제18회 한국정보처리학회 추계학술발표대회 논문집 제9권 제2호 (2002.11)

IBM p690 시스템에서 LoadLeveler를 사용한 Gang Scheduling과 Backfilling Scheduler 성능 분석

우준, 김중권, 이상산
한국과학기술정보연구원 슈퍼컴퓨팅센터
e-mail:wjnadia@hpcnet.ne.kr

An Performance Analysis for Gang Scheduling, and Backfilling Scheduler with LoadLeveler at the IBM p690

Joon Woo, Joong Kwon Kim, Sang San Lee
SuperComputing Center, Korea Institute of Science and Technology Information

요 약

분산 병렬 시스템에서 사용되는 배치 작업 스케줄링 기법으로 잘 알려진 것은 Gang Scheduling과 Backfilling Scheduler가 있다. 특히 IBM SP 시스템에서 주로 사용되는 작업 스케줄러인 LoadLeveler 최신 버전에서는 이전 버전에서도 지원하였던 Backfilling Scheduler 뿐만 아니라 Gang Scheduling 기법을 새롭게 지원하게 되었다. 이에 따라 KISTI 슈퍼컴퓨팅센터에서는 슈퍼컴퓨터 3호기로 신규 도입된 IBM p690 시스템에서 LoadLeveler의 Gang Scheduling 혹은 Backfilling Scheduler 중의 한 가지 기법을 선택하여 서비스 레벨 클래스를 구현하고자 하였다. 이러한 노력의 일환으로 두 가지 스케줄링 기법을 테스트 및 분석하였다. 이에 따르면 Gang Scheduler가 개념상 여러 가지 장점을 가지므로 Backfilling Scheduler에 비하여 서비스 레벨 클래스 구성에는 용이하지만, 불완전한 구현 및 특히 CPU Utilization이 저하되는 심각한 문제점을 가지고 있었다. 따라서 Backfilling Scheduler를 통한 제한적인 서비스 레벨 클래스를 구성하기로 결론지었다.

1. 서론

배치 시스템은 작업을 제출, 실행 및 공유 자원 정보를 추적 관리하며, 분산 시스템 환경에서 분산 자원에 대한 중앙 집중화된 접근을 가능하게 한다. 이러한 기능은 작업 관리와 전체 가용 계산 자원 측면에서 사용자에게 단일 시스템 이미지를 가능하게 하여 분산 자원의 활용을 상당히 간편하게 하지만, 분산 시스템의 글로벌한 관점 이상의 기능을 제공해야 한다.

많은 공유 시스템에서와 같이 공평하고 효율적으로 계산 자원을 활용하고자 할 때 복잡성이 발생한다. 이러한 복잡성은 성능의 저하와 불공평한 자원의 활용 문제를 야기한다.

배치 시스템에서 스케줄러는 분산 시스템의 활용을 극대화하기 위해서 작업을 할당받아 어느 시점에, 어떤 계산 자원을 할당하여, 어떻게 실행해야 하는지를 결정한다. 이러한 스케줄러의 결정은 다음과 같이 크게 세 가지 영역으로 구분된다.

■ 트래픽 제어(Traffic Control)

스케줄러는 작업간 충돌이 발생하지 않도록 중재하는 역할을 수행해야 한다. 일반적으로 작업들의 자원 경쟁을 허용하는 경우에는 분산 시스템의 성능 저하, 작업 실행 지연, 하나 혹은 그 이상 작업의 실행 불능 등의 문제를 야기한다. 이에 따라 스케줄러는 작업에 요청된 자원을 추적, 전용하여 다른 작업이 이러한 자원을 사용하지 못하게 한다.

■ 임무 정책(Mission Policies)

클러스터 혹은 HPC 플랫폼을 구성할 때는 한가지 이상의 목적을 가지며, 이것은 종종 어떻게 시스템이 사용되어야 하고, 누구에게 혹은 어떻게 사용하도록 허용해야 하는 지에 대한 다양한 원칙을 정의한다. 효율성을 위해서 스케줄러는 사이트 임무 정책을 스케줄링 행위와 연계하게 하는 일련의 정책을 제공해야 한다.

