SMP 슈퍼컴퓨터에서의 집합 IO 성능

차광호, 김성호, 이식
한국과학기술정보연구원 슈퍼컴퓨팅본부
e-mail : khocha@kisti.re.kr

Performance evaluation of collective I/O on an SMP supercomputer

Kwangho Cha, Sungho Kim, Sik Lee
Supercomputing Center, Korea Institute of Science and Technology Information

요 약

멀티 코어 또는 매니 코어 기반의 HPC 시스템 보급이 늘어나면서 HPC 어플리케이션이 사용하는 프로세스의 수 또한 증가하고 있다. 이런 경우, 기존의 IO 방식이 아닌 병렬 IO 의 사용을 고려하여야 하는데 그 중 특히 집합 IO 는 중요한 역할을 수행한다. 본 연구에서는 IBM Power 595 기반 슈퍼컴퓨터에서 집합 IO 특성을 알아 본다.

1. 서론

병렬 시스템에서 동작하는 많은 응용 프로그램들은 아직까지도 기존의 단일 IO 방식을 사용하여 IO 를 수행하고 있다. 그러나 프로세서 수와 시스템 규모가 증가될수록 기존 IO 방식의 사용에는 제약이 따르게 된다. 이를 고려하여 병렬 파일시스템에 보다 효과적인 병렬 IO 기법이 제안되었는데 MPI 환경에서 사용하는 MPI-IO 가 그 예이다[1,2].

특히 집합 IO(Collective IO)의 경우, 단일 파일 기반의 병렬 IO 를 지원하여, 성능과 관리의 효율성을 증대시키고 파일의 프로세스 수에 대한 의존성을 제거하였다[2]. 이처럼 대규모 프로세스 환경에서 적합하도록 설계된 집합 IO 의 성능을 본 연구에서 살펴보았다.

GPFS 를 사용하는 IBM 슈퍼 컴퓨터에서 프로세스 할당 방식과 파일시스템의 특징을 변경하면서 집합 IO 성능과의 연관 관계를 알아 보았다. 본 논문의 구성은 다음과 같다. 2 장에서는 MPI-IO 의 개념과 실험에 사용된 시스템을 3 장에서는 실험 환경을 보다 자세히 살펴보며, 4 장에서는 실험 결과에 대하여 설명한다. 5 장에서는 실험 결과가 지니는 의미를 살펴본다.

2. 관련 연구

2.1. MPI-IO

MPP(Massively Parallel Processing)시스템이나, Cluster System 등에서 많이 이용되는 MPI2 (Message Passing Interface Ver.2)는 병렬 IO 를 위한 IO 데이터 접근 방식을 지원하고 있다. 병렬 파일 시스템 환경에서는 병렬 프로세스가 단일 파일을 접근하는 경우, 각 프로세서들은 각자 파일 포인터를 갖게 되는데 이를 MPI-IO 에서는 유도 데이터형을 이용하여 사전에 정의할 수 있다. 이러한 유도 데이터형과 IO 를 동시에 수행하는 집합 IO 를 프로그램 작성시 선택적으로 적용할 수 있다[1].

MPICH 등의 대표적인 MPI 라이브러리는, 집합 IO 의 성능 향상을 위하여 2 phase IO 방식으로 구현되어 있다. HPC 분야의 응용 프로그램의 경우, 각 프로세스는 작은 크기의 IO 요청을 빈번하게 요청하는 특징을 가진다. 이때, 계산 노드 중 선택된 IO 집합자(IO Aggregator)가 이러한 요청들을 취합하여 처리하도록 하여 IO 데이터의 크기를 증가시키고 IO 요청 빈도는 낮추어 성능 개선을 가능하도록 한 것이 2 phase IO 이다. 이때 IO 집합자는 계산 노드의 역할도 동시에 수행하게 된다[2].

2.2. SMP 슈퍼컴퓨터

본 연구에서 사용된 슈퍼컴퓨터는 Power6 기반의 64way 시스템을 단위 노드로 사용하고 있으며, 노드들을 연결하는 주 네트워크는 인피니밴드를 이용하였다. 단위 계산 노드로 사용된 Power6 기반의 P595 시스템은 듀얼 코어로 구성된 CPU 4 개가 하나의 Processor book 을 구성하고 8 개의 Processor book 이 하나의 시스템을 구성하게 된다[3,4].

