운영 체제

Assignment 4 보고서

학과 : 컴퓨터정보공학부

학번 : 2018202018

이름 : 유승재

제출 일자 : 2023. 12. 06 (수)

담당 교수 : 김태석 교수님

1. Introduction

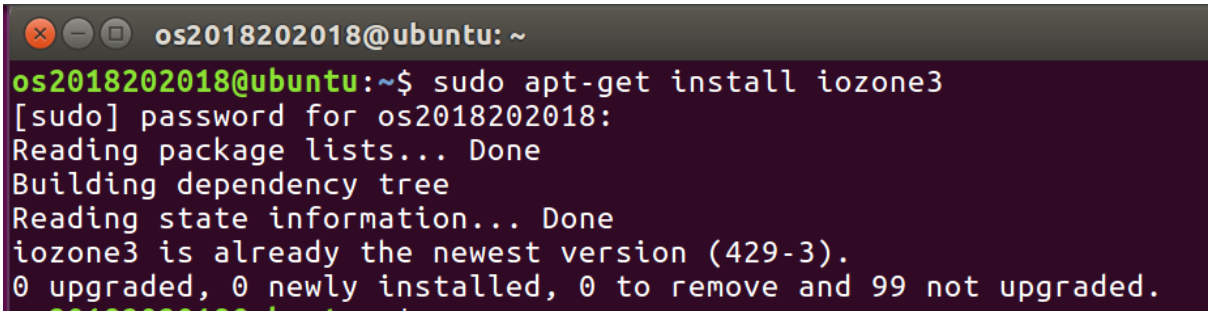
이번 과제는 I/O Zone을 이용한 세 가지 I/O Scheduler 성능을 테스트하고 결과를 비교하는 것이 목표인 과제입니다. Linux I/O Scheduler는 noop, deadline, cfq 세 종류로 나뉩니다. 이번 과제에서는 세 Scheduler 별 성능 결과를 I/O Zone을 통해 분석하고 이 성능 결과를 표와 그래프로 작성하여 보고서를 작성합니다. 이를 진행하며 record size를 변경해가며 테스트를 진행해서 차이점을 확인해봅니다.

Noop scheduler는 정렬을 하지 않는 scheduler입니다. 병합만을 수행하기 때문에 Random access를 하는 device를 위한 schedule입니다.

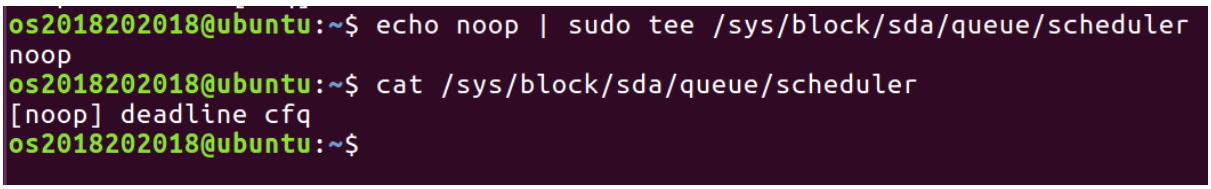
Deadline scheduler는 4개의 queue를 사용하여 기아 현상을 방지하고 읽기 우선 정책을 사용합니다. Deadline이 지난 요청이 없을 경우에는 정렬된 큐에서 요청을 꺼내서 처리합니다.

Cfq scheduler는 입출력을 요청하는 모든 프로세스들에 대해 디스크 I/O 대역폭을 공평하게 할당하는 것을 보장하는 기법입니다. I/O를 요청한 프로세스 별로 queue를 할당하고 각 프로세스에 관한 큐는 섹터 순으로 정렬합니다.

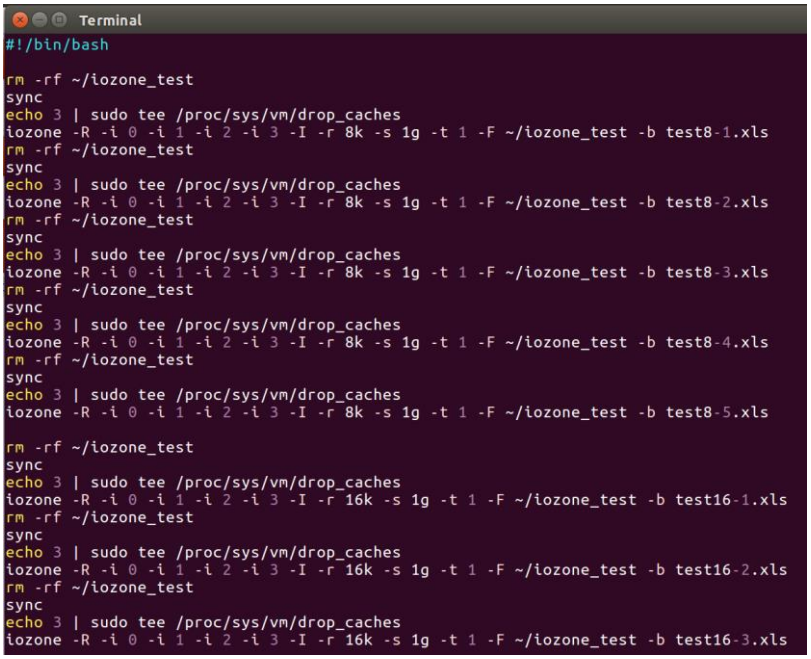
2. Conclusion & Analysis



Iozone을 설치했습니다. 이를 통해 Scheduler들의 성능 차이를 비교합니다.



그 후 scheduler를 해당 명령어를 통해서 변경을 진행해주었습니다.



그 후 많은 명령어를 실행해야 하기 때문에 저는 각 scheduler 별로 실행해야 하는 명령어를 하나의 쉘 스크립트 파일에 한번에 적어서 실행하여 결과를 출력하는 방식으로 진행했습니다. 해당하는 명령어는 iozone -R -i 0 -i 1 -i 2 -i 3 -I -r size -s 1g -t 1 -F ~/iozone_test -b 파일명 입니다.

해당 명령어는 각각의 기능을 합니다.

-R : 엑셀 report 생성

-b : 해당하는 이름으로 파일 생성

-i : 각각의 번호에 맞는 테스트 진행

위의 명령어에서는 0, 1, 2, 3을 진행했으므로 write/re-write, read/re-read, random-read/write, read-backwards를 진행했습니다.

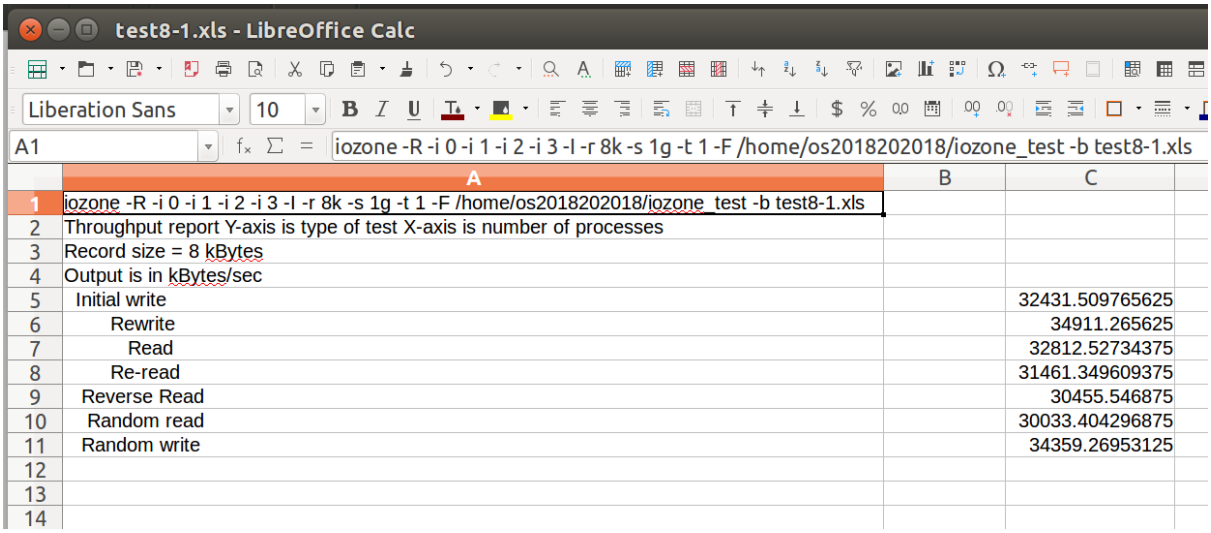
-I : 파일에 대한 모든 operation은 buffer cache를 우회 및 disk 직접 이동

-r : record size 설정

-s : 파일 사이즈 설정

이번 과제에서는 크기를 1g로 정하는 것을 권장하셨기 때문에 1g로 진행했습니다.

