

## 분할 방법에 따른 분산구조 해석법의 성능분석

### Effect of Substructuring Techniques on the Performance of Distributed Structural Analysis Method

성 창 원\*      박 효 선\*\*  
Sung, Chang Won    Park, Hyo Seon

#### ABSTRACT

Structural analysis of large-scale structures involving large amount of computational load and data storage requires high-performance computing resources. We have previously developed PC-level distributed structural analysis algorithms based on substructuring technique where each personal computer assigned to a slave node has been involved in the computations for single substructures. Recently, it has been proved by the authors that the performance of distributed structural analysis algorithm can be further enhanced by changing substructuring schemes. Therefore a new distributed structural analysis algorithm with one PC to multiple substructures scheme is presented in this paper. The algorithm is implemented on the network of multiple personal computers and applied to structural analysis of two dimensional frame structures.

#### 1. 서      론

최근 구조해석과 관련된 문제들의 규모가 점점 대형화되고 있으며 이러한 대형 구조물을 개인용 컴퓨터를 이용한 기존의 구조해석 알고리즘으로 해석하는 것은 효율적이라고 할 수 없다. 따라서 대형 구조물의 해석 및 설계에 관련된 효율적 기술을 개발하기 위해서는 구미 선진국과 같이 고성능 연산기의 사용을 보편화하거나 대형 구조물을 위한 효율적인 구조해석 알고리즘을 개발하는 것이 필수적이다. 그러나 국내 실정을 고려하면, 전자는 많은 비용과 이용성의 한계로 인해 비현실적이며, 개인용 컴퓨터를 이용하여 고성능 연산기 수준의 성능을 발휘할 수 있는 연산기법과 구조해석 알고리즘을 개발하는 것이 타당하다.<sup>1),2),3),4),5),6),7)</sup>

따라서 그 대안으로서, 다수의 개인용 컴퓨터가 구조해석에 필요한 연산을 부분적으로 할당받아 동시에 구조해석을 수행할 수 있는 가상의 병렬연산 시스템과 이에 적합한 대형 구조물을 위한 구조해석법을 기존의 연구<sup>8)</sup>에서 개발하였다. 기존의 연구에서는 현대의 개인용 컴퓨터에서 하나의 부구조물을 해석하도록 구성되어 있으며 이 경우 부구조물의 수는 병렬연산 시스템에 사용된 개인용 컴퓨터의 수에 의해 제약된다. 그러나 실험결과 대형 구조물의 해석시간은 일반적으로 부구조물의 수가 증가할수록 감소하다가 자유도의 수에 따라 변화하는 최적 부구조물의 수를 넘어서면 다시 증가하는 양상을 나타내기 때문에 병렬연산 시스템에 사용된 개인용 컴퓨터의 수에 의해서 부구조물의 수가 제약을 받게 되면 분산구조 해석법의 효율성은 저하될 수 있다. 따라서 본 연구에서는 병렬연산 시스템에 사용된 개인용 컴퓨터의 수에 제약을 받지 않고 부구조물을 나눌 수 있도록 알고리즘을 수정하여 하나의 개인용 컴퓨터가 여러 개의 부구조물을 해석하도록 재구성하였으며, 수정된 분산구조 해석 알고리즘을 기존의 분산구조 해석 알고리즘과 비교 분석하고자 한다.