■ 최적화(Optimization)

분산 시스템의 자원은 제한되어 있기 때문에 자원에 대한 요구는 항상 공급을 초과한다. 지능화된 스케줄링은 더 많은 작업을 더욱 빠른 turnaround 시간 안에 실행하게 하여 분산 시스템의 자원을 효율적으로 사용할 수 있게 한다.

이 논문에서는 위에서 설명한 배치 스케줄러 시스템으로 IBM SP 시스템에서 주로 사용되는 LoadLever를 기반으로 하여 Gang Scheduling과 Backfilling Scheduler 기법을 비교 테스트하고 성능을 분석하였다. KISTI 슈퍼컴퓨팅센터에서는 이러한 성능 분석을 통하여 2002년 1월에 도입된 슈퍼컴퓨터 3호기인 IBM p690 시스템의 배치 스케줄러 시스템을 최적 구현하는데 활용하였다. 논문의 구성은 2장에서 2가지 대표적인 스케줄링 기법을 설명하고, 3장에서 스케줄링 기법을 비교 테스트하고 성능을 분석하며, 4장에서 결론을 맺는다.

2. 스케줄링 기법

대부분의 다중 상용 시스템들은 각 스케줄링 결정이 미래에 예측 불가능한 영향을 미치는 공간 분할 기법을 사용한다. 이 기법은 '나중에 제출될 다른 더 큰 작업을 대기 상태에 머물게 하면서까지 현재 작업을 스케줄링해야 하는지?', 혹은 '그렇다고 나중에 제출될 작업을 위해서 현재의 프로세스 중의 일부를 idle 상태로 두어야 하는지?' 등과 같은 딜레마를 가져왔다. 이러한 딜레마를 해결하기 위한 대한 대안으로 Gang Scheduling과 Backfilling Scheduler 기법을 사용할 수 있으며, 이러한 기법을 구현하려는 다양한 시도가 있어 왔다. 이 장에서는 이 두 가지 기법에 대한 소개를 하고자 한다.

2.1 Gang Scheduling

Gang Scheduling은 다수의 프로세스가 하나의 CPU를 시간 분할하여 공유 수행하는 방식을 취한다. 각 스케줄링 결정에 따른 영향이 할당된 time slice에만 한정되며, 미래에 제출될 작업은 다른 time slice를 할당받아 실행된다. 또한 이러한 부가된 유연성은 전체 시스템의 이용률 및 응답 속도를 향상시키는 결과를 보여줄 수 있다.

2.2 Backfilling Scheduler

Backfilling Scheduler는 작은 작업이 큐에서 앞서 있는 다른 큰 작업의 실행 시간 전에 종료될 수 있는 경우에, 큰 작업 보다 먼저 스케줄링 되어 실행하도록 허용하는 스케줄링 기법이다. 또한 하나의 CPU를 하나의 프로세서만이 전용하여 실행하는 방식을 택하고 있다. 하지만 각 작업의 실행 시간 [wall_clock_time]에 대한 정확한 정보를 알고 있어야만 한다는 조건이 따른다.

3. 스케줄링 기법 비교 테스트 및 성능 분석

KISTI 슈퍼컴퓨팅센터에서는 2002년 1월에 슈퍼컴퓨터 3호기로 IBM p690 시스템을 도입하였으며, 배치 스케줄러 시스템으로 LoadLeveler V.3.1을 채용하였다. 이 버전의 LoadLeveler는 앞에서 설명한 Gang Scheduling과 Backfilling Scheduler 두 가지 스케줄링 기법 모두를 지원한다. 이에 따라 최적의 배치 스케줄러 구성 방안을 도출하고자 이 두 가지 스케줄링 기법을 테스트하고, 각 기법의 성능을 비교 분석하였다. 특히 서비스 레벨에 따라 차등화된 클래스 스케줄링 우선 순위 및 실행 응답 시간을 보장하는 서비스 레벨 클래스[1]의 구성의 기초 자료로 활용하는 데 목적이 있다.

3.1 테스트 환경

테스트 환경은 IBM의 p690 시스템을 기반으로 하였으며, Gang Scheduling과 Backfilling Scheduler 각각에 대하여 다음과 같은 테스트 환경을 구축하였다.