-F : throughput test에서 각 thread 또는 process 파일 이름



	A	B	C
1	iozone -R -i 0 -i 1 -i 2 -i 3 -I -r 8k -s 1g -t 1 -F /home/os2018202018/iozone_test -b test8-1.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		32431.509765625
6	Rewrite		34911.265625
7	Read		32812.52734375
8	Re-read		31461.349609375
9	Reverse Read		30455.546875
10	Random read		30033.404296875
11	Random write		34359.26953125
12			
13			
14			

위의 파일은 명령어를 실행한 후 나온 출력 결과 중 하나입니다. 위에서는 각 initial write, rewrite, read , re-read 등이 실행되는 동안 얼마만큼의 시간을 사용했는지를 보여줍니다.

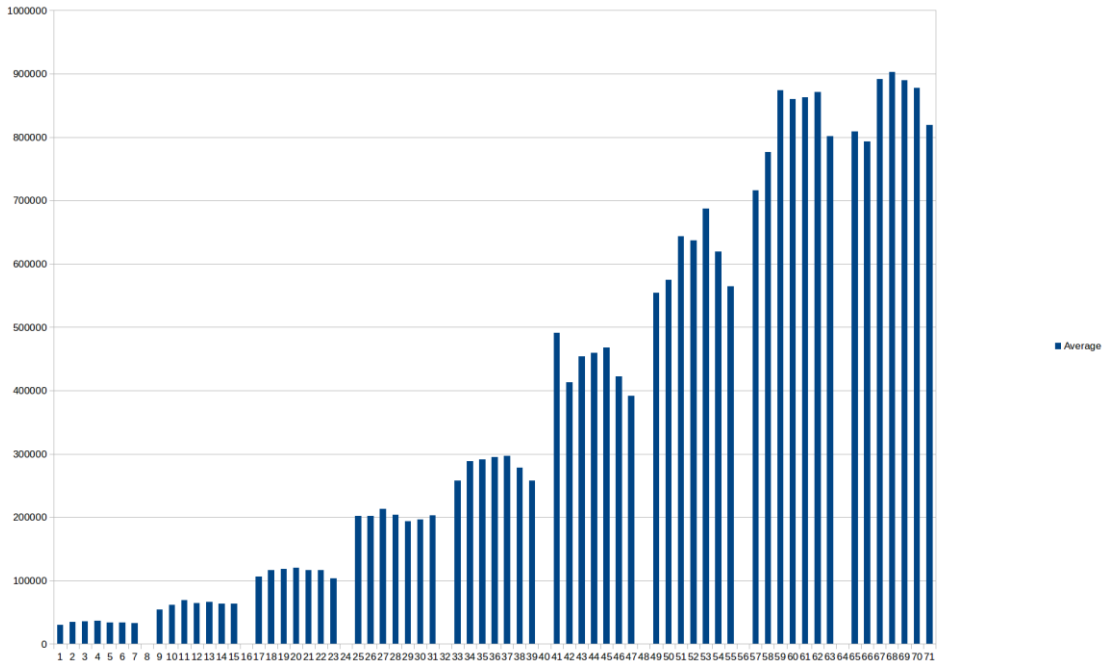
8k	1	2	3	4	5	Average
Initial write	32431.509765625	27102.771484375	30252.2734375	33360.27734375	28635.87109375	30356.540625
Rewrite	34911.265625	32995.85546875	34358.828125	36780.2734375	36741.75390625	35157.5953125
Read	32812.52734375	41256.5234375	38807.3828125	32201.3984375	34999.77734375	36015.521875
Re-read	31461.349609375	40926.60546875	41763.9375	37234.2265625	31910.76953125	36659.377734375
Reverse Read	30455.546875	33708.22265625	36365.6875	30029.791015625	38341.0390625	33780.057421875
Random read	30033.404296875	28494.45703125	39286.08984375	34134.0703125	36100.2109375	33609.646484375
Random write	34359.26953125	29826.3046875	35017.109375	35255.16796875	31117.166015625	33115.003515625
16k						
Initial write	57053.03515625	52154.31640625	61332.41796875	47412.890625	52808.078125	54152.14765625
Rewrite	60627.25390625	60339.64453125	74343.5859375	74049.8515625	41403.85546875	62152.83828125
Read	67172.484375	76036.8046875	76897.6484375	68353.2421875	59074.578125	69506.9515625
Re-read	63868.9140625	69344.03125	64350.8203125	62202.44921875	61830.61328125	64319.365625
Reverse Read	71211.375	68435.5625	68459.078125	60881.0546875	62446.58984375	66286.73203125
Random read	64644.9140625	60137.41796875	65880.140625	63850.0859375	61424.9140625	63187.49453125
Random write	64292.12890625	70765.8671875	47515.921875	69558.0625	65380.9609375	63502.58828125
32k						
Initial write	83805.015625	102330.0703125	108694.9765625	124409.8671875	113986.1796875	106645.221875
Rewrite	74670.5546875	122867.578125	124032.4453125	137511.484375	121619.9375	116140.4
Read	118819.0859375	119920.734375	114414.921875	114616.390625	122227.8203125	117999.790625
Re-read	117646.8515625	106517.921875	136629.796875	115584.03125	125693.625	120414.4453125
Reverse Read	118709.1015625	114460.3125	123592.265625	119717.171875	107731.9140625	116842.153125
Random read	128323.8515625	118953.7265625	128372.6484375	110894.0390625	95256.4765625	116360.1484375
Random write	91792.3671875	100812.515625	133639.921875	72796.5703125	119018.7890625	103612.0328125
64k						
Initial write	202626.546875	191255.703125	216512.078125	201023.4375	196946.9375	201672.940625
Rewrite	242398.078125	220084.953125	176834.0625	179855.703125	189068.71875	201648.303125
Read	197202.25	213222.125	278864.375	201803.921875	174498.1875	213118.171875
Re-read	210924.9375	207931.25	213281.4375	187146.0625	199606.15625	203777.96875
Reverse Read	140773.703125	223186.125	213616.03125	189118.765625	201207.15625	193580.35625
Random read	218381.046875	207412.171875	174161.75	197827.609375	183685.078125	196293.53125
Random write	245855.40625	216341.171875	153206.171875	177846.53125	223854.859375	203420.828125
128k						
Initial write	255659.59375	279645.6875	252340.4375	243204.6875	260319.234375	258233.928125
Rewrite	263991.40625	275966.90625	291152.46875	294352.65625	315797.71875	288252.23125
Read	261843.59375	329639.3125	265067.375	293132.25	305098.96875	290956.3
Re-read	287712.4375	324655.71875	301424.84375	262531.40625	298018.40625	294868.5625
Reverse Read	317407.5	322491.90625	264291.90625	281108.34375	297142.8125	296488.49375
Random read	266772.3125	312661.6875	307561.1875	240243.03125	261601.1875	277767.88125