\* 영남대학교 건축공학과 박사수료

\*\* 영남대학교 건축공학과 조교수

## 2. 수정된 PC level 분산구조 해석법

최근 상용화되어 있는 구조해석 프로그램들은 PC 한대에서 모든 구조해석 작업이 수행된다. 따라서 구조물의 자유도수가 커지고 부재수가 증가하면, 구조해석에 소요되는 시간이 비선형으로 증가하게 된다<sup>9)</sup>. 이것을 해결하기 위한 선행연구<sup>8)</sup>로서, 구조해석 프로그램을 네트워크로 연결된 다수의 개인용 컴퓨터에서 나누어 수행시킴으로서 구조해석에 소요되는 시간을 단축시키는 분산구조 해석법을 개발하였으며, 전체 구조물을 여러 개의 작은 부구조물로 나누어 해석하는 부구조 기법을 사용하였다. 앞서 개발된 분산구조 해석법은 PC 한대에서 하나의 부구조물을 해석하도록 구성되어 있기 때문에, 전체 구조물을 여러 개의 부구조물로 나눌 경우, 부구조물의 수가 PC의 수에 의해서 제약을 받게 된다. 이러한 제약은 구조물의 규모가 작을 경우에는 분산구조 해석법의 성능에 큰 영향을 미치지 않지만, 구조물의 규모가 대형화되면, 해석시간이 부구조물의 수가 증가함에 따라 계속적으로 감소하다가 최적 부구조물의 수를 넘어서면서 부터 다시 증가하는 양상을 나타낸다. 따라서 부구조물의 수가 제약될 경우, 대형 구조물의 해석시간을 단축함에 있어서 제약이 따르게 된다.

그러므로 PC 한대에서 여러 개의 부구조물을 해석할 수 있도록 알고리즘을 수정하면, 분산구조 해석법의 효율성이 증대되어 한층 더 고성능 연산기의 성능에 접근할 수 있으며, 대형 구조물을 더욱 효율적이고 합리적으로 설계할 수 있다.

### 2-1 기존의 부구조기법

본 논문에서 사용한 기존의 부구조 기법<sup>10),11),12)</sup>에 대해 STEP별로 간략하게 정리하면 다음과 같다.

STEP 1. 전체 구조물을 여러 개의 부구조물로 나눈다.

STEP 2. 나누어진 각 부구조물에 대해서, 부구조물 단위의 강성 매트릭스를 구성한다.

STEP 3. 강성 매트릭스를 이용하여, 각 부구조물의 경계부 강성을 추출해 낸다.

$$K_b^{(n)} = K_{bb}^{(n)} - K_{bi}^{(n)} (K_{ii}^{(n)})^{-1} K_{ib}^{(n)} \quad (1)$$

STEP 4. 강성 매트릭스를 이용하여 각 부구조물에 대한 해석을 수행한다.

$$[U_i^{(n)}]_{\text{경계부 구속시}} = (K_{ii}^{(n)})^{-1} P_i^{(n)} \quad (2)$$

$$R_b^{(n)} = K_{bi}^{(n)} (K_{ii}^{(n)})^{-1} P_i^{(n)} \quad (3)$$

STEP 2에서 STEP 4까지를 부구조물의 수만큼 반복한다.

STEP 5. 추출된 경계부 강성 매트릭스를 이용하여 경계부 해석을 수행한다.

$$K_b U_b = Q_b \quad (4)$$

$$Q_b = - \sum_{i=1}^n R_b^{(i)} + P_b \quad (5)$$

$$U_b = K_b^{-1} Q_b \quad (6)$$

STEP 6. 해석결과(경계부 변위)를 이용하여 부구조물의 내부변위를 수정한다.

$$U_i^{(n)} = -(K_{ii}^{(n)})^{-1} K_{ib} U_b + U_i^{(n)} \quad (7)$$

STEP 6을 부구조물의 수만큼 반복 수행한다.

STEP7. 계산된 변위를 이용하여 부재력을 계산한다.

## 2-2 수정된 PC level 분산구조 해석법

선행연구<sup>8)</sup>에서 개발한 분산구조 해석법은 문제의 규모가 큰 경우에, 그 문제를 여러 개의 작은 문제들로 나누고, 나누어진 작은 문제들을 ethernet network에 연결된 다수의 PC에 분산 할당하여 계산하도록 구성되어 있다. 따라서 본 연구에서는 작은 문제들을 다수의 PC에 분산할 때, 여러 개의 문제를 하나의 PC에 분산하도록 재구성하였다. 선행연구<sup>8)</sup> 및 본 연구에서 공통적으로 사용한 가상의 병렬연산 시스템은 그림 1과 같다.