3.1.1 Gang Scheduling

■ 시스템 구성 : 각 시스템 별 32CPU를 장착한 IBM p690 시스템 3대를 16CPU의 LPAR[2]* 노드로 재구성하여 총 6노드로 구성됨.

1) 클래스는 가용한 자원의 종류 및 필요량, 스케줄링 선택사항 등 구체적인 특성을 부여한 노드들의 집합이다.

2) LPAR[Logical Partitioning]는 하나의 물리적 시스템을 여러 개의 논리적 시스템으로 나누어 운영하는 것을 말하며, 각각의 파티션 노드는 독립된 운영체제, 프로세서, 메모리, I/O 장치 등을 가지게 된다.

■ 실행 프로그램 : p690 시스템 사용자에 의해 다양한 유형의 프로그램이 수행됨.

■ 클래스 구성

<표 1> Gang Scheduling을 이용한 클래스 구성

클래스 이름	서비스 레벨	Priority	Execution Factor	preemption 관계
p_express	1.5	2	2	p_normal 작업 선점 가능
p_normal	1	1	1	p_express 작업에 의해 선점 될 수 있음

※ Execution Factor : Time Slice 할당 비율 변수

※ preemption 관계 : 이미 수행되고 있는 작업의 자원을 선점하기 위한 관계 정의.

3.1.2 Backfilling Scheduler

■ 시스템 구성 : 32CPU를 장착한 IBM p690 시스템 1대로 구성됨.

■ 실행 프로그램 : 32CPU를 사용하는 MPI 프로그램 실행.

■ 클래스 구성

<표 2> Backfilling Scheduler를 이용한 클래스 구성

클래스 이름	서비스 레벨	Priority
p_express	1.5	2
p_normal	1	1

3.2 비교 테스트 및 성능 분석

본 비교 테스트에서는 두 가지 스케줄러의 성능을 비교하기 위한 기준으로 실행 응답 시간 보장을 위한 서비스 레벨 클래스 구현의 용이성 뿐만 아니라, 시스템 자원의 효율적 활용 기준인 CPU Utilization을 포함하였다. 다음은 이러한 기준에 따른 테스트 및 분석결과를 설명하고 있다.

3.2.1 서비스 레벨 클래스 구현 용이성

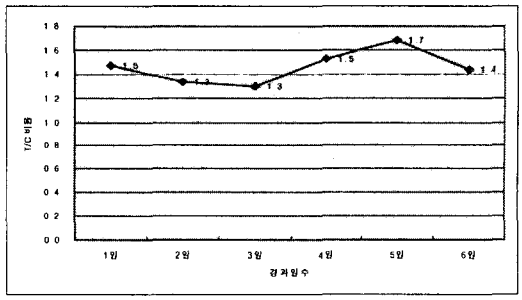
서비스 레벨에 따른 클래스 구현의 용이성을 판단하기 위해서 다음과 같은 수식을 사용하였다.

T/C=[Average Turnaround Time]/[Average CPU Time]

위와 같은 수식에 의해서 산출된 p_express와 p_normal T/C 값의 비율이 1:1.5 정도가 유지될 수 있다면 서비스 레벨 클래스의 구현이 용이한 것으로 결론 지을 수 있다. (그림1), (그림2)의 그래프는 p_express를 기준으로 한 p_normal의 T/C 값 비율의 일별 변화를 보여주고 있다.

■ Gang Scheduling

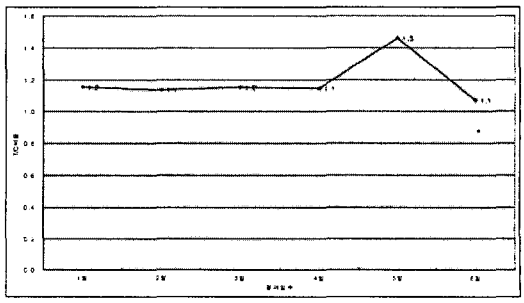
Gang Scheduling은 preemption과 execution factor를 사용하여 이미 실행중인 작업에 대한 제어가 가능하므로, (그림1)에서와 같이 서비스 레벨의 구현이 가능할 것으로 보였다. 하지만 preemption 및 execution factor의 불완전한 구현을 보완해야 하는 문제점을 가지고 있다.