256k						
Initial write	304081.5625	394639.75	416291.1875	403775.8125	935363.3125	490830.325
Rewrite	433409.21875	445105.6875	442655.25	389038.96875	353421.4375	412726.1125
Read	447475.46875	444348.03125	485943	457111.15625	435881.40625	454151.8125
Re-read	474961.9375	450326.5625	468716.75	475060.3125	429956.40625	459804.39375
Reverse Read	482153.75	485299.28125	452485.5625	457623.0625	461933.6875	467899.06875
Random read	364322.90625	434702.90625	436356.09375	455348.0625	419740	422093.99375
Random write	404936.09375	462014.46875	408692.25	272788.25	411595.125	392005.2375
512k						
Initial write	552460.4375	603991.625	551853.3125	544329.125	519941.03125	554515.10625
Rewrite	565936.0625	589992.5	609788.25	505025.5	601045	574357.4625
Read	669372	617184.125	645976.4375	658258.375	625501.75	643258.5375
Re-read	634951.4375	647386.1875	637626.375	632496.8125	631393.875	636770.9375
Reverse Read	720079.4375	681561.3125	670230.6875	664509.875	701548.5625	687585.975
Random read	610890.1875	618034.5625	636088.875	633030.6875	596983.5625	619005.575
Random write	585457.8125	543671.8125	563679.3125	585448.5625	544913.75	564634.25
8m						
Initial write	764846.8125	748205.1875	735440.6875	713796.3125	615621.3125	715582.0625
Rewrite	684274	802265.3125	808099.3125	760636.875	828538.5	776762.8
Read	876008.375	884845.125	833060.6875	894121.0625	880282.6875	873663.5875
Re-read	867265.5	899844.25	784710.0625	854822	893317.4375	859991.85
Reverse Read	876999	876897.0625	850280.4375	833870.8125	874496.125	862508.6875
Random read	887066.1875	811727	833465.1875	900760.25	924123.375	871428.4
Random write	851551.5	753213.8125	828194.3125	744985	827713.1875	801131.5625
16m						
Initial write	845555.375	799154.3125	823224.0625	791015.125	786711.3125	809132.0375
Rewrite	813968.4375	886545.625	856950.6875	797994.4375	610765	793244.8375
Read	866158.5625	952003.875	881307.25	843423.375	917330.6875	892044.75
Re-read	890467.3125	939597.5	888171.25	873333.6875	924013.3125	903116.6125
Reverse Read	872197.5625	906135.875	886148.5625	865312.4375	918489.3125	889656.75
Random read	911545.125	900580	837977.875	887400.875	851909.3125	877882.6375
Random write	771392.875	833768.6875	816455.8125	835680.5625	836694.0625	818798.4

위는 noop scheduler로 iozone을 실행했을 때의 결과값입니다.

이를 각각의 scheduler 별로 별도의 엑셀 파일을 만들어 이를 묶어서 각 명령어마다 실행된 시간의 평균을 계산하였습니다. 예시로 record size를 8k size로 진행한 5번의 소요 시간의 평균은 30356.540625 kbytes/sec이 소요된 것을 확인할 수 있었습니다.

1초에 얼마나 많은 양의 바이트를 읽을 수 있는지를 보여주는 것이기 때문에 값이 클수록 performance가 좋다고 해석할 수 있습니다.



해당 그래프는 위의 표를 그래프로 표현한 결과입니다.

순서대로 record size가 8k, 16k, 32k, 64k, 128k, 256k, 512k, 8m, 16m일 때의 각각의 명령어의 평균값을 나타내고 있습니다. 명령어는 순서대로 initial write, rewrite, read, re-read, reverse read, random read, random write를 의미합니다.

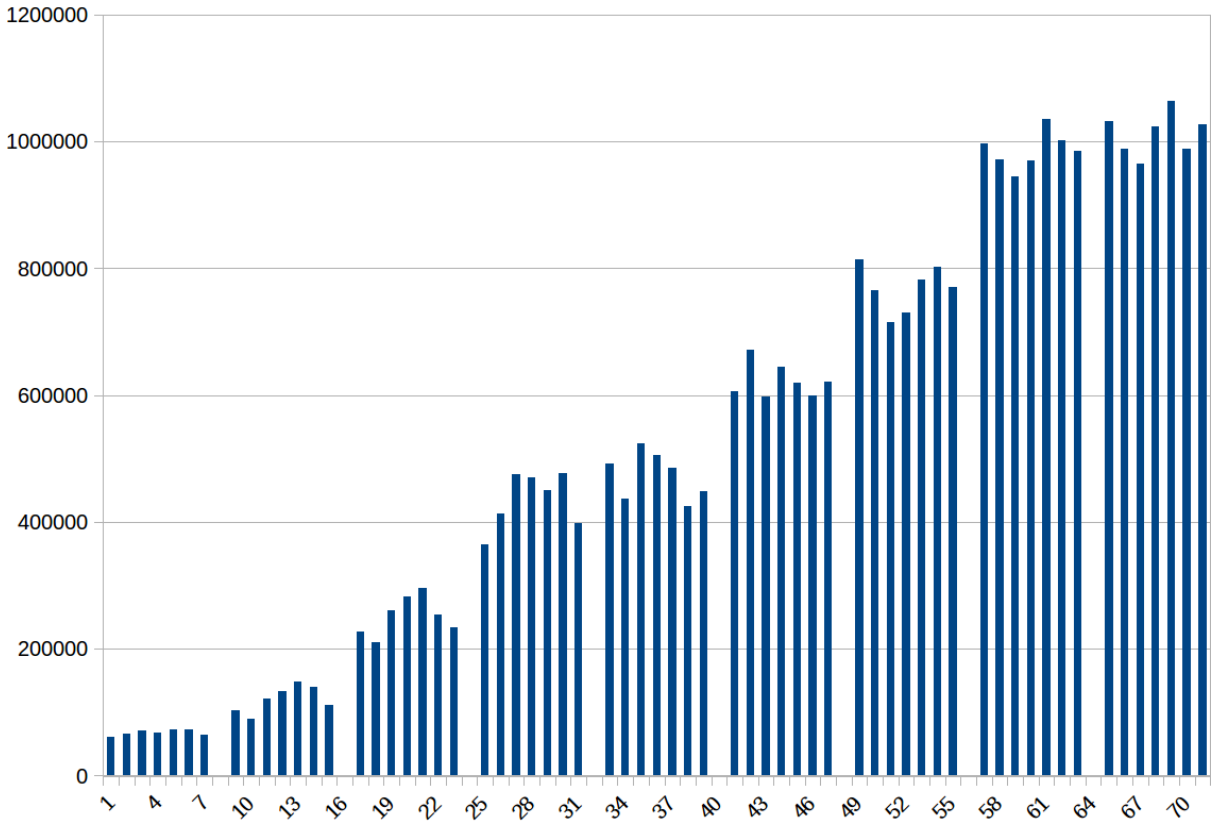
전체적인 record size가 커질수록 performance가 대체적으로 상승하는 결과를 확인할 수 있었습니다. 각 명령어에 따라 record size가 같더라도 소요되는 시간이 다르고 결과값이 가끔 튀는 경우도 발생하지만 기본적으로 같은 명령어 내에서는 record size가 커지면 반드시 performance가 좋아지는 것을 확인할 수 있었습니다. 하지만 낮은 record size에서는 record size가 2배가 될 경우 performance가 거의 2배 가까이 상승하는 것을 확인할 수 있었던 것을 볼 수 있는 방면, 8m와 16m는 그 정도까지의 상승량을 보이지 않는 것을 볼 수 있었습니다. 대부분의 모든 명령어가 균일한 수준으로 performance가 발생하는 것을 확인해볼 수 있었습니다. 이는 명령어를 따로 정렬하지 않는 noop의 특징이라고 생각합니다.

다음으로 deadline scheduler로 테스트한 결과입니다.