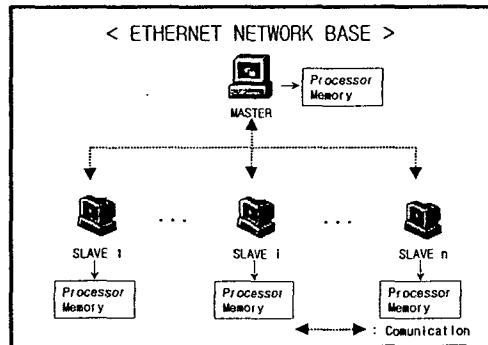


그림 1. 병렬 연산 시스템

그림 1에서, master와 slave로 사용된 실험대상 PC들은 윈도우 95를 operating system으로 사용하고 있으며, 펜티엄 166MHz의 CPU와 64Mbyte의 메인 메모리 그리고 10Mbps의 ethernet network카드로 구성되어 있다. 네트워크 상에 연결된 다수의 PC들 중에서 임의의 하나를 master로 설정하고 나머지 PC들 중에서 원하는 대수만큼을 slave로 설정하며, master와 slave의 연산에 필요한 정보 및 데이터는 WMPI(Windows Message-passing Interface)<sup>13),14)</sup>패키지를 이용하여 전송한다.

그림 1과 같이 구성된 병렬 연산 시스템을 이용하여, 선행연구<sup>8)</sup>에서 개발된 분산구조 해석법을 수정하여 수정된 분산구조 해석법을 개발하였다. 수정된 분산구조 해석법의 흐름도는 그림2와 같다.

앞 절에서 설명한 부구조 기법과 분산구조 해석법을 위한 병렬연산 시스템을 조합하면, 기존의 부구조 기법에서 부구조물의 수만큼 순차적으로 반복 해석하는 루틴을 분산구조 해석법의 각 slave에서 순차·반복적으로 계산하도록 분산 할당하여 동시에 수행함으로써 이론적으로 전체해석시간을 (1/부구조물의 수)로 줄일 수 있다.

그림 2의 흐름도를 살펴보면, master에서 전체 구조물의 해석에 필요한 절점, 부재, 경계조건 그리고 하중을 모두 입력받고, 입력받은 데이터를 부구조물의 수만큼 부구조 기법을 이용하여 나눈다. 나누어진 부구조물에 대한 데이터를 master에서 slave로 연속적으로 전송하며, 각 slave에서는 전송된 데이터를 이용하여 할당된 부구조물의 수만큼 해석을 동시에 수행한다. 해석 결과 중에서 반력과 경계부 강성 매트릭스를 다시 master로 전송하며 master에서는 전송된 반력과 경계부 강성매트릭스를 각각 조합한다. 조합된 외력항과 경계부 강성매트릭스를 이용하여 경계부 해석을 수행하고 그 결과 중에서 변위항을 다시 각각의 slave로 연속적으로 전송하게 된다.

Slave에서는 master에서 부터 전송된 경계부분의 변위를 이용하여 각각의 부구조물의 해석에서 계산된 부구조물의 내부 변위를 수정하고 부재력을 계산하여 구조해석을 종료하게 된다.

본 연구에서 수정된 분산구조 해석법은 다음의 가정 하에서 수행된다.

- 1) 각각의 부구조물 사이의 경계부분(Interface)은 하나의 층이나 최대 2개층에 걸쳐서 발생한다.
- 2) 부구조물의 수는 slave processor 수의 정수 배로 설정한다.

