132 대한기계학회논문집 A권 제22권 제1호, pp. 132~141, 1998

MPI를 이용한 판재성형해석 프로그램의 병렬화

김 의 중* · 서 영 성**

(1997년 3월 12일 접수)

Parallelization of Sheet Forming Analysis Program Using MPI

Eui Joong Kim and Yeong Sung Suh

Key Words: Sheet Forming(판재성형), Contact Algorithm(접촉 알고리듬), Massively Parallel Processors(MPP: 다중병렬 프로세서), Message Passing Interface(MPI: 메시지 패싱 인터페이스), Load Balancing(부하조정)

Abstract

A parallel version of sheet forming analysis program was developed. This version is compatible with any parallel computers which support MPI that is one of the most recent and popular message passing libraries. For this purpose, SERI-SFA, a vector version which runs on Cray Y-MP C90, a sequential vector computer, was used as a source code. For the sake of the effectiveness of the work, the parallelization was focused on the selected part after checking the rank of CPU consumed from the exemplary calculation on Cray Y-MP C90. The subroutines associated with contact algorithm was selected as targe parts. For this work, MPI was used as a message passing library. For the performance verification, an oil pan and an S-rail forming simulation were carried out. The performance check was carried out by the kernel and total CPU time along with theoretical performance using Amdahl's Law. The results showed some performance improvement within the limit of the selective paralellization.

1. 서 론

컴퓨터를 이용한 대표적인 공학 수치해석방법 중의 하나인 유한요소해석 기술은 구조해석을 포함한 여러 공학의 제반문제에 폭 넓게 활용되는 방법으로, 컴퓨팅 기술이 획기적으로 발전함에 따라 단순한 선형문제로부터 점차 고도의 복잡한 비선형 문제로 그 적용영역이 확대되고 있으며, 실제로 산업계의 설계업무에서 매우 중요한 역할을 담당하고 있다. 특히 자동차 산업에서는 충돌해석, 열유동해석 및 금속성형해석 등의 분야에 이러한 수치해석을 이용한 시뮬레이션기법을 적극 활용하며 실제로 제품의 설계시간 단축 및 원가절감 등의 효과를 얻고 있다. 충돌 및 판재성형해석 등과 같은 영역에서는 문제의 크기가 상대적으로 크고 특히 접촉해석을 위한 반복계산이 많은 계산시간을 요구하므로, 일반적으로 슈퍼컴퓨터와 같은 고성능 컴퓨터 상에서 해석하고 있다. 그러나, 점차 해석대상 모델의 크기가 대형화됨에 따라, 해석 소요시간 및 그에 따른 비용이 비약적으로 증가하여, 이들을 줄일 수 있는 효율적인 계산방법의 필요성이 대두되고 있다. 현재 많이 활용되고 있는 과학계산용 슈퍼컴퓨터들은 대부분 벡터 프로세서를 장착한 벡터 컴퓨터인데, 이러한 벡터 컴퓨터로 대표되는 순차처리 컴퓨터(sequential computer)들은 과거 몇십

*시스템공학연구소, 현재 Carnegie-Mellon Univ.
**회원, 한남대학교 기계공학과

년동안 눈부신 발전을 거듭해 왔던 것이 사실이나, 현재에는 근본적으로 내재하고 있는 하드웨어적인 한계성(회로(circuit)속의 전자의 전달속도의 한계, 전자의 이동 중 트랜지스터의 교체, 그리고 열손실에 따른 성능저하 등)으로 인해 그 발전속도가 점차로 둔화되고 있다.[1] 최근 이러한 순차 컴퓨팅의 한계성을 극복하고자 병렬 컴퓨팅기법이 도입되었으며, 이에 대한 연구가 세계적으로 매우 활발히 진행되고 있다. 이러한 결과로 최근에는 벡터 슈퍼컴퓨터와 비교하여 최고 성능(peak performance) 면에서 그 성능이 월등한 병렬 컴퓨터가 대거 등장하였다. 선진국에서는 이미 병렬 컴퓨팅분야에서의 제반 연구가 활발하게 진행되고 있으며 실제로 여러 응용분야에서 개발사례들이 보고되고 있으나 국내의 경우, 특히 응용 소프트웨어를 활용하는 병렬 컴퓨팅에 관한 연구는 선진국에 비해 아직도 미약한 실정이다.

본 연구에서는 판재성형해석 전용 프로그램인 SERI-SFA(SERI, Sheet Forming Analyzer)[2]를 병렬화하여 다중병렬 프로세서(Massively Parallel Processors 또는 축약하여 MPP)를 장착한 IBM SP2상에서 사용할 수 있도록 하였다. SERI-SFA는 비선형 외연적-동적 유한요소해석 코드인 DYNA3D 알고리듬을 근간으로 하는 벡터 컴퓨터 기반의 판재성형 전용해석 프로그램이다. 병렬 컴퓨터 상에서 프로그램의 병렬처리를 하기 위해서는 메시지 패싱 라이브러리(message passing library)가 필요한데, 본 연구에서는 현재 메시지 패싱 라이브러리의 표준으로 자리잡아 가고 있는 MPI(Message Passing Interface, 메시지 패싱 인터페이스)[3]를 이용하였다. 병렬화를 하는 과정에서는, 먼저 프로그램을 분석하여 가장 시간이 많이 걸리는 병목(bottle-neck)부분을 검색하고, 이 부분을 선택적으로 병렬화하는 방법으로 추진하였다. 병렬처리 영역 결정을 위해 우선 Cray 슈퍼컴퓨터의 운영체계인 UNICOS에서 지원하는 성능분석 도구인 플로우뷰(flowview)를 이용하여 성능분석을 수행하였다. 병렬화를 완료한 후 디버깅과 성능검증을 수행하였다.