8k	1	2	3	4	5	Average
Initial write	24902.72852	80280.49219	69000.95313	55297.87891	71948.8125	60286.17305
Rewrite	35994.73438	75877.66406	90573.61719	60798.46484	64917.53125	65632.40234
Read	32549	100764.8516	73781.46094	76198.48438	72390.88281	71136.93594
Re-read	62350.54688	97091.90625	63621	45375.57031	72179.28125	68123.66094
Reverse Read	82399.73438	96718.57031	65787.99219	35701.75781	86862.19531	73494.05
Random read	100498.4375	85665.71875	61334.84375	34208.30859	81641.60156	72669.78203
Random write	67638.71875	76491.60156	55627.39063	56994.875	66455.70313	64641.65781
16k						
Initial write	115036.1172	108292.1875	104949.0391	101800.2813	85312.99219	103078.1234
Rewrite	102753.8594	84237.39063	89314.36719	84062.86719	89494.60156	89972.61719
Read	136098.4375	91777.65625	121613.0391	113284.5703	141845.8125	120923.9031
Re-read	162707.7656	135277.8906	133258.5156	113770.5547	121932.2656	133389.3984
Reverse Read	139823.6563	138003.7813	136750.7344	162402.3125	161173.6563	147630.8281
Random read	157737.0313	129013.6406	128835.2734	158970.7344	129424.7891	140796.2938
Random write	123039.8984	106562.2813	110659.0391	121985.1328	97578.60156	111964.9906
32k						
Initial write	218998.5156	222228.3438	216136.5313	265792.8125	213590.6406	227349.3688
Rewrite	173684.5313	224977.0625	247359	198480.0313	207449.4844	210390.0219
Read	236012.4219	253337.4688	288575.9063	268829.5938	258161.1719	260983.3125
Re-read	249982.125	294406.4375	286791.9688	300185.4375	284600.3125	283193.2563
Reverse Read	251850.2656	297981.9063	323788.2813	302397.8438	304252.1875	296054.0969
Random read	239021.2813	282426.4375	277652.125	237171.2344	232784.1563	253811.0469
Random write	251119.6094	211215.7969	266959.25	220223.2344	215872	233077.9781
64k						
Initial write	352155.5625	352081.3125	394065.7813	371277.0938	355668.8125	365049.7125
Rewrite	381941.75	445003.8438	379434.6563	402257.375	460454.7188	413818.4688
Read	428465.8438	445753	537200.375	454670.2188	513697.4063	475957.3688
Re-read	544820.0625	425176.5938	438890.2188	439731.9688	506078.5313	470939.475
Reverse Read	437333.5938	432776.7188	470164.125	445494.6875	465951.125	450344.05
Random read	425153.9688	496092.7813	431965.5	529878.75	503814.7188	477381.1438
Random write	384916.3125	476985.3125	379504	370945.875	380572.8438	398584.8688
128k						
Initial write	560838.375	455377.5	590817.5625	425717.4063	430004	492550.9688
Rewrite	409001.2188	446144.0313	502491.9375	360387.5313	465888	436782.5438
Read	659742.375	622653.5	438060.375	434502.4063	466674.25	524326.5813
Re-read	631137	491222.0938	475449.3125	433755.0625	493070.0313	504926.7
Reverse Read	478546.5938	500497.5938	489127.3125	436382.8125	526139.1875	486138.7
Random read	445067.125	401583.2188	389849.5313	416359.6563	471954.1563	424962.7375
Random write	484009.4688	435784.5	440446.5625	451169.4375	430058.1563	448293.625

256k						
Initial write	675121.1875	654856.25	602453.9375	539050.4375	559165.125	606129.3875
Rewrite	813276.375	626129.1875	675525.25	527591.5625	718413.875	672187.25
Read	548493.5	611330.0625	559326.375	629867.3125	640094.125	597822.275
Re-read	596373.4375	582631.1875	835440.75	663418.375	549722.5625	645517.2625
Reverse Read	654342.9375	584289.875	515840.5938	860199.9375	482995.0313	619533.675
Random read	560212.1875	606924.0625	553359.5625	624868.8125	651240.875	599321.1
Random write	850343.0625	620445.5625	525114.8125	559225.625	553824.8125	621790.775
512k						
Initial write	854055.3125	748852.375	708516	874887.875	883737.625	814009.8375
Rewrite	786149.1875	745661.375	722681.375	823087.0625	747989.625	765113.725
Read	728431.25	683758.5625	806354.4375	642097.8125	715080.3125	715144.475
Re-read	636228.125	721013.125	695440.4375	866596	729813.25	729818.1875
Reverse Read	757972.3125	738009.25	769602.0625	876701.3125	767814.5625	782019.9
Random read	840562.8125	724472.875	880261.9375	908587.0625	655956.0625	801968.15
Random write	725423.6875	764513.875	798537.75	790203.3125	776454.375	771026.6
8m						
Initial write	981534.375	945540.3125	1035956	1046045.625	975705.375	996956.3375
Rewrite	1129915.125	1004292.625	971577.4375	804558.75	946127.5625	971294.3
Read	995754.9375	891863.875	977322.375	922875.75	935508.5625	944665.1
Re-read	897854.8125	914056.0625	1009691.75	909458.8125	1118307.125	969873.7125
Reverse Read	1043142.063	1028432.063	1111309.375	965600.0625	1030750.063	1035846.725
Random read	1049744.375	980730	939904.375	967974.75	1072299.625	1002130.625
Random write	987146.6875	931397.625	1010339.813	1061128	932418.125	984486.05
16m						
Initial write	1117993.5	1054524.625	1072244.75	998212.0625	920573.125	1032709.613
Rewrite	966863.5	964077.5625	963752.1875	964314.3125	1081291.625	988059.8375
Read	998726.625	909082.6875	976408.6875	994556.5625	950854.5625	965925.825
Re-read	1029866.375	998021.125	924851.1875	1174373.625	990561.75	1023534.813
Reverse Read	1094331.125	992095.9375	1041466.75	1082187.5	1112391.75	1064494.613
Random read	911626	906067.75	1015754.063	1075809.25	1029631.813	987777.775
Random write	988416.8125	1010898.563	885500.375	1042097.25	1206331.5	1026648.9

이를 그래프로 그렸을 때의 결과입니다.



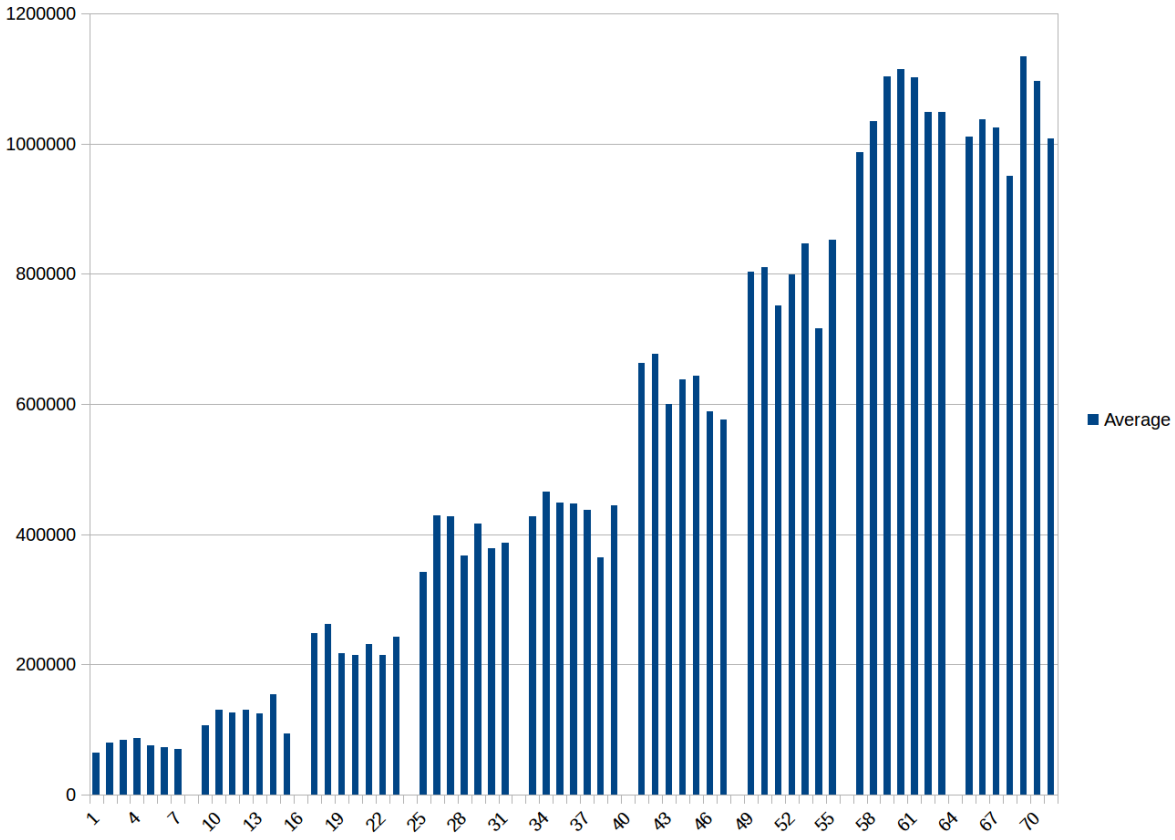
해당 결과 또한 위와 마찬가지로 record size별로 나누고 그 안의 명령어들의 평균을 나타낸 그래프입니다. noop와 동일한 양상을 볼 수 있었습니다. Deadline 또한 noop와 비슷하게 8m와 16m의 performance 차이가 크지 않은 것을 확인할 수 있었습니다. 또한 record size가 증가할수록 거의 2배씩 performance가 증가하는 모습을 확인할 수 있었습니다. Deadline은 read 정책을 사용하는 정책이고 그로 인해 read performance가 다른 명령어에 비해 좀 더 높은 performance가 나오는 것을 확인할 수 있었습니다.