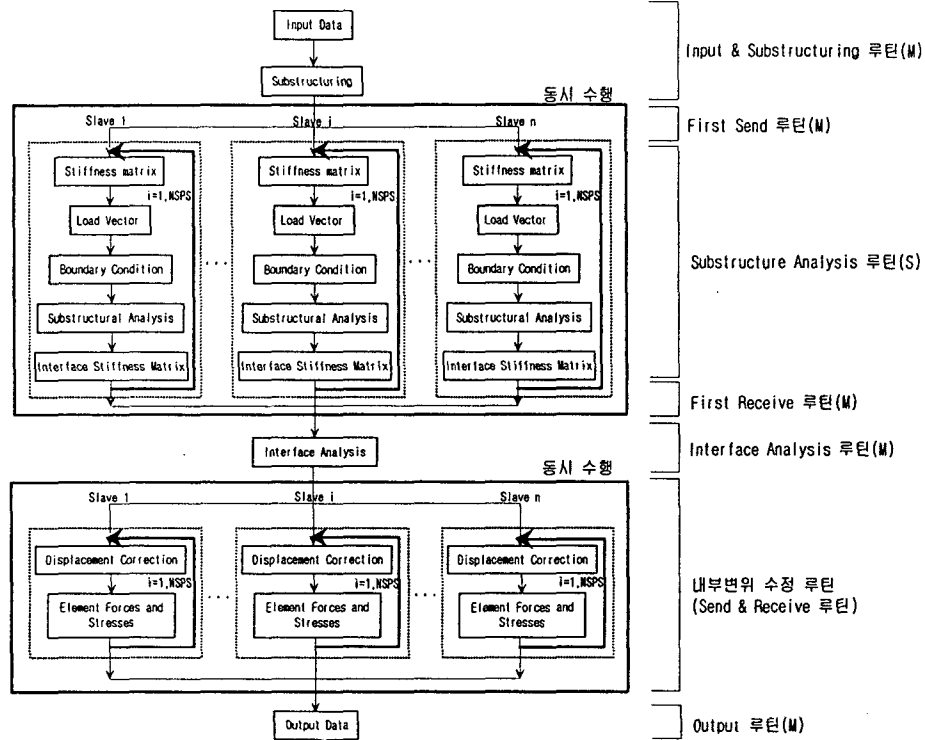


그림 2. 수정된 분산구조 해석법의 흐름도

### 3. 적용 예 제

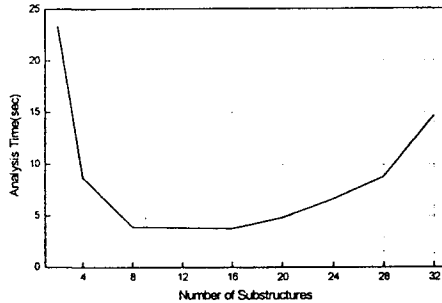
예제 적용을 위한 수정된 분산구조해석 기법은 각 부구조물 사이의 경계부분이 최대 2개층에서 발생하도록 그리고 부구조물의 수가 slave processor 수의 정수 배가 되도록 개발되었으며 적용예제는 평면골조의 3경간 54층 구조물 (전체 자유도수:660)과 7경간 90층 구조물 (전체 자유도수:2184)을 사용하며, 경간은 10m, 층고는 4m로 하였다. 이 두개의 예제를 수정된 분산구조해석법을 이용하여 해석하고 성능을 분석하였다.

#### 3-1. 직렬 알고리즘에서의 전체 해석시간

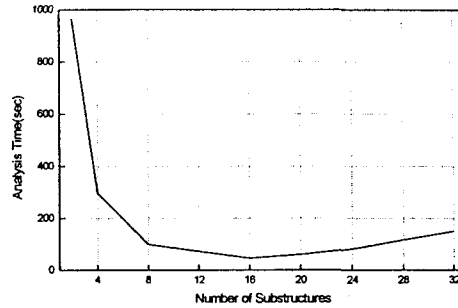
수정된 분산구조 해석법은 기존의 부구조 기법을 사용하였으므로, 기존의 부구조 기법이 가지고 있는 특성이 그대로 반영된다. 따라서 기존의 부구조 기법이 가지는 특성을 파악하기 위해서, 기존의 부구조 기법을 적용한 직렬 알고리즘을 구성하고 해석에 소요되는 시간을 평가한다. 전체 구조물을 여러 개의 부구조물로 분할할 때, 부구조물의 수에 따른 전체 해석 시간의 변화를 살펴보면 그림 3과 같다.

