한국정밀공학회지 제 22 권 제 1 호 (2005 년 1 월)
Journal of the Korean Society of Precision Engineering, Vol. 22, No. 1, January 2005.

클러스터 시스템에서 3차원 강소성 유한요소법의 병렬처리

최영#, 서용위*

Parallel Processing of 3D Rigid-Plastic FEM on a Cluster System

Young Choi# and Yongwie Seo*

ABSTRACT

On the cluster system, the parallel code of rigid-plastic FEM has been developed. The cluster system, Simforge, has 15 processors and the total memory is 4.5GBytes. In the developed parallel code, the distributed data of the column-wise partitioned stiffness are stored as the compressed row storage and the diagonal preconditioned conjugate gradient solver is applied. The analysis of block upsetting is performed with the parallel code on Simforge cluster system. In this paper, the analysis results are compared and discussed.

Key Words : Rigid-plastic FEM (강소성 유한요소법), Cluster System (클러스터 시스템), Stiffness Matrix(강성행렬), Parallel PCG Method (병렬화된 PCG 법)

1. 서론

1970년대 초에 금속 성형 공정에 대한 강소성 유한요소법(rigid-plasitic FEM)[1]이 개발된 이후, 최근 컴퓨터 산업의 발전을 바탕으로 다양한 3차원 성형문제에 대한 유한요소해석 및 설계기술이 개발되고 산업현장에서 이용되고 있다.[2]

최근, 금속성형 분야의 유한요소해석에 대한 병렬처리(parallel processing) 연구가 활발히 진행되고 있다.[3-8] 초기에는 2차원 유한요소법에 대한 병렬처리에 관한 연구가 진행되었으며[3-5], Yang[6] 등은 영역분할(domain decomposition)법을 이용하여 강소성 유한요소법을 병렬화하고 3차원 압출을 해석하였다. Im[7] 등은 영역분할법과 MBJ-PCG(Modified block Jacobi Preconditioned Conjugated)[7] 법을 이용하였으며, PC 클러스터 시스템을 구축하고 연립방정식의 해를 구하기 위해 LU 분해법[8]을 이용하여 3차원 단조공정을 해석 하였다.

본 연구에서는 클러스터 시스템(cluster system)을 이용하여 병렬처리를 수행하였다. 클러스터 시스템은 개인용 컴퓨터를 네트워크 장치를 이용하여 연결한 후 MPI[9] 혹은 PVM[10] 같은 병렬화 도구을 이용하여 분산시스템으로 구축한 병렬 컴퓨터 시스템(parallel computer system)이며 비용 대비 성능이 우수한 장점이 있다. 그러나 프로세서간 통신이 저속의 네트워크 장치(network device)를 이용하기 때문에 병목현상의 발생되는 단점이 있다. 이러한 클러스터 시스템에서의 병렬화된 유한요소법에서는 프로세서간 통신량을 최소화하는 알고리즘이 요구된다.

접수일: 2004년 3월 23일; 게재승인일: 2004년 10월 7일
교신저자: 인제대학교 기계자동차공학부
Email: ychoi@inje.ac.kr, Tel: (055) 320-3755
* 인제대학교 기계자동차공학부

클러스터 시스템을 이용한 3차원 유한요소법의 병렬화에 있어, 강성행렬의 분산저장과 선형화된 연립방정식 풀이법의 선택이 중요하다. 영역분할(domain decomposition)법은 해석할 전체영역을 부영역(sub-domain)으로 분할하고 부영역에 대한 강성행렬을 각 프로세서에 구성하여 연립방정식의 해를 구하는 것으로 영역분할 방법이 중요하다. 3차원 강소성 유한요소법에서 구성되는 강성 행렬이 희박행렬(sparse matrix)이기 때문에 Gauss소거법과 같은 직접법 보다 반복법이 연립방정식의 풀이에 보다 효과적이다. 본 연구에서는 반복법의 일종으로 행렬-벡터곱과 벡터의 내적만을 이용해서 연립방정식의 해를 구할 수 있는 PCG법을 사용하여 연립방정식의 해를 구하였다.

또한 강성행렬을 열방향으로 분할(column-wise partitioning)[11]하여 행압축 저장법(row compressed storage)[12]로 클러스터 시스템의 각 노드(프로세서)에 분산 저장하였다.

본 연구에서는 클러스터 시스템(cluster system)을 구축하고 병렬화된 강소성 유한요소(parallel rigid-plastic FEM) 프로그램을 개발하여 이를 블록 업세팅(block upsetting) 공정 해석에 적용하였다.

