운영체제

Assignment5

한준호
2018741035
최상호 교수님

Assignment 5

- Introduction

I/O Zone을 이용해 세 가지의 Linux I/O scheduler의 성능을 테스트 해본다. 이 때, File size, Buffer cache, record size, thread or process, thread or process의 파일 경로, 테스트 연산 항목 등을 바꿔거나 선택하면서 각 Linux I/O 스케줄러의 성능을 비교한다.

- Conclusion & Analysis

```
os2018741035@ubuntu:~$ sudo apt-get install iozone3
[sudo] password for os2018741035:
Reading package lists... Done
Building dependency tree
Reading state information... Done
The following NEW packages will be installed:
  iozone3
0 upgraded, 1 newly installed, 0 to remove and 98 not upgraded.
Need to get 417 kB of archives.
After this operation, 742 kB of additional disk space will be used.
Get:1 http://kr.archive.ubuntu.com/ubuntu xenial/multiverse amd64 iozone3 amd64
429-3 [417 kB]
Fetched 417 kB in 3s (118 kB/s)
Selecting previously unselected package iozone3.
(Reading database ... 218377 files and directories currently installed.)
Preparing to unpack .../iozone3_429-3_amd64.deb ...
Unpacking iozone3 (429-3) ...
Processing triggers for man-db (2.7.5-1) ...
Setting up iozone3 (429-3) ...
```

IOZone을 사용하기 위해 설치해주었다.

```
os2018741035@ubuntu:~$ iozone -R -i 0 -i 1 -i 2 -i 6 -I -r 8k -s 1g -t 1 -F ~/io
zone_test -b file_name.xls
```

- -R: I/O 실험 결과에 대한 excel file로 작성 및 file name: file_name.xls
- -i #: read, write, random fwrite 연산 수행
- -I: buffer cache 거치지 않고 연산 수행
- -r #: record size : 8KB ~ 16MB
- -s #: file size: 1GB
- -t #: thread or process 개수: 1
- -F #: thread or process의 파일 경로: ~/iozone_test

* I/O Zone 옵션 정리
-a : automatic mode
-A : -a와 비슷하나 더 자세하고 시간오래걸림. -a는 파일크기가 32MB를 넘으면 64k 미만으로 test를 하지않는다. (Iozone version 3.61이상에서는 이거대신 -z를 쓴다)
-b [filename] : excel과 호환가능한 binary file을 생성한다. (ex : iozone -Rab output.wks)
-B : mmap() file을 사용한다. 모든 임시파일들이 측정을 위해 생성되고 mmap() interface로 access된다
-c : timing, 계산에서 close()를 포함한다. 현재 test되고있는 OS에서 close()가 제대로 동작되는지 확실치 않을때 (NFS에서 유용)
-C : throughput testing에서 child끼리 전송되는 bytes를 보여준다. 프로세스관리나 파일 I/O에서 OS가 stravation problem이 있다면 유용
-d [Microsecond(delay)] : 프로세스나 스레드가 생성되는 때에 지정한 microsecond 만큼 delay준다 (보통은 같은시간이 생성됨)
-D : mmap file에 msync(MS_ASYNC)를 사용. mmap space에 있는 모든 data가 비동기식으로 disk에 쓰여진다.
-e : flush(fsync, fflush)를 포함함(timing 계산시에)
-E : extension test를 위해 사용됨. 특정 platform에서만 동작
-f [filename] : test의 임시파일에 특정파일이름의 사용한다.
-F [filename] [filename] ..... : -f와 비슷함. name의 개수는 프로세스나 스레드의 개수와 같아야된다.
-g : test에 사용되는 최대 파일 size설정
-G : mmap file에 msync(M_SUNC)를 사용(-D와 반대)
-h : help screen을 표시한다
-i # : 어떤 test를 할지 결정한다.(ex : -i 1 -i 2 -i 5 -i 6 -i7)
    숫자들의 의미 : 0 = write/re-write, 1 = read/re-read, 2 = random-read/write, 3 = read-backwards, 4 = re-write-record, 5 = stride-read, 6 = fwrite/re-fwrite, 7 = fread/re-fread, 8 = random mix, 9 = pwrite/re-pwrite, 10 = pread/re-pread, 11 = pwritev/re-pwritev, 12 = preadv/re-preadv
-I(대문자 아이) : VXFS filesystem에게 파일에 대한 모든 operation은 buffer cache를 우회할것이고 disk로 직접간다고 알려주는것
-j : file access의 stide를 설정(# * record size)

-J #(milliseconds) L I/O operation전에 계산 delay를 준다(-X / -Y로 써두됨, 같은기능)
-K : normal testing동안 random access를 발생시킨다.
-I(엘) # : 실행할 프로세스나 스레드의 최소 개수 제한
-L #(bytes) : 프로세스 캐쉬 라인 크기를 설정(적절하게 설정시 test speed up 가능!)
-m : multiple buffer를 사용, 프로그램에 따라 single/multiple사용, 둘 모두를 위해서
-M : uname()을 사용해서 output file에 현재 시스템에 대한 정보를 적어준다
-n # : 최소 파일크기를 설정
-N : operation마다 결과를 보고해준다
-o : disk로의 쓰기를 동기화 방식으로 한다
-O : 매초마다 operation결과를 준다
-p : 각 file operation전에 프로세스 캐쉬를 비운다
-P : 스레드나 프로세스를 특정 프로세서로 bind한다
-q #(Kbytes) : automode에서 최대 recordsize를 설정
-Q : offset/latency file을 설정한다
-r # : 특정 record size를 설정(보통은 4k)
-R : Excel report를 생성. 그래프 그릴때 사용
-s # : file size를 설정
-S # : 프로세서 캐쉬 사이즈 설정
-t # : throughput mode로 시작하게 설정, 숫자는 프로세스나 쓰레드 개수를 설정해줌
-T : test에 POSIX pthread를 사용한다
-u : 최대가능 프로세스나 스레드의 개수를 설정(-I option과 같이 사용한다)
-U [Mountpoint] : test동안 unmount하고 remount한 mount point를 설정한다. iozone이 각 test 시작전에 un/remount한다. 그러면 buffer cache에 아무것도 없이 시작하는 것을 보장한다.
-v : 버전을 보여준다
-V # : 임시파일에 쓰여질 patten을 지정해줘서, 각 test의 정확도를 검증해준다
-w : test동안 사용한 임시파일을 지우지 않고 남겨둔다
-W : read/write동안은 파일을 잠근다(LOCK)
-y # : 최소파일크기를 설정한다.
-z : -a랑 같이 쓰고 가능한 record size모두를 사용해서 test한다.
-Z : mmap I/O와 file I/O의 mixing을 가능하게한다

*사용한 테스트 연산(4개 연산 수행)

Number	Description	Number	Description	Number	Description
0	write/re-write	1	read/re-read	2	random-read/write
3	read-backwards	4	re-write-record	5	strid-read
6	fwrite/re-fwrite	7	frrad/re-fread	8	random-mix
9	pwrite/re-pwrite	10	pread/re-pread	11	writev/re-pwritev
12	preadv/re-preadv				

1. write / re-write (0)

• Write

- 새로운 파일 생성 측정

- 데이터 write 성능 뿐만 아니라 Storage Media 위치(Track) 선택을 위한 오버헤드 정보 포함 (Metadata - dentry, alloc, inode, …)

- Metadata 오버헤드 때문에 re-write 성능 대비 초기 성능이 낮음

• Re-write

- 이미 존재하는 파일에 대한 write 성능 측정

- Metadata가 이미 존재 하여 Metadata 생성 오버헤드 상쇄(Write 성능보다 높음)

2. read / re-read (1)

• Read

- 이미 존재는 파일에 대한 Read 성능 측정

• Re-read

- 최근에 읽었던 파일에 대한 성능 측정

- 캐시에 데이터가 유지 된 상태이면 높은 성능 유지

3. random-read / write (2)

• Random Read

- 파일 내 임의의 위치에 대한 Read 성능 측정

- 캐시 크기, 디스크 수, seek latency, … 영향 받을 수 있음

• Random Write

- 파일 내 임의의 위치에 대한 Write 성능 측정

- 캐시 크기, 디스크 수, seek latency, … 영향 받을 수 있음

4. fwrite / re-fwrite (6)

• Fwrite

- fwrite() Library 함수를 사용하여 측정
- Buffered Write Operation
- Buffer는 User Address Space
- Application이 매주 작은 크기의 데이터를 write 하는 경우 fwrite()는 데이터를 buffered 하고 blocked 함
- 이는 실제 OS 호출 수를 줄이는 효과
- 데이터가 어느 정도 쌓였을 때 OS call
- 새 파일을 생성 하는 것으로 Meatadata 생성 오버헤드 포함됨

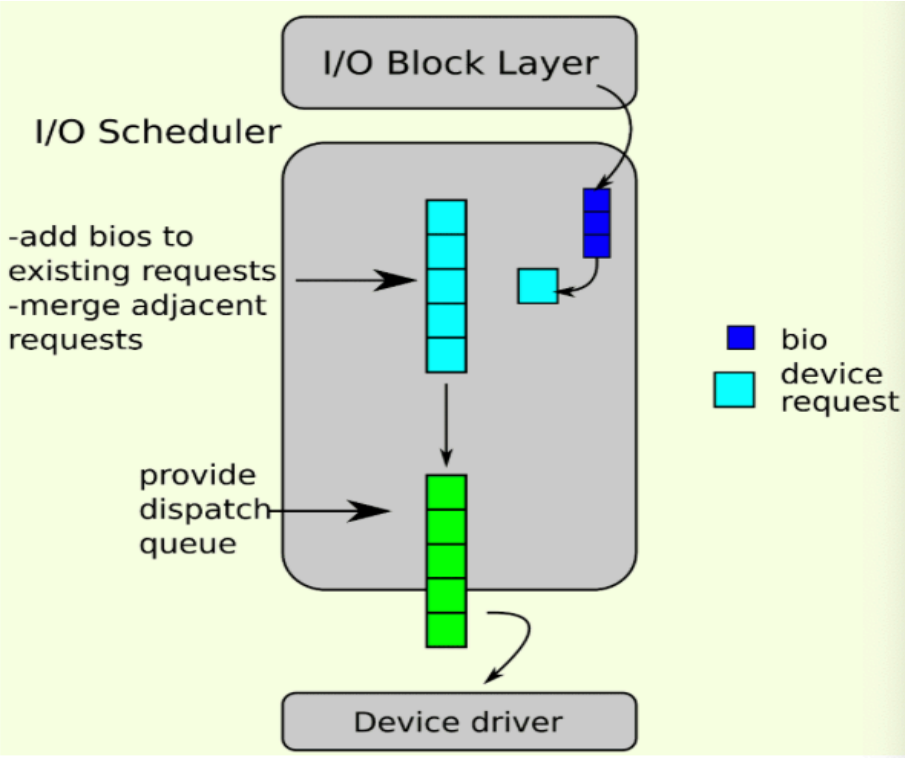
• Re-fwrite

- fwrite() Library 함수를 사용하여 측정
- Buffered Write Operation
- Buffer는 User Address Space
- Application이 매주 작은 크기의 데이터를 write 하는 경우 fwrite()는 데이터를 buffered 하고 blocked 함
- 이는 실제 OS 호출 수를 줄이는 효과
- 데이터가 어느 정도 쌓였을 때 OS call
- 기존 파일에 write 하므로 Meatadata 오버헤드 상쇄

write()가 높은 I/O traffic를 일으키기 때문에 3가지 스케줄러 대한 각각의 성능을 비교하기 더 쉬울 것 같다고 생각했기 때문에, fwrite() 함수를 사용해서 측정하는 fwrite / re-fwrite(6) 다른 연산 항목으로 선택하였다.