그림 3을 보면, 전체 자유도수가 660개인 예제에서는 부구조물의 수가 증가함에 따라 해석시간이 감소하다가 부구조물의 수가 16개 이상에서는 다시 증가하는 양상을 나타낸다. 이러한 현상은 경계부분 강성매트릭스의 크기가 커지기 때문에 발생하게 되며, 전체 자유도수가 2184개인 예제에서는 부구조물의 수가 16개 이

상일 때 해석시간의 증가양상이 많이 둔화된 것을 알 수 있다. 전체 자유도가 2184개인 예제에서, 부구조물



(a) 전체 자유도 : 660개



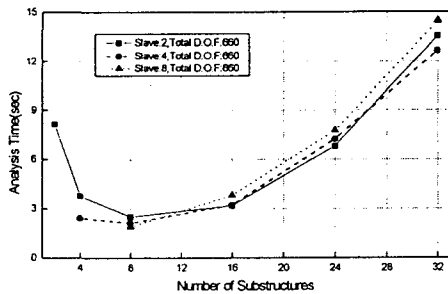
(b) 전체 자유도 : 2184개

그림 3. 직렬 알고리즘의 전체 해석시간

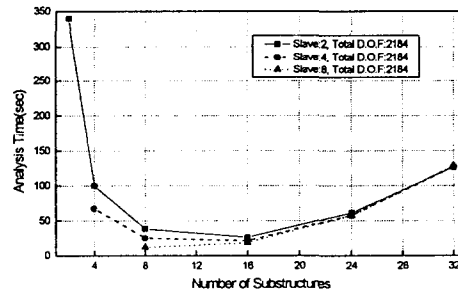
의 수가 증가함에 따라 부구조물의 해석시간은 감소하게 되고, 반대로 경계부분 강성 매트릭스의 크기는 증가하게 된다. 이러한 감소된 시간과 증가된 시간은 서로 상쇄되고 부구조물의 수가 16개 이상일 때는 전체 해석시간이 작은 비율로 증가하는 양상을 나타낸다.

### 3-2. 부구조물의 수에 따른 분산구조 해석법의 전체 해석시간

앞절에서 언급한 바와 같이, 기존 부구조 기법의 특징은 부구조물의 수에 따라 해석시간이 변화한다. 따라서 분산구조 해석법의 성능을 분석하기 위해서, 현대의 PC에서 해석하는 부구조물의 수가 변함에 따른 전체 해석시간의 변화를 나타내면 그림 4와 같다.



(a) 전체 자유도수 : 660개



(b) 전체 자유도 수 : 2184개

그림 4. 부구조물의 수가 변함에 따른 해석시간의 변화

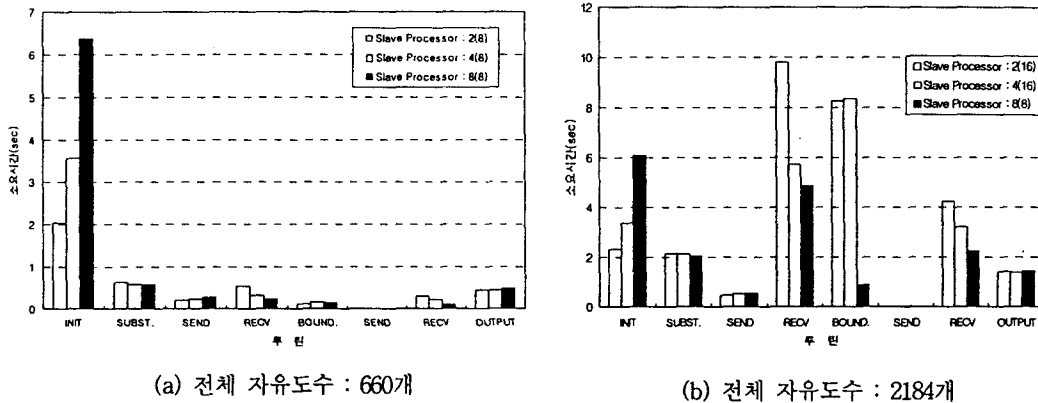
그림 4에서 해석시간의 변화 양상은 부구조물의 수가 8~16개 일 때까지 감소하다가 그 이상 증가하면 해석시간도 같이 증가하는 양상을 나타내고 있다. 이러한 증가 현상은 직렬 알고리즘에서 보다 분산구조 해석법에서 더욱 명확하게 나타날 뿐만 아니라, 부구조물의 수가 32개일 경우에는 직렬 알고리즘에서 소요된 해석시간이나, 분산구조 해석법에서 소요된 시간이 거의 비슷하게 나타나고 있다. 이러한 양상은 경계부 해석을 master processor에만 집중시킨 결과로 판단된다. 또한 직렬 알고리즘과 비교해 볼 때, 최소 해석시간이 나타나는 부구조물의 수가 다소 작은 양상을 나타낸다. 이러한 현상도 경계부 해석을 master processor에만 집중시켰기 때문에 발생한 현상으로 판단된다.