마지막으로 cfq의 결과를 확인해보았습니다.

8k	1	2	3	4	5	Average
Initial write	73758.48438	75925.63281	47221.42188	72158.51563	55821.66016	64977.14297
Rewrite	83317.8125	82089.82813	79628.14844	73744.69531	81111.10156	79978.31719
Read	88654.57813	105501.6719	64795.19531	90666.17969	67900.53906	83503.63281
Re-read	82192.20313	109443.8672	70198.92188	83388.76563	85942.59375	86233.27031
Reverse Read	73408.47656	93782.6875	59203.89453	80945.25	73610.17188	76190.09609
Random read	65213.51563	73308.86719	56059.01172	87134.80469	81584.70313	72660.18047
Random write	76675.54688	79500.25	65381.72266	75958.42969	49432.36328	69389.6625
16k						
Initial write	102195.9531	142655.0313	101447.5859	68218.72656	117360.4844	106375.5563
Rewrite	160077.9688	160378.4375	80158.5	91207.4375	157160.7656	129796.6219
Read	141479.6094	130194.5391	107101.6797	116609.6719	133608.2813	125798.7563
Re-read	164799.8125	127874.5156	120023.0234	104112.6719	136735	130709.0047
Reverse Read	160401.3906	110313.9531	126337.2266	110099.8516	119776.9453	125385.8734
Random read	114252.125	154653.875	202028.5781	186015.3125	116445.2656	154679.0313
Random write	98909.35156	78479.63281	77967.71094	76599.92188	140531.1719	94497.55781
32k						
Initial write	247190.6406	264713.2188	260803.3281	243311.5781	227492.1719	248702.1875
Rewrite	248877.2344	282507.0313	248486.3906	279209.7813	248734.4219	261562.9719
Read	218047.3906	198475.5156	218336.7656	225398.6719	226618.875	217375.4438
Re-read	191294.3281	207163.4531	222876.3594	235409.8594	216250.7344	214598.9469
Reverse Read	224518.7656	226162.625	261976.6719	240287.3906	201768.0469	230942.7
Random read	194966.8594	244960.4063	207132.3438	205283.5156	217767.5	214022.125
Random write	244200.8438	279707.2188	251128.9219	191652.8438	248455.1875	243029.0031
64k						
Initial write	334806.5313	363044.625	352521.0625	326656.0313	330798.1875	341565.2875
Rewrite	411252.2188	472060.5938	503804.0938	440446.5625	320041.9063	429521.075
Read	374306.5313	425864.1563	456496.4375	422564.9688	456632.625	427172.9438
Re-read	383044.5313	377594.7188	315979.375	361197.1563	396738.0938	366910.775
Reverse Read	398597.1563	427460.5313	395234.5938	424324.7188	438075.0313	416738.4063
Random read	366291.9688	358686.9063	406019.5625	397328.4375	360950.25	377855.425
Random write	375086.5938	402606.7188	397080.1875	351878.0938	405570.4063	386444.4
128k						
Initial write	381236.1875	420176.7813	469819.6563	444681.75	418270.4063	426836.9563
Rewrite	436365.5625	549943.4375	478741.4688	511092.3125	352208.3125	465670.2188
Read	426648.5938	414364.375	442975.5938	523528.3438	432139.8438	447931.35
Re-read	461592.2188	499025.8438	374300.125	492203.6563	405616.875	446547.7438
Reverse Read	437676.5938	382847.3438	379981.9063	548068.625	438941.3438	437503.1625
Random read	428762.3125	491933.7813	401045.1875	17336.67969	482248.5	364265.2922
Random write	504263.9063	471844.5938	415866.9375	432265.2188	399833	444814.7313

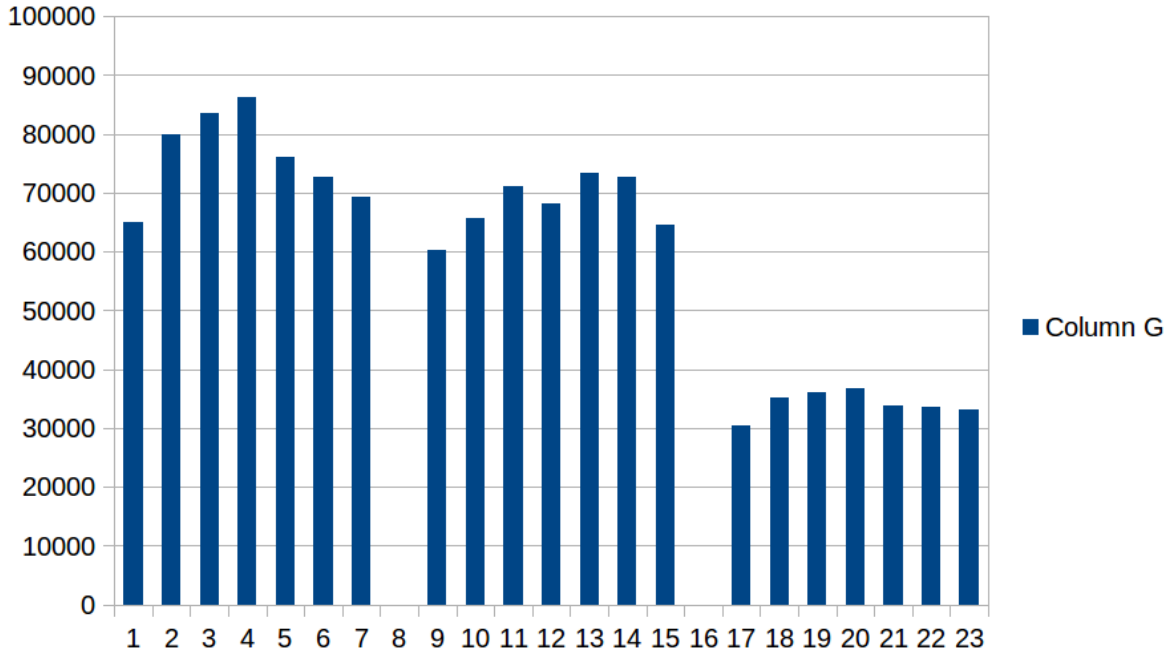
256k						
Initial write	644878.25	617647.875	813129.4375	611346.875	627136.875	662827.8625
Rewrite	621564.75	597452	705641.8125	673784.25	783508.25	676390.2125
Read	727914.875	519231.75	583975.5625	598954.1875	571176.4375	600250.5625
Re-read	690168.125	658474.1875	504383.9688	711726.5625	624156.875	637781.9438
Reverse Read	722628.625	754034.375	533592.25	670951.25	535624.5	643366.2
Random read	541815.6875	549037.1875	614676.8125	534027.8125	703196.1875	588550.7375
Random write	586696	517904.5625	577124.9375	579324.0625	616626.5625	575535.225
512k						
Initial write	705893.5625	876638.875	889883	743708.25	801257.5625	803476.25
Rewrite	732231.4375	834328.75	843346.75	800604.0625	843974.5	810897.1
Read	714528.25	737143.25	651876.875	774806.625	878419.25	751354.85
Re-read	768716.3125	814698.8125	746827.1875	732266.3125	929166.8125	798335.0875
Reverse Read	915120.375	880011.4375	865185.25	764970.625	807821.75	846621.8875
Random read	657193.5625	781642.0625	714078.125	807344.6875	618405.625	715732.8125
Random write	851384.1875	830959.3125	974983.3125	789459.625	816278.0625	852612.9
8m						
Initial write	1035253.438	1046699.125	957002.5	879891.75	1014337	986636.7625
Rewrite	1015624.813	1073657.75	1021444.438	974578.25	1089493.25	1034959.7
Read	1633125.75	866947.875	978642.1875	1006922.563	1032933.375	1103714.35
Re-read	1717695.875	935151.5	895675.1875	1059964.125	966748.3125	1115047
Reverse Read	1109188.875	1109611.375	1111031.375	1091098.5	1088149	1101815.825
Random read	958986.5625	967925.5625	1084889.25	1194650.75	1036932.25	1048676.875
Random write	1012744.938	1090370	1049268.5	986725	1101891	1048199.888
16m						
Initial write	911367.75	998047.8125	1121193.375	1012538.625	1008394.375	1010308.388
Rewrite	950417.9375	964740.9375	1256780.875	1000815.938	1016999.188	1037950.975
Read	1073200.875	1116136.875	925839.25	983789.4375	1026534.25	1025100.138
Re-read	844289.8125	964560.0625	1024529.188	992356.9375	923673.875	949881.975
Reverse Read	1188572	1196017.5	978915.375	1175246.375	1128009.125	1133352.075
Random read	1036079.75	1200499.25	1064961.375	980774.1875	1199392.25	1096341.363
Random write	1009185.5	1047800.563	1009483.625	983444.5	987869.3125	1007556.7

위와 마찬가지로 표로 나타낸 결과입니다. 이를 그래프로 표현해보았습니다.