• 'noop'

- No Operation. 아무것도 하지 않은 스케줄러로 인접한 요청 병합만 수행하고 그 외에 아무 작업을 하지 않는다.
- Requeset FIFO 큐만 유지.
- Random access 하는 device를 위한 스케줄러로, 탐색에 대한 부담이 없기 때문에 정렬이 필요없고, 주로 플래시 메모리에서 사용한다.
- 주로 지능형 RAID 컨트롤러있거나, SSD사용하거나, 반도체 디스크 등 성능 좋은 디스크를 사용할 경우 선택되어지는 스케줄러로 커널은 아무것도 하지 않은 것이 부하를 줄일 수 있다는 생각이 기저에 있다.




```
os2018741035@ubuntu:~$ echo noop | sudo tee /sys/block/sda/queue/scheduler
[sudo] password for os2018741035:
noop
os2018741035@ubuntu:~$ cat /sys/block/sda/queue/scheduler
[noop] deadline cfq
```

스케줄러를 noop으로 변경해주었다.

```
os2018741035@ubuntu:~$ rm -rf ~/iozone_test
os2018741035@ubuntu:~$ sync
os2018741035@ubuntu:~$ echo 3| sudo tee /proc/sys/vm/drop_caches
[sudo] password for os2018741035:
3
```

실험에 영향을 주는 요소를 제거하기 위해 실행 할 때마다 캐시 및 버퍼를 비워주었다.

〈Noop Scheduler 성능 테스트 결과〉

• Record size: 8KB

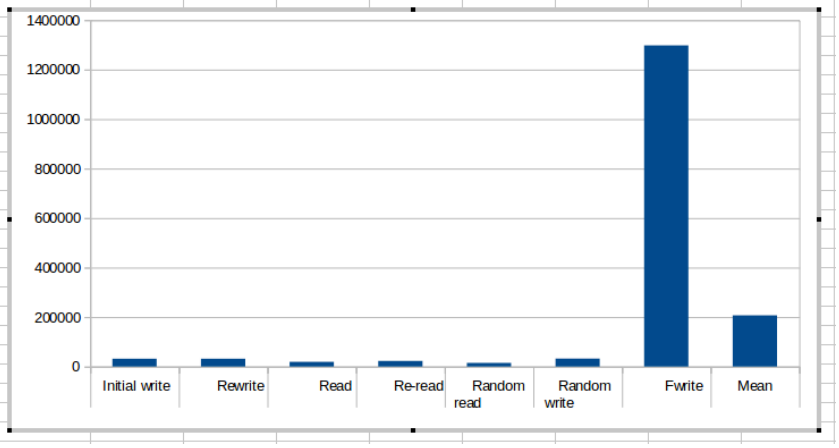
```
os2018741035@ubuntu:~$ iozone -R -i 0 -i 1 -i 2 -i 6 -I -r 8k -s 1g -t 1 -F ~/io
zone_test -b file_name.xls
```

```
        Excel chart generation enabled
        O_DIRECT feature enabled
        Record Size 8 kB
        File size set to 1048576 kB
        Command line used: iozone -R -i 0 -i 1 -i 2 -i 6 -I -r 8k -s 1g -t 1 -F
/home/os2018741035/iozone_test -b file_name.xls
        Output is in kBytes/sec
        Time Resolution = 0.000001 seconds.
        Processor cache size set to 1024 kBytes.
        Processor cache line size set to 32 bytes.
        File stride size set to 17 * record size.
        Throughput test with 1 process
        Each process writes a 1048576 kByte file in 8 kByte records
```

```
        Children see throughput for  1 initial writers  =   31530.08 kB/sec
        Parent sees throughput for  1 initial writers   =   31524.91 kB/sec
        Min throughput per process                      =   31530.08 kB/sec
        Max throughput per process                      =   31530.08 kB/sec
        Avg throughput per process                      =   31530.08 kB/sec
        Min xfer                                        = 1048576.00 kB

        Children see throughput for  1 rewriters        =   31596.84 kB/sec
        Parent sees throughput for  1 rewriters         =   31590.41 kB/sec
        Min throughput per process                      =   31596.84 kB/sec
        Max throughput per process                      =   31596.84 kB/sec
        Avg throughput per process                      =   31596.84 kB/sec
        Min xfer                                        = 1048576.00 kB

        Children see throughput for  1 readers          =   19191.10 kB/sec
        Parent sees throughput for  1 readers           =   19190.04 kB/sec
        Min throughput per process                      =   19191.10 kB/sec
        Max throughput per process                      =   19191.10 kB/sec
        Avg throughput per process                      =   19191.10 kB/sec
        Min xfer                                        = 1048576.00 kB

        Children see throughput for 1 re-readers        =   22520.92 kB/sec
        Parent sees throughput for 1 re-readers         =   22519.95 kB/sec
        Min throughput per process                      =   22520.92 kB/sec
        Max throughput per process                      =   22520.92 kB/sec
        Avg throughput per process                      =   22520.92 kB/sec
        Min xfer                                        = 1048576.00 kB

        Children see throughput for 1 random readers    =   14760.46 kB/sec
        Parent sees throughput for 1 random readers     =   14760.09 kB/sec
        Min throughput per process                      =   14760.46 kB/sec
        Max throughput per process                      =   14760.46 kB/sec
        Avg throughput per process                      =   14760.46 kB/sec
        Min xfer                                        = 1048576.00 kB

        Children see throughput for 1 random writers    =   32065.15 kB/sec
        Parent sees throughput for 1 random writers     =   32058.35 kB/sec
        Min throughput per process                      =   32065.15 kB/sec
        Max throughput per process                      =   32065.15 kB/sec
        Avg throughput per process                      =   32065.15 kB/sec
        Min xfer                                        = 1048576.00 kB

        Children see throughput for  1 fwriters         = 1298031.75 kB/sec
        Parent sees throughput for  1 fwriters          =  643316.39 kB/sec
        Min throughput per process                      = 1298031.75 kB/sec
        Max throughput per process                      = 1298031.75 kB/sec
        Avg throughput per process                      = 1298031.75 kB/sec
        Min xfer                                        = 1048576.00 kB
```

```
"Throughput report Y-axis is type of test X-axis is number of processes"
"Record size = 8 kBytes "
"Output is in kBytes/sec"

"   Initial write "    31530.08

"         Rewrite "    31596.84

"            Read "    19191.10

"         Re-read "    22520.92

"     Random read "    14760.46

"    Random write "    32065.15

"          Fwrite " 1298031.75


iozone test complete.
```

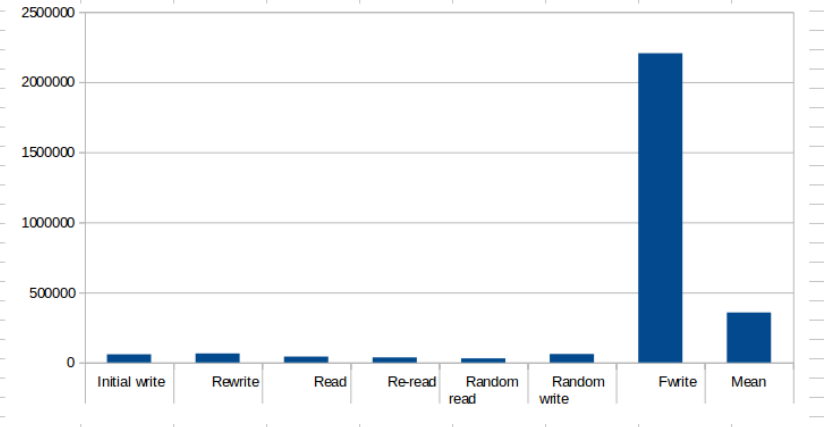
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 8k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 8 kBytes		
4	Output is in kBytes/sec		
5	Initial write		31530.07617
6	Rewrite		31596.83984
7	Read		19191.09766
8	Re-read		22520.92188
9	Random read		14760.45703
10	Random write		32065.15234
11	Fwrite		1298031.75
12	Mean		207099.4707



평균: 207099.4707 KB/sec

• Record size: 16KB

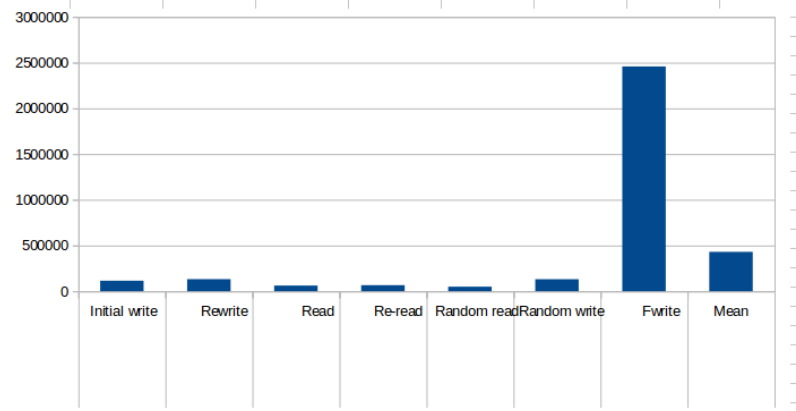
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 16k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 16 kBytes		
4	Output is in kBytes/sec		
5	Initial write		58999.51953
6	Rewrite		64755.40234
7	Read		42542.41016
8	Re-read		36959.83984
9	Random read		30075.07031
10	Random write		60998.75781
11	Fwrite		2205625
12	Mean		357136.5714