2. 클러스터 시스템(Cluster System)

2.1 클러스터 시스템

본 연구에서 개발한 클러스터 시스템(Simforge cluster system)은 개인용 컴퓨터 15대를 네트워크 연결을 이용하여 구축하였다. 개발한 클러스터 시스템 사양을 Table 1에 나타낸다.

Table 1 Specification of Simforge cluster system

	CPU	RAM	HDD	VGA
Master 1 node	Athlon XP 2400+*	1G	80G	16M AGP
Slave 14 nodes	Athlon XP 1800+	256M	30G	16M AGP

* Athlon XP 1800+ setting is used for comparison.

클러스터 시스템의 네트워크 대역폭을 늘리기 위해, 각 노드(node)에 100Mbps 랜카드 2장을 설치하고 16포트 스위치(16 ports switch) 2와 연결하여 채널본딩(channel bonding)으로 내부 네트워크를 구성하였다. 마스트(master) 노드만 외부 네크워크인 인터넷(internet)에 연결되어 있다. 클러스터 시스템의 OS 는 리눅스(kernel 2.4.20)를 이용하였고 i686 프로세스에 최적화하여 성능향상을 도모하였다. 클러스터의 병렬화 도구로 MPI(MPICH 1.2.5)[9]를 이용하였다.

개발된 클러스터 시스템인 Simforge를 Fig. 1에 나타낸다. 이러한 소규모 클러스터 시스템은 개인 혹은 중소규모의 연구집단에서 사용하는데 적합할 것으로 판단된다. Simforge 클러스터 시스템의 구축비용은 노드당 60만원 정도이다.

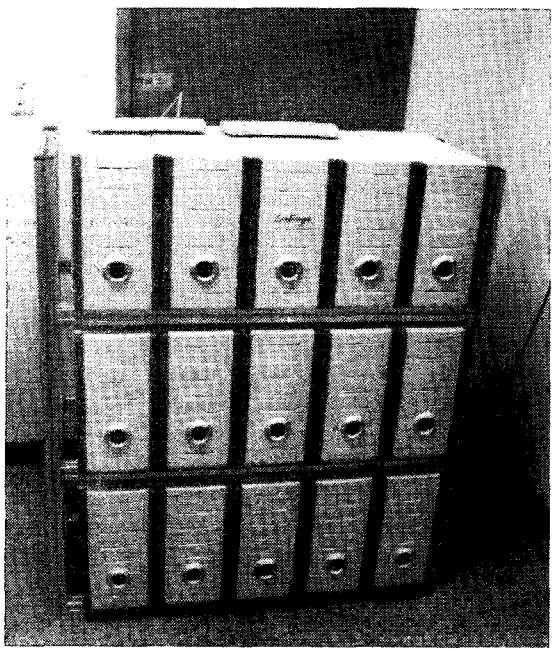


Fig. 1 Simforge cluster system

3. 강소성 유한요소법의 병렬처리

3.1 강소성 유한요소법

강소성 유한요소법에서 비압축성 조건을 포함하는 범함수의 변분은 아래와 같다[13].

$$\delta\pi = \int_V \bar{\sigma}\delta\dot{\bar{\varepsilon}}dV + K_p\int_V \dot{\varepsilon}_V\delta\dot{\varepsilon}_V dV - \int_{S_F} \bar{t}_i\delta v_i dS = 0 \quad (1)$$

여기서, $\bar{\sigma} = \sqrt{(3/2)\sigma'_{ij}\sigma'_{ij}}$, $\dot{\bar{\varepsilon}} = \sqrt{(2/3)\dot{\varepsilon}_{ij}\dot{\varepsilon}_{ij}}$ 이며, $\dot{\varepsilon}_v = \dot{\varepsilon}_{ii}$ 이다.

식(1)은 유한요소 이산화 절차를 통해 비선형 연립방정식으로 변환되고, Newton 법을 이용하여 선형화된 연립방정식를 얻는다.

$$\left[\frac{\partial^2 \pi}{\partial v_I \partial v_J}\right]_{v=v_0} \Delta v_J = -\left[\frac{\partial \pi}{\partial v_I}\right]_{v=v_0} \tag{2}$$

식(2)를 아래와 같이 표현하면,

$$\boldsymbol{K}\Delta v = \boldsymbol{f} \tag{3}$$

여기서, $\boldsymbol{K}$는 강성행렬, $\boldsymbol{f}$는 절점력 벡터이며 Δv는 속도증분이다. 각 시간스텝(time step)에서 식(3)의 해가 수렴할 때까지 반복적으로 식(3)을 구성하고 해를 구하는 과정을 반복한다.