이처럼 SMP 시스템을 계산 노드로 이용하는 경우에는 CPU 내부, CPU 간, 시스템 내부 그리고 시스템 간의 통신 성능에 차이를 갖게 되므로 CPU 를 할당하는 방식에 따라 성능 차이가 발생할 수 있다. 본 연구에서는 이와 같은 사항을 고려하여 CPU 할당에 차이를 두어 집합 IO 의 성능을 분석하였다.

3. 실험 환경

3.1. 시스템 환경

집합 IO 의 성능을 분석하는데 사용된 하드웨어와

소프트웨어의 구성은 표 1 과 같다.

본 연구에서는 가능한 한 노드 안에 위치하는 프로세서를 사용하는 경우와, 일정 수의 계산 노드에서 균등하게 프로세서를 사용하는 경우의 집합 IO 성능을 측정하고 분석하였다. 단위 노드 내에 프로세서를 할당하는 경우, 프로세서의 위치에도 차이를 줄 수 있는데 인접한 프로세서를 우선으로 배치하는 방법과 노드내에서 최대한 프로세서를 흩어지게 위치시키는 방법도 함께 적용하여 성능을 분석하였다. 16 개의 단위 시스템을 이용하여 1024 개의 프로세서까지 테스트하였다.

<표 1> 하드웨어 및 소프트웨어 구성

System	Power 595
CPU	IBM 64bit POWER6
CPU Clock Speed	5.0 GHz
Cache / core	L1: 64/64KB(I/D), L2: 8MB, L3: 32MB
OS	AIX Version 6.1
File System	GPFS Version 3.2.1
Programming Language	IBM XL Fortran Version 12.1 IBM XL C/C++ Version 10.1
MPI	PE Version 5.1.1
Scheduler	LoadLeveler Version 3.5.1
Benchmark Program	NPB 3.3

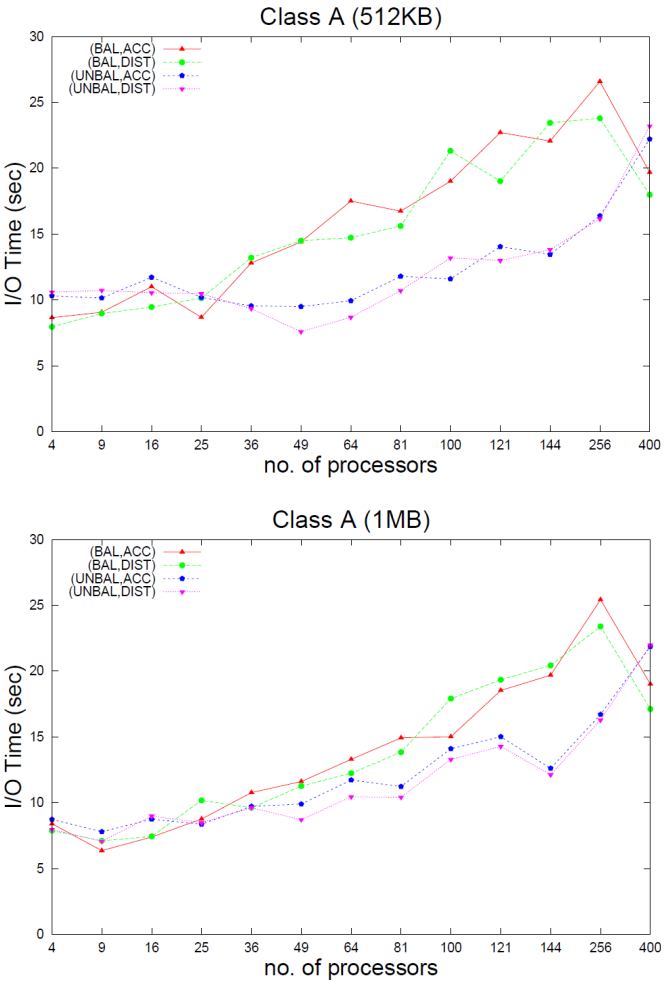
3.2. BTIO

NASA 에서 개발한 NPB(NAS Parallel Benchmarks)는 8 종류 이상의 테스트 프로그램으로 구성되어 있다. 특히 BT(Block-Tridiagonal) 테스트는 MPI-IO 에 대한 테스트가 가능한 형태로 확장될 수 있는데, 이것이 BTIO 이다[5].

Block-Tridiagonal Solver 와 관련된 연산과 MPI-IO 를 이용한 쓰기 작업을 반복적으로 수행하고 최종 단계에서는 테스트 중에 기록된 내용에 대한 읽기 작업을 수행하고 테스트를 종료한다. 다른 NPB 테스트처럼 문제 크기의 선택이 가능하여 시스템의 규모에 맞게 문제 크기를 선택할 수 있다. 본 연구에서는 3.3 버젼을 사용하였다.