평균: 357136.5714 KB/sec

• Record size: 32KB

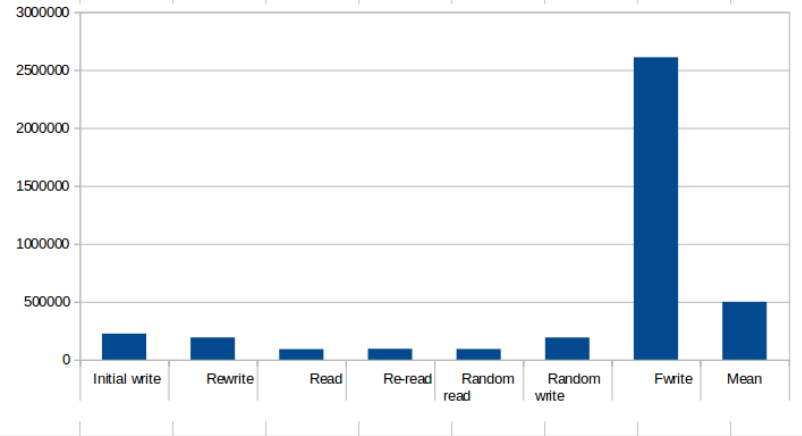
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 32k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 32 kBytes		
4	Output is in kBytes/sec		
5	Initial write		115235.7266
6	Rewrite		132091.1406
7	Read		61910.66016
8	Re-read		66006.42188
9	Random read		49899.08594
10	Random write		131291.1406
11	Fwrite		2457977.75
12	Mean		430630.2751



평균: 430630.2751 KB/sec

• Record size: 64KB

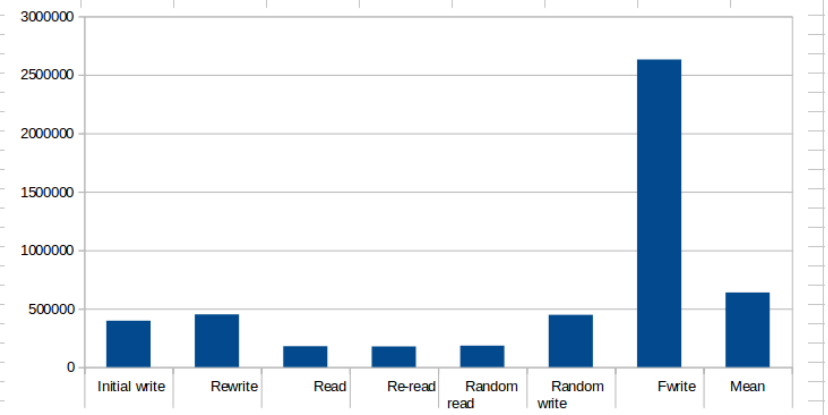
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 64k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 64 kBytes		
4	Output is in kBytes/sec		
5	Initial write		225170.3438
6	Rewrite		192077.0156
7	Read		91040.14063
8	Re-read		93990.60156
9	Random read		92734.60938
10	Random write		191302.625
11	Fwrite		2609502.25
12	Mean		499402.5123



평균: 499402.5123 KB/sec

• Record size: 128KB

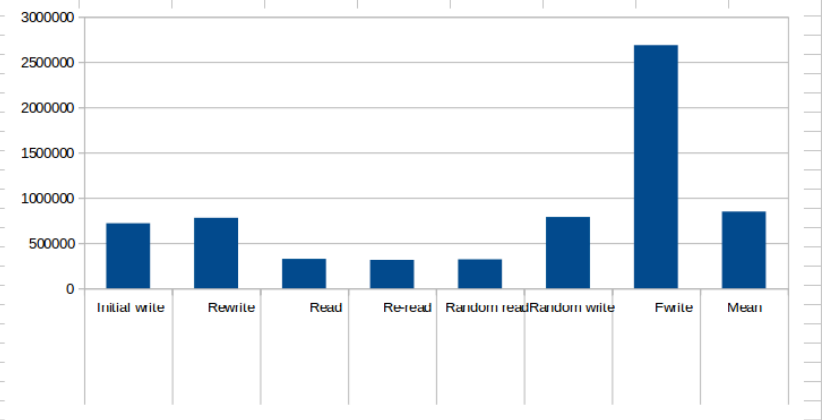
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 128k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 128 kBytes		
4	Output is in kBytes/sec		
5	Initial write		397435.0625
6	Rewrite		451788.75
7	Read		179122.0313
8	Re-read		177602.6719
9	Random read		183809.625
10	Random write		447281.5313
11	Fwrite		2631004.25
12	Mean		638291.9888



평균: 638291.9888 KB/sec

• Record size: 256KB

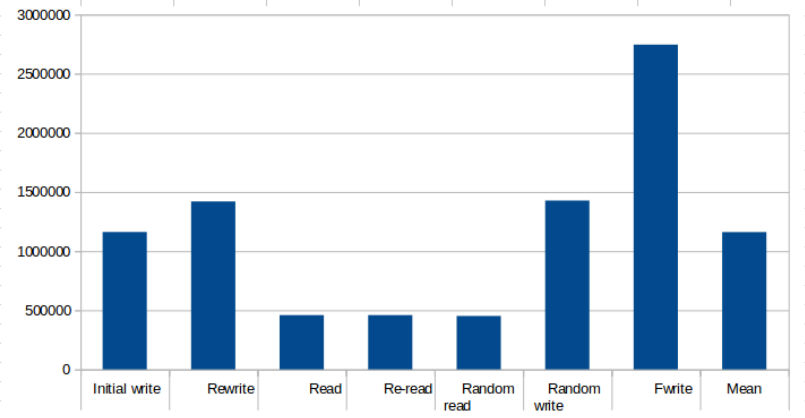
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 256k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 256 kBytes		
4	Output is in kBytes/sec		
5	Initial write		719002.4375
6	Rewrite		778836.6875
7	Read		325994.9063
8	Re-read		314112.9688
9	Random read		320494.125
10	Random write		788336.625
11	Fwrite		2686273.25
12	Mean		847578.7143



평균: 847578.7143 KB/sec

• Record size: 512KB

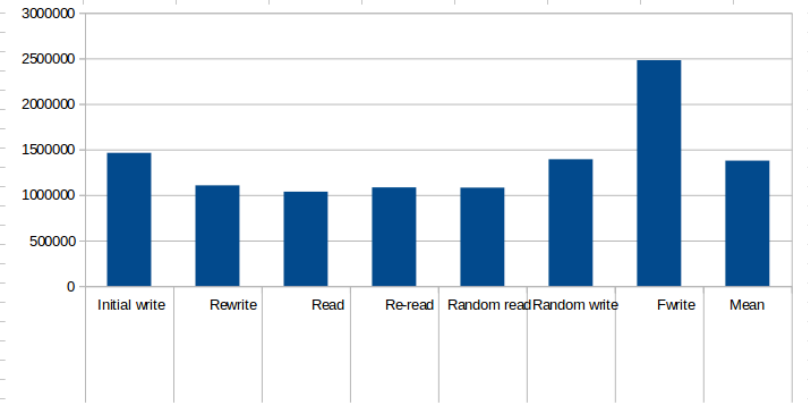
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 512k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 512 kBytes		
4	Output is in kBytes/sec		
5	Initial write		1161528.875
6	Rewrite		1420407.75
7	Read		459572.8125
8	Re-read		459264.8125
9	Random read		452358.6563
10	Random write		1427986
11	Fwrite		2745538
12	Mean		1160950.987



평균: 1160950.987KB/sec

• Record size: 8MB

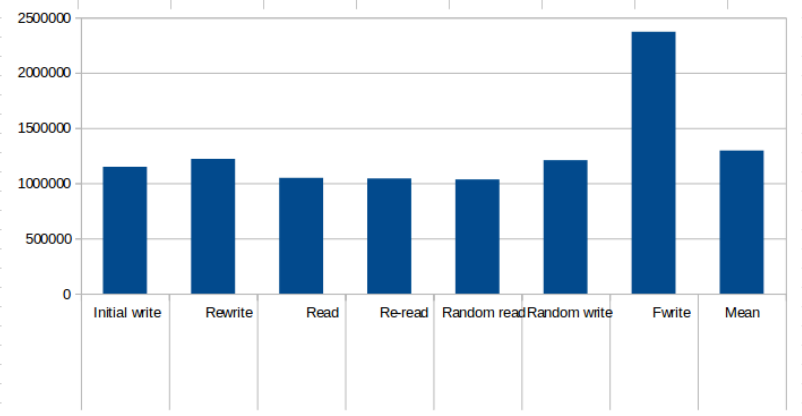
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 8m -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 8192 kBytes		
4	Output is in kBytes/sec		
5	Initial write		1463093.875
6	Rewrite		1105486.25
7	Read		1036075.563
8	Re-read		1083021.625
9	Random read		1080469.25
10	Random write		1392422.375
11	Fwrite		2479442.25
12	Mean		1377144.455



평균: 1377144.455 KB/sec

• Record size: 16MB

	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 16m -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 16384 kBytes		
4	Output is in kBytes/sec		
5	Initial write		1148192.125
6	Rewrite		1220162.625
7	Read		1048690.375
8	Re-read		1043299.938
9	Random read		1034546.5
10	Random write		1208716.25
11	Fwrite		2371641
12	Mean		1296464.116

