IBM p690에서 작업 스케줄링 알고리즘에 따른 시스템 효율성 벤치마크

우준, 김중권
KISTI 슈퍼컴퓨팅센터
e-mail:wjnadia@kisti.re.kr

A System Effectiveness Benchmark for Job Scheduling Algorithms on the IBM p690

Joon Woo, Jung-Kwon Kim
Supercomputing Center, KISTI

요 약

ESP는 Effective System Performance의 약자로 NERSC에서 개발한 HPC 시스템에 대한 새로운 성능 측정 기준이다. 기존 HPC 시스템에서는 주로 성능 측정의 대상으로 시스템[프로세서]의 계산 성능에 주안점을 두었지만 시스템의 효율성은 무시되는 경향이 있었다. ESP는 실제 운영환경에서 배치 작업 스케줄러 및 병렬 작업 환경에 영향을 받는 시스템 효율성(ESP:Effective System Performance)을 측정하는 데 주안점을 두고 있다. KISTI 슈퍼컴퓨팅센터는 2003년 7월 국내 최고 성능의 슈퍼컴퓨터인 IBM p690+ 시스템의 도입을 완료하고 ESP를 사용하여 배치 작업 스케줄러인 LoadLeveler의 스케줄링 알고리즘에 따른 시스템 효율성 벤치마크를 수행하였다. 이 벤치마크를 통해서 효율적인 시스템 자원 활용을 위한 작업 스케줄링 알고리즘의 적용 근거를 마련하게 되었다.

1. 서론

기존 HPC 시스템에서는 주로 성능 측정의 대상으로 시스템[프로세서]의 계산 성능에 주안점을 두었지만 시스템의 효율성은 무시되는 경향이 있었다. 하지만 다수의 노드로 구성된 대규모 HPC 시스템의 실제 운영환경에서는 병렬 작업 환경 및 배치 작업 스케줄러의 효율성이 시스템의 전반적인 성능에 많은 영향을 끼치는 것이 사실이다. 이에 따라 지금까지 구체적인 성능 측정 기준이 없었던 시스템 효율성을 NERSC(National Energic Research Supercomputing Center)에서 개발한 ESP라는 측정 도구를 사용하여 KISTI 슈퍼컴퓨팅센터에 신규 도입된 국내 최대 규모 슈퍼컴퓨터인 IBM p690+ 시스템에서 배치 스케줄링 알고리즘에 따른 시스템 효율성 벤치마크를 수행하였다.

2. 시스템 효율성

본 연구에서는 시스템의 효율성을 측정하기 위한 도구로 ESP를 사용하였으며, 시스템 효율성에 많은 영향을 미치는 배치 작업 스케줄러로 LoadLeveler를 사용하고 있다. 이 장은 ESP와 LoadLeveler 및 ESP를 사용하여 수행된 벤치마크 결과에 따른 시스템 효율성 계산 방식을 설명하고 있다.

1) ESP

ESP는 Effective System Performance의 약자로

NERSC에서 개발한 새로운 성능 측정 기준이다. ESP는 실제 운영환경에서의 시스템 효율성(ESP:Effective System Performance)을 측정하는데 주안점을 두고 있다.

ESP는 주로 운영 시스템의 특성에 기반을 두고 있는 production 기반 병렬형 시스템을 위한 성능 측정 기준을 제공하도록 고안되었다. 이러한 특성은 병렬작업의 수행 시간, 작업 스케줄링과 선점형 작업 수행을 포함하고 있다. ESP 테스트의 주요 목적은 최소 경과 시간 내에 배치 스케줄러를 통해서 고정된 숫자의 병렬 작업을 수행하는 것이다.

또한, 작업들의 경과 시간이 고정된 목표 실행 시간에 근접할 수 있도록 특별히 구성되어 있다. 그래서 전체 테스트 경과 시간은 프로세서 속도와는 무관하고 주로 스케줄러의 효율성과 병렬 작업의 수행에 따른 오버헤드에 의해서 결정된다. 테스트에서는 14가지 작업 유형중에서 추출된 230개의 작업이 있다.

실제로 제출된 각각의 작업은 기본적으로 다음과 같은 일련의 통신 및 연산 작업을 반복하도록 구성되어 있다. 다만 각각의 작업은 서로 다른 목표 수행 시간을 가지고 이 시간이 도달할 때까지 수행된다. 각각의 타스크는 랜덤 메시지 문자열을 생성하고 digest를 계산한다.

Z ----> task ----> Y

각 타스크는 랜덤 메시지를 Y에게 보내고 유사한 메시지를 Z로부터 수신한다. Y&Z의 랭크는 랜덤하게 생성된다. 각각의 수신된 메시지로 타스크는 초기에 랜덤 메시지를 가지고 있던 메시지 버퍼에서 XOR 연산을 한다.