3.2 PCG 법

본 연구에서는 식(3)의 해을 얻기 위해 최적화 기법(optimaization) 중 반복법의 일종인 PCG법을 이용하였다. 식(4)의 최소점은 식(3)의 해을 만족한다.

$$\Phi(\Delta v) = \frac{1}{2}\Delta v^t \boldsymbol{K}\Delta v - \Delta v^t \boldsymbol{f} \tag{4}$$

PCG(preconditioned CG, 예조건화 CG)법에서는 공액경사도법(CG, conjugate gradient method)으로 식(4)의 최소점을 구할 때, 수렴속도와 수렴성을 증가시키기 위해, 예조건화 기법(preconditioning)을 사용한다.

PCG법은 벡터-벡터 내적 및 행렬-벡터 곱으로 연립방정식의 해을 구하기 때문에 병렬화가 용이한 장점이 있다. PCG 알고리즘[10]을 Fig. 2에 나타낸다. $\boldsymbol{M}$은 예조건화 행렬(preconditioning matrix)이다.

PCG 알고리즘을 살펴보면, 스텝 3, 6, 9 및 10에서 예조건화 행렬($\boldsymbol{M}$)의 역행렬($\boldsymbol{M}^{-1}$)이 필요함으로 예조건화 행렬의 선택이 수렴성과 계산속도에서 중요한 사항임을 알 수 있다.

본 연구에서는 예조건화 행렬의 역행렬을 구하는 과정을 생략하도록 강성행렬의 대각성분을 예조건화 행렬(diagonal preconditioning matrix)로 사용하였다.

Step 1. Choose Δv_0 that is initial guess of Eq. (3)
Step 2. Set residual vector, $\boldsymbol{r} = \boldsymbol{f} - \boldsymbol{K}\Delta v$
Step 3. Set initial search direction, $\boldsymbol{p} = \boldsymbol{M}^{-1}\boldsymbol{r}$
Step 4. For i = 0, 1, 2, until convergence
  do step 5 ~ step 11
  Step 5. Perform matrix-vector multiple, $\boldsymbol{q} = \boldsymbol{K}\boldsymbol{p}$
  Step 6. Perform dot product, $\alpha = (\boldsymbol{r}, \boldsymbol{M}^{-1}\boldsymbol{r})/(\boldsymbol{p},\boldsymbol{q})$
  Step 7. Update, $\Delta v = \Delta v + \alpha \boldsymbol{p}$
  Step 8. Set new residual, $\boldsymbol{r}_{new} = \boldsymbol{r} - \alpha \boldsymbol{q}$
  Step 9. Perform, $\beta = (\boldsymbol{r}_{new}, \boldsymbol{M}^{-1}\boldsymbol{r}_{new})/(\boldsymbol{r}, \boldsymbol{M}^{-1}\boldsymbol{r})$
  Step 10. Update search direction, $\boldsymbol{p} = \boldsymbol{M}^{-1}\boldsymbol{r}_{new} + \beta \boldsymbol{p}$
  Step 11. Update residual, $\boldsymbol{r} = \boldsymbol{r}_{new}$

Fig. 2 PCG Algorithm[11]

3.3 강성행렬(Stiffness Matrix)

행렬-벡터 곱과 벡터 내적의 계산에서 프로세서간 통신량을 최소화하도록 강성행렬을 분산 저장하는 것이 필요하다. 강소성 유한요소법에서 강성행렬는 대칭행렬임으로 상삼각(upper triangular) 성분만이 필요하지만, 행렬-벡터 곱의 연산에서 상삼각 성분만을 이용하게 될 경우, 프로세서간 통신량이 증가하게 된다.

