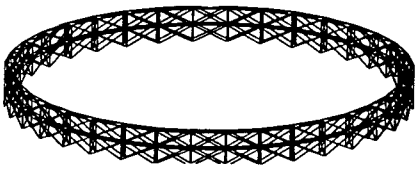
그림 3에 나타난 트러스 구조물은 본 연구에서 구성된 분산시스템의 분산성능을 분석하기 위해 제작된 3차원 트러스 구조물을 구성하는 2개층 단위의 단위 모듈이며 이러한 모듈을 적용하여 30층, 60층 그리고 90층 규모의 트러스 구조물을 구성하였다. 이들 구조물의 절점수는 각각 2,232개, 4,392개, 6,552개이며, 총자유도수는 각각 6,696개, 13,176개, 19,656개로 구성되어 있다.

5.2 144층 가새튜브구조

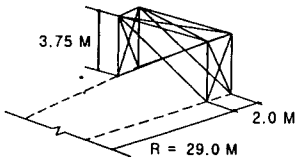
그림 4는 526.7m의 높이를 가지는 144층의 초고층 강구조물을 보여준다. 이 구조물은 튜브인튜브(tube in tube) 구조시스템으로 구성되어 있으며, 8,463개의 절점과 20,096개의 부재로 구성되어 있다. 총 자유도수는 50,778개이며 적용된 하중은 Section 1 부분에서는 고정하중 50 psf와 적재하중 50psf, Section 2와 3부분에서는 고정하중 50psf와 적재하중 40psf이며, 횡하중으로는 UBC(1994)에 의한 풍하중을 적용하였다. 적용된 풍하중은 노풍도 C(개방된 지역), 중요도 1, 기본 풍속 113Km/h로 가정하였다. 이 구조물의 상세도는 참고문헌[14]에 나타나 있다.

5.3 적용 결과 및 분석

한 대의 컴퓨터에서 순차적으로 실행하도록 프로그래밍된 SPCG 알고리즘과 본 논문에서 개발된 두 개의 DPCG 알고리즘을 4가지 예제의 구조해석에 적용한 결과 절점변위 및 응력이 동일하게 나타났다. 본 논문에서는 개인용 컴퓨터들로 구성된 분산연산 시스템의 성능과 이러한 분산 연산 시스템에 적용되는 분산 방정식해법의 성능에 초점을 맞춰 결과를 분석하였으며 그 결과는 표 1, 그림 5와 그림 6에 나타나 있다. 표 1은 각 알고리즘이 1회 반복하는데 소요되는 시간과 전체 계산시간을 보여 주며 그림 5는 표 1을 도식화한 것이다. 그리고 그림 6은 DPCG-I 알고리즘과 DPCG-II 알고리즘에 대한 성능향상도(speed-up)를 보여 준다. DPCG-I 알고리즘은 144층 프레임 예제를 제외한 나머지 예제 모두에서 SPCG보다 더 많은 해석시간이 소요되었으며 또한 Slave 수를 증가시킴에 따라 해석시간도 증대됨을 알 수 있었다.



(a) 2개층 단위 모듈



(b) 단위 모듈 상세

그림 3 3차원 truss구조물의 구성 모듈

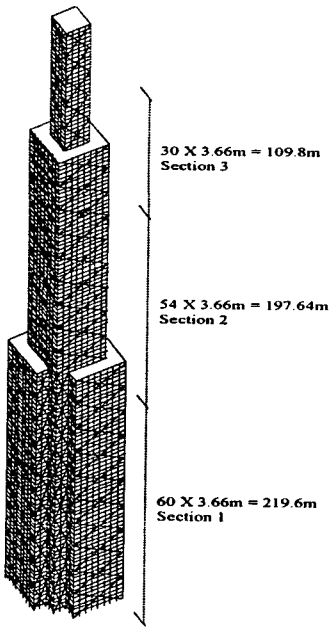


그림 4 144층 프레임 구조물

표 1 각 알고리즘의 1회 반복시 소요시간(단위:초)

적용예제 / 층수(자유도수)	알고리즘	1대	Slave 2대	Slave 4대	Slave 8대
30층 트러스 (6,696)	SPCG	0.67 (442)	-	-	-
	DPCG-I	-	1.21 (799)	1.51 (997)	2.94 (1,940)
	DPCG-II	-	0.58 (383)	0.62 (409)	0.85 (561)
60층 트러스 (13,176)	SPCG	1.35 (1,755)	-	-	-
	DPCG-I	-	2.33 (3,029)	2.76 (3,588)	5.18 (6,734)
	DPCG-II	-	1.09 (1,417)	0.83 (1,079)	0.98 (1,274)
90층 트러스 (19,656)	SPCG	2.38 (4,522)	-	-	-
	DPCG-I	-	3.85 (7,315)	4.03 (7,657)	7.43 (14,117)
	DPCG-II	-	1.30 (2,470)	0.98 (1,862)	1.38 (2,622)
144층 프레임 (50,778)	SPCG	29.2 (18,250)	-	-	-
	DPCG-I	-	7.8 (4,875)	9.8 (6,125)	18.6 (11,625)
	DPCG-II	-	2.4 (1,500)	1.94 (1,212)	2.2 (1,375)

※ ( )안의 수는 전체 계산에 소요된 시간을 나타낸다.