4. 실험 결과

실험은 BTIO 의 IO 시간을 측정하는 방식으로 진행되었다. 프로세스 할당 방식과 GPFS 파일시스템 구성 방식의 2 가지 설정을 변경하면서 성능을 측정하였다. 64 CPU 를 사용하는 단위 노드 16 개를 실험에 이용하였는데, CPU 를 단위 노드를 먼저 채우고 다음 노드를 채우는 방식과 16 노드에 균등하게 CPU 를 배정하는 방식을 고려하였다. 또한 단위 노드 내에서는 8 개의 Processor book 에 8 개의 CPU core 가 존재하는데 Processor book 을 모두 채우고 다음 Processor book 을 채우는 방식과 Processor book 에 균등하게 CPU 를 사용하는 방식을 고려하였다. GPFS 의 경우에는 블록사이즈의 크기를 512KB 와 1MB 로 바꾸어 가면서 테스트 하였다.

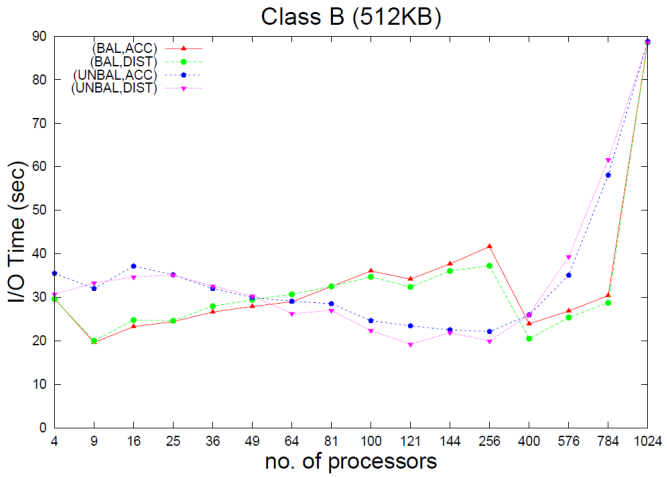


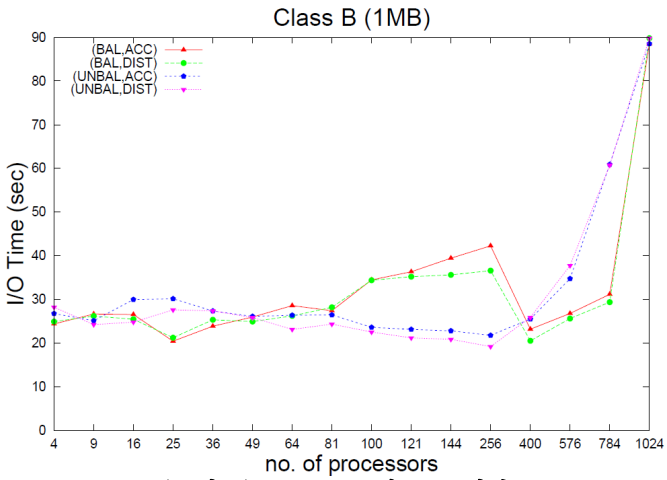
(그림 1) BTIO A 클래스 IO 시간:

BAL: 노드를 균등하게 사용
UNBAL: 한노드를 모두 채우고, 다음 노드 사용
DIST: Processor book 을 균등하게 사용
ACC: 한 Processor book 을 우선 채우고 그 다음 Processor book 을 사용