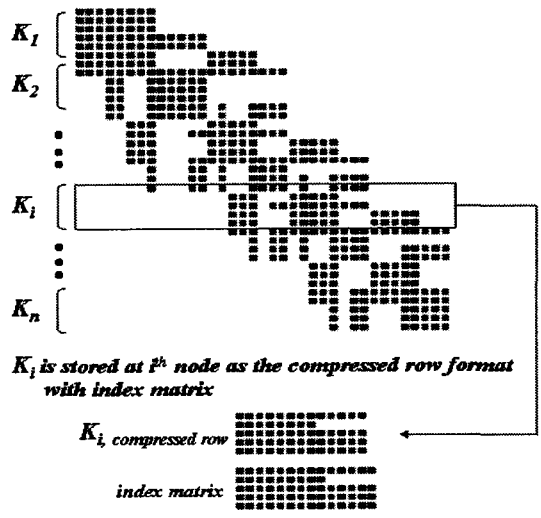


Fig. 3 Distribute storage of stiffness matrix

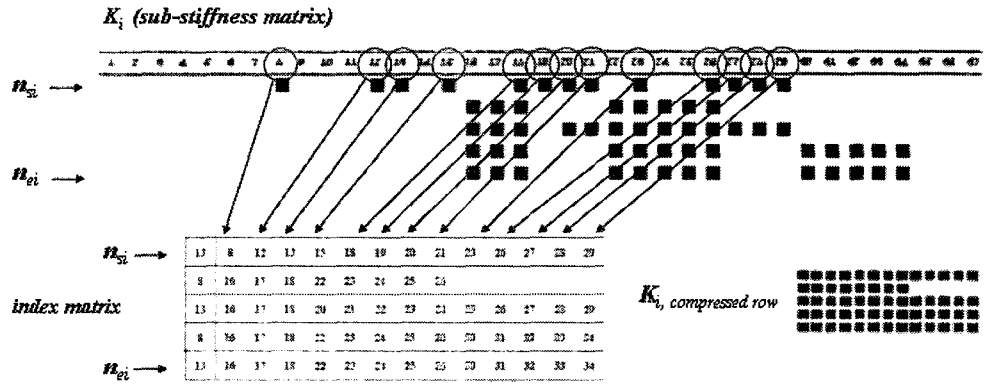


Fig. 4 Compressed row storage of sub-stiffness

본 연구에서는 전체 강성행렬을 분산 저장하여 행렬-벡터 곱의 연산에 이용한다. 또한, 행렬-벡터 곱의 연산에서 강성행렬의 행과 벡터가 곱해짐으로 강성행열을 열방향으로 나누어(column-wise partitioning)[11], 각 프로세서에 행압축법(compressed row storage)[12]으로 분산 저장하는 것이 효과적이다. 강성행렬의 분산 저장을 Fig. 3에 나타낸다. 절점수(total node number)를 병렬연산에 참여하는 프로세서 수로 나누어서 각 프로세서가 연산해야 할 절점들(n_{si} ~ n_{ei})을 결정하고 이에 따라 강성행렬을 분할 한다(column-wise partitioning). 각 프로세서에 저장되는 부강성행렬(sub-stiffness matrix, K_i)은 행압축법으로 저장된다. 각 행(row)에서 연산에 필요없는 제로(zero) 항을 제외한 요소들이 저장되고, 요소의 열(column) 위치가 위치행렬(index matrix)에 저장된다. 위치행렬에서 각 행의 첫번째 열은 대응하는 강성행열 각 행의 제로가 아닌 항의 수를 갖고 있다. 이를 Fig. 4에 나타낸다.

3.4 유한요소 데이터의 분산저장과 계산

병렬화된 강소성 유한요소법을 구현하기 위해, 열방향 분할에 따른 강성행렬의 분활을 포함하여 전체 유한요소 모델의 분산 저장이 필요하다. 이를 Table 2에 나타낸다.

열방향 분할(column-wise partitioning)로 전체 강성행렬는 각 프로세서에 부강성행렬로 분산저장된다. 즉, 각 프로세서에는 절점 시작번호(n_{si})와 절점 끝번호(n_{ei})를 결정하여 연산해야 할 절점들(n_{si} ~ n_{ei})을 분할한다.

각 프로세서에서는 전체 요소에 대한 계산을 수행하지 않고 연산해야 할 절점(n_{si} ~ n_{ei})을 포함한 요소(the elements related to the column-wise partitioned nodes)에 대해서만 연산을 수행하여 부강성행렬을 구성한다. 각 프로세서에서 부강성행렬을 구성할 때 프로세서간 통신을 사용하지 않는다. 각 프로세서에서 요소값을 갖는 변수들에 대해서도 열방향으로 분할된 절점을 포함하는 요소에 대해서만 저장되고 계산된다.