2. MPI 개요

메시지 패싱은 분산 메모리형 병렬 컴퓨터에서 사용되는 하나의 패러디임으로, 현재 여러 형태의 메시지 패싱기법이 구현되고 있으나 그 기본 개념은 병렬 컴퓨터의 프로세서들 간의 커뮤니케이션이라 할 수 있다. 지난 수 년 동안 이 메시지 패싱이라는 패러디임을 통해서 많은 응용 프로그램들이 개발되어 왔으며, 각각의 하드웨어 개발업체들은 자신의 병렬 컴퓨터에 적합한 메시지 패싱기법들을 개발해 왔다. 다양한 종류의 병렬 컴퓨터에서 효율적으로 이식이 가능하며 좋은 성능의 응용 프로그램을 개발할 수 있는 라이브러리 커널의 개발이 요구됨에 따라 MPI가 탄생되었다. MPI는 분산 메모리 병렬 컴퓨터의 각 프로세서간의 메시지 패싱을 통한 커뮤니케이션을 위해 고안된 최초의 표준 응용 프로그램 인터페이스(Application Programming Interface, 또는 축약하여 API)이다. MPI의 목적은 실용적이고, 유연하며, 이식성이 좋은 메시지 패싱 표준을 사용한 API를 제공함으로써 다양한 분산 메모리형 병렬 컴퓨터에서 모두 동작하는 병렬 프로그램을 쉽게 만들 수 있도록 하는 데에 있다. MPI는 FORTRAN 90이나 HPF(High Performance FORTRAN)와 같은 컴퓨터 언어가 아니므로 표준 컴파일러를 사용하여 여러 하드웨어로의 손쉬운 이식이 가능한 병렬 프로그램을 작성할 수 있도록 해준다. 또한, 프로세서의 메모리에 직접 데이터를 쓰는 회수를 최소화하는 효율적인 커뮤니케이션을 제공함으로써 커뮤니케이션에 의한 부하를 줄여주는 장점이 있다. MPI는 ANSI나 ISO와 같은 표준화 기구에서 고안된 것은 아니며, 많은 하드웨어 및 소프트웨어 개발자, 학계, 연구소들로 구성된 MPI 포럼의 결론을 토대로 탄생되었다. "MPI standard"는 1994년 5월 MPI 포럼에서 시작되어 1995년 6월에 최종 1.1 버전이 발표되었으며, 미국 알곤(Argonne) 국립연구소의 웹 페이지의 "MPI: A Message-Passing Standard"를 통해 온라인 문서로 배포되고 있다. 1995년 4월에 시작된 MPI-2 포럼에서는 1997년 6월까지 현재의 MPI 1.1 버전에 포함되어 있지 않은 동적 프로세스 관리(dynamic process management), 일방향 커뮤니케이션(one-sided communication), 병렬파일 입출력을 추가하고 1.1 버전의 집단작업(collective operation)을 확장하는 한편, FORTRAN 90 및 C++을 사용할 수 있도록 MPI를 확장하고 있다.[3]

병렬 프로그램을 MPI로 작성함으로써 소프트웨어 개발자들은 여러 가지 혜택을 얻게 된다. 먼저 MPI는 여러 병렬 컴퓨터 개발회사들이 참여하여

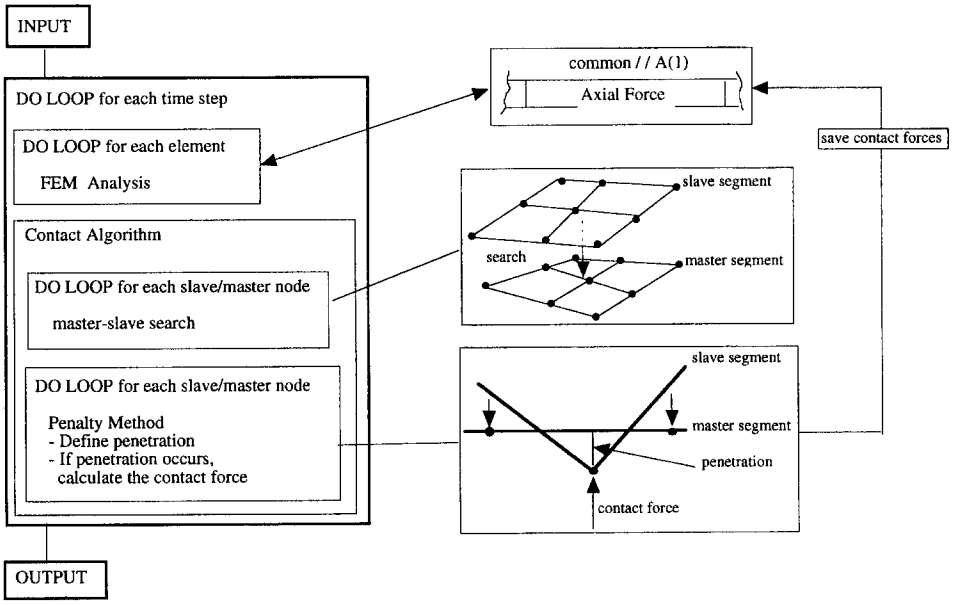


Fig. 1 Basic contact algorithm of SERI-SFA.