반면, DPCG-II 알고리즘은 30층 트러스의 경우 Slave 수가 2일 때 최대의 성능향상을 보였으며 60층 트러스, 90층 트러스 그리고 144층 프레임의 경우 Slave가 4대 일 때 최대의 성능향상을 보였다. 특히, 전체 자유도수가 50,778개인 144층 프레임 구조물 예제의 경우, Slave 수를 4대로 하였을 때, DPCG-II 알고리즘은 SPCG 알고리즘 보다 15배나 더 빠르게 계산할 수 있었다. 이러한 이유로는 SPCG 알고리즘의 경우, 144층 프레임 구조물의 많은 데이터를 한 대의 컴퓨터에서 모두 저장하고 계산해야 하기 때문이다. 즉, 계산상 필요한 기억요구량이 한 대의 컴퓨터가 보유하고 있는 주기억장치의 한계를 벗어나게 되고 이로 인한 하드디스크의 가상 메모리화와 이러한 가상메모리의 매우 느린 전송속도로 때문에 많은 시간이 소요된다[15].

반면, 분산연산 시스템을 구성하여 DPCG-I 또는 DPCG-II와 같은 분산 알고리즘을 적용하면 전체 계산에 사용되는 기억량이 분산연산 시스템을 구성하는 개개의 컴퓨터들로 분산되기 때문에 한

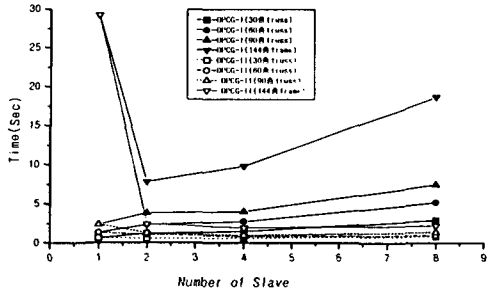


그림 5 각 알고리즘의 1회 반복시 소요시간

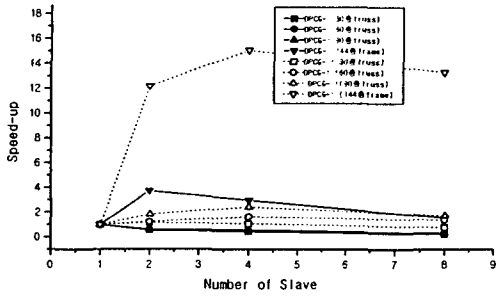


그림 6 DPCG 알고리즘의 성능향상도

표 2 1회 반복시의 각 알고리즘의 통신량(KByte)

적용예제 / 층수(자유도수)	알고리즘	Slave 2대	Slave 4대	Slave 8대
30층 트러스 (6,696)	DPCG-I	314	523	942
	DPCG-II	10.2	22.7	48.8
60층 트러스 (13,176)	DPCG-I	618	1029	1853
	DPCG-II	10.2	22.7	48.8
90층 트러스 (19,656)	DPCG-I	921	1536	2764
	DPCG-II	10.2	22.7	48.8
144층 프레임 (50,778)	DPCG-I	2380	3967	7141
	DPCG-II	17.72	38.35	81.21

대의 컴퓨터가 보유하고 있는 기억장치의 한계 내에서 대부분의 계산이 가능하므로 이러한 부가적인 시간의 발생을 줄일 수 있다.

DPCG-I 알고리즘과 DPCG-II 알고리즘을 비교해 보면, 모든 예제에서 DPCG-II 알고리즘이 더 효율적으로 적용되었음을 알 수 있다. 이러한 결과의 주된 요인으로는 표 2에서 알 수 있듯이 통신량의 현저한 차이 때문이다. 동일예제에서 동일한 Slave 수의 분산시스템에 적용된 두 알고리즘의 연산량이 거의 동일한데 반해 통신량은 상당한 차이가 있음

을 알 수 있다. DPCG-II 알고리즘의 반복구간에서의 통신량이 DPCG-I 알고리즘보다 144층 구조물의 경우 88배~134배 정도가 적으며 이러한 반복구간 내에서의 통신량에 의해 알고리즘의 성능이 좌우된다는 것을 알 수 있다.