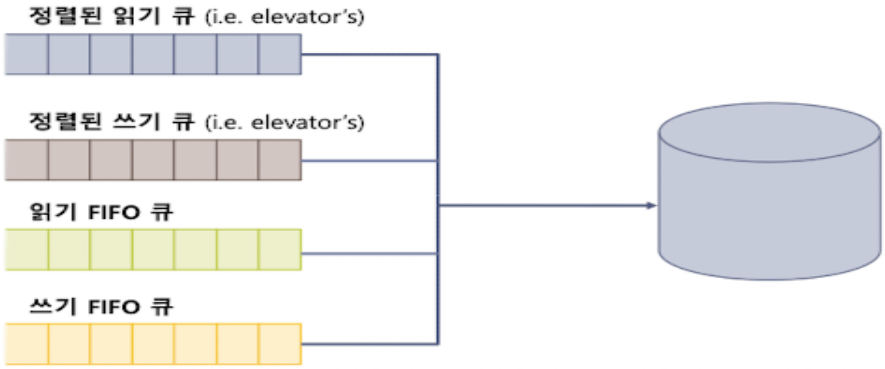


평균: 1296464.166 KB/sec

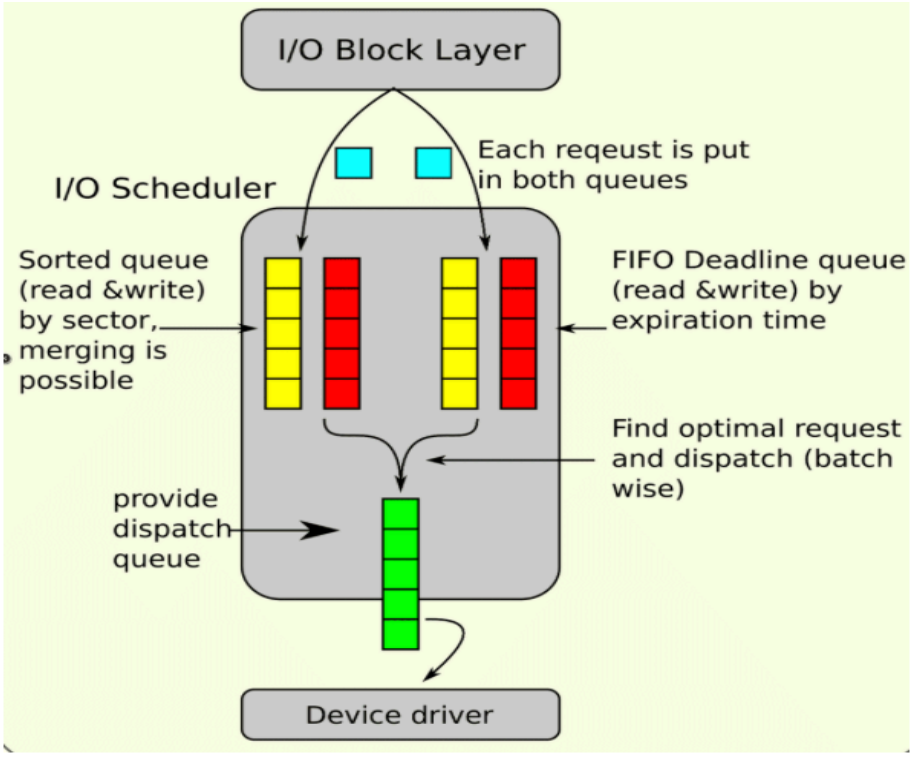
Record size를 크게 설정 할수록 IOZone의 출력 값인 프로세스 당 처리율 (KB/sec)이 대체로 크게 나왔다. 예외적으로 Record size를 16MB로 설정 했을 때가 8MB로 설정 했을 때보다 처리율이 작게 나왔다. 결과적으로 Record size를 8MB로 설정 했을 때 가장 빠른 처리율이 나왔다.

Record size를 작게 설정 했을 때는, 추가적으로 수행하는 연산인 fwrite의 프로세스 당 처리율 값만 다른 연산에 비해 너무 크게 나와서 그래프를 이용한 시각적인 비교가 어려웠는데 Record size를 크게 설정 할수록 나머지 연산도 프로세스 당 처리율이 크게 나와서 그래프를 이용한 시각적인 비교가 쉬웠다.

• ’deadline’



- 정렬된 읽기 큐, 정렬된 쓰기 큐, 입력 FIFO 큐, 출력 FIFO 큐로 4개의 큐를 사용한다. Deadline이 지안 요청이 없을 경우에 정렬된 큐에서 요청을 꺼내서 처리한다.
- 모든 요청은 마감시간을 가진다.(읽기 요청: 500ms, 쓰기 요청: 5s)
- 읽기 우선 정책으로, 기아 현상(Starvation)을 방지한다.



- I/O 대기 시간의 한계점(deadline)을 마련하고, 그것이 가까워 온 것 부터 먼저 처리한다.
- 처리량보다 지연에 최적화된 스케줄링을 한다.
- 읽기 및 쓰기를 모두 균형있게 처리한다.

- 몇몇 프로세스가 많은 수의 I/O를 발생시키는 환경에 적합하다.
- 데이터 베이스의 파일 시스템에 많이 사용된다.

```
os2018741035@ubuntu:~$ echo deadline | sudo tee /sys/block/sda/queue/scheduler
[sudo] password for os2018741035:
deadline
os2018741035@ubuntu:~$ cat /sys/block/sda/queue/scheduler
noop [deadline] cfq
```

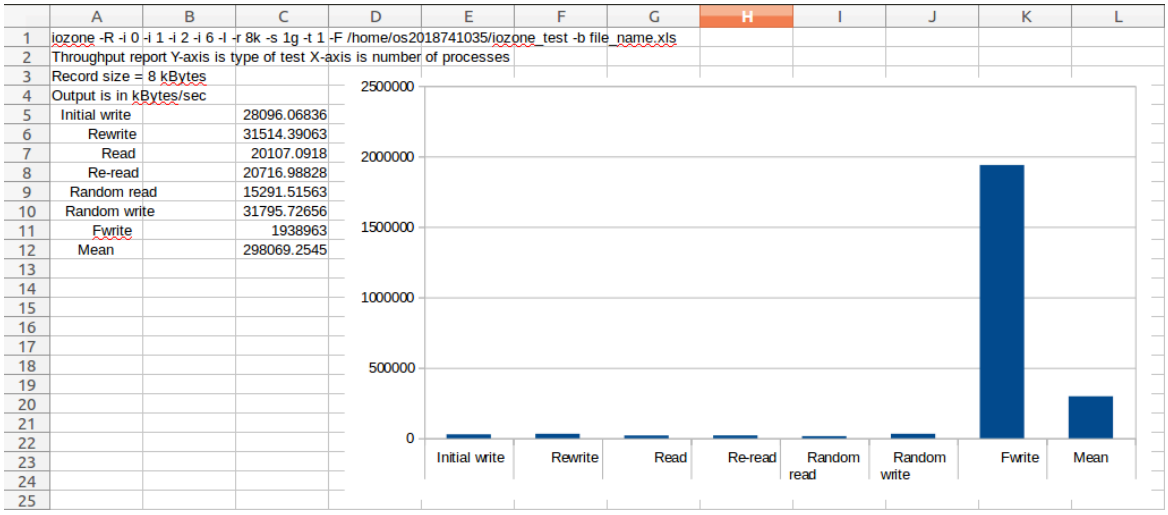
스케줄러를 deadline으로 변경해주었다.

```
os2018741035@ubuntu:~$ rm -rf ~/iozone_test
os2018741035@ubuntu:~$ sync
os2018741035@ubuntu:~$ echo 3| sudo tee /proc/sys/vm/drop_caches
[sudo] password for os2018741035:
3
```

실험에 영향을 주는 요소를 제거하기 위해 실행 할 때마다 캐시 및 버퍼를 비워주었다.

〈Deadline Scheduler 성능 테스트 결과〉

• Record size: 8KB

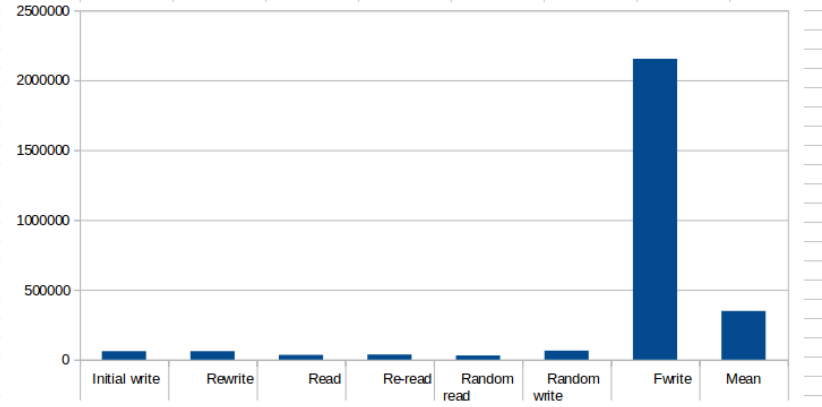


	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 8k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 8 kBytes		
4	Output is in kBytes/sec		
5	Initial write		28096.06836
6	Rewrite		31514.39063
7	Read		20107.0918
8	Re-read		20716.98828
9	Random read		15291.51563
10	Random write		31795.72656
11	Fwrite		1938963
12	Mean		298069.2545

평균: 298069.2545 KB/sec

• Record size: 16KB

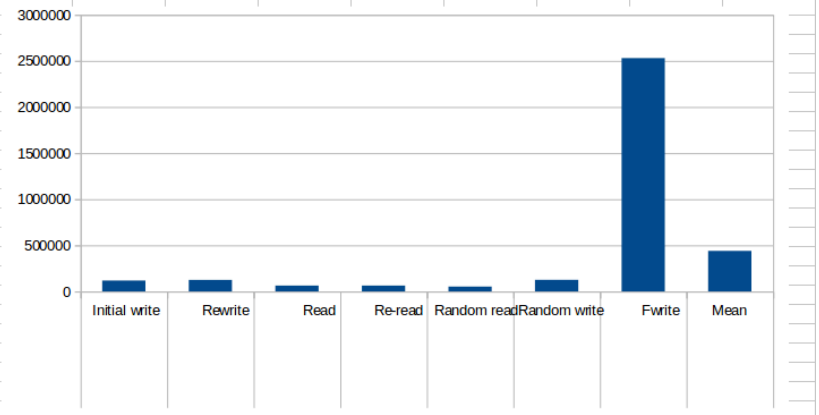
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 16k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 16 kBytes		
4	Output is in kBytes/sec		
5	Initial write		61162.96875
6	Rewrite		60892.14063
7	Read		34231.6875
8	Re-read		36948.33984
9	Random read		30057.49023
10	Random write		64413.97266
11	Fwrite		2153615.5
12	Mean		348760.2999



