Adam Holt & Joe Cosenza
CSCI3753 Programming Assignment 4 Report

Test Environment: Course VM – Ubuntu 14.04
Memory: 3.8 GiB
Processor: Intel Core i7 2.3 GHz
OS Architecture: 64 bit

Abstract:

We ran a series of tests in order to compare the run times and number of context switches for processes that are primarily CPU-bound, I/O-bound, and a combination of the two, by using different scheduling policies and a varying number of simultaneous processes. On average, for CPU-bound processes, the CFS scheduling policy completed the fastest. For I/O-bound processes, a CFS scheduling policy had longer run times than the other two and simultaneously more context switches. For mixed processes CFS was again slower to complete than the other two policies and also had more context switches. In all cases, I/O-bound and mixed processes resulted in significantly more voluntary context switches than involuntary.

Introduction:

Linux utilizes several different scheduling policies depending on whether the tasks fall into Real Time (RT) or Completely Fair Scheduler (CFS) classes. For these tests, we compared the results of processes running on **SCHED_RR** and **SCHED_FIFO** which are in the RT class and **SCHED_OTHER** which is in the CFS class.

Utilizing these three policies, we ran tests in order to analyze the run times and number of context switches the occurred during process execution. We created benchmark tests to test each of the scheduling policies based on operations that are primarily CPU-bound, primarily I/O-bound, and a combination of the two. Additionally, we ran each of these three process types using low (10), medium (100), and high (1000) number of simultaneous processes. With the data generated from these benchmark tests we created a series of graphs in order to visually represent the data and analyze the results.

Method:

In order to compare the performance of three different scheduling policies, First-in First-out (FIFO), Round Robin (RR), and Completely Fair Scheduler (CFS), we created a series of benchmark tests to run processes that are primarily CPU-bound, I/O-bound, and a combination. For CPU-bound processes we used an algorithm to calculate the value of pi, for I/O-bound processes we used a function that reads and writes data to a series of files. For mixed processes we used a program that performs a series of instruction using both of the previous tasks. Each of the programs takes parameters that determine the scheduling policy to use

(FIFO, RR, or CFS) as well as the number of processes to fork (low(10), medium(100), or high(1000)).

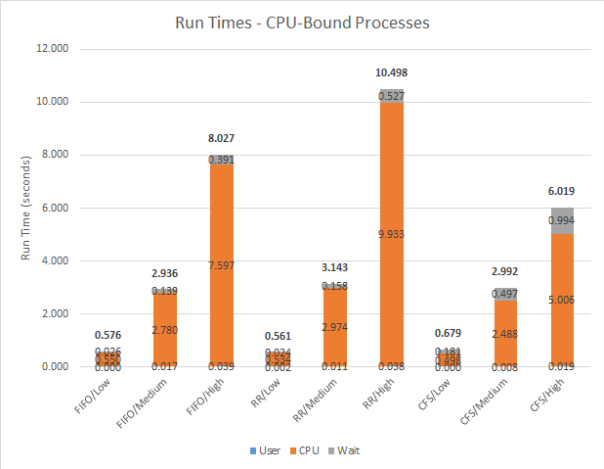
Since we have three different scheduling policies, three types of process, and three quantities of simultaneous processes we end up with 27 uniques tests. In order to run all of the tests in order we used a shell script that calls each of the functions, runs them using the appropriate scheduling policy and number of processes, then outputs the results to a series of .csv files. Each test was run a total of nine times in order to receive an array of data that we could take averages values for.

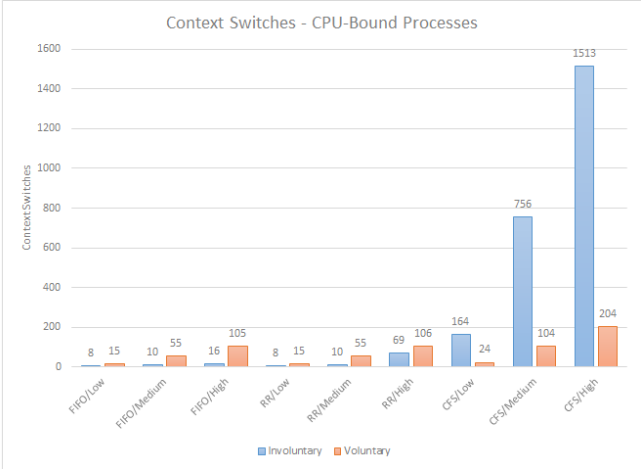
Once we had the data in .csv format we imported each set into Microsoft Excel where we took the average results for each of the nine tests and created graphs based on run times and context switches. Each graph contains data for one process types and is compared in terms of scheduling policy and number of processes used.