결정한 표준이므로, 하드웨어의 종류와 무관한 API를 가지고 있다. 따라서 하나의 병렬 컴퓨터에서 MPI 표준 API를 사용하여 작성된 병렬 프로그램들은 다른 모든 병렬 컴퓨터에서 각 하드웨어의 특성에 최적화된 상태에서 실행될 수 있다는 것을 의미하며, 이것은 하드웨어마다 서로 다른 API를 가지고 있는 PVM(Parallel Virtual Machine)과는 다른 점이라고 할 수 있다.

현재 MPI는 IBM RS/6000, SP, Intel i860, Delta, Paragon, Thinking Machines CM5, Cray T3D/E, XMP, YMP, C90, nCUBE, SunOS 4.x, Solaris, HP UX, SGI, DEC alpha, Fujitsu AP1000 등의 다양한 병렬 컴퓨터에서 구현되어 있다. 또한 MPI의 성능은 기존의 PVM, Express, P4 등보다 월등하며, 비동기 커뮤니케이션(asynchronous communication), 효율적인 메시지 버퍼관리(message buffer management), 그룹(group), 다양한 커뮤니케이션 모드를 제공하므로 보다 효율적인 병렬 프로그램을 작성할 수 있도록 해주게 된다.[4~5] 현재 여러 연구소 및 대학교에서 제공하는 MPI들이 공개되어 무료로 이용이 가능하다.[5]

3. SERI-SFA의 병렬화

3.1 SERI-SFA의 개요

병렬화 목적 프로그램으로 채택된 SERI-SFA는 벡터 슈퍼컴퓨터 Cray C90에서 실행되는 비선형 외연적 적분법의 동적 유한요소해석 프로그램인데, 기본적으로 Belytschko-Tsay 셸요소를 채택한 DYNA3D 알고리듬을 사용하고 있으며 접촉알고리듬으로는 주종탐색법 및 벌칙함수법을 사용하고 있다.[6] 주종탐색법은 Hallquist[6]가 제안한 접촉 탐색 알고리듬으로서, 판재성형해석의 경우 주종탐색법으로 접촉처리를 하는 부분의 계산 소요시간은 전체의 약 60~70%를 차지하고 있다. SERI-SFA의 기본적인 접촉 알고리듬 구조는 Fig. 1과 같다. 이 부분은 본 연구에서 병렬화를 수행한 영역인데, 접촉여부 판정 및 접촉력 계산 등을 수행한다.

3.2 SERI-SFA의 병렬화

3.2.1 병렬화 영역 결정

먼저, Cray 슈퍼컴퓨터의 운영체계인 UNICOS에서 지원되는, 성능분석 도구 플로우뷰를 통하여 SERI-SFA의 병렬화 영역을 결정하였다. 플로우뷰는 여러 가지 성능분석에 관련된 정보를 제공하는데, 본 연구에서는 병렬화 영역을 선택하기 위해

Table 1 CPU rating by subroutine from an S-rail forming simulation.
Flowtrace Statistics Report
Showing Routines Sorted by CPU Time(Descending)
(CPU Times are Shown in Seconds)

Routine name	Tot. time	# Calls	Avg. Time	Percentage	Accumg.%	
SLAVE1	1.14 E+03	28320	4.01 E−02	36.92	36.92	*********
CROSS	7.57 E+02	1.45 E+08	5.23 E−06	24.60	61.51	******
SLAVE2	4.24 E+02	28308	1.50 E−02	13.79	75.30	***
PTIME	2.11 E+02	88139914	2.39 E−06	6.84	86.03	*
STIFFS	1.20 E+02	6	2.00 E+01	3.89	86.03	
TRNFBS	7.14 E+01	226464	3.15 E−04	2.32	88.35	
UNPK	6.00 E+01	31310733	1.92 E−06	1.95	90.30	
SHL24S	5.31 E+01	679392	7.82 E−05	1.73	92.03	
STEX	3.83 E+01	4118158	9.30 E−06	1.24	93.27	
FEM3D	3.34 E+01	1	3.34 E+01	1.09	94.36	
			⋮			
			⋮			
PENSTF	1.78 E−06	1	1.78 E−06	0.00	100.00	
RDETPT	1.77 E−06	1	1.77 E−06	0.00	100.00	
LOADP	1.76 E−06	1	1.76 E−06	0.00	100.00	
MOMDEP	1.76 E−06	1	1.76 E−06	0.00	100.00	
RBCC	1.76 E−06	1	1.76 E−06	0.00	100.00	
JOYINF	1.75 E−06	1	1.75 E−06	0.00	100.00	
RGIDWI	1.75 E−06	1	1.75 E−06	0.00	100.00	
Totals	3.08 E+03	2.77 E+08				

서, 그중 각 부 프로그램마다의 CPU 사용현황에 관한 정보를 이용하였다. Table 1은 플로우뷰를 이용하여 NUMISHEET '96[7]에서 벤치마크 문제로 제시되었던 S자형 레일(S-rail) 성형 시뮬레이션 시 CPU의 소요상황에 대한 성능분석이다. Table 1에서 보는 것과 같이 가장 CPU의 소요가 많은 곳은, 접촉 알고리듬 관련 부 프로그램들의 일부인 slave1.f, cross.f와 slave2.f임을 알 수 있다. 이 때 cross.f는 slave1.f와 slave2.f에 종속된 부 프로그램이어서 slave1.f나 slave2.f만 병렬화를 하여도 자동적으로(자료의 종속성이 없으면) 병렬화가 되므로 본 연구에서는 이를 바탕으로 slave1.f와 slave2.f를 병렬화 영역으로 결정하였다. 일반적인 판재성형 시뮬레이션에서도 이와 유사한 플로우뷰 결과를 보여줄 것으로 예측되며, 후에 오일 팬(oil pan) 성형 시뮬레이션[8]의 예를 통하여 그 사실을 확인하였다.