평균: 348760.2999 KB/sec

• Record size: 32KB

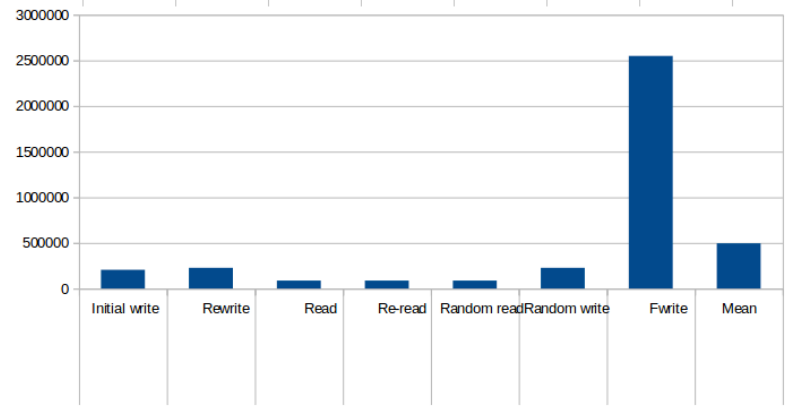
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 32k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 32 kBytes		
4	Output is in kBytes/sec		
5	Initial write		118100.0938
6	Rewrite		124683.9844
7	Read		63483.85156
8	Re-read		63595.53516
9	Random read		52932.34375
10	Random write		125852.6875
11	Fwrite		2531307
12	Mean		439993.6423



평균: 439993.6423 KB/sec

• Record size: 64KB

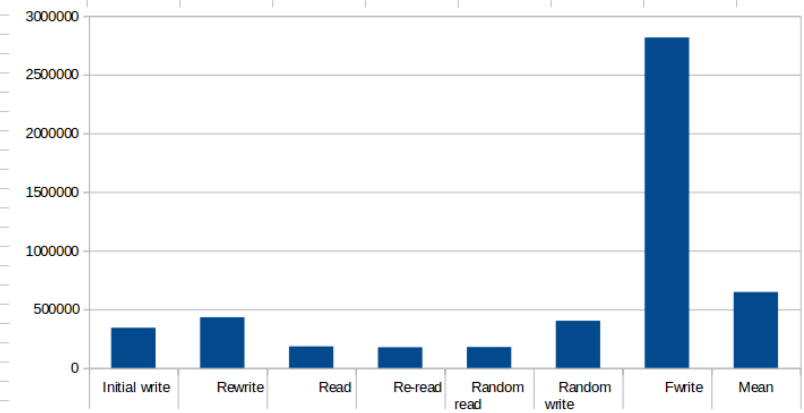
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 64k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 64 kBytes		
4	Output is in kBytes/sec		
5	Initial write		205877.2813
6	Rewrite		226952.375
7	Read		87597.89063
8	Re-read		88006.625
9	Random read		87719.75781
10	Random write		227431.0938
11	Fwrite		2548432.75
12	Mean		496002.5391



평균: 496002.5391 KB/sec

• Record size: 128KB

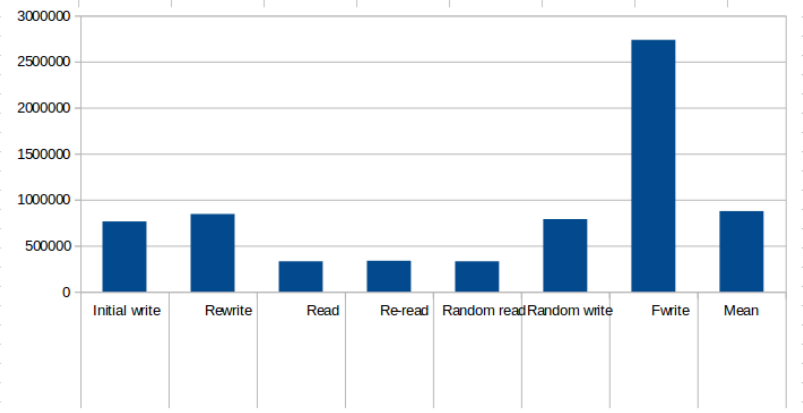
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 128k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 128 kBytes		
4	Output is in kBytes/sec		
5	Initial write		343316.5625
6	Rewrite		432461.5
7	Read		184883.3906
8	Re-read		176930.3594
9	Random read		178828.2031
10	Random write		402348.9063
11	Fwrite		2816361.75
12	Mean		647875.8103



평균: 647875.8103 KB/sec

• Record size: 256KB

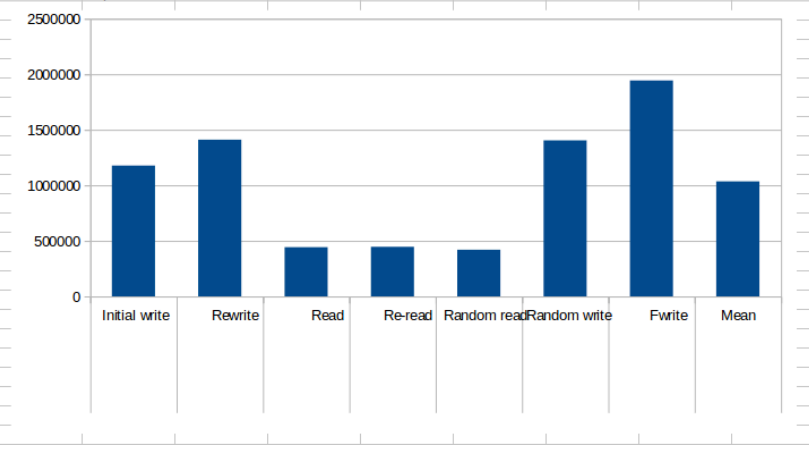
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 256k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 256 kBytes		
4	Output is in kBytes/sec		
5	Initial write		762932.5
6	Rewrite		844128.0625
7	Read		330431.1563
8	Re-read		336121.1563
9	Random read		329894
10	Random write		788994.9375
11	Fwrite		2736203.25
12	Mean		875529.2946



평균: 875529.2946 KB/sec

• Record size: 512KB

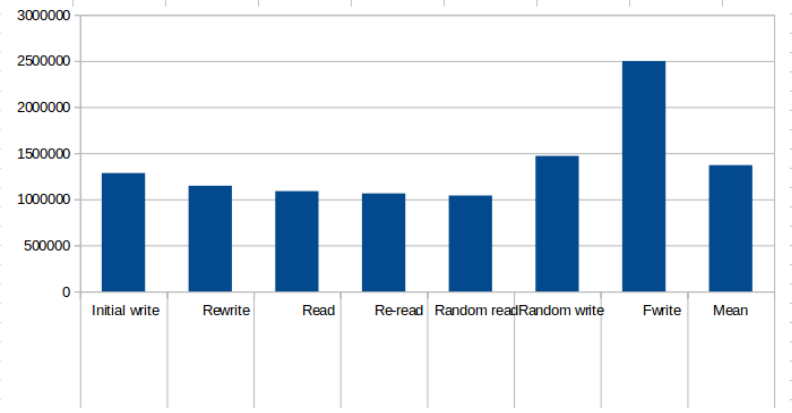
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 512k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 512 kBytes		
4	Output is in kBytes/sec		
5	Initial write		1178090.375
6	Rewrite		1410835
7	Read		443617.0938
8	Re-read		447064.5
9	Random read		420113.1563
10	Random write		1405068.375
11	Fwrite		1944025.625
12	Mean		1035544.875



평균: 1035544.875 KB/sec

• Record size: 8MB

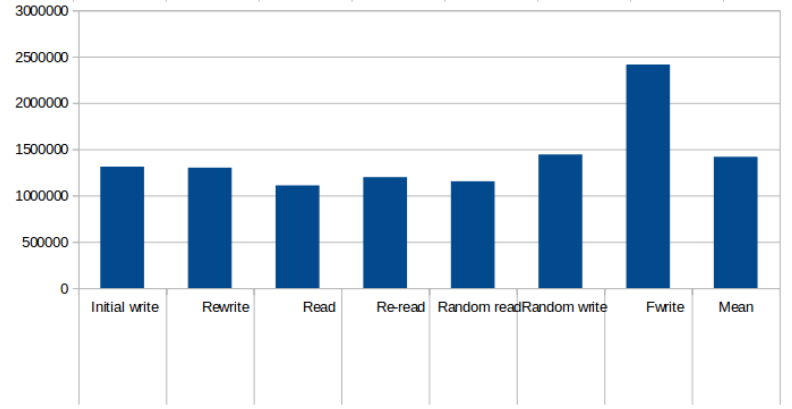
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 8m -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 8192 kBytes		
4	Output is in kBytes/sec		
5	Initial write		1284190.125
6	Rewrite		1146409.625
7	Read		1087717.625
8	Re-read		1063396.625
9	Random read		1040750.563
10	Random write		1470326.125
11	Fwrite		2498661.75
12	Mean		1370207.491



평균: 1370207.491 KB/sec

• Record size: 16MB

	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 16m -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 16384 kBytes		
4	Output is in kBytes/sec		
5	Initial write		1311702.125
6	Rewrite		1301158
7	Read		1109334.25
8	Re-read		1198616.25
9	Random read		1153910.75
10	Random write		1444102.75
11	Fwrite		2414132.5
12	Mean		1418993.804

