다분야통합해석에 기반한 설계문제의 병렬처리를 위한 부하분산알고리즘

Load Balancing Algorithm for Parallel Computing of Design Problem involving Multi-Disciplinary Analysis

조 재 석* 주 민 식** 송 용 호*** 최 동 훈†
Cho, Jae-Suk Chu, Min-Sik Song, Yong-Ho Choi, Dong-Hoon
(논문접수일 : 2007년 3월 23일 ; 심사종료일 : 2007년 5월 19일)

요 지

다분야통합해석에 기반한 설계문제는 일반적으로 전체 설계과정에서 매우 큰 계산시간을 요구하며, 이러한 계산시간을 단축하기 위해 병렬처리시스템을 도입하는 것이 필수적이다. 그러나 다분야통합해석에 기존의 병렬처리기법을 적용하기 위해서는 해석에 필요한 모든 CAE 소프트웨어들이 병렬처리시스템의 모든 서버에 설치되어 있어야 하며, 이는 매우 큰 CAE 소프트웨어의 비용을 필요로 한다. 본 논문에서는 이러한 문제점을 해결하기 위해 가중치 기반 멀티큐 부하분산 알고리즘을 제안하였다. 제안된 알고리즘은 서버들의 성능과 설치된 CAE 소프트웨어들의 종류가 각기 다른 이종 병렬처리시스템을 고려하였으며 성능검증을 위해 선입선출(First Come First Servre) 알고리즘을 적용한 경우와 비교한 전산실험을 수행하였다.

핵심용어 : 다분야통합해석, MDA, 병렬처리시스템, MPS, 병렬처리, 부하분산알고리즘

Abstract

An engineering design problem involving Multi-Disciplinary Analysis(MDA) generally requires a large amounts of computing time for the entire design process, and therefore it is essential to introduce a Multiple Processor System (MPS) for reducing the computing time. However, when applying conventional parallel processing techniques, all of the CAE S/W required for the MDA should be installed on all the servers making up MPS because of characteristic of MDA and it would be a great expense in CAE S/W licenses. To solve this problem, we propose a Weight-based Multiqueue Load Balancing algorithm for a heterogeneous MPS where performance of servers and CAE S/W installed on each server are different of each other. To validate the performance, a computational experiments comparing the First Come First Serve algorithm and our proposed algorithm was accomplished.

Keywords : *multi-disciplinary analysis, mda, multiple processor system, mps, parallel computing, load balancing*

1. 서 론

최근 제조업계의 제품설계단계에서의 요구사항은 제품의 성능을 극대화하면서 설계에 소요되는 시간을 최소화하는 것이다. 그러나 만족할 만한 제품성능을 얻기 위해서는 많은 수의 설계안에 대한 시스템의 반응을 구해야 하며, 이로 인해 설계안의 해석을 수행하는데 많은 시간이 소요된다. 설계안에 대한 시스템의 반응을 구하기 위해 여러 분야의 공학적 원리들을 동시에 고려하는 다분야통합해석(Multi-Disciplinary Analysis; MDA)을 수행해야 하는 설계문제일 경우 설계모델의 해석에 소요되는 시간은 더욱 커지게 된다.

이러한 소요시간을 단축하기 위해 병렬처리시스템(Multiple

† 책임저자, 정회원 · 한양대학교 최적설계신기술연구센터 소장
Tel: 02-2220-0478 ; Fax: 02-2291-4070
E-mail: dhchoi@hanyang.ac.kr
* 한양대학교 최적설계신기술연구센터
** 한양대학교 최적설계신기술연구센터
*** 한양대학교 정보통신공학부

• 이 논문에 대한 토론을 2007년 8월 31일까지 본 학회에 보내주시면 2007년 10월호에 그 결과를 게재하겠습니다.

Processor System; MPS)을 도입하고자 하는 연구가 진행되고 있다. 그러나 MDA에 기반한 설계문제의 경우, 일반적인 병렬처리기법을 적용하여 병렬처리를 효율적으로 수행하기 위해서는 MPS의 모든 서버에 MDA에 필요한 모든 CAE S/W들을 설치해야 하며, 설계문제의 MDA가 해석에 상용 CAE S/W를 필요로 하는 모델로 구성된 경우 이는 매우 큰 S/W 비용을 필요로 한다.