Table 2 Distributed data of each processor

Distributed data	Distribution
Stiffness Matrix	Column-wise partitioning (n_{si} ~ n_{ei})
Elements Elemental values	Related to n_{si} ~ n_{ei} nodes
Nodes Nodal values	Related to the distributed elements

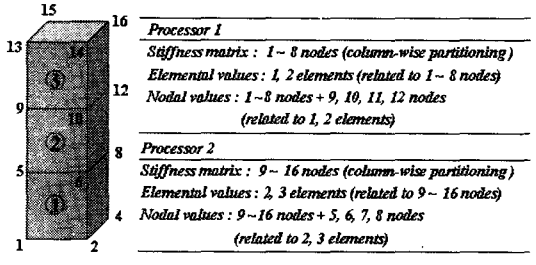


Fig. 5 FE-model with 3 brick elements

육면체 요소 3개(총 절점수는 16개, 총 요소수는 3개)인 유한요소 모델을 프로세서 2개로 분산 저장한 경우, 프로세서 1 및 프로세서 2의 분산 저장된 데이터와 유한요소 모델을 Fig. 5에 나타낸다. 프로세서 1의 부강성행렬은 요소 1과 2의 요소에 대해 부강성행렬(elemental stiffness)을 계산하여 구성되며, 프로세서 2의 부강성행렬은 요소 2와 3의 요소에 대해 부강성행렬을 계산하여 구성된다.

3.5 PCG 법의 병렬처리

PCG법의 병렬처리에 있어, Fig. 2에 나타낸 PCG 알고리즘에서 사용되는 벡터들은 열방향으로 분할된 절점들(n_{si} ~ n_{ei})에 대해 저장되고 계산된다. 즉, 스텝 2, 3, 5, 7, 8, 10 및 11이 열방향으로 분할된 절점들에 대해 계산된다. Fig. 5에 나타낸 유한요소 모델에서는 절점 1 ~ 8에 대한 값은 프로세서 1에서 계산되고 저장된다. 절점 9 ~ 16은 프로세서 2에서 계산되고 저장된다.

PCG법의 스텝 6 및 9의 벡터-벡터 내적 계산은 마스터-슬래이브(master-slave) 방식의 병렬화 기법을 이용한다. 각 프로세서에서 열방향으로 분할된 절점들(n_{si} ~ n_{ei})에 대해 부분적인 내적 값이 계산된 후 마스터 노드로 전송하고 마스트 노드에서 내적 값의 합을 계산하여 각 노드로 다시 전송한다. 벡터-벡터 내적 계산의 병렬화를 Fig. 6에 나타낸다.

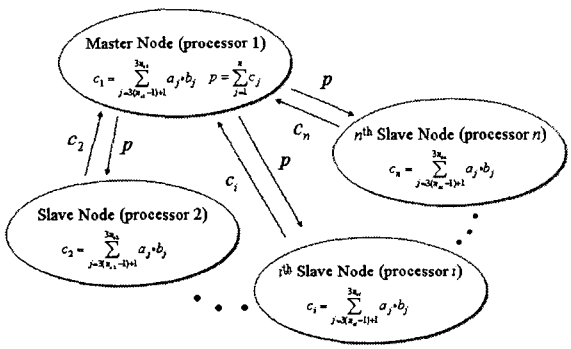


Fig. 6 Master-slave model for the calculation of the dot product, $p = \vec{a} \cdot \vec{b}$

Fig. 5의 경우, 9 ~ 16 절점들에 대한 부분적인 내적 값을 프로세서 2에서 계산한다. 이를 마스트 노드인 프로세서 1로 전송하고 프로세서 1에서 1 ~ 8 절점들에 대한 부분적인 내적 값을 계산하고 전송 받은 값과 합산하여 필요한 내적 값을 계산한 후 프로세서 2로 전송한다.

스텝 3에서 강성행렬-벡터 곱의 연산을 수행하기 위해서는 곱해지는 벡터 성분 중 열방향으로 분할된 절점(n_{si} ~ n_{ei}) 외의 값이 필요하다. 필요한 벡터 성분 값들은 다른 프로세서로부터 전송 받아 강성행렬-벡터 곱의 연산을 수행한다. Fig. 5에 보인 유한요소 모델의 경우, 프로세서 1은 9, 10, 11 및 12 절점들에 대한 벡터 성분이 필요하고 프로세서 2는 5, 6, 7 및 8 절점들에 대한 벡터 성분이 필요하다. 각 프로세서는 필요한 벡터 성분을 갖고 있는 프로세서로부터 데이터를 전송받아 강성행렬-벡터 곱의 연산을 수행한다. 이때 각 프로세서에서 수행되는 강성행렬-벡터 곱의 결과로 얻어지는 벡터는 열방향으로 분할된 절점(n_{si} ~ n_{ei})에 대한 값을 갖는 부분 벡터이다. 강성행렬-벡터 곱의 연산을 Fig. 7에 나타낸다.