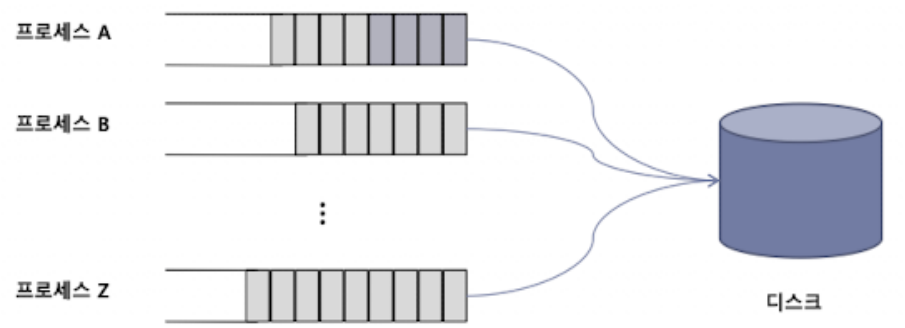


평균: 1418993.804 KB/sec

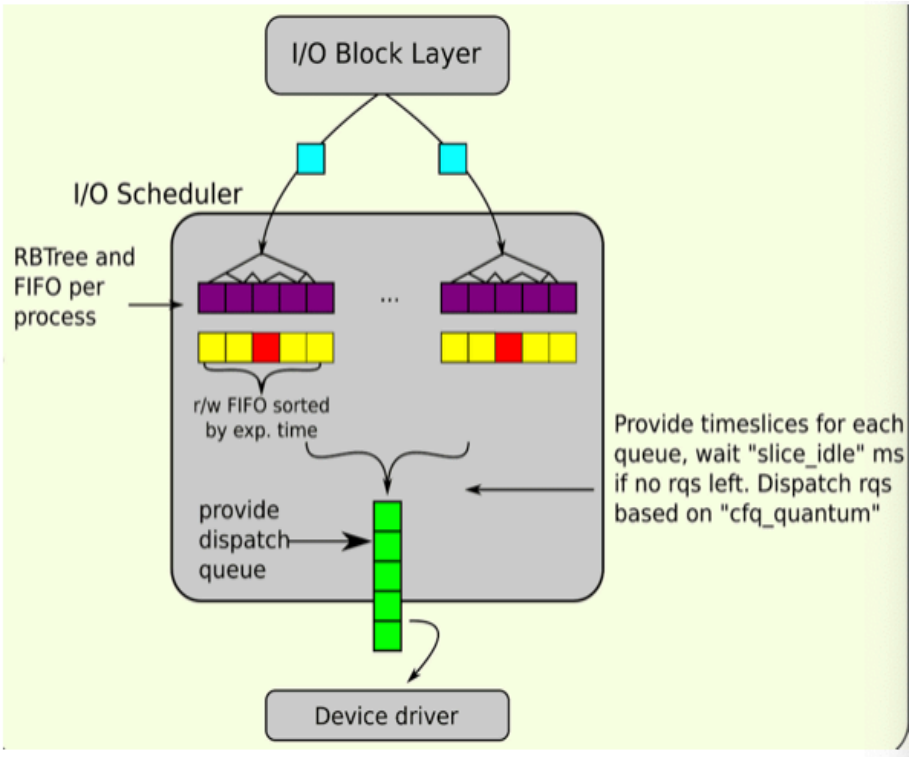
Record size를 크게 설정 할수록 IOZone의 출력 값인 프로세스 당 처리율 (KB/sec)이 크게 나왔다.

Record size를 작게 설정 했을 때는, 추가적으로 수행하는 연산인 fwrite의 프로세스 당 처리율 값만 다른 연산에 비해 너무 크게 나와서 그래프를 이용한 시각적인 비교가 어려웠는데 Record size를 크게 설정 할수록 나머지 연산도 프로세스 당 처리율이 크게 나와서 그래프를 이용한 시각적인 비교가 쉬웠다.

• ’CFQ’



- 입출력을 요펑하는 모든 프로세스들에 대해 디스크 I/O 대역폭을 공평하게 할당하는 것을 보장하는 기법.
- I/O를 요청한 프로세스 별로 큐를 할당한다.
- 각 프로세스에 관한 큐는 섹터 순으로 정렬한다.
- 각 큐는 라운드 로빈으로 처리한다.



- Completely Fair Queuing 스케줄러.
- 프로세스마다 I/O 대기열을 가지고 최대한 균등하게 예약을 한다.
- 많은 프로세스들이 세세한 I/O를 많이 발생시킬 때 사용하면 좋다.
- Fedora Core 커널 패키지의 기본이다.

```
os2018741035@ubuntu:~$ echo cfq | sudo tee /sys/block/sda/queue/scheduler
cfq
os2018741035@ubuntu:~$ cat /sys/block/sda/queue/scheduler
noop deadline [cfq]
```

스케줄러를 cfq로 변경해주었다.

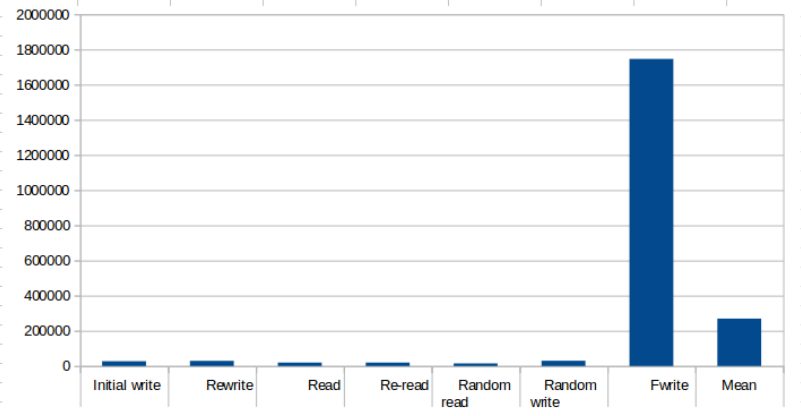
```
os2018741035@ubuntu:~$ rm -rf ~/iozone_test
os2018741035@ubuntu:~$ sync
os2018741035@ubuntu:~$ echo 3| sudo tee /proc/sys/vm/drop_caches
[sudo] password for os2018741035:
3
```

실험에 영향을 주는 요소를 제거하기 위해 실행 할 때마다 캐시 및 버퍼를 비워주었다.

〈CFQ Scheduler 성능 테스트 결과〉

• Record size: 8KB

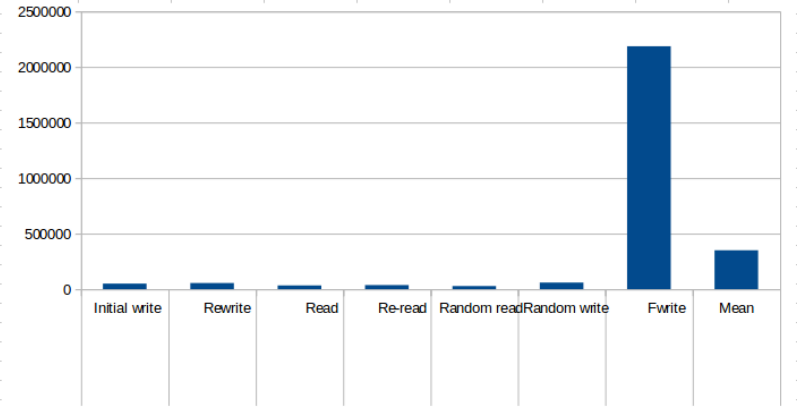
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 8k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 8 kBytes		
4	Output is in kBytes/sec		
5	Initial write		26920.85156
6	Rewrite		29480.22656
7	Read		19263.82422
8	Re-read		19615.49805
9	Random read		14737.59082
10	Random write		29799.23438
11	Fwrite		1745976.625
12	Mean		269399.1215



평균: 269399.1215 KB/sec

• Record size: 16KB

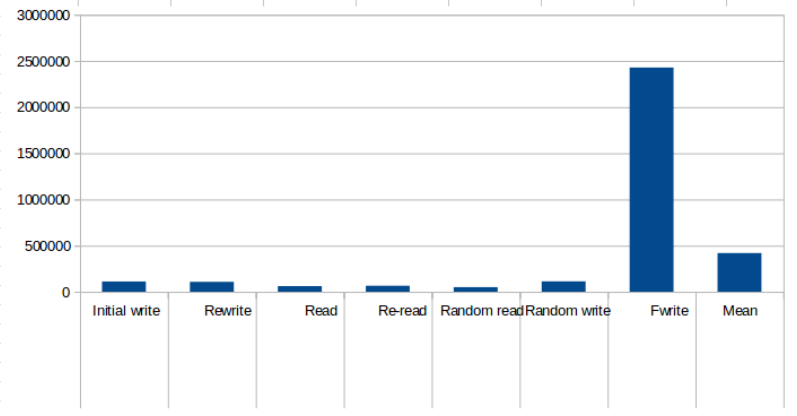
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 16k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 16 kBytes		
4	Output is in kBytes/sec		
5	Initial write		50238.86719
6	Rewrite		55703.78516
7	Read		33707.05469
8	Re-read		36969.78125
9	Random read		27985.17188
10	Random write		59592.57031
11	Fwrite		2186368.75
12	Mean		350080.8544



평균: 350080.8544 KB/sec

• Record size: 32KB

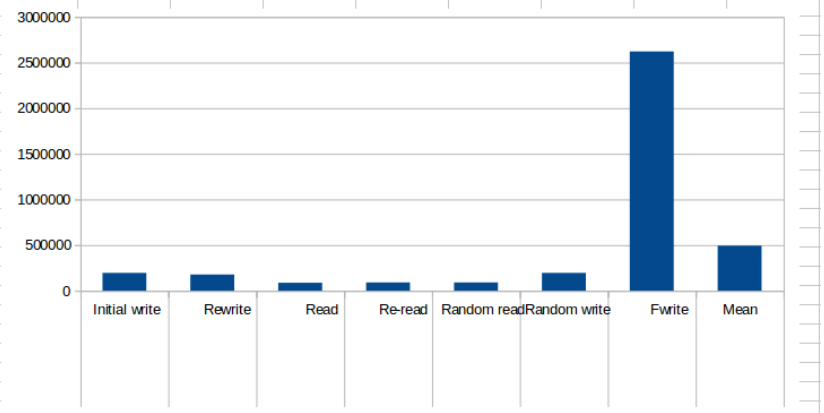
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 32k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 32 kBytes		
4	Output is in kBytes/sec		
5	Initial write		110716.4453
6	Rewrite		108248.6641
7	Read		61431.05859
8	Re-read		65861.39844
9	Random read		50256.78906
10	Random write		112862.2656
11	Fwrite		2428501.25
12	Mean		419696.8387



평균: 419696.8387 KB/sec

• Record size: 64KB

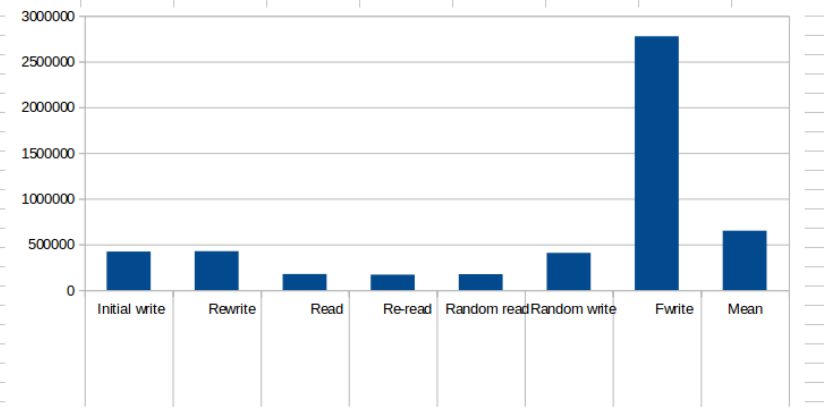
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 64k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 64 kBytes		
4	Output is in kBytes/sec		
5	Initial write		195794.9688
6	Rewrite		177699.4844
7	Read		87536.35156
8	Re-read		90573.40625
9	Random read		90107.64844
10	Random write		195495.7031
11	Fwrite		2621407
12	Mean		494087.7946