한편, 현대의 PC에서 하나의 부구조물을 해석했을 때(1 PC : 1 Substructure)와 비교해 보면, 전체 자유도 수가 660개인 예제에서는 수정된 분산구조 해석법의 해석시간이 감소한다는 것을 알 수 있으며, 전체 자유도

수가 2184개인 예제에서는 이러한 현상이 더욱 뚜렷이 나타나고 있다. 이러한 양상은 전체 자유도 수가 작은 구조물에서는 뚜렷하게 나타나지 않지만 전체 자유도가 많은 구조물에서는 뚜렷하게 나타난다.

### 3-3. 각 루틴별 소요시간

수정된 분산구조 해석법에 있어서, 해석시간이 최소로 소요되는 경우를 기준으로, 각 루틴(그림 2의 7개 루틴)별 소요시간을 분석해 보면 그림 5와 같다.



(a) 전체 자유도수 : 660개

(b) 전체 자유도수 : 2184개

그림 5. 각 루틴별 소요시간

분산구조 해석법의 초기화 부분(가상의 병렬 시스템을 구축하는데 소요되는 루틴)의 소요시간이 slave processor 수에 비례하여 증가함을 알 수 있으며, 특히 자유도 수가 660개인 구조물에서는 초기화에 소요되는 시간이 부구조물의 해석시간보다 훨씬 커지게 된다. 그러나 분산구조 해석법을 적용하는데 있어서, 초기화 부분은 가상 병렬 시스템을 구축하는데 소요되는 시간이므로, 분산구조 해석법의 전체 해석 시간에 포함시키지 않는 것이 타당할 것으로 판단된다. 그림 5(a)에서 전체 구조물을 부구조물로 분할하는 루틴에 소요되는 시간이 상당히 큰 것으로 나타나고 있으며, 이로 인해 전체 자유도 수가 작은 구조물에서는 전체 해석 시간을 증가시키는 요인이라고 볼 수 있다.

한편 그림 5(b)에서는 부구조물 해석시간과 경계부 해석시간 그리고 내부 변위 수정시간이 전체 해석시간의 80%이상을 차지하고 있는 것으로 나타난다. 이러한 시간들 중에서 경계부 해석시간만 제외하면, 다른 시간들은 모두 slave processor에서 동시에 작업이 이루어지는 시간이므로, 분산구조 해석법의 효율성을 발휘할 수 있다. 따라서 분산구조 해석법은 자유도 수가 많은 구조물에서 더욱 실용적일 것으로 판단된다.

또한 분산구조 해석법의 결과치를 출력하는 루틴은 직렬 알고리즘에서나 분산구조 해석법에서나 동일한 시간이 소요되는 루틴이며, 분산구조 해석 알고리즘의 효율성과는 관련이 없으므로, 분산구조 해석법의 효율성을 평가하는 기준은 될 수 없는 것으로 판단된다.