Results:

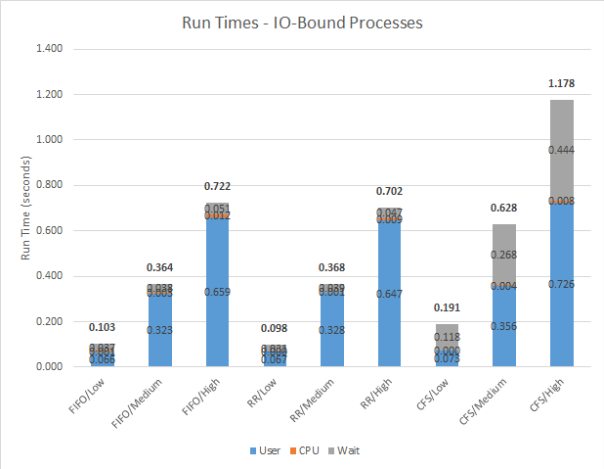
Each pair of graphs represents one process type. The data is divided into run time and context switches. Within the run time graphs, the CPU, User, and Wait times are stacked on top of each other in order to depict the total run time. In the context switch graphs, involuntary and voluntary switches are displayed side-by-side.

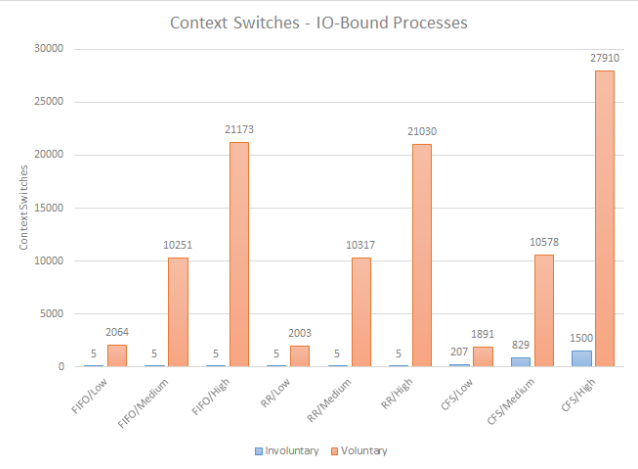
CPU-Bound Processes



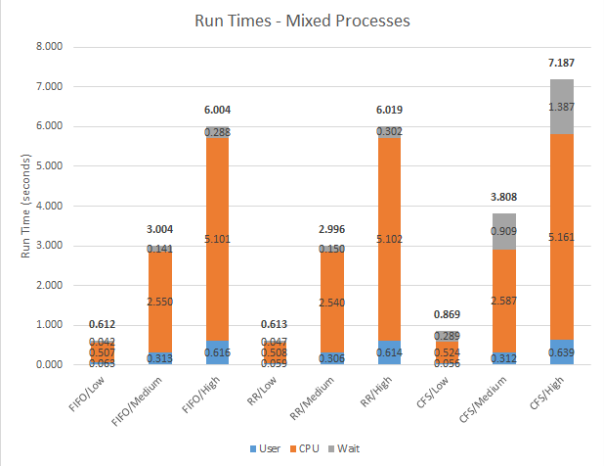


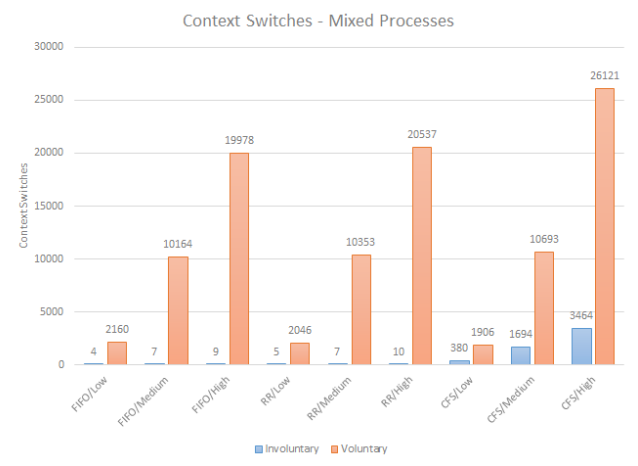
I/O-Bound Processes





Mixed Processes





Analysis:

For CPU-bound processes, the amount of time spent in the CPU predictably increased as the number of simultaneous processes did. As the processes increased, the amount of CPU time for the CFS scheduler increased at a slower pace than the other two policies so that even though the CFS policy spent more wait time caused by more context switches, it was consistently faster.

Context switches for CPU-bound processes, especially involuntary ones were noticeably higher for a CFS scheduler. It makes sense that there would be primarily involuntary context switches since a CPU-bound process would need to retain control over the CPU until the operation is complete, so it would be rare for a process to surrender its control before the end of a time slice. The higher number of context switches correlates to the longer wait time for the CFS scheduler and both increase sharply at higher number of processes.

For I/O-bound processes the runtimes again increased as the number of processes did, though this time with the bulk of the time spent on user time. For these processes, both FIFO

and RR scheduling policies completed in roughly the same amount of time while the CFS scheduler became progressively slower, partially due to increasingly long wait time.

Context switches were again higher for the CFS scheduling policy, though for these processes the overwhelming majority of context switches were voluntary. Since these processes relied so heavily on I/O operations, it follows that there would be more voluntary context switches as each operation can complete prior to the end of its time slice. Even though the majority of context switches for the CFS scheduling policy were voluntary, the total of context switches is still significantly higher than the other two policies, resulting in longer wait time, and thus longer overall run time.

Processes that combined CPU- and I/O-bound actions were similar to the I/O-bound operations, except for an increase in CPU and Wait times. It makes sense that the user time should be the same when comparing I/O-bound and mixed processes since adding the CPU-bound operations would not impact that time. It also follows that run times for mixed processes should be a sum of the times for CPU- and I/O-bound processes since our test operations combine those functions.

The number of voluntary context switches for mixed processes is roughly the same as for I/O-bound operations while involuntary switches seem to be a combination of those for CPU- and I/O-bound. Again, this follows logically since there were very few voluntary context switches for the CPU-bound tests.

Conclusion:

For CPU-bound operations, the CFS scheduler is faster in general, and increasingly so as the number of processes increases. As the number of processes increases, the number of context switches, and thus wait time, increases, however, since the CPU time decreases at a faster rate, it remains a faster scheduling policy than FIFO or RR.

For I/O-bound operations, the CFS scheduler is slower for all levels of process and becomes increasingly slower as the number of context switches increases and creates more wait time. While the User times remains more or less consistent regardless of scheduling policy, the increasing in switching makes the CFS scheduler slower and slower compared to the other two policies.

Based on the way that we set up our tests, mixed processes behave, as expected, as a sum of the CPU- and I/O-bound tests. While the CPU times for the CFS scheduler are similar to the other policies, the increases in context switching which results in longer wait times makes the CFS scheduler increasingly slower than the other two policies.

A CFS scheduler seems well-suited to CPU-intensive tasks, such as mathematical calculations, but the number of context switches inherent in the policy make it less suited to tasks that require lots of user input or other I/O. FIFO and RR scheduling policies both behaved largely the same in our tests and seem to lend themselves well to more interactive and real time types of processes as well as a combination of CPU- and I/O-bound processes. These would both be good scheduling policies for running video games.

References:

Silberschatz, Abraham, Galvin, Peter Baer and Gagne, Greg. 2013. *Operating System Concepts.* s.l. : John Wiley & Sons, Inc., 2013.

