IBM 멀티 노드에서의 LoadLeveler 최적 작업환경 구현

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

LoadLeveler Optimization Job Environment Implement in IBM Multi Node

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

요 약

다수의 사용자가 사용하는 시스템의 자원을 프로그램을 실행 시 한정된 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, 멀티 노드 시스템에서 이러한 LoadLeveler에서의 클래스를 구분하여 시스템의 이용 효율을 높이고 사용자게 다양한 선택의 폭을 가질 수 있게 최적 환경을 구성하였다.

작업관리 시스템의 주요한 환경변수는 CPU와 메모리 그리고 작업 실행시간이다, 이러한 작업환경 변수에 따라 클래스의 종류를 구분하여 KISTI IBM의 1, 2차 시스템에서 이러한 환경을 사용자의 이용률과 배분정책에 따라 알맞게 설계하여 시스템의 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.

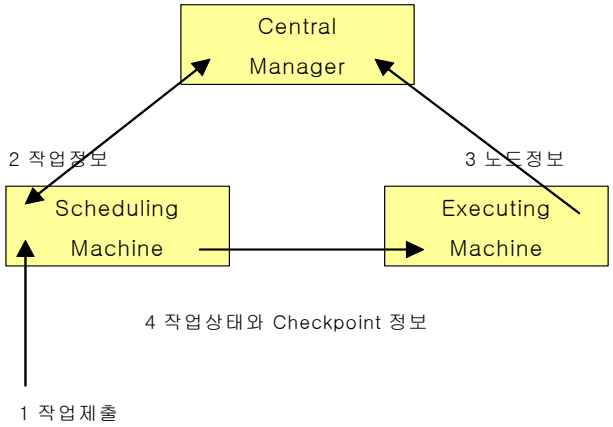
2.3 작업 진행 단계

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

<표 1> 작업 실행 순서

제출 → 대기 → 시작 → 수행 → 종료

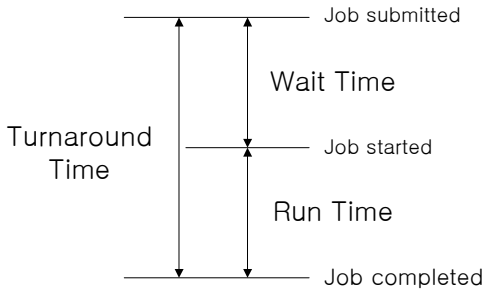
(그림 1)은 LoadLeveler의 구조를 나타낸 것이다. 작업을 제출하면 LoadLeveler의 관리자가 작업정보와 노드정보를 분석하여 적당한 자원을 찾아서 해당 큐에 작업을 할당한다.



(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.

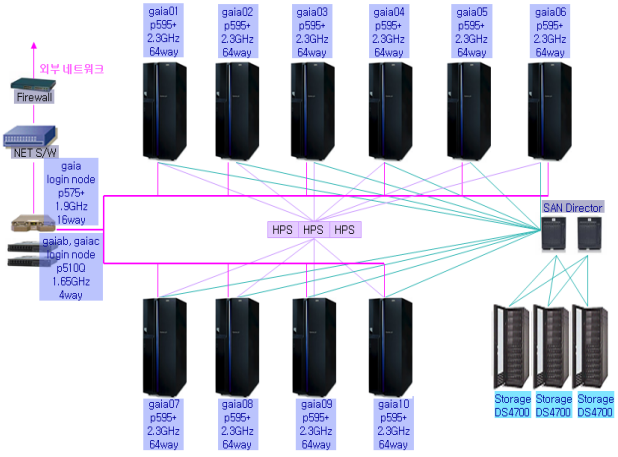
<표 3> 작업의 CPU 할당

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

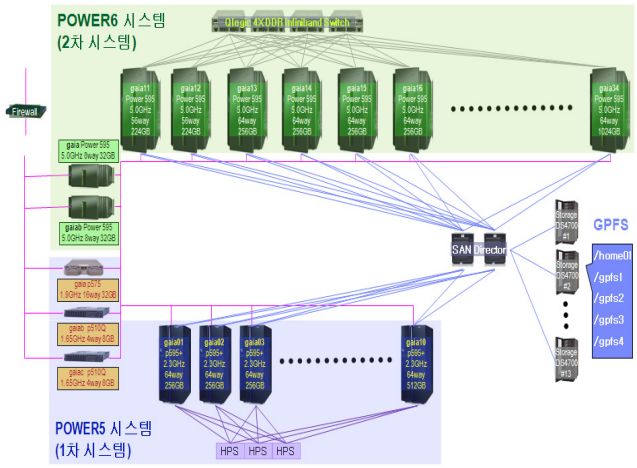
IBM 시스템 1차와 2차 시스템으로 구성되어 있고 1, 2차 구분은 시스템의 도입연도에 따라 구분하였다. 1차 시스템은 p595 시스템으로서 (그림 3)과 같이 10개의 노드로 구성되어 있으며 홈디렉토리는 SAN을 통한 GPFS로 연결되어 있고 사용자 파일의 큰 작업을 위하여 별도의 스크래치 디스크가 3개 GPFS로 연결되어 있다.