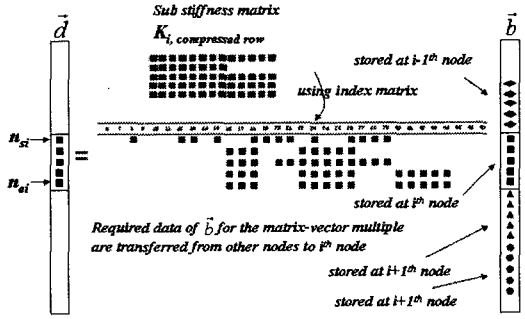


Fig. 7 Partitioned computing model for the stiffness matrix-vector multiple, $\vec{d} = K_i \cdot \vec{b}$

PCG에서 해가 수렴하여 각 프로세서에서 열방향으로 분할된 절점들에 대한 속도증분이 얻어지면 필요한 절점들에 대한 속도증분을 다른 프로세서로부터 전송받게 된다. 즉, 프로세서 1은 9, 10, 11 및 12 절점에 대한 속도증분을 프로세서 2로부터 전송 받고, 프로세서 2는 5, 6, 7 및 8 절점들에 대한 속도증분을 프로세서 1로부터 전송 받는다.

*N*개의 프로세서로 PCG법을 병렬처리 할 경우, 각 프로세서는 필요한 데이터를 다수의 다른 프로세서로 전송받게 된다. 본 연구에서의 병렬처리는

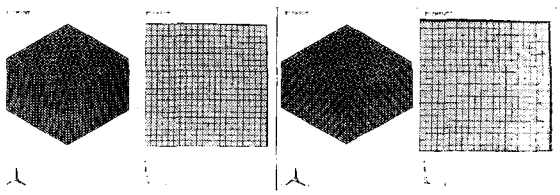
프로세서간 전송되는 데이터 크기가 작아지고 동시에 다수의 프로세서에서 데이터 전송이 이루어짐으로 클러스터 시스템과 같이 프로세서간 데이터 전송 속도가 늦은 병렬컴퓨팅 환경에 적합한 방법이라 생각된다. 이러한 분할 연산(partitioned computing model) 방법은 분할된 데이터를 마스트 프로세서로 전송하여 마스터 프로세서에서 전체 데이터를 구성한 후, 각 프로세서로 데이터를 보내는 마스터-슬레이브(master-slave) 방식의 병렬화 보다 효율적이다.

4. 블록 업세팅 병렬해석

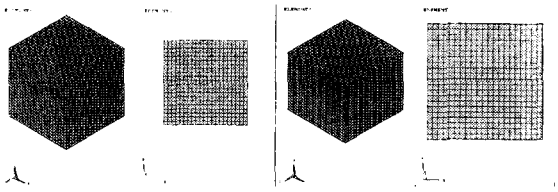
4.1 블록 업세팅(Block Upsetting)

본 연구에서 개발한 클러스터 시스템과 병렬유한요소 프로그램을 3차원 성형공정의 하나인 블록 업세팅(block upsetting) 공정에 적용하였다. 1/8모델을 해석대상으로 하였으며 원점(0,0,0)과 꼭지점(1,1,1)을 갖는 정육면체를 선형육면체요소로 모델링 하였다. 모델링 된 절점수는 10648이며, 요소수는 9261이다. 소재는 이상적인 점소성(viscoplastic) 재료로 가정하였다. 또한 모델링 된 유한요소 격자는 요소번호를 기준으로 각 프로세서에 균등한 개수로 분할된다.

$$\bar{\sigma} = 10\bar{\varepsilon}^{0.1}\dot{\bar{\varepsilon}}^{0.1}\,(MPa) \qquad (5)$$