본 연구에서는 MPS의 각 서버들에 MDA에 필요한 CAE S/W들이 각각 모두 설치되어 있지는 않은 경우를 고려한 가중치 기반 멀티큐 부하분산알고리즘을 제안하였다. 제안된 알고리즘은 각 CAE S/W모델별로 별도의 큐를 관리하고, 각 모델들의 해석에 가중치를 부여하였다. 알고리즘의 성능을 검증하기 위해 각 모델의 해석과 MPS의 각 서버들을 모델링한 시뮬레이터를 구현하여 전산실험을 수행하였다.

2. 연구배경 및 연구목적

실제 산업현장의 설계분야에서 일반적으로 사용되는 설계기법인 반응표면법(Response Surface Method), 다구찌기법(Taguchi Method), 매개변수분석기법(Parametric Study) 등을 적용하기위해서는 설계기법이 요구하는 다수의 실험점(Sampling Point)에 대한 시스템의 반응을 구해야 한다. 산업현장에서는 일반적으로 상용 CAE S/W로 구현한 정밀한 설계모델을 사용함을 고려할 때 이는 매우 긴 시간을 필요로 한다. 이러한 소요시간을 단축하기 위해 MPS를 이용하고자 하는 연구(Kato, 2005; Kim *et al.*, 2005; Koch *et al.*, 2000)가 수행되어 왔다. 설계문제가 단일분야의 해석(e. g. 구조해석, 유동해석, NVH해석)에 기반한 일반적인 문제일 경우 MPS의 각 서버에 해당 CAE S/W 하나만 설치되어 있으면 선입선출(First Come First Serve; FCFS) 알고리즘과 같은 일반적인 병렬처리기법을 적용하여도 MPS의 최대성능을 이용할 수 있다.

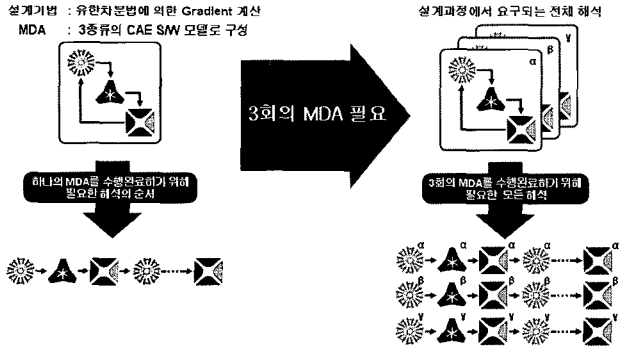


그림 1 설계기법에 의해 생성된 MDA들을 수행하기 위한 각 CAE 모델들의 해석순서

그러나 MDA에 기반을 둔 설계문제일 경우 단일분야의 해석에 기반한 설계문제에 적용된 병렬처리 개념을 확장하는 형태로 선입선출알고리즘을 적용하기 위해서는 MDA에 필요한 모든 CAE S/W가 MPS의 각 서버에 모두 설치되어야 한다. 일반적으로 산업현장에서는 고가의 상용 CAE S/W를 사용함을 고려할 때 이는 매우 큰 S/W비용을 필요로 한다. 그러나 MDA는 그림 1과 같이 해석들이 순차적으로 수행되어야 하는 부분이 많으며, 모든 CAE S/W를 동시에 실행하는 것은 아니므로 MPS의 각 서버에 모든 CAE S/W가 설치되지는 않았고, 서버들의 성능도 상이한 Heterogeneous MPS를 이용하더라도 다수의 시스템반응값을 구하기 위해 병렬처리를 수행할 수 있다.

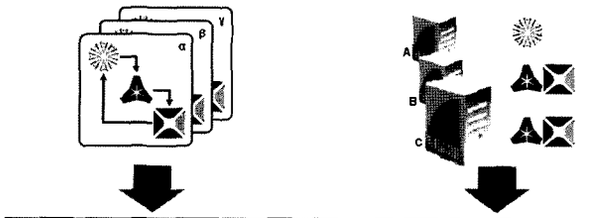
Heterogeneous MPS를 이용하여 MDA에 기반한 설계문제의 다수의 시스템반응값을 구하기 위해 병렬처리를 수행할 경우, 일반적인 병렬처리 문제에서 고려하지 않는 세 가지 조건을 고려해야 하는데 첫 번째는 MDA를 구성하는 각 CAE 모델들의 해석을 완료하는데 필요한 요구계산량과 MPS를 구성하는 각 서버들이 제공할 수 있는 계산성능이 각기 다르다는 점이고 두 번째는 MDA의 특성상 각 CAE 모델들이 연성(Coupling)관계를 가진다는 점이며 세 번째는 MPS의 각각의 서버들이 해석을 수행할 수 있는 CAE 모델들의 종류가 각기 다르다는 점이다. 위 세 가지 조건으로 인해 해석들의 수행순서를 어떻게 결정하느냐에 따라 전체 설계과정을 완료하는데 소요되는 시간이 크게 달라지게 되므로 이러한 조건들을 고려한 스케줄링 알고리즘이 필요하다.