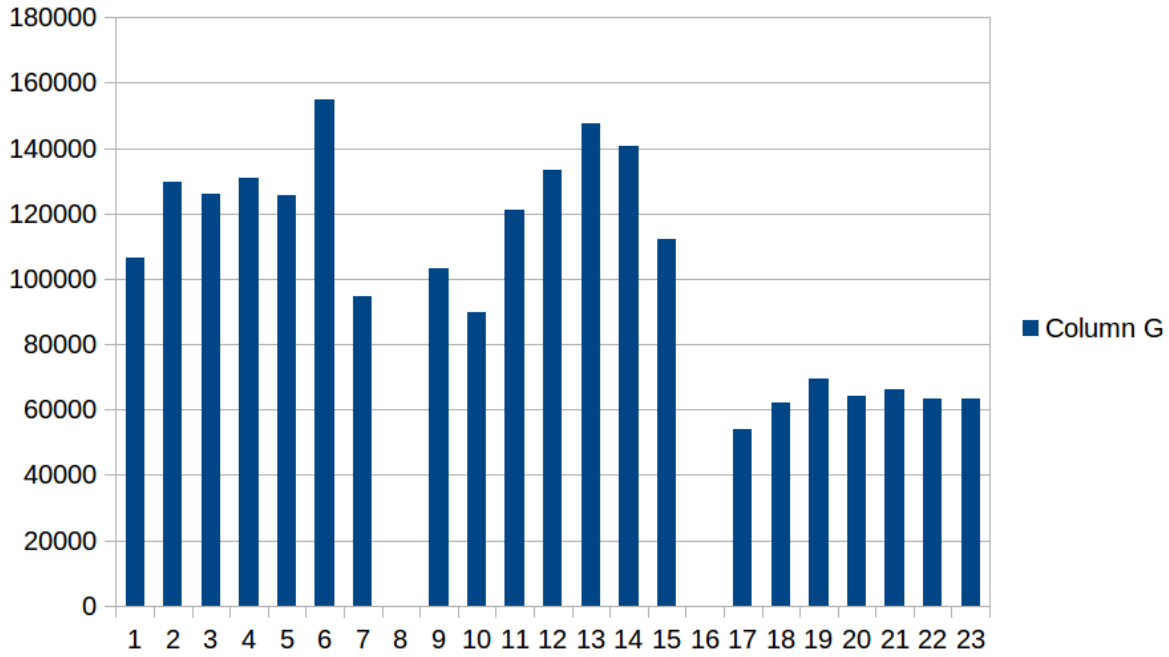
cfq에서는 record size가 낮은 경우에는 이전과 비슷한 양상을 보였습니다. 하지만 cfq가 유난히 performance가 중간중간 낮은 결과가 자주 나온 경향이 있었습니다. 또한 cfq에서만 record size가 16m일 경우에 8m일 경우보다 오히려 대체적으로 낮은 모습이 많이 보였습니다. 하지만 cfq scheduler는 대역폭을 공평하게 할당하는 기법이므로 record size가 커질수록 전체적인 performance는 차이가 커지는 모습을 확인할 수 있었습니다.

이제 각각의 scheduler 별로 각 record size에서 얼마나 performance의 차이가 있는지를 그래프를 통해서 확인해보겠습니다

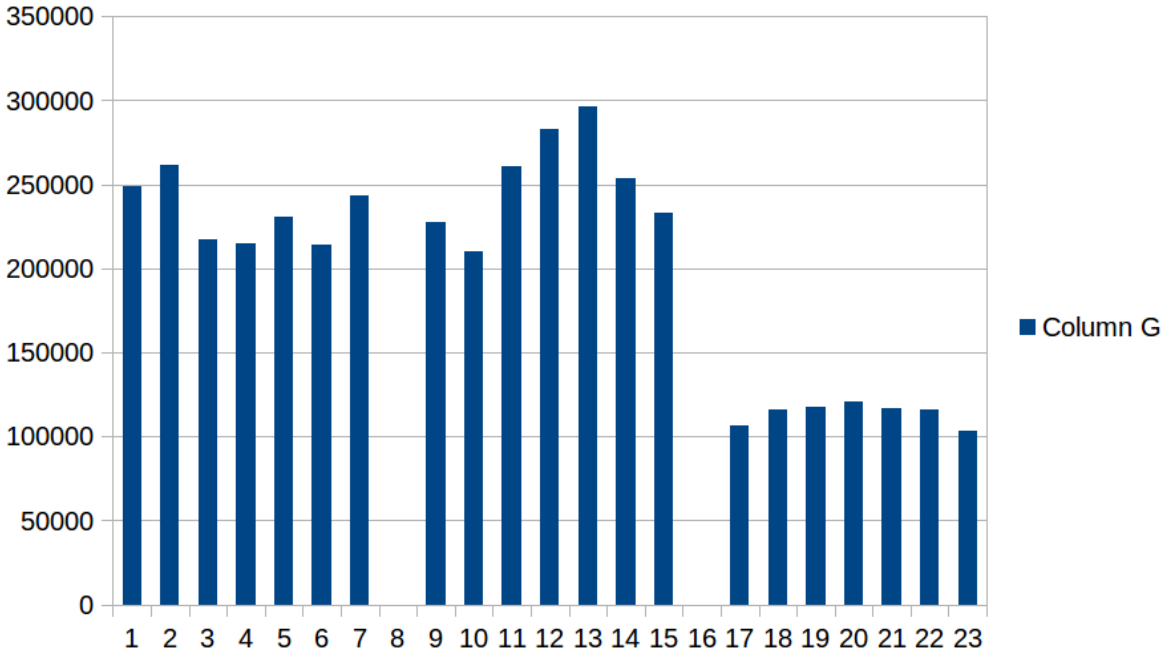


왼쪽부터 cfs, deadline, noop 순으로 record size가 8k 일 때의 performance를 확인했습니다.

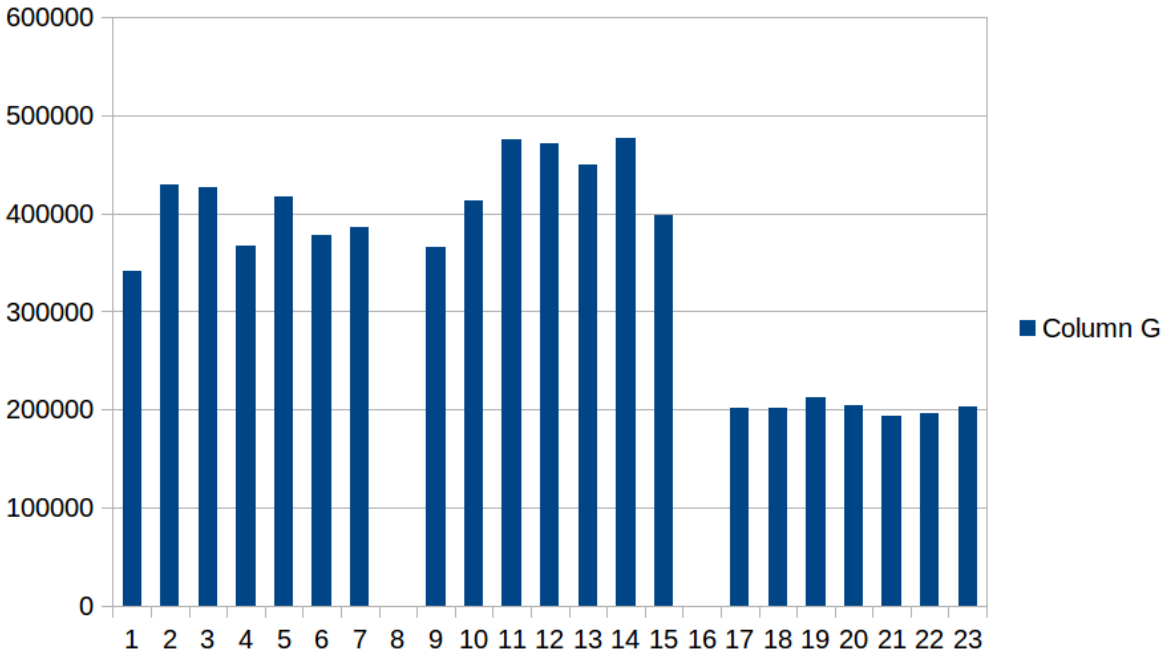
전체적으로 cfs가 deadline보다 performance가 높고, noop가 많이 낮은 것을 확인할 수 있었습니다.