### 3-4. Relative Speed-up의 비교

본 절에서는 기존의 relative speed-up<sup>8)</sup>을 수정된 분산구조 해석법의 relative speed-up과 비교 분석한다. Relative speed-up(이하 RSP)은 식(8)로 정의하여 평가하고, 그래프로 나타내면 그림 6과 같다.

$$RSP = \frac{\text{직렬 알고리즘의 최소 해석시간}}{\text{분산구조 해석법의 최소 해석시간}} \quad (8)$$

그림 6의 (a)와 (b)에서 알 수 있듯이, 전체 구조물의 자유도 수가 많아질수록 수정된 분산구조 해석법의

RSP가 기존의 RSP보다 높게 나타나는 것을 알 수 있다.

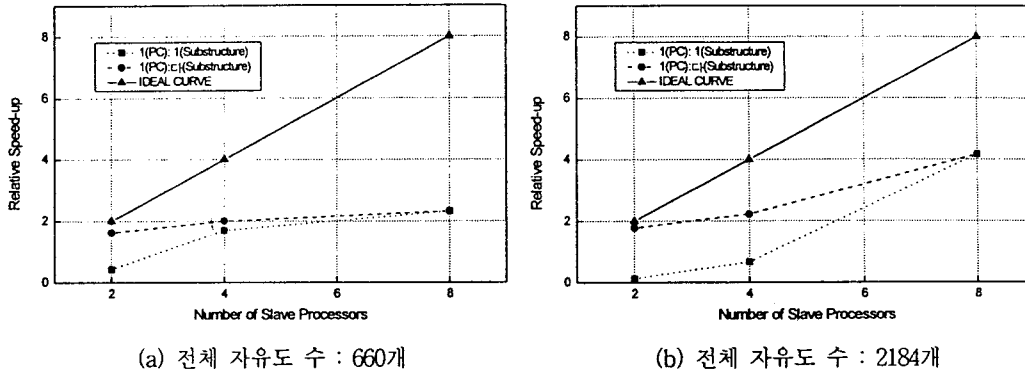


그림 6. Relative Speed-up의 비교

수정된 분산구조 해석법에서의 RSP를 살펴보면, 전체 자유도수가 660개인 구조물에서는 slave processor의 수가 8개 일 때 최대 2정도의 매우 낮은 RSP를 나타내고 있으며, 반면 전체 자유도수가 2184개인 구조물에서는 slave processor의 수가 8개 일 때 약 4정도의 다소 증가된 RSP를 나타내고 있다. 또한 하나의 PC에서 하나의 부구조물을 해석하는 경우와 여러 개의 부구조물을 해석하는 경우를 살펴보면, 전체 자유도 수가 증가할수록 RSP의 차이가 커지는 것을 알 수 있다. 따라서 전체 자유도 수가 많은 대형 구조물에서 수정된 분산구조 해석법은 더욱 효율성이 증가한다는 것을 알 수 있다.

#### 4. 결 론

대형 구조물을 효율적으로 해석하고 설계하기 위한 수정된 분산구조 해석법에 대한 결론으로서 다음을 얻을 수 있다.