병렬처리의 목적, 작업들의 특성과 MPS의 구성방식에 따라 매우 다양한 부하분산알고리즘(Load Balancing)이 개발되었으나 이러한 조건들을 모두 고려한 스케줄링 알고리즘은 발표된 바가 없으며, 알려진 바와 같이 병렬처리문제의 조건이 다를 경우 기존 알고리즘을 적용하면 효율성을 기대할 수 없으므로 (Kameda *et al.*, 1997; BARAK *et al.*, 1993), 본 논문에서는 전술한 세 가지 조건을 모두 고려한 부하분산알고리즘을 제안하였다.

3. 가중치 기반 멀티큐 부하분산알고리즘

3.1. 다분야통합해석에 기반한 설계문제와 병렬처리시스템의 모델링 방법

본 논문에서 제안된 알고리즘은 설계문제의 MDA가 I개의 CAE 모델들로 구성되며 모델의 해석순서는 설계구조행렬(Design Structure Matrix; DSM)에 정의되고, 설계기법에 정의된 N개의 실험점(Sampling Point)에 의한 시스템



		δ	C	μ_1	μ_2	μ_3
	J_1^n	2	3	0	1	0
	J_2^n	2	3	0	0	1
	J_3^n	2	6	1	0	0

	P	B_1	B_2	B_3
R_1	1	1	0	0
R_2	1	0	1	1
R_3	2	1	1	1

그림 2 MDA와 MPS의 모델링 예

의 반응값을 구하기 위해 N개의 MDA를 수행해야 하는 경우를 고려한다. 또한 병렬처리를 수행할 MPS는 각 서버들의 성능이 각기 다르며, 설치된 CAE S/W들의 종류도 각기 다른 M개의 서버로 구성된 Heterogeneous MPS인 경우를 고려하였다. n번째 MDA의 DSM에 속한 i번째 모델의 k번째 해석은 작업 ${}_kJ_i^n$으로 모델링되며, ${}_kJ_i^n$의 수행완료에 필요한 연산의 예상치인 요구계산량은 ${}_kJ_i^n{:}C$ (unit : $GHz \cdot sec$)로, DSM의 j번째 모델이 J_i^n의 해석결과를 입력변수로 사용하는지 여부는 $J_i^n{:}\mu_j$로, MDA가 수렴하기까지 수행해야 하는 J_i^n의 반복회수의 예상치를 $J_i^n{:}\delta$로 정의한다. 또한 MPS의 m번째 서버는 R_m으로 모델링되며, R_m의 성능은 $R_m{:}P$ (unit : GHz)로, R_m에 DSM의 i번째 모델의 해석을 수행할 수 있는 CAE S/W의 설치여부는 $R_m{:}B_i$로 정의한다. MDA와 MPS를 모델링 예는 그림 2와 같다.

해석소요시간 ${}_kT_i^n$은 ${}_kJ_i^n$이 R_m에서 수행되었을 때 완료에 소요되는 시간이며 식 (1)과 같이 정의된다.

$$T_i^n = \frac{J_i^n{:}C}{R_m{:}P} = c/p \ (\text{sec}) \qquad (1)$$

〔where $J_i^n{:}C$ = c $GHz \cdot sec$, $R_m{:}P$ = p GHz 〕

실제 설계문제에서는 실험점에 의해 정해지는 입력변수의 초기값에 따라 ${}_kJ_i^n{:}C$와 $J_i^n{:}\delta$의 크기가 모두 다르므로 ${}_kT_i^n$도 모두 다르며 각 MDA가 언제 수렴할 지도 정확히 예측할 수 없다. 따라서 설계과정의 초기에는 $J_i^n{:}\delta$와 ${}_kJ_i^n{:}C$를 추정하여 부여하고 설계과정 중 수렴한 MDA들의 $J_i{:}\delta$의 평균값 $J_i^{avg}{:}\delta$을 아직 수렴하지 않은 MDA들의 $J_i{:}\delta$의 예상치로, 마찬가지로 각 모델의 수행 완료된 해석들에 사용된 연산횟수의 평균값 $J_i^{avg}{:}C$을 아직 수행되지 않은 해석의 예상치로 이용해야 한다.