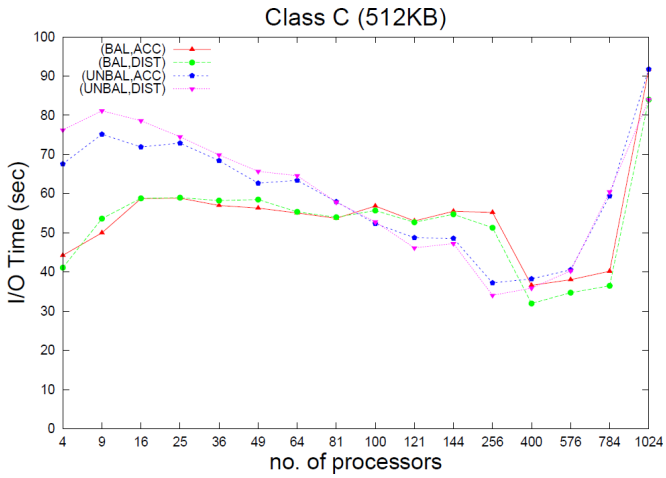
그림 1 은 클래스 A 에서의 IO 시간을 보여준다. 클래스 A 의 경우에는 문제크기가 너무 작아지는 이유로 400 개 이상의 프로세스를 활용하는 테스트는 진행 할 수 없다. 상대적으로 문제 크기가 작은 클래스 A 의 경우 작은 크기의 파일을 생성하는 이유로, 병렬성이 증가되어도 IO 시간의 개선은 나타나지 않는다. 그림 2 와 3 을 보면 C 클래스에서 프로세스의 수가 증가하는 경우, 일부 구간에서 IO 시간의 개선이 나타남을 확인할 수 있다.

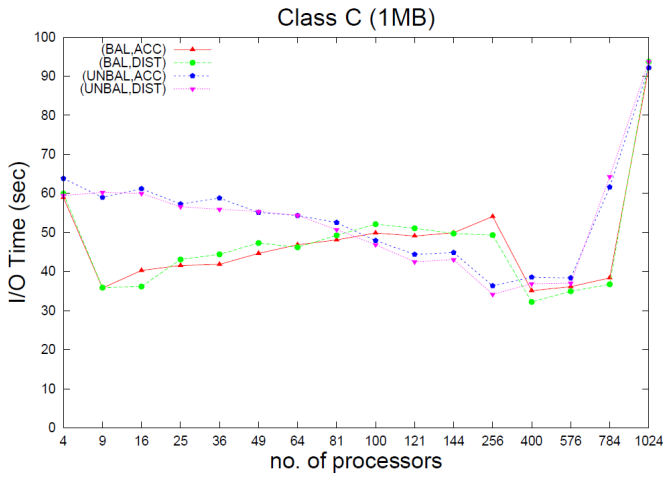
설정 변경에 따른 성능 결과는 클래스 A,B,C 모두 유사한 특징을 보이고 있다. 우선 노드 내에서 프로세스 할당 방식은 성능에 크게 영향을 미치지 않았다. 노드 할당 방식은 문제 크기가 커질수록 성능 패턴이 상이하게 나타났는데 C 클래스의 경우, 균형에 맞추어 노드를 할당하는 경우, 좋은 IO 성능을 보여주었다. 이와 같이 프로세스를 할당하는 방식은 수반되는 통신패턴의 변화를 가져오기 때문에 프로세스의 수, 문제의 크기에 따른 성능 변화 구간이 다양하다는 특징이 있다.





(그림 2) BTIO B 클래스 IO 시간





(그림 3) BTIO C 클래스 IO 시간

반면 GPFS 의 설정 변경은, 진행된 테스트 전반에 걸쳐 공통된 특정을 보여주었다. 즉 BTIO 의 경우에는 512KB 보다는 큰 1MB 의 블록 사이즈를 유지하는 경우, 보다 나은 성능을 보여 주었다.

5. 결론

본 연구에서는 SMP 를 사용하는 슈퍼컴퓨터에서 집합 IO 의 특성을 살펴 보았다. BTIO 를 이용하는 테스트에서 노드 할당 방식에 따라 성능의 차이가 나타났으며, 특히 파일시스템인 GPFS 의 설정이 MPI-IO 의 성능에 직접적인 영향을 미치고 있음을 확인하였다. 이에 향후, 응용프로그램의 IO 특징에 따라 다양한 설정의 GPFS 파티션을 제공할 수 있어야 할 것이다.

참고문헌

[1] Hakan Taki, Gil Utard, "MPI-IO on a parallel file system for cluster of workstations," Proc. of the 1st IEEE Computer Society International Workshop on Cluster Computing, 1999.

[2] Rajeev Thakur, and Alok Choudhary, ``An Extended Two-Phase Method for Accessing Sections of Out-of-Core Arrays," in Scientific Programming, vol. 5, no. 4, pp. 301~317, 1996.

[3] IBM Redbooks, "IBM Power 595 Technical Overview and Introduction," http://www.redbooks.ibm.com/abstracts/redp4440.html?Open

[4] H. Q. Le, et al, "IBM POWER6 microarchitecture," IBM Journal of Research and Development, Vol. 51(6), pp. 639~662, 2007

[5] NAS Parallel Benchmarks, http://www.nas.nasa.gov/Resources/Software/npb.html