6. 결 론

다수의 개인용 컴퓨터들을 동일한 구조물의 구조해석에 동시에 사용할 수 있는 가상병렬연산시스템을 개발하였다. 그리고 이에 적합한 분산구조해석 알고리즘을 반복법인 PCG 알고리즘을 기본으로 하여 DPCG-I과 DPCG-II로 나누어 개발하였으며, 개발된 분산구조해석법은 4가지 3차원 트러스 구조물과 프레임 구조물의 해석에 적용하였다. 분산연산시스템의 성능과 각 분산구조해석 알고리즘의 성능을 분석한 결과를 요약하면 다음과 같다.

1) 해석하고자 하는 구조물의 규모(자유도 수)가 적은 경우, 연산의 분산에 의한 계산시간의 감소 보다 분산연산에 소요되는 통신량의 증가가 많아지기 때문에 분산구조해석기법의 효율성을 기대하기 힘들다. 그러나 해석 대상구조물의 자유도 수가 많은 대형구조물의 경우 분산구조해석법의 효율성은 높아지게 된다.

2) 개인용 컴퓨터들로 구성된 분산연산시스템에서의 분산구조해석법의 성능은 구조물을 구성하는 전체자유도 및 부재의 분할 방식과 이에 관련한 데이터의 통신량과 통신회수에 좌우된다.

3) DPCG-II 알고리즘을 개인용 컴퓨터 4대를 이용하여 분산구조해석한 결과, 144층 초고층 가새 튜브 프레임 구조물(자유도수: 50,778개)의 경우 한 대의 개인용 컴퓨터를 이용할 경우 해석에 소요되는 시간을 5시간 정도에서 20분 정도로 감소시킴으로서 분산구조해석법의 효율성을 확인할 수 있었다.

감사의 글

이 논문은 1999학년도 영남대학교 학술연구조성비 지원에 의한 것임을 밝히며 이 지원에 깊이 감사드립니다.

참 고 문 헌

1. Adeli. H. and Kamal. O., *Parallel Proceing in Structural Engineering*, Elsevier Applied Science, New York, 1993
2. K. Mani Chandy and Stephen Taylor, *An Introduction to Parallel Programming*, Jones and Bartlett Publishers, 1992
3. A. I. Khan& B. H. V. Topping, "Parallel Finite Element analysis using Jacobi-conditioned Conjugate Gradient Algorithm", *Advances in Engineering Software*, No. 25, 1996, pp.309~319
4. E. Barragy and Graham F. Carey, "A Parallel Element by Element Solution Scheme", *Int. J. Numer. Methods Eng.*, Vol. 26, 1998, pp.2367~2382
5. Al Geist, *PVM : Parallel Virtual Machine, A Users' Guide and Tutorial for Networked Parallel Computing*, The MIT Press, 1994
6. Marc Snir 외 4, *MPI : The Complete Reference*, The MIT Press, 1996
7. 박효선, 박성무, 권윤한, "PC level 병렬 구조해석법 개발을 위한 PCG 알고리즘", 한국전산구조공학회 가을 학술발표회 논문집, 제11권 2집, 1998, pp.362~369
8. 박효선, 박성무, 성창원, 김재홍, "부구조기법을 이용한 PC level 분산구조해석법", 한국전산구조공학회 가을 학술발표회 논문집, 제11권 2집, 1998, pp.53~60
9. 성창원, 박효선, "분할방법에 따른 분산구조 해석법의 성능분석", 한국전산구조공학회 발표논문집, 제12권,제1집, 1999, pp.50~57
10. 성창원, 박효선, "PC level 분산구조 해석법의 효율성 예측", 대한건축학회학술발표논문집, 제19권, 제1호, 1999, pp.81~86
11. G. H. Golub and G. F. Van Loan, *Matrix Computations*, 2nd Ed., The Johns Hopkins University Press, Baltimore, Md., 1989
12. Foresti, S., Hassanzadeh, S., Murakami, H., and Sonnad, V., "A Comparison of

Preconditioned Iterative Methods using Rapid Operation Application against Direct Solution Methods", *Int. J. Numer. Methods Eng.*, Vol. 32, 1991, pp.1137~1144

13. Dickinson, J. K. and Forsyth, P. A., "Preconditioned Conjugate Gradient Methods for Three-Dimensional Linear Elasticity", *Int. J. Numer. Methods Eng.*, Vol. 37, 1994, pp.2211~2234
14. Hojjat Adeli and Hyo Seon Park, *Neurocomputing for Design Automation*, CRC Press, 1998
15. A. K. Noor, "New Computing Systems and Future High-Performance Computing Environment and Their Impact on Structural Analysis and Design", *Computers & Structures*, Vol. 64, No. 1997, 1-4, pp.1~30

(접수일자 : 1999. 4. 2)