Appendix A:

CPU-Bound							
Scheduler	**Processes**	**Wall**	**CPU**	**User**	**Processor**	**Involuntary**	**Voluntary**
FIFO	Low	0.57	0.55	0	97%	8	15
		0.58	0.55	0	95%	9	15
		0.58	0.55	0	95%	9	15
		0.58	0.56	0	96%	10	15
		0.56	0.53	0	96%	6	15
		0.56	0.55	0	97%	7	15
		0.57	0.55	0	96%	8	15
		0.61	0.57	0	93%	5	15
		0.57	0.54	0	96%	7	15
	Medium	2.86	2.73	0.01	96%	9	55
		2.81	2.7	0.02	96%	7	55
		2.92	2.76	0.01	95%	8	55
		3.16	2.94	0.03	93%	10	55
		2.7	2.59	0.01	96%	10	55
		2.92	2.74	0.02	94%	11	55
		3.07	2.91	0.02	95%	13	55
		2.98	2.82	0.01	95%	13	55
		3	2.83	0.02	95%	12	55
	High	7.22	6.85	0.03	95%	15	105
		7.78	7.35	0.04	95%	14	105
		7.52	7.14	0.04	95%	18	105
		7.63	7.2	0.03	94%	15	105
		7.9	7.48	0.03	95%	16	105
		8.2	7.76	0.05	95%	15	105
		8.22	7.79	0.03	95%	18	105
		8.42	7.93	0.05	94%	19	105
		9.35	8.87	0.05	95%	17	105
RR	Low	0.56	0.54	0	96%	8	15
		0.57	0.54	0	97%	6	15
		0.58	0.55	0	95%	8	15
		0.58	0.55	0	94%	6	15
		0.55	0.53	0	96%	8	15
		0.54	0.51	0.01	96%	8	15
		0.57	0.55	0	97%	7	15
		0.55	0.53	0	95%	10	15
		0.55	0.51	0.01	95%	7	15
	Medium	2.97	2.82	0.01	95%	10	55
		2.95	2.78	0.02	95%	7	55
		2.98	2.82	0.01	95%	9	55
		2.97	2.82	0.01	95%	8	55
		3.13	2.98	0.01	95%	8	55
		3.34	3.12	0.02	94%	10	55
		3.44	3.28	0.01	95%	14	55
		3.27	3.12	0	95%	11	55
		3.24	3.03	0.01	93%	10	55
	High	9.44	8.9	0.05	94%	41	106
		9.45	8.97	0.03	95%	43	106

		10.22	9.69	0.04	95%	54	106
		10.2	9.67	0.04	95%	64	106
		11.63	11	0.03	94%	97	106
		10.64	10.07	0.03	94%	75	106
		10.61	10.07	0.04	95%	80	106
		11.24	10.61	0.03	94%	82	106
		11.05	10.42	0.05	94%	87	106
		0.67	0.5	0	75%	163	24
		0.85	0.5	0	59%	169	24
		0.79	0.5	0	64%	170	24
		0.68	0.5	0	74%	160	24
	Low	0.62	0.5	0	81%	161	24
		0.65	0.49	0	77%	169	24
		0.63	0.49	0	78%	165	24
		0.62	0.5	0	80%	160	24
		0.6	0.5	0	84%	159	24
		3.4	2.5	0.01	74%	773	104
		3.08	2.5	0	81%	780	104
		2.88	2.48	0.01	86%	750	104
		3	2.48	0.01	83%	749	104
CFS	**Medium**	2.96	2.47	0.02	84%	759	104
		2.86	2.49	0.01	87%	745	104
		2.92	2.49	0	85%	734	104
		2.96	2.5	0	84%	758	104
		2.87	2.48	0.01	86%	755	104
		5.86	4.98	0.02	85%	1488	204
		6.18	4.99	0.02	81%	1525	204
		5.92	4.98	0.02	84%	1509	204
		6.16	5	0.02	81%	1510	204
	High	6.23	5.09	0.01	81%	1556	204
		6.07	4.98	0.04	82%	1517	204
		5.92	5.01	0.01	84%	1487	204
		5.89	4.98	0.02	84%	1498	204
		5.94	5.04	0.01	85%	1529	204
IO-Bound							
Scheduler	**Processes**	**Wall**	**CPU**	**User**	**Processor**	**Involuntary**	**Voluntary**
		0.1	0	0.06	68%	5	1998
		0.11	0	0.07	65%	4	2331
		0.12	0	0.07	65%	5	2280
		0.1	0	0.07	69%	5	1998
	Low	0.09	0	0.06	71%	5	2006
		0.1	0	0.06	67%	5	1990
		0.1	0.01	0.06	71%	4	1987
		0.1	0	0.07	67%	5	2003
		0.11	0	0.07	64%	5	1982
		0.36	0.01	0.32	92%	6	10022
FIFO		0.37	0	0.32	91%	3	10025
		0.38	0	0.34	89%	6	9996
		0.36	0	0.33	91%	5	10013
	Medium	0.38	0.01	0.33	89%	5	9991
		0.36	0	0.33	90%	3	10104
		0.35	0.01	0.3	89%	6	10022
		0.35	0	0.32	90%	3	11716
		0.37	0	0.32	88%	4	10367
	High	0.75	0.02	0.66	92%	8	20125
		0.73	0	0.68	93%	5	20002