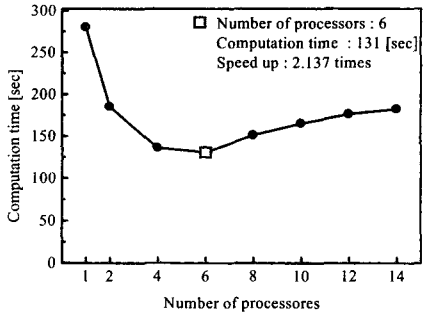


(a) Initial mesh system (b) H.R 10%

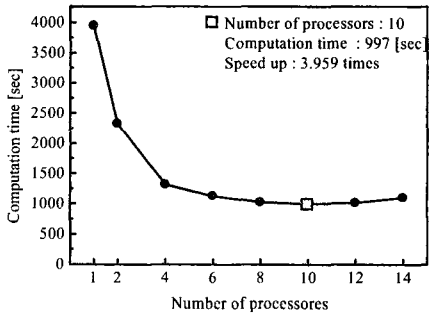


(c) H.R 20% (d) H.R 30%

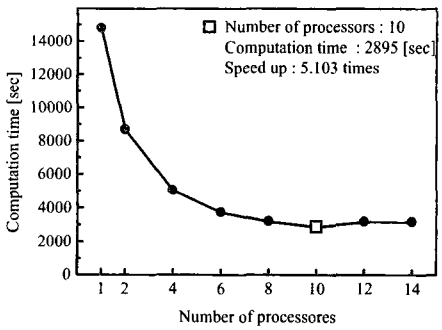
Fig. 8 Deformed mesh system



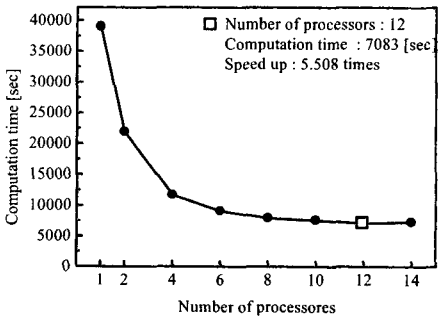
(a) 3K DOF (729 elements)



(b) 31K DOF (9261 elements)



(c) 98K DOF (29719 elements)



(d) 206K DOF (64000 elements)

Fig. 9 Parallel computation time of rigid plastic FEM

4.2 해석 결과

해석결과를 Fig. 8에 나타낸다. Fig. 8 (a)는 초기모델이며, Fig. 8 (b), (c) 및 (d)는 각각 높이 감소율 10%, 20% 및 30% 일 때의 변형된 유한요소격자이다.

4.3 병렬계산 결과비교

개발한 병렬 강소성 유한요소 코드의 실행결과를 Fig. 9에 나타낸다. 병렬계산에 이용된 프로세서 수에 대해 병렬 계산시간(parallel computation time)을 나타내었다. 해석단계(step)의 증분은 초기 높이의 1%로 하였으며 30% 높이 감소율까지 해석하였다. Table 3에 해석에 사용된 유한요소 모델들을 나타내었다.

Table 3 FE models for the analysis of block upsetting

DOF of Models	No. of Nodes	No. of Elements
3K	1000	729
31K	10648	9261
98K	32768	29719
206K	68929	64000

병렬해석결과, 해석에 이용된 프로세서 수가 증가하면 해석시간이 감소한 후 다시 증가하는 것을 알 수 있다. 이는 프로세서 수가 증가하면, 프로세서간의 전송 데이터 양이 증가하게 되어 분산하여 해석하는 병렬화 효과를 감쇠시키기 때문이다. 해석시간이 최소가 되는 프로세서 수는 해석 모델이 커질수록 증가하는 경향을 보인다.

자유도가 3K인 유한요소 모델의 경우, 프로세서 1개를 이용한 해석시간이 280[sec]이며 프로세서 6개의 경우 해석시간이 131[sec]로 약 2.1배 해석시간의 단축이 있었으며, 자유도가 206K인 경우, 프로세서 12개를 이용할 때, 계산시간이 가장 짧았으며, 약 5.5배의 해석시간의 단축이 있었다. 이는 유한요소 모델의 자유도가 큰 경우, 병렬화 효과가 우수함을 나타낸다.

병렬화 효율을 Table 4에 나타낸다. 병렬화 효율은 아래 식으로 평가한다.

$$\eta = \frac{Ctime_1 / Ctime_{np}}{np} \tag{6}$$

여기서, np는 병렬계산에 사용된 프로세서 수이며, $Ctime$은 계산시간을 나타낸다.

병렬화 효율은 프로세서를 한 개 사용한 경우를 기준으로, 프로세서 수만큼 계산시간이 빨라지는 이상적인 병렬화에 대한 실제 병렬 해석시간의 비로써 병렬계산에 사용된 각 프로세서의 효율성을 나타낸다.

병렬화 효율은 프로세서 수가 많을수록 감소하며, 유한요소 모델의 자유도가 증가할수록 병렬화 효율이 증가함을 알 수 있다.