- (1) 전체 구조물의 자유도수가 660개인 구조물의 경우, 본 알고리즘의 RSP는 기존의 RSP보다는 높거나 같게 나타나고 있다. 또한 전체 자유도 수가 2184개인 구조물에서는 이러한 RSP의 차이가 더 크게 된다. 이러한 RSP의 차이는 기존의 부구조 기법이 부구조물의 수가 증가할수록 해석시간이 감소되는 경향 때문으로 평가된다. 또한 두 해석법에서의 RSP 차이는 구조물의 자유도 수가 증가할수록 더욱 크게 나타난다.
- (2) 하나의 slave processor에서 해석하는 부구조물의 수에 따른 전체 해석시간의 변화는 전체 자유도수에 관계없이 직렬 알고리즘에서 나타나는 양상과 비슷한 양상을 나타낸다. 그러나 부구조물의 수가 8~16개 이상인 경우에 대해서는 후미의 증가 양상이 직렬 알고리즘보다 더 증가된 양상을 나타내고 있으며, 이러한 양상은 경계부분 해석을 master processor에만 의존하기 때문에 발생된다.
- (3) 본 연구에서 개발한 분산구조 해석법의 각 루틴별 소요시간을 비교 분석한 결과, 전체 자유도 수가 660개인 구조물에서는 초기화시간이 전체 해석시간의 70~90%를 차지하고 있으며, 이러한 초기화 시간은 분산구조 해석법의 효율성 평가 대상이 아니며, 단지 가상의 병렬 시스템을 구축하는데 소요되는 시간이므로 제외한다. 그리고 기존의 부구조 기법을 적용하는데 있어서, 전체 구조물을 여러 개의 작은 부구조물로 나누는데 소요되는 시간이 전체 해석시간의 10~20%정도 차지한다. 이러한 부구조물로 나누는 시간은 분산구조 해석법의 효율성을 저감시키는 원인이 되므로, 부구조 기법의 적용을 위한 효

올적인 부구조 기법화가 필요할 것으로 판단된다.

- (4) 본 연구에서 개발한 분산구조 해석법은 전체 자유도 수가 비교적 적은 660개의 예제에서는 slave processor가 2, 4, 8개 일 때, 1.627, 1.999, 2.328로 매우 낮은 RSP를 나타내고 있지만, 전체 자유도수가 2184개인 예제에서는 1.785, 2.228, 4.171로 증가함을 알 수 있다.

## 참 고 문 헌

1. M. Papadrakakis, Solving Large-scale Problems in Mechanics, John Wiley & Sons, 1993
2. H. Adeli and O. Kamal, Parallel Processing in Structural Engineering, Elsevier Applied Science, 1993
3. Hyo Seon Park and Hojjat Adeli, "Distributed Neural Dynamics Algorithms for Optimization of Large Steel Structures", Journal of Structural Engineering, ASCE, Vol. 123, No. 7, pp. 880-888, July, 1997
4. David E. Keyes, Ahmed Sameh and V. Venkatakrishnan, Parallel Numerical Algorithms, Kluwer Academic Publishers, 1997
5. Hyo Seon Park and Hojjat Adeli, Neuro-computing for Design Automation, CRC Press, 1998
6. Lou Baker, B. J. Smith, Parallel Programming, McGraw-Hill, 1997
7. 박효선, "대형 구조물을 위한 병렬 구조해석 및 설계", 전산구조공학회지, 제9권, 제3호, pp 47-53, 1996년 9월
8. 박효선, 박성무, 성창원, 김재홍, "부구조기법을 이용한 PC level 분산구조해석법", 한국전산구조공학회 발표논문집, 제11권, 제2집, pp. 53-60, 1998년 10월
9. 박효선, 박성무, 권윤환, "PC level 병렬구조해석법 개발을 위한 PCG 알고리즘", 한국전산구조공학회 발표논문집, 제11권, 제2집, PP.362-369, 1998년 10월
10. J. S. Przemieniecki, Matrix Structural Analysis of Substructures, Journal of the AIAA, Vol. 1, No. 1, pp. 138-147, January 1963
11. A. K. Noor, H. A. Kamel and R. E. Fulton, Substructuring Techniques - Status and Projections, Computers & Structures, Vol. 8, pp. 621-632, 1978
12. A. K. Noor and H. E. Lowder, Approximate Reanalysis Techniques with Substructuring, Journal of the Structural Division, Proceedings of the American Society of Civil Engineers, Vol. 101, No. ST8, August 1975
13. Message Passing Interface Forum, MPI-2: Extensions to the Message-Passing Interface, University of Tennessee, July, 1997
14. Message Passing Interface Forum, MPI-2 Journal of Development, University of Tennessee, July, 1997
15. William Weaver, Jr. and Paul R. Johnston, Structural Dynamics by Finite Elements, Prentice-Hall INC., 1987