		0.73	0	0.67	91%	5	22346
		0.73	0.01	0.67	93%	4	20034
		0.71	0.01	0.66	94%	5	21796
		0.71	0.01	0.65	93%	6	22119
		0.69	0.04	0.61	93%	4	20236
		0.72	0.01	0.66	93%	6	22226
		0.73	0.01	0.67	92%	5	21675
		0.1	0	0.07	70%	6	1995
		0.1	0	0.07	66%	4	2004
		0.1	0	0.06	70%	4	2007
		0.1	0	0.06	69%	4	2003
	Low	0.09	0	0.07	74%	4	2002
		0.1	0	0.06	68%	5	2004
		0.09	0	0.07	71%	6	1999
		0.1	0	0.07	69%	4	2005
		0.1	0	0.07	68%	5	2004
		0.36	0	0.33	91%	5	10131
		0.38	0.01	0.33	90%	5	10304
		0.38	0	0.33	87%	5	10495
		0.38	0	0.34	90%	5	10426
RR	**Medium**	0.37	0	0.33	88%	5	10382
		0.36	0	0.32	89%	5	10236
		0.36	0	0.33	91%	4	10359
		0.35	0	0.32	90%	5	10312
		0.37	0	0.32	88%	5	10209
		0.71	0.01	0.66	94%	3	20238
		0.71	0.01	0.65	93%	4	20986
		0.71	0	0.66	94%	5	21667
		0.71	0.01	0.64	91%	4	20184
	High	0.69	0.01	0.64	95%	5	21731
		0.7	0.01	0.65	93%	5	21258
		0.69	0	0.65	94%	6	19967
		0.7	0.01	0.64	93%	5	21909
		0.7	0.02	0.63	92%	5	21333
		0.19	0	0.08	43%	223	1781
		0.19	0	0.07	39%	204	1840
		0.18	0	0.08	43%	181	1765
		0.22	0	0.08	36%	194	1907
	Low	0.16	0	0.07	47%	198	1900
		0.22	0	0.08	37%	206	2001
		0.2	0	0.07	39%	218	1970
		0.17	0	0.06	41%	210	1896
		0.19	0	0.07	39%	229	1958
		0.76	0	0.38	50%	672	9312
CFS		0.7	0	0.38	55%	695	9130
		0.61	0	0.37	61%	602	8236
		0.65	0	0.36	56%	927	11338
	Medium	0.54	0.02	0.32	65%	732	9781
		0.6	0	0.35	60%	972	12267
		0.59	0.01	0.35	61%	995	12386
		0.57	0.01	0.34	63%	838	10665
		0.63	0	0.35	57%	1029	12089
		1.37	0	0.76	56%	1605	28799
	High	1.22	0.01	0.73	61%	1343	25007
		1.28	0.01	0.74	59%	1284	24535
		1.21	0	0.73	61%	1855	32222

