IBM 시스템의 LoadLeveler 최적 작업환경 구현

이영주*, 성진우*, 김성준*, 장지훈*
*한국과학기술정보연구원
e-mail:yjlee@kisti.re.kr

LoadLeveler Optimization Job Environment Implement in IBM System

Young-Joo Lee*, Jin-Woo Sung*, Sung-Jun Kim*, Ji-Hoon Jang*
*Korea Institute of Science Technology Information

요 약

시스템의 한정된 자원을 다수의 사용자들이 프로그램을 실행 시 자원을 효율적으로 배분하기 위하여 작업관리 시스템을 사용한다. 이러한 작업관리 시스템은 여러가지 종류가 있으며 사용하는 시스템의 환경과 작업의 특성에 따라 적당한 작업관리 시스템을 선택하여 사용한다. IBM 시스템은 자체로 제공하는 작업관리 시스템으로서 LoadLeveler를 사용하고 있는데, 이러한 LoadLeveler에서의 클래스를 설계하여 작업의 처리 효율을 높혔으며 계정별 작업 우선순위를 부여하여 사용자게 선택의 폭을 넓히고 최적 환경을 구성하였다.

작업관리 시스템의 주요한 시스템 환경변수는 CPU와 메모리이고, 작업환경 변수는 작업 실행시간이다. 따라서 KISTI IBM 시스템에서는 이러한 환경을 사용자의 배분정책에 맞게 설계하여 시스템의 안정성을 유지하고 사용자의 선택에 따라 전체 작업처리 효율을 증가하였다.

1. 서론

작업관리 시스템은 한정된 전체 시스템의 자원을 다수의 사용자가 요구한 자원에 따라서 효율적으로 배분하고 관리할 수 있는 시스템이다. 이러한 작업관리 시스템은 전체 시스템 자원을 하나의 사용자가 독점 사용하는 것을 막고 동시에 다수의 사용자가 각각의 프로그램에서 처리할 자원을 요구할 때 각각의 요구자원을 배분한다. 이러한 작업관리 시스템은 여러 종류가 있으며 시스템의 종류와 작업의 특성에 따라서 적당한 작업관리 시스템을 사용되고 있다. KISTI에 설치된 IBM p595 시스템에서는 LoadLeveler, SUN 시스템에서는 SunGridEngine, 클러스터 시스템은 PBS 등이 사용되고 있다. 이러한 작업관리 시스템은 전체 작업의 흐름을 원활하게 유도하고 시스템의 작업처리에 많은 영향을 준다. 작업관리 시스템을 설계하고 관리하는 데 많은 환경변수를 가지고 있지만 그 중에서도 CPU와 메모리가 가장 큰 환경 변수 요인이다.

본 논문에서는 작업관리 시스템을 통하여 작업을 실행할 때 클래스를 시스템의 특성에 맞게 구현하고 사용자계에 따라서 작업의 우선 순위를 부여하여 시스템의 안정성을 유지하고 사용자의 클래스 선택의 폭을 넓혀서 전체 시스템의 작업 처리 효율성을 높이고자 한다.

2. 관련 연구

2.1 LoadLeveler

LoadLeveler는 분산컴퓨터를 위한 작업로드 관리 시스템으로서 사용자가 여러 노드 또는 시스템을 하나의 컴퓨터처럼 사용할 수 있게 한다. 또한 여러 시스템의 작업 부하를 균형 있게 관리하며, 순차 프로그램과 병렬 프로그램을 모두 지원한다. 이 작업관리 시스템은 원래는 위스콘신 대학교의 Condor Job Scheduler를 기초로 한 것으로 IBM에 의해 사용자 기반 우선 순위, NFS/AFS 지원, GUI 등의 기능이 추가되었다.

2.2 LoadLeveler 구조

LoadLeveler는 사용자의 프로그램을 실행하기 위해 명시한 요구자원을 시스템 자원으로부터 할당받아 처리한다. 사용자가 LoadLeveler를 통하여 작업을 제출하면 작업은 일단 LoadLeveler의 큐에 들어가 대기하다가 해당 작업의 처리 조건이 가능한 큐를 찾으면 해당 큐에서 작업을 실행한다.