그러나 과거이력을 사용할 경우 알고리즘의 성능을 정확히 비교평가하기 어려워지므로 동일한 CAE 모델이면 모델의 입력변수값이 다르더라도 $J_i^n{:}\delta$와 $J_i^n{:}C$의 값이 동일하며 초기에 추정하여 부여한 예상치에서 변하지 않는다고 가정한다.

3.2. 제안된 알고리즘의 개념

MDA의 DSM이 그림 1과 같고 설계기법에 의해 3회의 MDA를 수행해야 하며 $J_1^n{:}\delta$, $J_2^n{:}\delta$, $J_3^n{:}\delta$가 모두 2회인 설계문제를 살펴보자.

만약 $R_A{:}P$, $R_B{:}P$, $R_C{:}P$가 2 GHz로 모두 동일하고 $J_1^n{:}C$, $J_2^n{:}C$, $J_3^n{:}C$가 10 $GHz \cdot sec$로 모두 동일하다면 각 서버가 작업들을 수행한 이력은 그림 3과 같다. 모든 작업이 완료된 시점을 비교해보면 그림 3b)와 같이 R_A가 J_1^n, R_B가 J_2^n, R_C가 J_3^n을 각각 수행할 수 있는 경우

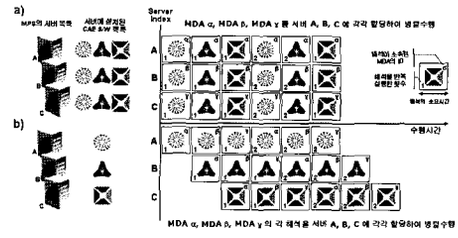


그림 3 MPS의 서버들의 성능과 MDA의 해석들의 요구계산량이 모두 동일한 설계문제의 병렬처리 예

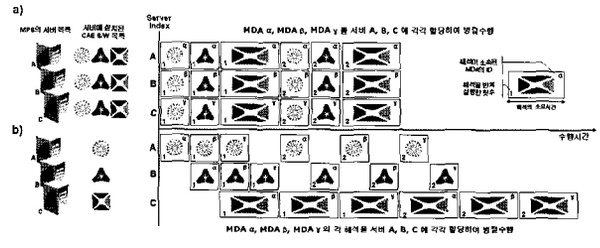


그림 4 MPS의 서버들의 성능은 동일하나 MDA의 해석들의 요구계산량은 상이한 문제의 병렬처리 예

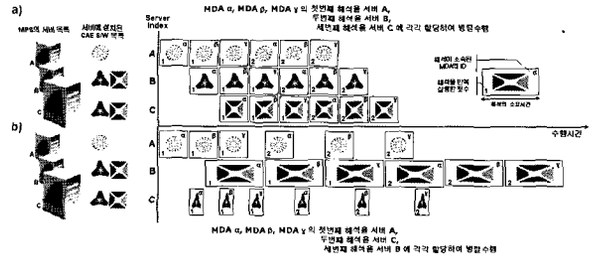


그림 5 MPS의 서버들의 성능과 MDA의 해석들의 요구계산량이 모두 상이한 설계문제의 병렬처리 예

정해진 해석순서로 인해 R_B, R_C가 유휴상태(Idle)로 대기하는 시간이 약간 존재한다 하더라도 그림 3a)와 같이 각 서버가 모든 작업을 수행할 수 있는 경우에 근접하는 병렬처리 성능을 보일 수 있게 된다.

그러나 $R_A{:}P$, $R_B{:}P$, $R_C{:}P$는 2 GHz로 모두 동일하며 $J_1^n{:}C$, $J_2^n{:}C$도 10 $GHz\cdot sec$로 동일하지만 $J_3^n{:}C$가 20 $GHz\cdot sec$일 때 R_A가 J_1^n, R_B가 J_2^n, R_C가 J_3^n을 수행할 수 있을 경우 그림 4b)와 같이 T_1^n, T_2^n에 비해 T_3^n가 두 배이기 때문에 병렬처리 과정 중간에 R_A, R_B가 유휴상태로 대기하는 현상이 발생하며 이로 인해 성능이 그림 4a)에 비해 현저히 떨어지게 된다.