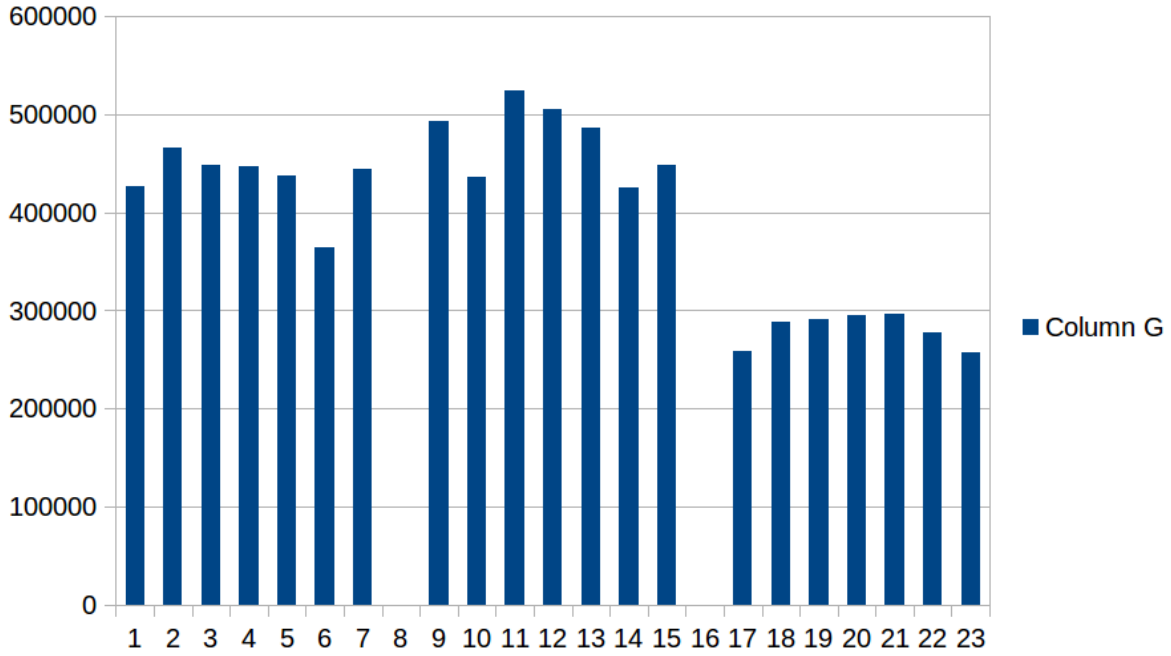
16k의 performance 입니다. 여전히 noop가 가장 낮은 performance를 보여주고 있는 것을 확인할 수 있었습니다. 또한 cfs와 deadline의 performance가 비슷하지만 cfs가 근소하게 더 좋은 것을 확인할 수 있었습니다.



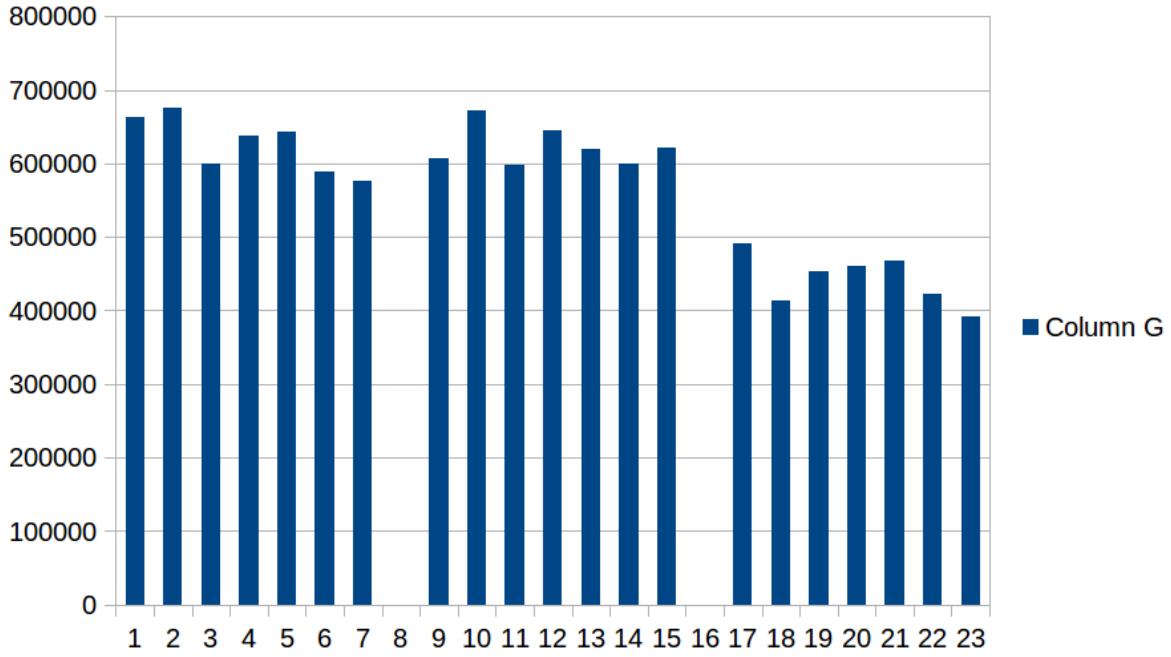
Record size가 32k일 경우입니다. Noop가 가장 performance가 낮고 deadline이 cfs보다 조금 더 좋은 performance가 나타나는 것을 확인할 수 있었습니다.



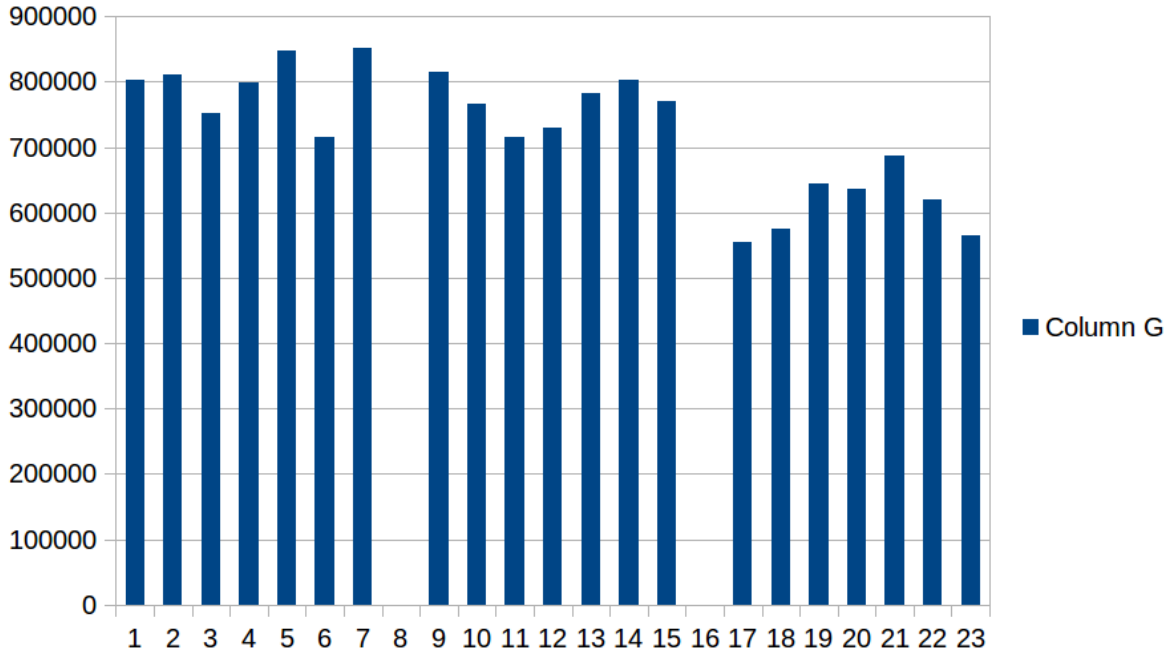
Record size가 64k일 경우입니다. Noop는 여전히 가장 낮은 performance를 가지고 있고 64k 부터는 deadline이 cfs보다 더 높은 performance를 보여주는 것을 확인할 수 있었습니다.