3.2.2 병렬화 작업

성능분석을 통하여 병렬화 영역을 slave1.f와 slave2.f로 선택하고, 이 부분에 대한 병렬화 방법을 결정하였다. 부 프로그램 slave1.f와 slave2.f는 모두 종속절점(nsn)의 개수만큼 반복계산되는 DO 루프(loop)를 가지고 있는데, 바로 이 DO 루프에서의 계산시간이 slave1.f와 slave2.f의 계산시간과 거의 일치한다. 본 연구에서는 이 주(main) DO 루프를 가용 프로세서의 개수만큼 분할하여 병렬처

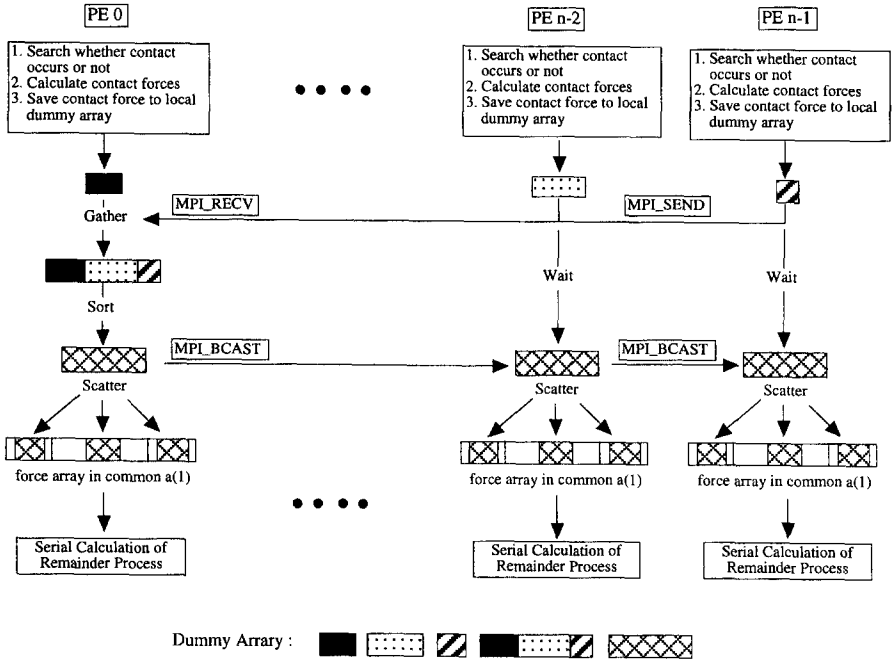


Fig. 2 Parallel processing concept for the contact algorithm.

리를 수행하게 하였다. 이와 같이 병렬화 방법을 결정한 후, 이 부분에 대한 자료의 종속성(data dependency) 분석을 수행하였다. 여기서 자료의 종속성이란 병렬처리를 수행했을 때 계산과정에서 시간적인 순서나 기억장소의 사용에 관련하여 병렬처리의 결과가 순차처리의 결과와 다를 때 이를 자료의 종속성이 있다고 말한다. slave1.f는 접촉탐색을 수행하는 부 프로그램으로 종속절점(slave node)에서 가장 가까운 주 절점(master node)을 찾고, 이 주 절점을 공유하는 세그먼트(segment)중 종속절점에서 가장 가까운 주 세그먼트(master segment)를 찾는 역할을 하며, 자료의 종속성은 없는 것으로 나타났다. 부 프로그램 slave2.f는 slave1.f에서 결정된 종속 절점에서 가장 가까운 주 세그먼트(master segment)와 종속 절점의 거리를 계산하고 관통여부를 판정하여 벌칙함수방법에 의한 접촉력을 계산한다. 여기서, 계산된 접촉력은 common//a(1)의 force라는 배열(array)에 저장된다. 이때 자료의 종속성이 발생되는데, 본 연구에서는 이를 피하기 위해 각 프로세서에서 계산된 접촉력을 일시적으로 허배열(dummy array)에 저장한 후 이것을 주 프로세서(master processor, PE0)로 보내어 취합한 후 이것을 다시 각 프로세서로 보내는 방법을 이용하였다. 즉 각 프로세서에서 국부적으로 계산된 접촉력을 주 프로세서에서 모아 이 값들을 공유하는 방법이다. 이때 각 프로세서에서 개별적으로 계산된 접촉력을 주 프로세서로 보내거나, 또는 각 프로세서에서 보낸 접촉력 정보를 주 프로세서에서 취합한 후 이것을 다시 각 프로세서로 보내기 위해서는 메시지 패싱을 통한 자료의 전송이 필요한데 이 과정에서 MPI 라이브러리를 이용하였다. Fig. 2는 본 연구에서수행한 SERI-SFA의 병렬화 작업과정의 개념도 이고, 이때 사용된 MPI 라이브러리는 다음과 같다.