$R_A{:}P$, $R_B{:}P$가 2 GHz로 동일하나 $R_C{:}P$는 4 GHz이고 $J_1^n{:}C$, $J_2^n{:}C$도 10 $GHz\cdot sec$로 동일하나 $J_3^n{:}C$가 20$GHz\cdot sec$이며 R_A가 J_1^n을, R_B, R_C가 각각 J_2^n, J_3^n을 수행할 수 있는 경우를 고려해보자. 그림 5에서 각 서버가 작업들의 병렬처리를 수행한 이력을 살펴보면 J_2^n와 J_3^n을 R_B, R_C 중 어떤 서버에서 수행하느냐에 따라 전체 병렬처리 성능이 매우 달라진다. 그림 5b)와 같이 R_C가 J_2^n, R_B가 J_3^n을 수행할 경우 T_3^n가 T_1^n의 두 배, T_2^n의 네 배가 되어 그림 5a)와 같이 R_B가 J_2^n, R_C가 J_3^n을 수행하여 T_1^n, T_2^n, T_3^n가 모두 동일한 경우보다 성능이 현저히 떨어지게 된다.

따라서 병렬처리성능을 최대화하기 위해서는 MPS의 각 서버들의 가동률을 최대화해야 하고, 이를 위해 각 모델들의 전체해석소요시간 중 최대값인 $Max(\sum T_i^n)$을 최소화해야 한다. 따라서 각 모델들의 $J_i^n{:}C$와 각 서버들의 $R_m{:}P$을 고려하여 가능한 $J_i^n{:}C$이 큰 모델부터 J_i^n을 수행할 수 있는 서버들 중 $R_m{:}P$이 높은 서버에 할당함으로서 서버들의 부하(Workload)를 조절해야 한다. 이를 위해 제안된 알고리즘은 각 모델의 서버할당의 우선순위로 사용되는 가중치를 부여한다.

또한 MPS가 하나의 큐만으로 모든 작업들을 관리할 경우, 만약 큐의 최상단에는 J_1이, J_1보다 하단에 J_2가 대기중이라면 J_2를 수행할 수 있는 서버가 유휴상태가 된다 하더라도 다른 서버가 J_1의 수행을 시작할 때 까지 유휴상태로 대기해야 한다. 이러한 현상이 발생할 경우 서버가동률이 떨어지게 되며 전체적인 병렬처리의 효율성이 떨어지게 된다. 이를 방지하기 위해서는 MPS에 작업종류별로 독립적인 큐를 유지하는 멀티큐를 도입해야 한다.

3.3 가중치 부여방법

제안된 알고리즘은 식 (2)를 통해 가중치를 각 CAE 모델에 부여하며 각 모델의 가중치를 구하기 위해서는 각 서버들이 어떤 작업들을 담당할지 결정해야한다. 각 서버들이 담당할 작업들은 $Max(\sum T_i^n)$을 최소화할 수 있는 조합을 찾음으로서 결정된다. 가중치 ${}_kW_i^n$은 ${}_kJ_i^n$의 해석결과를 입력변수로 사용하는 다른 모델들 중 ${}_kJ_i^n$보다 DSM에서 상단에 위치하는 모델이 존재하는지 여부에 따라 다른 식이 적용된다.

a) [if $all\ of\ J_i{:}\mu_D = 0\ (1 \le D < i)$]

$${}_kW_i^n = {}_kT_i^n + \sum_{d=i+1}^{I} ({}_kJ_i^n{:}\mu_d \times {}_kT_d^n)$$

b) [if $any\ of\ J_i{:}\mu_D = 1\ (1 \le D < i)$]

$${}_kW_i^n = {}_kT_i^n + \sum_{d=i+1}^{I} ({}_kJ_i^n{:}\mu_d \times {}_kT_d^n) + ({}_kJ_i^n{:}\delta - k) \times \sum_{d=D}^{i} {}_kT_d^n \qquad (2)$$