		1.1	0.01	0.71	66%	1476	27167
		1.06	0	0.72	68%	1436	27776
		1.09	0.02	0.7	65%	1519	27103
		1.12	0	0.73	65%	1527	29714
		1.15	0.02	0.71	63%	1451	28865
Mixed							
Scheduler	**Processes**	**Wall**	**CPU**	**User**	**Processor**	**Involuntary**	**Voluntary**
FIFO	**Low**	0.61	0.5	0.07	93%	6	2020
		0.62	0.5	0.07	92%	5	2111
		0.62	0.52	0.06	93%	4	2282
		0.6	0.49	0.07	94%	4	2059
		0.61	0.52	0.05	93%	4	2161
		0.6	0.5	0.07	95%	5	2232
		0.63	0.51	0.06	92%	4	2085
		0.61	0.51	0.06	94%	4	2209
		0.61	0.51	0.06	93%	4	2285
	Medium	3	2.54	0.32	95%	6	10142
		3.02	2.55	0.32	95%	8	10021
		3	2.54	0.32	95%	8	10194
		2.98	2.54	0.33	96%	7	10147
		3	2.53	0.32	95%	6	10100
		3.05	2.59	0.31	95%	6	10297
		2.98	2.55	0.3	95%	7	10019
		3.02	2.56	0.31	95%	7	10127
		2.99	2.55	0.29	95%	7	10429
	High	5.98	5.1	0.61	95%	9	20006
		6	5.08	0.63	95%	9	20099
		6.05	5.14	0.61	95%	9	19985
		5.99	5.12	0.61	95%	7	19982
		6.02	5.07	0.64	95%	9	19875
		6	5.11	0.6	95%	12	20026
		5.99	5.11	0.6	95%	8	19956
		6	5.08	0.61	94%	11	19859
		6.01	5.1	0.63	95%	10	20011
RR	**Low**	0.63	0.51	0.06	91%	4	2056
		0.61	0.5	0.07	93%	4	1998
		0.61	0.51	0.06	93%	4	2005
		0.62	0.5	0.06	92%	5	2066
		0.61	0.5	0.07	93%	4	1992
		0.61	0.51	0.06	93%	5	1990
		0.62	0.52	0.04	91%	6	2305
		0.61	0.51	0.06	93%	4	1998
		0.6	0.51	0.05	94%	5	2002
	Medium	3	2.54	0.31	95%	9	10635
		3	2.53	0.31	95%	7	10421
		2.98	2.53	0.31	95%	6	10257
		3.01	2.55	0.31	95%	7	10681
		3	2.55	0.3	95%	6	10101
		3.01	2.55	0.31	95%	6	10087
		2.99	2.54	0.3	95%	7	10631
		2.98	2.54	0.29	95%	7	10200
		2.99	2.53	0.31	95%	6	10164
	High	6.06	5.14	0.61	95%	9	21125
		6.02	5.1	0.62	95%	11	23418
		6.03	5.09	0.63	95%	10	20284
		6.02	5.12	0.6	95%	10	20023

		6.02	5.09	0.63	95%	10	20034
		6.02	5.12	0.6	94%	10	19911
		6.02	5.08	0.64	95%	10	19981
		5.99	5.09	0.6	95%	9	20044
		5.99	5.09	0.6	95%	10	20017
CFS	**Low**	1.18	0.54	0.05	50%	370	1826
		0.91	0.53	0.05	64%	338	1905
		0.92	0.53	0.04	63%	287	2052
		0.78	0.51	0.07	74%	418	1890
		0.83	0.51	0.06	69%	419	1829
		0.74	0.51	0.06	78%	354	1857
		0.86	0.54	0.05	69%	433	1959
		0.79	0.52	0.06	73%	363	1879
		0.81	0.53	0.06	73%	441	1959
	Medium	4.1	2.62	0.3	71%	1480	8874
		5.13	2.63	0.29	56%	1811	12930
		3.93	2.62	0.28	74%	1622	10676
		3.41	2.58	0.31	84%	1878	11603
		3.55	2.55	0.32	81%	1613	10375
		3.65	2.59	0.33	79%	1743	9597
		3.62	2.62	0.31	80%	1669	11028
		3.53	2.55	0.33	81%	1730	9873
		3.35	2.52	0.34	85%	1697	11280
	High	7.83	5.15	0.66	74%	3444	28646
		7.89	5.21	0.64	74%	3513	23281
		7.52	5.2	0.61	77%	3095	26018
		7.01	5.15	0.64	82%	3797	25637
		7.05	5.13	0.66	82%	3550	27186
		6.92	5.13	0.67	84%	3829	30374
		6.53	5.16	0.61	88%	3326	26133
		7.04	5.17	0.63	82%	3330	24649
		6.89	5.15	0.63	83%	3295	23161

Appendix B:

C files:

- cpu-process.c - Program that uses CPU-bound processes
- io-process.c - Program that uses I/O-bound processes
- mixed-process.c - Program that uses a combination of CPU- and I/O- bound processes

Other:

- testscript - Shell script to run the suite of tests for the project
- README - description of how to build the project
- Makefile - file to make and clean the project