MPIINIT : 모든 MPI 라이브러리보다 앞서 불려지며 MPI 환경을 초기화한다.

MPICOMMRANK : 이 라이브러리가 사용된 프로세서의 정보를 제공한다.

MPICOMMSIZE : 현재 사용하고 있는 프로세서 총 개수를 알려준다.

MPISEND : 메시지를 한 개의 프로세서에서 다른 한 개의 프로세서로 보낸다

MPIRECV : 메시지를 한 개의 프로세서가 다른 한 개의 프로세서로부터 받는다. 모든 통신이 완료되면 MPI 환경을 정리한다.

3.2.3 부하조정

부하조정(load balancing)이란 병렬처리시 각 프로세서마다 작업분량을 균일하게 할당하여 작업 대기시간을 최소화하는 것을 말하는데, 이것은 병렬화 작업에서 매우 중요한 부분 중의 하나이다. 이는 아무리 병렬화 작업을 잘 수행하였다 하더라도 각 작업(task) 간에 부하조정이 이루어지지 않으면 병렬처리의 효율이 저하되고 만족할 만한 성능향상을 얻을 수 없기 때문이다. 본 연구에서는 slave1.f와 slave2.f에서, 종속절점의 개수(nsn)만큼 반복계산되는 DO 루프계산을, 각 프로세서마다 현재 작업중인 자신의 프로세서 번호(ip)로부터 현재 사용중인 프로세서의 개수(np)만큼 증가시켜 계산함으로써 부하조정을 유도하였다. 즉, DO 루프의 정적 인접분할법(contiguous static partitioning, Fig. 3(a))에 의한 병렬처리기법은 접촉지점을 나타내는 종속절점이 특정한 프로세서에 편중이 되면서 부하가 불균형하게 걸리는 문제를 야기할 수가 있으므로 본 연구에서는 이를 피하기 위하여 정적 간격분할법(interleaved static partitioning, Fig. 3(b))을 이용하였다. 병렬화된 SERI-SFA는 PARA-SFA (PARAllel Sheet Forming Analyzer)로 명명하였다.

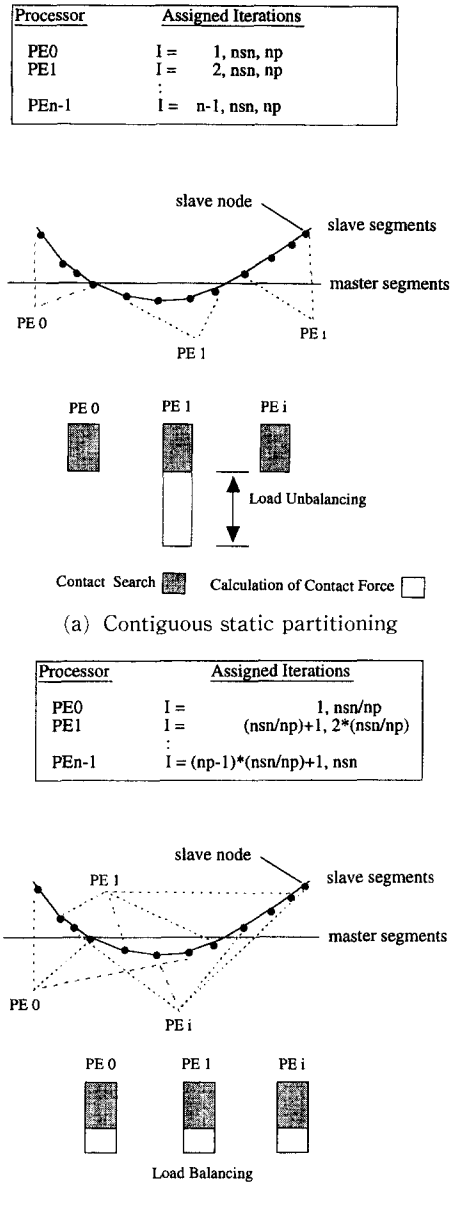


Fig. 3 Do loop. partitioning of subroutines slavel.f and slave2. f.

4. 결과 및 고찰

위와 같은 병렬화과정을 통하여 완성된 병렬처리 프로그램 PARA-SFA에 대하여 오일 팬과 S자형 레일의 성형 시뮬레이션을 통한 성능평가를 수행하였다. 이에 앞서 PARA-SFA의 성공적인 병렬화 여부를 확인하기 위하여 PARA-SFA의 계산결과와 순차 프로그램 SERI-SFA의 결과를 비교하였다.

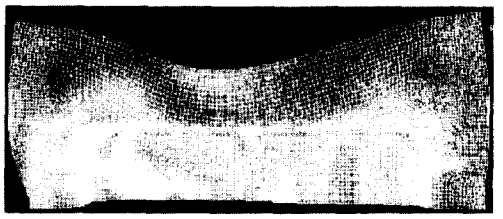
4.1 예제 개요

4.1.1 오일 팬 성형 시뮬레이션

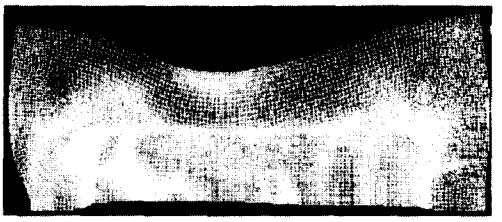
오일 팬 성형해석 예제[8]에 사용된 절점 수는 10,273개이고, 요소 수는 셸요소 9,881개이다. 판재의 질량밀도는, 7,800 Kg/m^{3}으로 가정하였고, 탄성계수는 210 GPa, 푸아송비는 0.3으로 가정하였다. 성형해석조건으로는 블랭크 홀더력이 697,480 N, 쿨롬마찰계수가 0.12, 펀치성형깊이를 143 mm로 가정하였다. 펀치속도는 계산시간의 절약을 위해 가상적으로 100 m/s로 증가하여 계산하였다. 이는 물리적으로는 타당한 수치가 아니지만, 계산결과의 단순한 수치적 비교로서는 이상이 없는 것으로 보았기 때문이다.