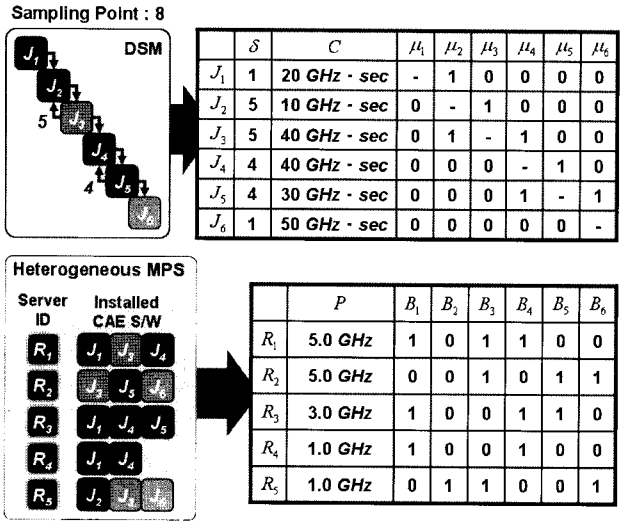
[where, $T_i^n = \dfrac{J_i^n{:}C}{\sum_{m=1}^{M}(R_m{:}B_i \times R_m{:}P)}$]

각 모델의 가중치가 정해지면 DSM에 정의된 모델들의 해석순서에 따라 생성되는 각 작업들을 해당 작업의 큐에 등록한다. 서버들 중 유휴상태로 대기중인 서버가 발생하면 각 큐들의 최상단에 대기중인 작업들을 비교하여 해당서버에서 수행할 수 있고 가중치가 가장 높은 작업을 할당하며 또한 각 작업의 가중치들을 주기적으로 갱신한다.

4. 전산실험 결과

4.1 실험환경 및 비교평가방법

제안된 알고리즘의 성능을 평가하기 위해 C^{++}로 구현한 시뮬레이터를 작성하였다. 3.1절에 설명한 모델링 방법으로 작업과 자원, 큐를 구현하고, 3.3절에 설명한 동작방식으로 각 CAE S/W모델의 해석이 서버에서 수행되는 과정을 모사하였으며 일반적인 병렬처리기법인 싱글큐 선입선출 알고리즘의 성능과 비교분석하였다. 싱글큐 선입선출알고리즘은 큐의 최상단에 대기중인 작업을 수행할 수 있는 서버가 발생하면 작업을 서버에 할당하게 되며 작업을 수행할 수 있는 서버가 동시에 2개 이상 발생할 경우 서버의 성능에 대한 고려없이 서버의 ID가 앞쪽에 있는 서버부터 할당하도록 구현하였다.



	δ	C	μ_1	μ_2	μ_3	μ_4	μ_5	μ_6
J_1	1	20 GHz · sec	-	1	0	0	0	0
J_2	5	10 GHz · sec	0	-	1	0	0	0
J_3	5	40 GHz · sec	0	1	-	1	0	0
J_4	4	40 GHz · sec	0	0	0	-	1	0
J_5	4	30 GHz · sec	0	0	0	1	-	1
J_6	1	50 GHz · sec	0	0	0	0	0	-

	P	B_1	B_2	B_3	B_4	B_5	B_6
R_1	5.0 GHz	1	0	1	1	0	0
R_2	5.0 GHz	0	0	1	0	1	1
R_3	3.0 GHz	1	0	0	1	1	0
R_4	1.0 GHz	1	0	0	1	0	0
R_5	1.0 GHz	0	1	1	0	0	1

그림 6 전산실험 예제문제의 모델링

4.2 실험결과 및 고찰

MDA의 DSM의 구조와 MPS의 형태는 그림 6과 같다. DSM은 CAE S/W모델의 해석순서에 두 개의 루프가 존재하는 비교적 복잡한 형태의 문제이며 가장 효율적으로 병렬처리를 수행할 수 있는 이상적인 조건은 그림 7의 Ideal MPS의 경우로서 서버들의 개수와 각 서버들의 성능이 동일할 경우 이 보다 더 빠르게 병렬처리를 수행할 수 있는 조건은 없다.

MDA의 DSM과 MPS의 형태를 고려하여 문제의 특징을 살펴보면 각 MDA의 수행을 완료하기 위해 수행해야 하는 작업들 중 대부분은 4종류의 작업(J_2, J_3, J_4, J_5) 중 하나이며 이 중 상당부분을 차지하는 J_2를 수행할 수 있는 서버는 1GHz의 성능을 가지는 R_5 하나뿐이다. 따라서 서버들이 유휴상태에서 대기하는 시간을 줄이고 서버가동율을

	Total Time	Time Ratio	Rate of Operation
Ideal MPS	340.25 sec	43.58 %	94.05 %
MQ+Weight	528.00 sec	67.63 %	60.62 %
Multi Queue FCFS	602.75 sec	77.20 %	53.11 %
Single Queue FCFS	780.75 sec	100.0 %	40.99 %

그림 8 전산실험 결과

높이기 위해서는 R_1, R_2, R_3, R_4가 처리할 수 있는 작업들이 큐에 최대한 빨리 등록되어야 하며 이를 위해서는 R_5가 J_2를 집중적으로 처리하여야 한다.