그리고 위 연산을 반복하지만 Y와 Z의 랭크 순서를 바꾸어 거꾸로 반복한다. 마지막 XOR 연산 후에 업데이트된 메시지는 정확하게 처음 메시지와 동일해야 한다. 아니면, 다이제스트가 일치해야 한다. 다이제스트가 일치하면 상태를 리턴한다.

이 코드는 의미 있는 작업을 수행하지는 않지만, 통신 서브 시스템에 부하를 가하고 실행의 정확성을 보장한다.

2) LoadLeveler

IBM SP 시스템에서 주로 사용되는 배치 작업 스케줄러인 LoadLeveler는 스케줄링 알고리즘으로 Backfilling Scheduler 뿐만 아니라 Gang Scheduling 기법을 새롭게 지원하게 되었다.

다음은 효율성 테스트에서 주요 비교 대상 알고리즘인 Gang Scheduling과 Backfilling Scheduler에 대하여 설명하고 있다.

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 Backfilling Scheduler

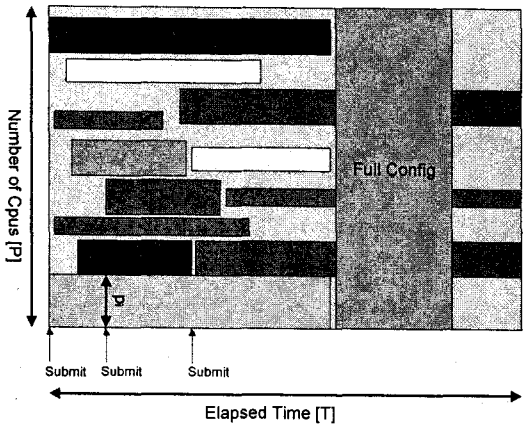
Backfilling Scheduler는 작은 작업이 큐에서 앞서 있는 다른 큰 작업의 실행 시간 전에 종료될 수 있는 경우에, 큰 작업 보다 먼저 스케줄링 되어 실행하도록 허용하는 스케줄링 기법이다. 또한 하나의 CPU를 하나의 프로세서만이 전용하여 실행하는 방식을 택하고 있다. 하지만 각 작업의 실행 시간 [wall_clock_time]에 대한 정확한 정보를 알고 있어야만 한다는 조건이 따른다.

3) 시스템 효율성

ESP 수행을 통한 시스템 효율성[E]은 다음과 같이 계산된다. 계산결과로 E는 '1'에 근접할수록 작업 수행효율성이 좋은 시스템이다. (그림 1)은 ESP를 통해서 수행되는 작업들이 주어진 전체 CPU 자원을 얼마나 효율적으로 사용하고 판단하기 위한 각 변수들의 의미를 도식적으로 설명하고 있다.

$$E = \frac{p_1 t_1 + p_2 t_2 + \ldots + p_n t_n}{PT}$$

pi = number of processors utilized by job
ti = wall clock run time in seconds required by job i on a dedicated system
n = number of individual jobs run during the test
P = total number of processors in the system
T = total elapsed wall clock time for test (not counting shutdown and reboot)



(그림 1) 시스템 효율성 벤치마크

3. 벤치마크 수행 및 결과

1) 벤치마크 환경

① 시스템 구성

테스트 시스템은 2003년 7월에 도입된 국내 최고 성능의 1.7GHz POWER4 CPU로 구성된 IBM 32-way p690+ 12대로 이루어지며, 각 노드는 최소 196GB 이상의 주기억장치 및 2TB 이상의 로컬 스크래치 파일 시스템을 지원한다. 또한, 각 노드는 두 채널씩의 SP SW2에 연결되어 전체 시스템은 클러스터링되어 있다.

② LoadLeveler 클래스 구성

<표 1>은 본 벤치마크 수행을 위해서 구성될 LoadLeveler의 스케줄링 알고리즘에 따른 클래스 구성의 차이점을 설명하고 있다.

<표 1> LoadLeveler 클래스 구성

	Backfilling	Gang Scheduling
bmt	exp_bmt 보다 낮은 우선순위를 부여하고 wall_clock_time을 target run time에 근접하게 설정한다.	exp_bmt에 의해서 선점당한다.
exp_bmt	bmt 큐 보다 높은 우선순위를 부여하고 예상시간보다 wall_clock_time을 상당히 크게 조절한다.	bmt의 모든 리소스를 선점한다.

③ ESP 작업 구성

테스트 시스템의 전체 CPU 수는 384개이며 <표 2>는 각 작업 유형에 따라 할당될 CPU 및 노드 수와 큐 등에 대한 구성 정보를 보여주고 있다.