4.1.2 S자형 레일 성형 시뮬레이션

본 예제는 NUMISHEET '96[7] 벤치마킹문제 중



(a) Results from SERI-SFA, a sequential program



(b) Results from PARA-SFA, a parallel program

All the results were the same no matter how many - 2, 4, 8, and 16 - processors were used.

Fig. 4 Plastic strain distribution after forming of the oil pan Compare the degree of brightness at local position.

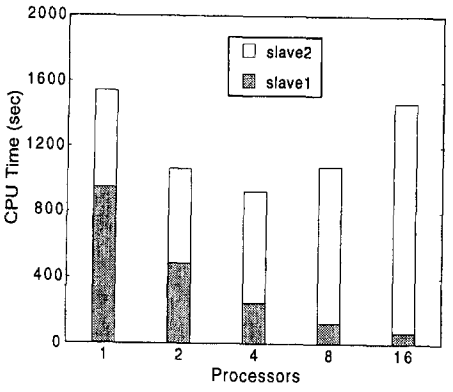
의 하나(SB-2)로서, 자동차의 바디구조물의 하나인 프레임 멤버와 유사한 형상이다. 사용된 절점수는 6,020개이며, 요소 수는 5,730개의 셸요소가 사용되었다. 판재의 질량밀도는 역시 7,800 Kg/m^{3}으로 가정하였고, 탄성계수는 210 GPa, 푸이송비 0.3으로 가정하였다. 성형해석조건으로는 블랭크홀더력이 10,000 N, 쿨롬마찰계수가 0.11, 펀치성형깊이를 37 mm, 펀치속도는 10 m/sec로 가정하였다.

4.2 성능 검증

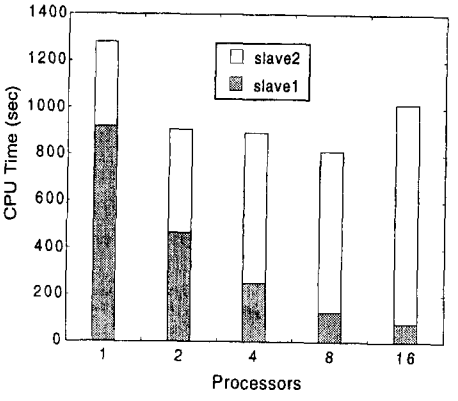
먼저, 프로그램의 정확도를 확인하기 위하여 순차 프로그램으로의 해석 결과와 병렬 프로그램으로의 결과를 비교하는 성능검증을 실시하였다. Fig. 4에 오일 팬 성형 시뮬레이션결과를 비교하였다. 그림으로 정확히 파악하기는 어려워, 몇몇 지점에서의 변형이나 응력의 수치를 직접 비교하여 해석결과가 거의 일치함을 확인하였다. S자형 레일의 성형 시뮬레이션에서도 동일한 결과를 얻었다.

4.3 성능 비교

Fig. 5에서 보는 것과 같이 두 가지 시뮬레이션의 경우 모두 자료의 교환에 따른 통신이 거의 발



(a) Oil pan forming simulation

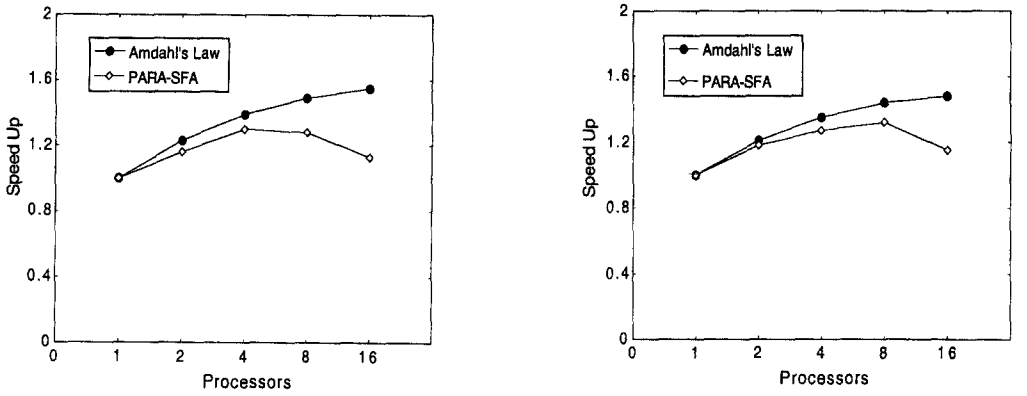


(b) S-rail forming simulation

Fig. 5 CPU performance by subroutine with PARA-SFA.