싱글큐 FCFS의 경우, 모든 종류의 작업들을 하나의 큐에서 관리하며, 큐의 하단에서 대기중인 작업을 상단에서 대기중인 작업보다 먼저 수행할 수 없는 큐의 특성상 ${}_1J_2^1$의 수행이 완료되면 ${}_1J_3^1$가 큐의 최하단에 들어오게 되므로, ${}_1J_2^1$부터 ${}_1J_2^8$까지 모든 실험점의 J_2가 수행 완료되어야만 J_3를 수행할 수 있으며 마찬가지로 ${}_1J_3^1$부터 ${}_1J_3^8$까지 모든 J_3의 수행이 완료되어야 ${}_2J_2^1$부터 ${}_2J_2^8$까지의 작업을 수행할 수 있게 된다. 이로 인해 그림 7과 같이 R_5가 J_2를 집중적으로 처리하지 못하므로. R_1, R_2, R_3, R_4가 유휴상태에서 대기하는 시간이 길어지게 되며 전체 작업의 수행완료시간이 늦어진다.

멀티큐 FCFS의 경우, ${}_1J_2^1$의 수행이 완료되면 ${}_1J_3^1$가 J_3를 관리하는 큐의 최상단에 들어오게 되므로, ${}_1J_2^1$의 수행이 완료되는 즉시 ${}_1J_3^1$를 수행할 수 있게 되며 마찬가지로 ${}_1J_3^1$의 수행이 완료되면 ${}_2J_2^1$를 수행할 수 있게 되므로 그림 7과 같이 R_5가 J_2를 집중적으로 처리하므로 R_1, R_2, R_3, R_4가 유휴상태에서 대기하는 시간이 싱글큐 FCFS보다 짧아지게 되므로 전체적인 효율성이 약 12% 정도 향상되고 전체 작업완

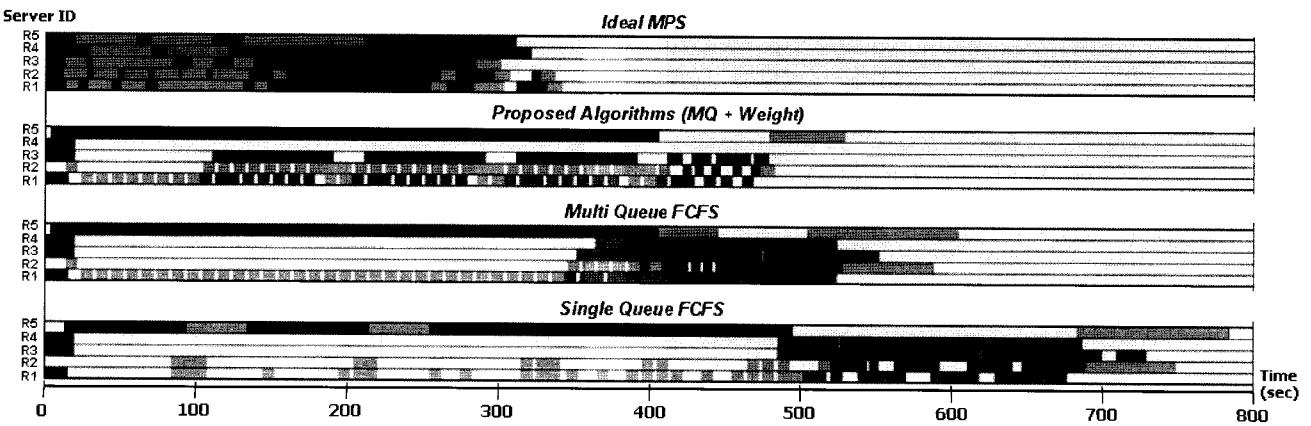


그림 7 전산실험 예제문제의 실행내역

료시간도 약 23% 정도 단축된다.