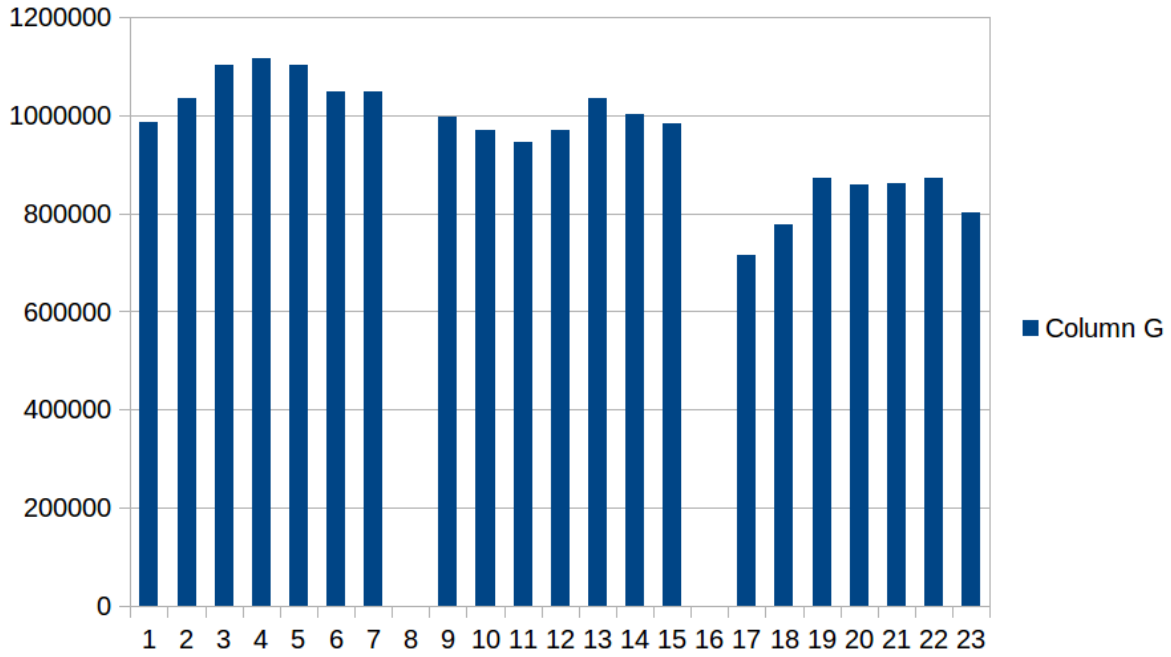
Record size가 128k인 경우입니다. Deadline이 cfs보다 확실히 더 좋은 performance를 보여주며 noop은 나머지 두 scheduler보다 거의 절반에 해당하는 performance를 보여주고 있습니다.



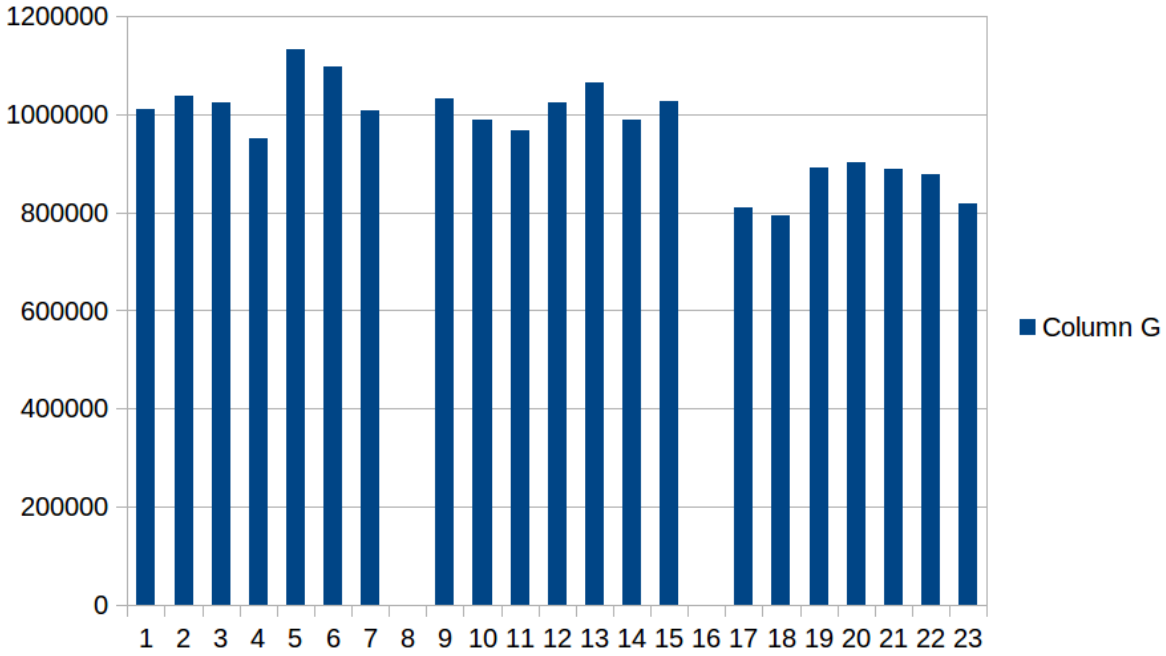
Record size가 256k인 경우입니다. Noop가 이전에 비해서 performance가 더 좋아진 것을 확인할 수 있었습니다.



Record size가 512k인 경우입니다. Deadline과 cfs의 큰 차이가 보이지 않습니다.



Record size가 8m인 경우입니다. Cfs가 record size가 커지면서 deadline보다 더 큰 차이를 보이는 것을 확인할 수 있었습니다. 또한 noop의 performance는 여전히 두 scheduler에 비해서 낮은 것을 확인할 수 있었습니다.



Record size가 16m인 경우입니다. Cfs가 8m보다 16m에서 더 좋지 않은 performance를 보여주지만 deadline보다 좀 더 좋은 performance를 보여주는 것을 확인할 수 있었습니다.

3. 고찰

이번 과제에서는 각 scheduler에 대해서 45번의 테스트를 통해서 결과를 압축하고 평균을 구하여 각 scheduler 별로 비교하는 과정을 거쳤습니다. 처음에는 많은 테스트를 거치고 오랜 시간을 기다린다고 생각하니 쉽지 않다고 생각했지만 테스트한 결과를 확인하면서 performance가 유난히 낮게 나오는 값이 은근히 많은 것을 확인하고 이런 결과는 제 컴퓨터 성능에 따라서 발생하는 오차 같은 것이라고 생각했습니다. 이러한 오차가 발생하는 경우가 생각보다 종종 있었기 때문에 테스트를 끝내고 보니 각 5번씩 테스트한 결과도 너무 적은 시행 횟수라고 생각했고 정확한 테스트를 위해서는 더 많은 테스트를 진행해보아야 할 것 같다고 생각했습니다. 하지만 시간이 부족하여 더 많은 테스트를 진행해보지는 못하여 아쉬움이 남습니다. 또한 이론적으로 알고 있는 scheduler의 기능과 실제로 관측되는 performance의 값이 다르게 나오는 것이 생각보다 적지 않아 아쉬움이 남았습니다.

4. Reference