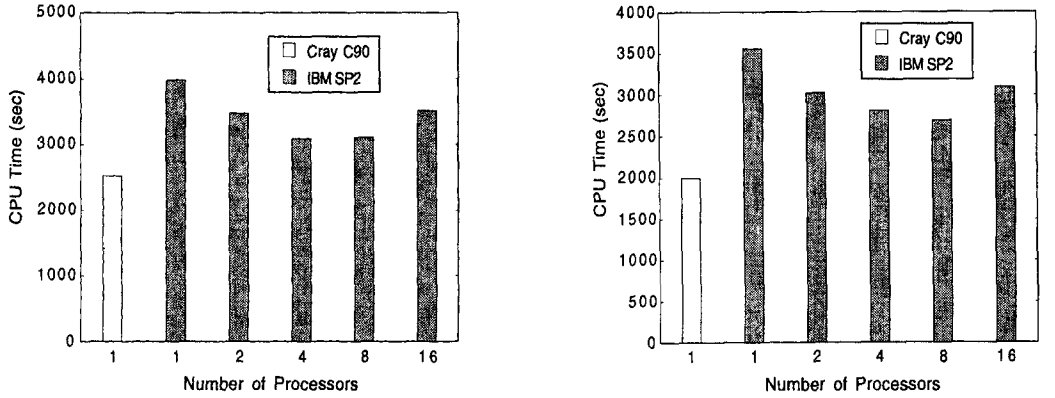
생하지 않는 slave1.f에서는 프로세서의 개수에 비례하는 성능향상을 얻을 수 있었으며, 접촉력을 계산하고 이것을 다른 프로세서와 공유하는 과정을 수행하는 slave2.f에서는 통신량이 많아짐으로 인하여 프로세서의 개수가 증가하면서 성능이 저하됨을 관찰할 수 있었다. 이 두 계산과정을 함께 고려할 때, 참 성능향상의 정도를 나타낸다. 오일 팬의 경우 프로세서가 4개일때 최대 성능이 나타났으며, 그 이후로는 통신 량의 증가에 따른 포화현상으로 성능이 감소함을 보여주었다. S자형 레일의 경우에는 프로세서가 8개일때 최고 성능을 나타냈으며 그 이후에는 오일 팬의 경우와 마찬가지로 포화현상으로 인한 성능저하가 발생되었다.

Fig. 6은 PARA-SFA의 전체적인 성능향상을 암달의 법칙(Amdahl's law)에 의한 이론적인 성능향상과 비교한 것이다. 여기서 암달의 법칙이란 순차프로그램(sequential program)을 병렬화했을 때



(a) Oil pan forming simulation (b) S-rail forming simulation

Fig. 6 Speed-up comparison between PARA-SFA results and predictions form Amdahl's law.



(a) Oil pan forming simulation (b) S-rail forming simulation

Fig. 7 Performance comparison of the results from SERI-SFA on Cray Y-MP C90 and PARA-SFA on IBM SP2.

얻을 수 있는 프로세서의 개수에 대한 성능향상을 예측하는 공식으로, 다음의 식과 같다.

$$S=\frac{T_t}{T_s+\frac{T_p}{n}} \tag{1}$$

여기서,

S : 증속 성능(speed-up)

T_t : 전체 계산시간, $T_t=T_s+T_p$

T_s : 순차부분의 계산시간

T_p : 병렬부분의 계산시간

n : 프로세서 개수

Fig. 6(a)는 오일 팬 성형 시뮬레이션시, 암달의 법칙의 이론 값과의 비교 결과이다. 이 때의 전체 계산시간에 대한 순차처리영역과 병렬처리 영역의 비가 각각 62%와 38%였다. 프로세서의 개수가 4개일 때까지는 거의 이론 값에 근접한 결과를 보여주다가 프로세서의 개수가 8개로 넘어서면서 성능저하가 나타난다. 이때 암달의 법칙은 통신에 의한 시간지연(latency) 등을 포함하여 여러 가지 지연현상을 고려하지 않은 이상적인 조건으로 가정하였기 때문에 실제 병렬처리 결과는 이보다 낮게 예측된다. Fig. 6(b)는 순차처리 영역과 병렬처리 영역의 비가 각각 65%와 35%인 S자형 레일의 시뮬레이션시 비교 결과이다. 프로세서의 개수가 8개일때까지는 거의 이론 값에 근접한 결과를 보여주다가 프로세서가 16개로 넘어서면서 포화현상에 의한 성능저하가 나타난다.

Fig. 7은 오일 팬 및 S자형 레일 시뮬레이션을 벡터 슈퍼컴퓨터인 Cray Y-MP C90 상에서 순차 프로그램 SERI-SFA로 수행하는데 소요되는 시간과, 다중병렬 컴퓨터인 IBM SP2 상에서 프로세서 개수 1, 2, 4, 8, 16개를 사용하여 병렬 프로그램

PARA-SFA로 수행하는데 소요되는 시간을 비교한 것이다. Cray Y-MP C90 프로세서 한 개의 최고 성능이 1 Flops(초당 10억회 실수 연산능력)이고, IBM SP2 프로세서 한 개의 최고 성능이 그 절반인 0.5 Flops인 것을 고려할 때, 선택적인 병렬화라는 제한성을 고려한다면 어느 정도 만족할 만한 병렬처리의 효과를 보여준다 할 수 있다.