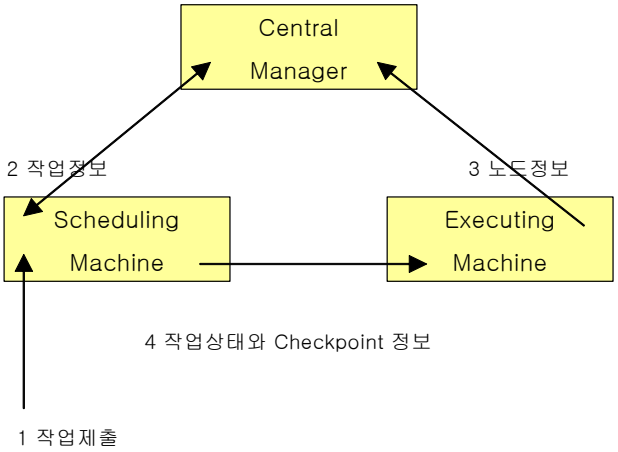
2.3 작업 진행 단계

작업관리 시스템에 작업을 제출하면 일반적으로 <표 1>과 같이 5 단계의 과정을 거쳐 실행된다.

<표 1> 작업 실행 순서

제출 → 대기 → 시작 → 수행 → 종료

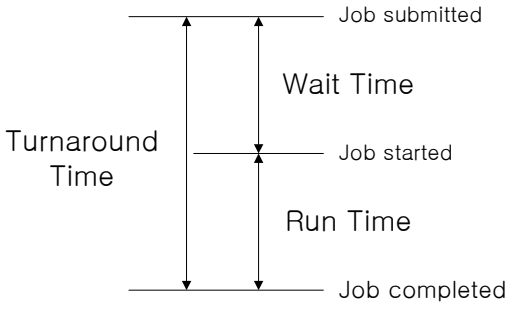
(그림 1)은 LoadLeveler의 구조를 나타낸 것이다. 작업을 제출하면 LoadLeveler의 관리자가 작업정보와 노드정보를 분석하여 적당한 자원을 찾아서 해당 큐에 작업을 할당한다.



(그림 1) LoadLeveler job cycle

2.4 작업 소요 시간

전체 작업 소요시간은 작업의 대기시간+실행시간이므로 실행시간은 컴퓨터의 성능에 따라 이미 정해지기 때문에 작업의 대기시간에 따라 작업수행 효율이 결정되기 때문에 이에 대한 최적화가 필요하다.



(그림 2) 작업실행시간

2.5 Scheduling

LoadLeveler job scheduling에는 크게 두 가지가 있다.

□ Backfill Scheduler

일단 Run Queue에 들어가 수행을 시작하면 종료될 때까지 요청된 CPU를 점유한다. 즉 태스크당 CPU 사용 우선순위는 클래스 구분 없이 동등한 것으로 간주한다. 따라서 CPU 사용효율은 좋으나 수행중인 작업의 클래스별 서비스 레벨 적용은 불가능하다.

<표 2> 큐의 작업 실행 순서

	P1	P2	P3	P4	P5	P6	P7	P8	P9
Run Queue	A	A	A	A	A	B	B	B	B

□ Gang Scheduler

Job Matrix를 구성하여 정해진 순서에 따라 일정한 Time Slice 동안 CPU를 번갈아 점유한다. 이때 Time Slice 할당 개수를 execution_factor라고 하며 1, 2, 3이 가능하다.

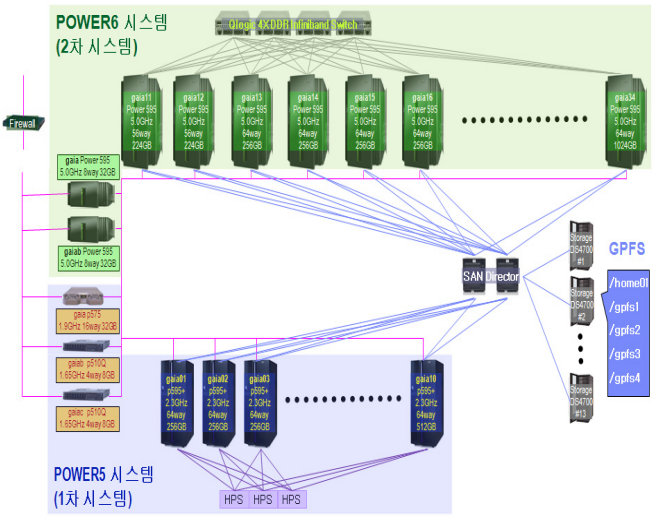
<표 3> 작업의 CPU 할당

Time slice	P1	P2	P3	P4	P5	P6	P7	P8	P9
T1	A	A	A	A	A	A	B	B	B
T2	A	A	A	A	A	A	C	C	C
T3	A	A	A	A	A	A	D	D	D

3. 시스템 구현

3.1 시스템 구성

IBM 시스템 구성은 (그림 3)과 같이 24개의 노드로 구성되어 있으며 외장 디스크는 SAN을 통한 GPFS로 구성하였다.