그러나 그림 7에서 보는 바와 같이 멀티큐를 도입하였다 할지라도 R_5가 J_2를 수행하는 동안 R_1이 모든 J_3를 처리할 수 있기 때문에 모든 J_2, J_3의 수행이 완료될 때 까지 약 350초 동안 R_2, R_3, R_4는 유휴상태에서 대기하게 된다. 이러한 현상을 최소화 위해서는 최대한 빨리 J_4, J_5, J_6이 큐에 등록되어 R_2, R_3, R_4가 작업을 수행 할 수 있도록 해야 한다. 이를 위해 식 (2)를 이용하여 각 작업에 가중치를 부여하고 가중치가 작은, 즉 해당 작업이 소속된 실험점의 전체 해석순서가 많이 진행된 작업부터 처리함으로서 일부 실험점의 J_4, J_5, J_6을 먼저 큐에 등록시킬 수 있게 된다. 이를 통해 그림 7과 같이 멀티큐 FCFS보다 효율성이 약 7% 정도, 전체 작업 수행완료시간이 약 10% 단축되게 된다.

실험결과들을 정리하면 그림 8과 같다. 제안된 알고리즘을 적용했을 때, 동일한 조건의 MPS를 이용했을 경우 전체 작업의 수행완료시간이 싱글큐 FCFS를 적용했을 경우 보다 약 33% 정도 단축되었으며, Ideal MPS를 이용했을 경우보다 절반정도의 CAE S/W 개수를 사용하여 약 61%의 서버 가동률을 보일 수 있었으며 Ideal MPS 대비 약 64.4%의 병렬처리성능을 나타낼 수 있었다.

5. 결 론

Heterogeneous MPS를 이용하여 복수의 MDA들의 병렬처리를 수행할 경우 효율성을 향상시키기 위한 가중치기반 멀티큐 부하분산알고리즘을 제안하였으며 알고리즘의 특징은 다음과 같다.

① MDA를 구성하는 각 CAE 모델들이 서버를 할당받을 수 있는 우선순위를 나타내는 가중치를 계산하고 MPS의 각 서버들 중 유휴상태(Idle)로 대기하는 서버가 발생하게 되면 MPS의 큐들을 검색하여 해당서버에서 해석을 수행할 수 있는 CAE S/W모델들 중 가장 가중치가 높은 모델을 할당한다.

② CAE S/W모델의 해석의 가중치는 각 CAE 모델의 해석에 필요한 연산의 크기와 MPS의 각 서버들의 성능, 해당 모델의 해석결과를 입력변수로 사용하는 모델들에 대한 가중치를 고려하여 부여한다.

③ CAE S/W모델의 종류에 기반한 멀티큐를 도입하여 수행대기중인 작업들을 종류별로 독립적으로 관리함으로서 유휴상태의 서버 발생 시 서버가 유휴상태로 대기하게 되는 시간을 최소한으로 줄였다.

작업과 서버, 큐를 모델링하였으며 C^{++}를 이용하여 구현한 시뮬레이터를 구축하였으며 알고리즘의 성능을 싱글큐 선입선출알고리즘과 비교, 평가한 결과 제안된 알고리즘의 성능이 전반적으로 우수함을 확인하였다.

감사의 글

본 연구는 최적설계신기술연구센터와 2007년도 2단계 두뇌한국21사업, 산업자원부의 국제공동기술개발사업에 의하여 지원되었으며 지원해주신 각 당국에 감사의 뜻을 표합니다.

참 고 문 헌

BARAK, A., S. GUDAY, R.G. WHEELER(1993) *The MOSIX Distributed Operating System - Load Balancing for UNIX*, Springer-Verlag.

Kameda, H. *et al.*(1997) *Optimal Load Balancing in Distributed Computer Systems*, Springer.

Kato, T.(2005) Realizing Grid Computing as Engineering System for Collaborative Parameter Study, *GridWorld/GGF15*, Boston, MA, USA.

Kim, H. *et al.*(2005) A Parallel Trade Study Architecture for Design Optimization of Complex Systems, *46th AIAA/ASME/ASCE/AHS/ASC Structures, Structural Dynamics & Materials Conference.*

Koch, P.N., B. Wujek, O. Golovidov.(2000) A Multi-Stage, Parallel Implementation of Probabilistic Design Optimization in an MDO Framework, *8th AIAA/USAF/NASA/ISSMO Symposium on Multidisciplinary Analysis and Optimization.*