평균: 494087.7946 KB/sec

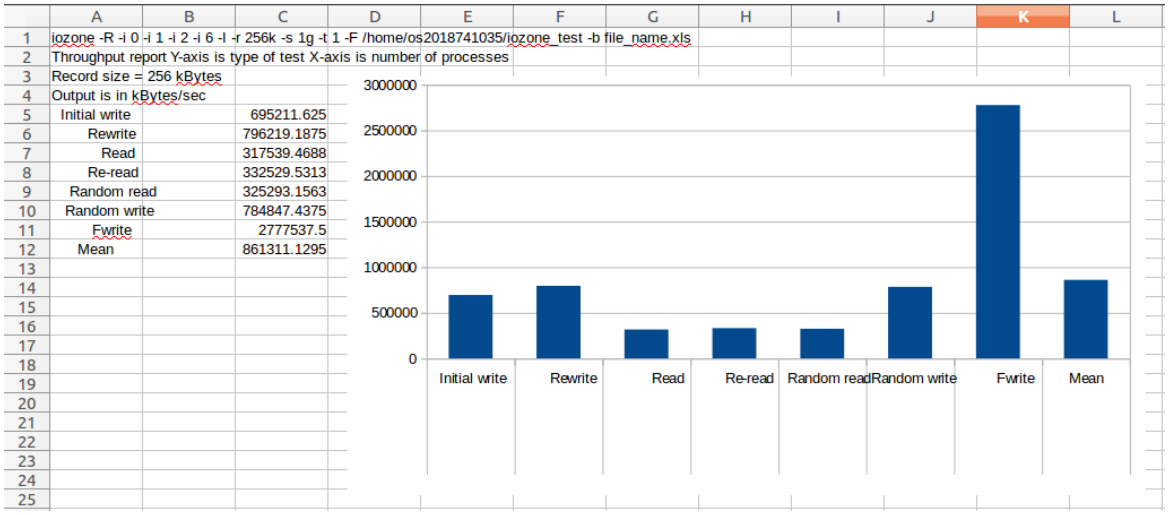
• Record size: 128KB

	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 128k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 128 kBytes		
4	Output is in kBytes/sec		
5	Initial write		421632.8125
6	Rewrite		425722.5938
7	Read		176092.5469
8	Re-read		169770.8906
9	Random read		174552.2969
10	Random write		408048.2813
11	Fwrite		2775921
12	Mean		650248.6317



평균: 650248.6317 KB/sec

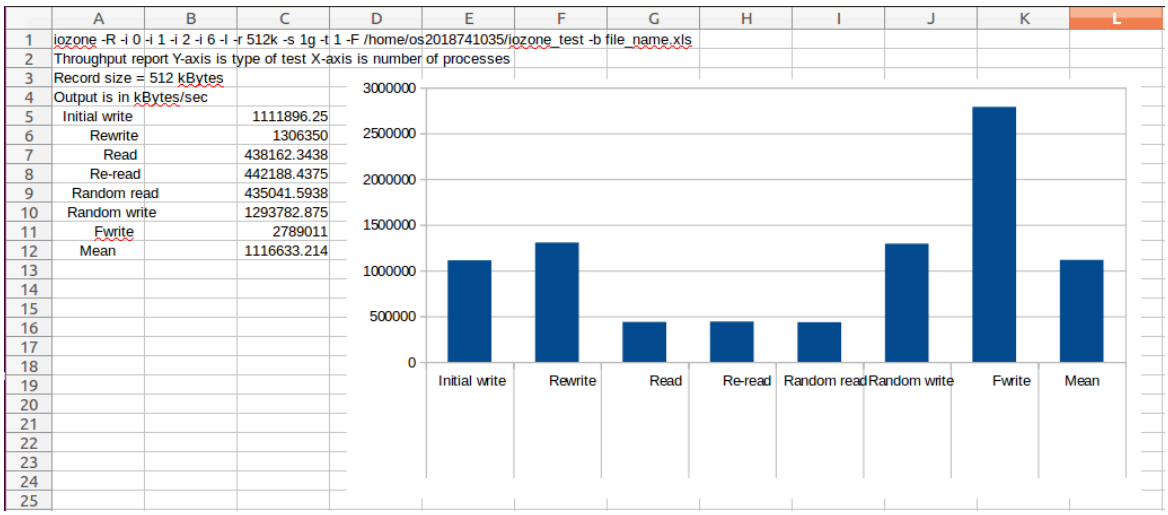
• Record size: 256KB



	A	B	C
1	iozone -R -l 0 -i 1 -i 2 -i 6 -l -r 256k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 256 kBytes		
4	Output is in kBytes/sec		
5	Initial write		695211.625
6	Rewrite		796219.1875
7	Read		317539.4688
8	Re-read		332529.5313
9	Random read		325293.1563
10	Random write		784847.4375
11	Fwrite		2777537.5
12	Mean		861311.1295

평균: 861311.1295 KB/sec

• Record size: 512KB

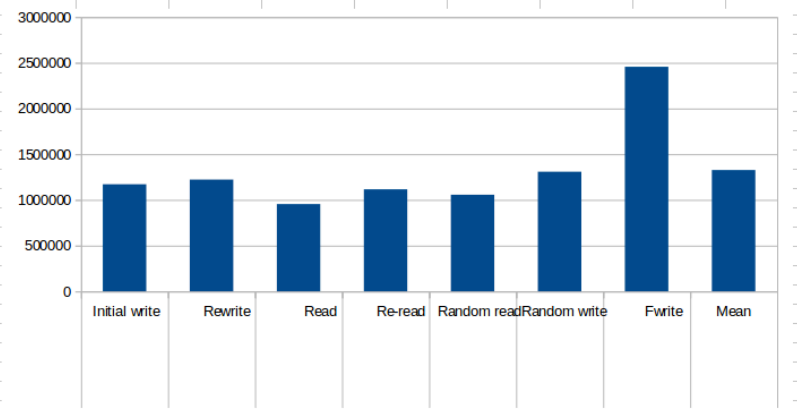


	A	B	C
1	iozone -R -l 0 -i 1 -i 2 -i 6 -l -r 512k -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 512 kBytes		
4	Output is in kBytes/sec		
5	Initial write		1111896.25
6	Rewrite		1306350
7	Read		438162.3438
8	Re-read		442188.4375
9	Random read		435041.5938
10	Random write		1293782.875
11	Fwrite		2789011
12	Mean		1116633.214

평균: 1116633.214 KB/sec

• Record size: 8MB

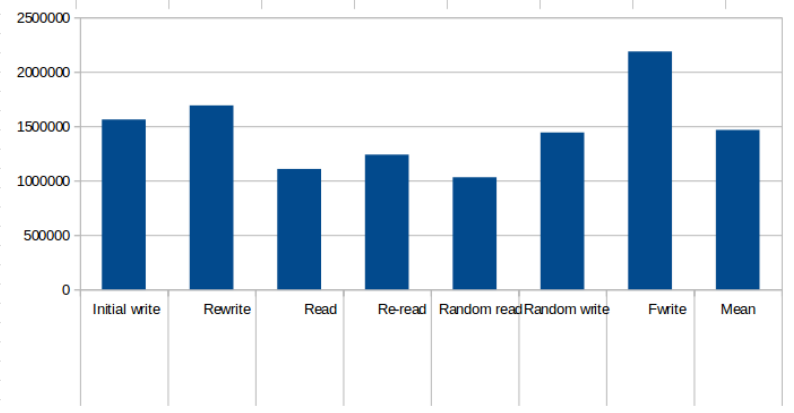
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 8m -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 8192 kBytes		
4	Output is in kBytes/sec		
5	Initial write		1171350
6	Rewrite		1222916.125
7	Read		954834.1875
8	Re-read		1115607.125
9	Random read		1056394.375
10	Random write		1308961.25
11	Fwrite		2455928.5
12	Mean		1326570.223



평균: 1326570.223 KB/sec

• Record size: 16MB

	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 16m -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 16384 kBytes		
4	Output is in kBytes/sec		
5	Initial write		1561698.25
6	Rewrite		1690850.875
7	Read		1106843.75
8	Re-read		1238671.125
9	Random read		1030163.813
10	Random write		1442289.5
11	Fwrite		2186296
12	Mean		1465259.045



평균: 1465259.045 KB/sec

Record size를 크게 설정 할수록 IOZone의 출력 값인 프로세스 당 처리율 (KB/sec)이 크게 나왔다.

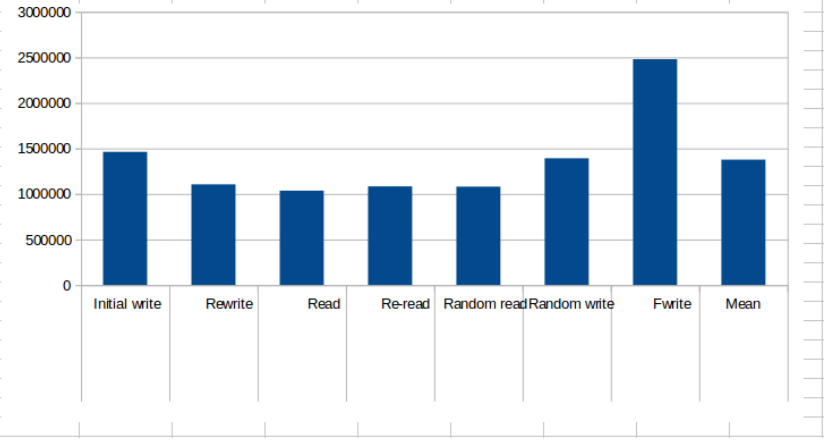
Record size를 작게 설정 했을 때는, 추가적으로 수행하는 연산인 fwrite의 프로세스 당 처리율 값만 다른 연산에 비해 너무 크게 나와서 그래프를 이용한 시각적인 비교가 어려웠는데 Record size를 크게 설정 할수록 나머지 연산도 프로세스 당 처리율이 크게 나와서 그래프를 이용한 시각적인 비교가 쉬웠다.