Table 4 Efficiency of parallel computation

np	Efficiency of parallel computation [%] DOF of FE models			
	3K	31K	98K	206K
2	75.7	85.4	85.2	89.1
4	51.5	75.0	73.6	83.7
6	35.6	58.3	66.2	71.8
8	23.2	48.3	57.9	61.8
10	16.0	40.0	51.0	51.7
12	13.3	32.5	38.8	45.9
14	11.0	25.6	33.2	38.5

프로세서 수가 많은 경우, 병렬화 효율이 급격히 작아지는 것은 프로세스 수가 많아져 프로세스당 연산량이 줄어들었으나, 프로세스간 통신량의 증가와 병목현상이 발생하여 병렬화 효과를 상쇄하는 것으로 생각된다.

병렬화 효율을 향상시키기 위해서는 네트워크 관련 하드웨어(hardware)의 업그레이드와 예조건화 행렬(preonditioner)의 개선 및 병렬화 알고리즘의 개선 등이 필요한 것으로 판단된다.

5. 결론

본 연구에서 15노드의 Simforge 클러스터 시스템을 개발하고 강성행렬을 행압축방식으로 분산저장고 PCG법을 이용하여 3차원 강소성 유한요소법을 성공적으로 병렬처리하였다. 3차원 금속성형공정의 예로, 블록업세팅을 해석하여 사용한 프로세서 수에 따른 해석시간을 비교하고 병렬화 효율을 비교한 결과, 병렬처리를 통해 대규모 문제를

보다 효과적으로 해석할 수 있음을 보였다. 향후, 효율적인 예조건화 행렬(preconditioner)을 이용하고 병렬화 알고리즘의 개선한다면 보다 효율성 높은 클러스터 시스템과 강소성 유한요소법의 병렬처리가 가능할 것으로 판단된다.

후 기

이 논문은 2002년도 한국학술진흥재단의 지원에 의하여 연구되었음 (KRF-2002-003-D00026)

참고문헌

1. Lee, C. H., Kobayashi, S., "New Solutions to Rigid Plastic Deformation Problems Using a Matrix Method," Trans. ASME, Engr. For Ind., Vol. 95, pp. 865-873, 1973.
2. Kim, H. C., Kim, Y. C., Choi, Y., Kim, B. M., "A Study on the Drawing Process of Square Rod from Round Bar by Using the Rigid-Plastic Finite Element Method," J. of KSPE, Vol. 15, No. 11, pp. 145-151, 1998.
3. Kato, T., Ishikawa, T., Yukawa, N., Niwa, K., "Application of parallel computer to rigid plastic fem analysis for saving computating time," Proceedings of the Numiform'89, Balkema, Rotterdam, 1989.
4. Kopp, R., Thomas, R., Bünten, R., "Optimizing of multi-stage metal forming processes using parallel computers," Simulation of Materials Processing: Theory, Methods and Applications, She & Dawson (eds), Balkema, Rotterdam, pp. 445-449, 1995.
5. Bubak, M., Chrobak, R., Kitwoski, J., Moscinski, J., Pietrzyk, M., "Pallel finite element calculation of plastic deformations on Exemplar SPP1000 and on networked workstations," J. Mater. Process. Technol., Vol. 60, pp. 409-413, 1996.
6. Park, K., Yang, D. Y., "Domain decomposition using substructuring method and parallel computation of the rigid-plastic finite element analysis," J. of KSTP, Vol. 7, No. 5, pp. 474-480, 1988.
7. Kim, S. Y., Im, Y. T., "Parallel processing of 3D rigid-viscoplastic finite element analysis using domain decomposition and modified block Jacobi preconditioning technique," J. Mater. Process. Technol., Vol. 134, pp. 254-264, 2003.
8. Cheon, J. S., Kim, S. Y., Im, Y. T., "Three-dimemsional bulk forming simulations under a PC cluster environment," J. Mater. Process. Technol., Vol. 140, pp. 36-42, 2003.
9. Gropp, W., Lusk, E., "Installation and User's Guide to MPICH, a Portable Implementation of MPI Version 1.2.5," ANL/MSC-TM-ANL-01, 2003.
10. Geist, A., Beguelin, A., Dongarra, J., Jiang, W., Mancheck, R., Sunderam, V., "PVM 3 Guide and Reference Manual," 1994.
11. Dincer, K., Hawick, K. A., Choudhary, A., Fox, G. C., "Implementation of Conjugate Gradient Algorithms in Fortran 90 and HPF and Possible extensions towards HPF-2," NPAC Technical Reports SCCS 639, 1995.
12. Saad, Y., "Iterative Methods for Sparse Linear Systems," Boston, PWS Publishing Co., 1996.
13. Kobayashi, S., Oh, S. I., Altan, T., Metal forming and the finite element method. New York: Oxford University Press, pp. 90-130, 1989.