5. 결 론

순차벡터 컴퓨터인 Cray Y-MP C90상에서 사용되는 판재성형 전용해석 프로그램인 SERI-SFA를 다중병렬 컴퓨터인 IBM SP2상으로 이식하는 병렬화 작업을 수행하였다. 병렬처리를 위한 메시지 패싱 라이브러리로는 현재 병렬화 목적으로 널리 사용되고 있는 MPI를 사용하여, 다양한 기종에서 사용할 수 있는 호환성을 추구하였다. 개발된 병렬처리 판재성형해석 프로그램 PARA-SFA는 순차프로그램 SERI-SFA와 같은 계산의 정확성을 보여주었고, 두 가지 예제 시뮬레이션을 통해 성능비교를 실시한 결과 제한적으로 부분적인 영역을 선택하여 병렬처리한 것으로서는 어느 정도 만족할 만한 성능향상을 얻을 수 있었다. 전체 계산시간에 대하여 병렬화 영역이 약 32%의 계산시간 비를 차지하는 오일 팬 성형 시뮬레이션의 경우, 프로세서의 개수가 4개일 때 암달의 법칙에 의한 이론적인 성능향상비가 약 39%인데 PARA-SFA는 약 30%의 성능향상비를 나타냈다. 한편 시간비로 약 35%의 병렬처리 영역을 가지는 S자형 레일의 성형 시뮬레이션의 경우에, 프로세서의 개수가 8개일때 이론적인 성능향상비가 44%인데 반하여 PARA-SFA에서는 약 32%의 성능향상비를 나타내었다. 부분적인 병렬화에 의한 성능향상은 어느 정도 만족할 만한 결과로 판단되나 전체적인 성능향상 및 프로세서수의 증가에 따른 성능향상비의 저하는 개선할 여지가 있는 것으로 본다. 본 연구는 최근에 개발되어 세계적으로 메시지 패싱 라이브러리의 표준으로 부각되고 있는 MPI를 이용하여 실제로 실무에서 사용되고 있는 판재성형해석 프로그램을 병렬화했다는 것과, 특히 미래의 컴퓨터 환경이라고 예측되는 병렬컴퓨터에 순차처리 프로그램을 이식하여 비교적 효과적인 결과를 얻을 수 있었다는 점에서 큰 의미가 있다고 할 수 있다.

보다 괄목할만한 성능향상의 증가를 위해서는 병렬처리 영역이 확대되어야 하는데, 이는 영역분할법(domain decomposition)의 적용 및 효과적인 병렬처리 영역의 분할[9~11] 그리고 통신에 따른 시간 소비를 효율적으로 통제하기 위한 최적의 메시지 패싱 라이브러리 기능의 활용 등으로 해결할 수 있으리라 생각된다. 또한 보다 근본적으로는 유한요소해석 프로그램 개발시 병렬처리를 고려한 효율적인 알고리즘을 개발하여야 할 것이다.

후 기

본 연구는 시스템공학연구소 슈퍼컴퓨터센터의 재정적 지원으로 수행하였으며, 연구결과의 정리 및 발표는 한남대학교 교비 연구비 계정에서 부분적으로 지원되었습니다. 관계자 여러분과 IBM SP2를 사용할 수 있도록 도움을 주신 한국 IBM, Cray Y-MP C90 상의 SERI-SFA 안정화 작업을 성공적으로 수행해 주신 이정열 과장(청남엔지니어링)에게 감사를 드립니다.

참고문헌

(1) Foster, I., 1995, *Designing and Building Parallel Programs*, Addison Wesley.

(2) Suh, Y. S., Lee, J. Y., Kim, S. W. and Lim, D. S., 1996, *Development of 3 Dimensional Sheet-Forming Analysis System* (Ⅱ), Cray Research Inc., Final Report.

(3) MPI 1.1 on-line document, http://www.mcs.anl.gov/mpi/mpi-report-1.1/ node*.html#Node*.

(4) Cornell Theory Center, 1996, "Introduction to Parallel and Distributed Memory Programming," *CTC Virtual Workshop Topics in High Performance Computing*.

(5) Cornell Theory Center, 1996, "The Message-Passing Interface Standard(MPI)," *CTC Virtual Workshop Topics in High Performance Computing*.

(6) Hallquist, J. O., 1983, "DYNA3D THEORETICAL MANUAL," Method Development Group, University of California, Lawrence Livermore National Laboratory, Report UCRL 19401.

(7) NUMISHEET '96: *3rd International Confer-*

ence on Numerical Simulation of 3-D Sheet Metal Forming Processes, 1996, Dearborn, Michigan, U. S. A.

(8) Suh, Y. S. and Lee, J. Y., 1995, "A Virtual Manufacturing of an Automobile Oil Pan made from the Vibration Damping Steel Sheet," Chung, H. H. and Kwon Y. W. (eds.) in PVP-Vol. 321/NE-Vol. 18, *Recent Advances in Solids and Structures*, ASME.

(9) Schweizerhof, K., Muller, G., Weiner, K., Wainscott, B. and Hallquist, J. O., 1995, "Experiences with LS-DYNA3D on Various Parallel Computers," *3rd International LS-DYNA3D Conference*, Kyoto, Japan.

(10) Farhat, C., Wilson, E. and Powel, G., 1987, "Solution of Finite Element Systems on Concurrent Computers," *Engineering with Computers*, Vol. 2, pp. 157~165

(11) Fox, G. C., 1988, *Numerical Algorithms for Modern Parallel Computers*, Springer-Verlag, Berlin.