(그림 3) IBM 2차 시스템 구성도

3.2 클래스 구성

클래스는 크게 시스템의 구분에 따른 p595의 1차

(Power5), 2차(Power5)로 구분하였다 1차 클래스의 특징은 클래스를 CPU 수행시간으로 나누었고 2차 시스템은 CPU 수에 의하여 클래스를 나누었다. 2차 시스템의 클래스의 특징은 노드가 많기 때문에 사용자 작업의 CPU 확보에 중점을 두었다. 하지만 CPU의 효율적인 면에서는 CPU의 전체 사용률이 떨어질 수 있다.

<표 4> Class table

Queue name	가용 CPU수	가용 노드수	Wall_clock_limit	시스템
large	1~448	7	3일	1차
long	1~64	2	10일	
class.1-2	1~2	2	2일	2차
class.2-32	2~32	10	2일	
class.32plus	32~768	12	2일	

3.3 사용자별 작업 우선순위 부여

시스템 사용자 중에서 유료계정과 무료계정으로 나눌 수 있는데 유료계정 사용자에게 작업의 큐 선점 우선권을 부여하기 위하여 사용자별 우선순위 설정을 하였다.

□ 현재

SYSPRIO: (ClassSysprio * 129600) - (QDate)

□ 우선순위 부여

유료계정이 무료계정보다 3일의 작업 우선 순위를 부여하는 공식은 아래와 같다.

SYSPRIO: (UserSysprio*259200) +(GroupSysprio*10) - (QDate)

부여된 숫자는 아래의 공식에 의해서 계산되어 진다.

259200 = 3일 * 24시간 * 60분 * 60초

3.4 실행

<표 5> 1차 시스템의 큐 통계

큐이름	작업개수	평균 CPU 개수	평균대기 시간 (Hour)	평균 Turnaround 시간 (Hour)	평균 CPU시간 (Hour)	평균 T/C
large	55,602	11.6	0.68	1.30	8.58	2.55
long	2,468	3.9	3.07	15.89	32.50	1.43
rokaf	247,444	2.4	0.00	0.06	0.57	1.21
총 합	305,514					

<표 5>는 1차 시스템의 큐 통계이다. long 큐는 large 큐에 지하여 평균 대기시간이 약 3배 긴 것을 알 수 있다. 이는 사용자들이 작업의 수행 시간이 긴 큐를 선호하고 있음을 나타낸다. 그러나 작업이 긴 큐를 많이 두면 시스템의 장애 또는 부팅 시 긴 작업이 종료된다는 문제점을 가지고 있다.

<표 6>은 2차 시스템의 큐 통계이다. 1차 시스템의 평균 CPU 수를 보면 최대 100배의 CPU가 큰 작업을 수행하고 있다는 것을 알 수 있다. 따라서 CPU를 많이 사용자는 작업을 2차 시스템에서 수행하고 있다는 것을 알 수 있다.

<표 6> 2차 시스템의 큐 통계

큐이름	작업개수	평균 CPU 개수	평균대기 시간 (Hour)	평균 Turnaround 시간 (Hour)	평균 CPU시간 (Hour)	평균 T/C
class.1-2	6,267	1.1	0.11	0.67	0.56	1.6
class.2-32	47,376	20.2	0.10	0.67	6.88	1.44
class.32plus	5,886	68.6	6.97	8.51	106.72	5.57
class.post	83	5.1	0.64	28.05	142.90	1.09
rokaf2	103,590	2.1	0.00	0.03	0.44	1.26
총 합	165,680					

4. 결론

IBM 시스템에서 작업관리 시스템을 1차, 2차 각각 구현하고 구현된 기능 중에서 유료계정의 작업 우선순위를 부여한 결과에 대하여 각각 테스트 한 결과 첫 번째 1, 2차 시스템의 클래스 특징이 잘 나타났다. 1차 시스템에서는 1차 시스템에 비하여 약 100배의 CPU 수 작업이 수행된 것을 알 수 있다. 그리고 두 번째 유료계정의 우선권을 부여함으로서 계정 간에 형평성을 유지할 수 있게 되었다.

따라서 IBM 관업관리 시스템을 이용하여 시스템별로 구분하여 클래스를 나누고 계정별로 구분하여 기능을 부여함으로서 사용자에게 더 넓은 선택의 기회를 부여할 수 있는 최적 환경을 구현하였다.

향후에는 더 다양한 작업관리 시스템을 구현하여 비교 분석하고자 한다.

참고문헌

[1] IBM, AIX Wprkload Manager.
[2] IBM, Using and Administering
[3] KISTI, IBM System User Guide, 2006
[4] KIPS, IBM 시스템에서 WLM을 이용한 LoadLeveler 최적 환경 구현, 2007