<표 2> ESP 작업 구성

Job-type	Fract-Size	Size	Nodes	Queue	Count	Target Run Time
A	0.03125	12	1	bmt	75	257
B	0.06250	24	1	bmt	9	341
C	0.50000	192	6	bmt	3	536
D	0.25000	96	3	bmt	3	601
E	0.50000	192	6	bmt	3	312
F	0.06250	24	1	bmt	9	1846
G	0.12500	48	2	bmt	6	1321
H	0.15820	60	2	bmt	6	1078
I	0.03125	12	1	bmt	24	1438
J	0.06250	24	1	bmt	24	715
K	0.09570	36	2	bmt	15	495
L	0.12500	48	2	bmt	36	369
M	0.25000	96	3	bmt	15	192
Z	1.00000	384	12	exp_bmt	2	100
Total					230	

2) 벤치마크 방법

테스트는 14가지 유형을 가지 총 230개의 작업에 대하여 전체 시스템 자원을 사용하는 2개의 작업을 제외하고는 난수 생성기에 의하여 정해진 순서대로 배치 작업 스케줄러인 LoadLeveler에 제출하는 것으로 수행된다.

또한, 테스트는 <표 2>의 Z 작업을 배제한 throughput mode와 Z 작업을 포함한 multimode로 구분하여 수행된다.

3) 벤치마크 결과

<표 3>은 Z작업을 수행하지 않는 throughput mode에서의 ESP 수행결과로 Backfilling에 의하여 Backfilling Scheduler가 근소한 우위를 보이고 있다.

<표 4>는 전체 CPU 자원을 요구하는 Z작업이 함께 수행되는 multimode에서의 ESP 수행결과로, Z 작업의 효율적 수행을 위해서 Gang Scheduling이 Preemption을 통하여 기존 작업의 CPU 자원 전부를 선점하므로 전체적인 시스템 효율성에서 Backfilling에 비하여 상당히 우위에 있다.

① throughput mode

<표 3> throughput mode에서의 테스트 결과

	Backfilling Scheduler	Gang Scheduling
가용 프로세서수:P[개]	384	384
작업 경과 시간:T[초]	15,659	15,780
PT	6,013,056	6,059,520
전체 작업 수행 시간:[초] $\sum_{i=0}^{228} Pit_i$	4,152,321	4,150,152
시스템 효율성 지수	0.69	0.68

② multimode

<표 4> multimode에서의 테스트 결과

	Backfilling Scheduler	Gang Scheduling
가용 프로세서수:P[개]	384	384
작업 경과 시간:T[초]	16,955	15,787
PT	6,510,720	6,062,208
전체 작업 수행 시간:[초] $\sum_{i=0}^{230} Pit_i$	4,201,678	4,301,950
시스템 효율성 지수:E	0.65	0.71

4. 결론

본 연구에서는 IBM p690 시스템상에서 ESP를 통한 새로운 형태의 시스템 효율성 벤치마크를 수행하였다. 벤치마크 결과를 고찰하였을 때, 배치 스케줄러인 LoadLeveler의 스케줄링 알고리즘으로 Gang Scheduling은 기존 자원의 선점이 요구되는 환경에서 월등한 우위에 있음을 파악할 수 있었고, 자원 선점이 요구되지 않는 일반적인 작업 환경에서는 Backfilling이 근소한 차이로 우위에 있음을 알 수 있었다. 이러한 결과를 근거로 IBM p690 시스템 자원의 효율적 활용을 위한 작업 스케줄링 알고리즘의 적용 근거를 마련할 수 있게 되었다.

향후 본 연구에서 확장하여 KISTI 슈퍼컴퓨팅센터가 가지고 있는 다양한 시스템 및 배치 스케줄러 하에서의 ESP 벤치마크를 수행하여 전반적인 시스템 간의 효율성을 비교 분석할 필요가 있다.

참고문헌

[1] Adrian Wong, Leonid Oliker, William Kramer, Teresa Kaltz and David Bailey, System Utilization Benchmark on the Cray T3E and IBM SP, The 6th Workshop on Job Scheduling Strategies for Parallel Processing, April, 2000

[2] Adrian T.Wong, Leonid Oliker, William T. C. Cramer, Teresa L. Kaltz and David H. Bailey, Evaluating System Effectiveness in High Performance Computing Systems, 1999

[3] Adrian T. Wong, Leonid Oliker, William T. C. Kramer, Teresa L. Kaltz and David H. Bailey, ESP: A System Utilization Benchmark, IEEE, 2000

[4] 우준, 김중권, 이상산, IBM p690 시스템에서 LoadLeveler를 사용한 Gang Scheduling과 Backfilling Scheduler 성능 분석, 추계 정보처리학회 학술발표대회, 제9권, 제2호, 2002

[5] NERSC ESP Guide
:http://www.nersc.gov/aboutnersc/esp.html