(그림 3) IBM 1차 시스템 구성도

2차 시스템도 p595 시스템으로서 (그림 4)와 같이 14개의 노드로 구성되어 있으며 홈디렉토리는 사용자의 편의성을

고려하여 1차 시스템의 홈디렉토리를 공유하고 있고 사용자의 큰 파일 작업을 위하여 별도의 스크래치 디스크가 4개 GPFS로 연결되어 있다. 1, 2차 시스템 각각의 노드에는 별도의 로컬 디스크가 연결되어 있다.



(그림 4) IBM 2차 시스템 구성도

3.2 클래스 구성

IBM 시스템은 큐를 클래스라 부른다. 이러한 클래스의 구성은 크게 1, 2차 시스템의 특성에 맞게 구분하여 구성하였다. 1차 시스템은 클래스를 CPU 수행시간으로 나누었고, 2차 시스템의 클래스는 CPU 수에 의하여 나누었다.

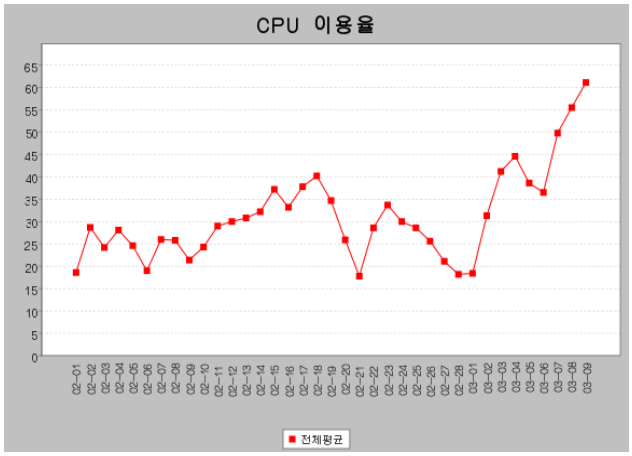
1차 시스템에서처럼 클래스를 수행시간만으로 구분할 경우는 작업은 수행시간에 따라 분류된 큐를 할당 받는데 이럴 경우 시스템의 CPU 이용률을 높일 수 있고, 2차 시스템처럼 CPU의 수에 의하여 구분할 때는 CPU 수에 의해서 클래스를 할당받게 되는데 이럴 경우는 사용자 작업의 많은 CPU 수 확보에 중점을 두었다. 1, 2차 시스템의 클래스 구성은 시스템 CPU의 이용률과과 사용자 편의성을 고려하여 서로 보완적인 효율적으로 구성하였다.

<표 4> Class table

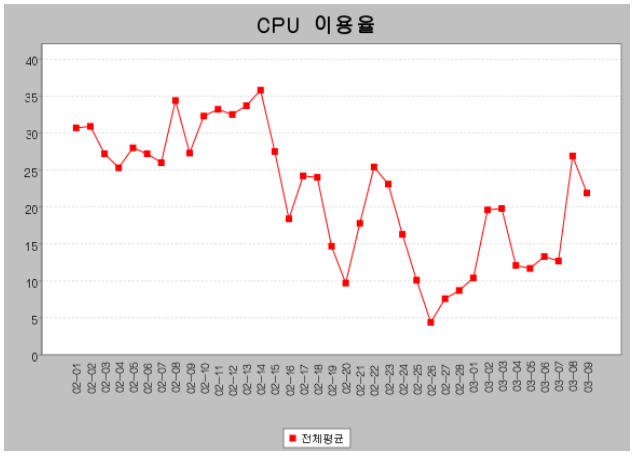
Queue name	가용 노드수	가용 CPU수	Wall_clock_limit	시스템
large	7	1~448	3일	1차
long	2	1~64	10일	
class.1-2	2	1~2	2일	2차
class.2-32	10	2~32	2일	
class.32plus	12	32~768	2일	

4. 실행결과

(그림 5)와 (그림 6)은 같은 기간 내의 1차 시스템과 2차의 시스템 각각의 CPU 이용률이다. 각각의 통계 표를 보면 CPU의 이용률이 서로 보완적인 것을 알 수 있다. 이것은 사용자들의 시스템 사용 패턴이 서로 다르다는 것을 알 수 있다.



(그림 5) IBM 1차 시스템 CPU 이용률



(그림 6) IBM 1차 시스템 CPU 이용률

5. 결론

IBM 전체 시스템에서 작업 큐를 구성할 때 큐의 구분을 전체 노드의 활용성을 고려하여 서로 다르게 구현한 결과 각각의 시스템의 CPU의 이용률이 서로 보완적으로 사용되는 것을 알 수 있었다. 따라서 전체 시스템을 효율적으로 활용할 수 있었다.

향후에는 통계기간을 더 길게 하여 각각의 큐에 대한 상세 시스템의 이용률을 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
[5] KIPS, IBM 시스템의 LoadLeveler LoadLeveler 최적 환경 구현, 2010