• **결과 분석**

- noop

• Record size: 8MB

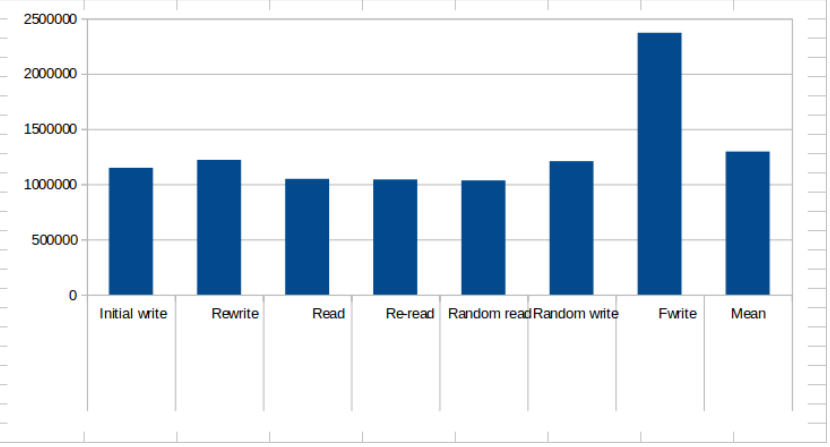
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 8m -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 8192 kBytes		
4	Output is in kBytes/sec		
5	Initial write		1463093.875
6	Rewrite		1105486.25
7	Read		1036075.563
8	Re-read		1083021.625
9	Random read		1080469.25
10	Random write		1392422.375
11	Fwrite		2479442.25
12	Mean		1377144.455



평균: 1377144.455 KB/sec

• Record size: 16MB

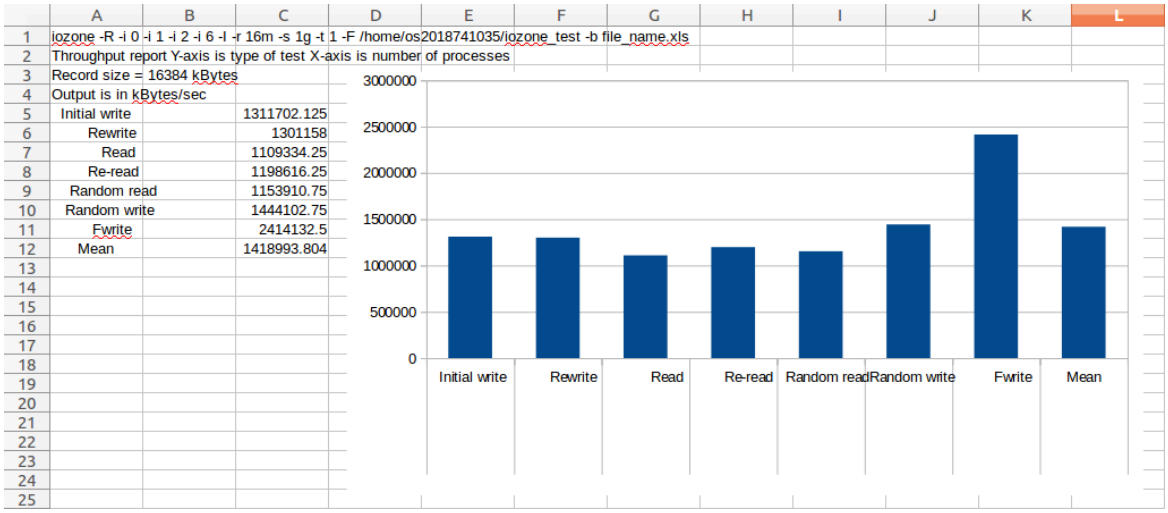
	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 6 -l -r 16m -s 1g -t 1 -F /home/os2018741035/iozone_test -b file_name.xls		
2	Throughput report Y-axis is type of test X-axis is number of processes		
3	Record size = 16384 kBytes		
4	Output is in kBytes/sec		
5	Initial write		1148192.125
6	Rewrite		1220162.625
7	Read		1048690.375
8	Re-read		1043299.938
9	Random read		1034546.5
10	Random write		1208716.25
11	Fwrite		2371641
12	Mean		1296464.116



평균: 1296464.166 KB/sec

- deadline

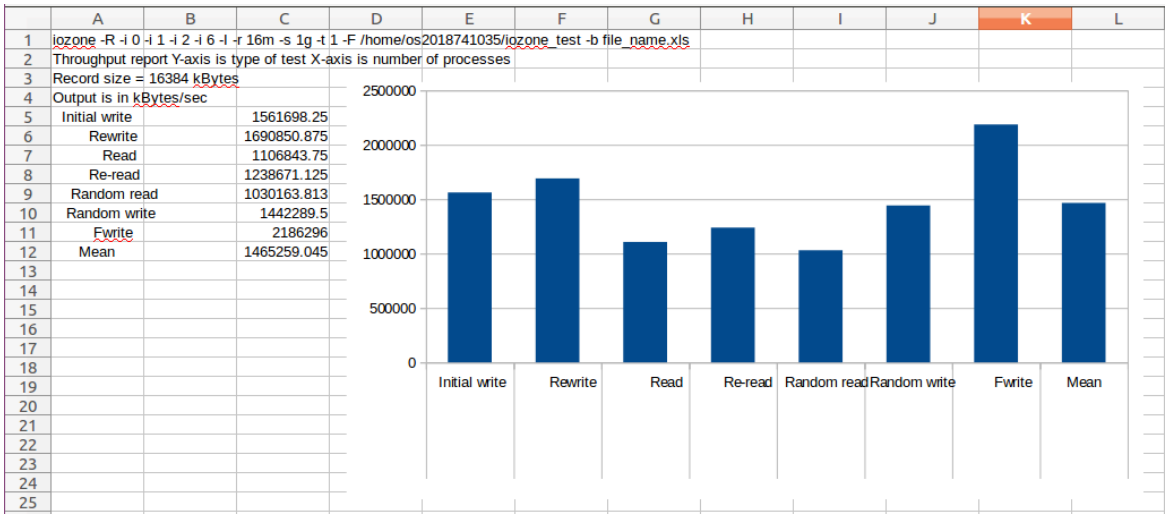
• Record size: 16MB



평균: 1418993.804 KB/sec

- cfq

• Record size: 16MB



평균: 1465259.045 KB/sec

'noop' Scheduler(16MB) : 1296464.166 KB/sec < 'noop' Scheduler(8MB) : 1377144.455 KB/sec < 'deadline' Scheduler(16MB) : 1418993.804 KB/sec < 'cfq' Scheduler(16MB) : 1465259.045 KB/sec

'noop' 스케줄러를 제외하고, 'deadline' 스케줄러와 'cfq' 스케줄러는 Record size를 크게 설정 할수록 IOZone의 출력 값인 프로세스 당 처리율 (KB/sec)이 크게 나왔다.

예외적으로 'noop' 스케줄러는 Record size를 16MB로 설정 했을 때가 8MB로 설정 했을 때보다 처리율이 작게 나왔다. 결과적으로 Record size를 8MB로 설정 했을 때 가장 빠른 처리율이 나왔다.

측정 하는 시점의 시스템 상황이나 환경에 따라 차이가 있을 수 있기 때문에, 3개의 Scheduler 중에서 무엇이 최고라고 하나를 딱 정하기는 어렵지만, 선택한 4개의 연산에 대한 성능 결과의 평균이 가장 좋은 것은 CFQ Scheduler인 것을 확인하였다.

- 고찰

I/O Zone을 설치하고 사용하는 방법과 사용하는 과정에서 옵션의 의미를 알 수 있었다. 또 성능 실험 결과를 excel file로 연동할 수 있는 방법과 연동된 결과를 이용해서 성능 결과 그래프를 작성하는 방법을 학습할 수 있었다.

초기 성능 테스트를 진행할 때, -r # 옵션을 통해 record size를 설정해주었지만 record size가 바뀌지 않고 기본 설정인 4KB로 실행되었다. 이는 buffer cache를 거치지 않고 연산을 수행하는 옵션인 -I(대문자 i)를 -l(소문자 L)로 착각하고 설정했었기 때문에 옵션들이 꼬여서 정상적으로 출력이 되지 않은 것이었고, 수정하여 정장석으로 실험은 진행할 수 있다.

테스트 연산을 어떤 것으로 수행하는지에 따라 Scheduler에 성능 결과가 다르게 나오기 때문에, 읽기 속도의 비교를 원한다던가, 쓰기 성능의 비교를 원한다던가 등과 같은 성능 실험 목적에 따라 적절한 테스트 연산을 선택하고 수행하는 것이 중요할 것 같다는 생각이 들었다.

I/O Zone은 다른 형태의 파일시스템 성능 측정 값에 대해 벤치마크를 한다. 예를 들어 읽기, 쓰기, 임의 읽기 등 특정 파일시스템에 적용하려는 응용프로그램에 따라 적절한 항목에 대해 주의를 기울여야 한다. 예를 들어 읽기 작업이 많은 OLTP 데이터 베이스가 있는 파일시스템이라면 Random Read, Random Write와 혼합 작업량에 주의를 기울여야 한다.

'deadline' 스케줄러와 'cfq' 스케줄러는 Record size를 크게 설정 할수록 IOZone의 출력 값인 프로세스 당 처리율 (KB/sec)이 크게 나왔지만, 'noop' 스케줄러는 Record size를 16MB로 설정 했을 때가 8MB로 설정 했을 때보다 처리율이 작게 나왔다. 이를 토대로, 무조건 Record size를 크게 설정 할수록 좋은 성능을 보여주는 것이 아니고 그렇기 때문에 측정 하는 시점의 시스템 상황이나 환경에 따라 적절하게 Record size와 같은 파라미터들을 설정해주는 것이 중요할 것이라고 생각이 들었다.

명령어를 통해 일일이 스케줄러를 설정하고, 실험에 영향을 주는 요소를 제거하기 위해 실행 할 때마다 캐시 및 버퍼를 비워주고, Record size를 변경하여 실험을 진행해야 했기 때문에 조금 어려움을 느꼈다.

- Reference

• 디스크 IO 성능 - I/O 스케줄러, https://www.mimul.com/blog/io-scheduler/

• iozone File System Benchmark 사용, https://m.blog.naver.com/kmk1030/220988845035