(그림 1) Gang Scheduling에서 T/C 비율 그래프

■ Backfilling Scheduler

(그림 2)에서와 같이 Backfill Scheduler에서는 일정한 T/C 비율의 유지가 가능할 것으로 보인다. 하지만 각 클래스의 priority는 클래스에서의 대기 시간만을 제어하고 실행중인 잡업의 제어가 불가능하므로, 경우에 따라서는 T/C 비율의 제어가 용이하지 않을 수도 있다.



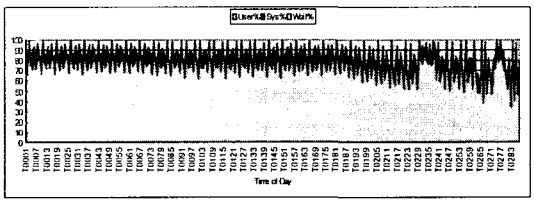
(그림 2) Backfilling Scheduler에서 T/C 비율 그래프

3.2.2 CPU Utilization

다음의 CPU Utilization 그래프는 두 가지 스케줄링 기법 모두 최대 실행 가능한 작업 수를 초과 제출한 경우의 통계 데이터를 보여주고 있다.

■ Gang Scheduling

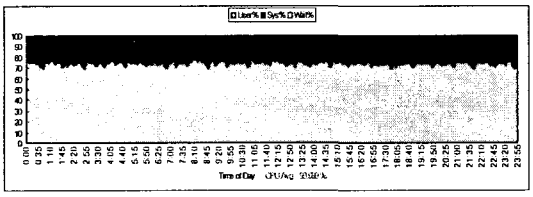
(그림 3)에서 일정한 주기를 가지고 CPU Utilization의 증감이 반복되며 이로 인하여 전체 CPU Utilization이 감소되었음을 확인할 수 있었다. 이것은 할당 할당된 Time Slice의 수행을 완료하고 다른 작업으로 전환하는 데 따른 일시적인 CPU idle 상태로 인하여 발생한다.



(그림 3) Gang Scheduling에서의 CPU Utilization 그래프

■ Backfilling Scheduler

(그림 4)는 병렬작업의 효율성에 따라서 정도의 차이를 보일 수 있지만, CPU Utilization을 상당히 높게 유지할 수 있음을 확인할 수 있다.



(그림 4) Backfilling Scheduler에서의 CPU Utilization 그래프

4. 결론

본 논문에서는 Gang Scheduling과 Backfilling Scheduler를 서비스 레벨 클래스 구현 용이성 및 CPU ·Utilization이라는 측면에서 테스트 및 분석하였다.

Gang Scheduling은 Backfilling Scheduler에 비하여 서비스 레벨 클래스 구현에 좀더 용이한 장점을 가지고 있지만, 아직은 구현이 불완전한 점과 특히 CPU Utilization이 저하되는 치명적인 약점을 가지고 있었다. 이에 따라 자원의 효율적 활용에 상대적인 장점을 가지는 Backfilling Scheduler를 적용하여 제한적인 서비스 레벨 클래스를 구현하기로 결론지었다.

하지만 향후 완전한 서비스 레벨 클래스 구현을 위하여 IBM LoadLeveler의 Gang Scheduling의 미비점을 보완하여야 하며, Maui 및 Catalina Scheduler와 같은 다른 배치 스케줄러 시스템도 테스트할 필요가 있다.

참고문헌

[1] Ahuva W. Mu'alem and Dror G. Feitelson, "Utilization, Predictability, Workloads, and User Runtime Estimates in Scheduling the IBM SP2 with Backfilling", IEEE Transactions on parallel and distributed systems, Vol.12, No.6, June 2001

[2] Dror G. Feitelson, Larry Rudolph, Uwe Schwiegelshohn, Kenneth C. Sevcik, and Parkson Wong, "Theory and Practice in Parallel Job Scheduling"

[3] Dror G. Feitelson and Morris A. Jette, "Improved Utilization and Responsiveness with Gang Scheduling"

[4] Dror G. Feitelson, " Metrics for Parallel Job Scheduling and their Convergence"

[5] IBM, " Using and Administering LoadLeveler Ver.3.1", First Edition, Dec. 2001