Rapport operativsystem lab2

Authors: Johnny Nguyen & Filip Hinderup

Home Assignment 3

Study the small test programs in Listing 1 (test1.c**) and Listing 2 (**test2.c**), so you understand what they are doing.**
How much dynamic memory is allocated in the test1 **program (i.e., how large is** struct link**)?**

Allocated memory: 4GB

How much data is written to the temporary file/tmp/file1.txt **in each iteration of the program** test2**?**

Amount of data written: 1 GB

Home Assignment 4

Study the man pages of vmstat **and** top **so you understand how these two commands work. Specifically, understand the parameters that you can give to the commands and the output format of the commands.**
In which output columns of vmstat **can you find the amount of memory a process uses, the number of swap ins and outs, and the number of I/O blocks read and written?**

Under swpd, buff, cache, si, so, bi, bo.

Where in the output from top **do you find how much cpu time and cpu utilization a process is using?**

Under %CPU and TIME+.

Task 1

Compile the test programs test1 **and** test2**. Execute "**top**" and "**vmstat 1**" in two new terminal windows.**
Execute the test program test1**. How much memory does the program use? Does it correspond to your answer in Home Assignment 3?**

Memory usage: 4GB

Memory usage was the same as the prediction from home assignment 3.

What is the cpu utilization when executing the test1 **program? Which type of program is** test1**?**

CPU utilization: 100%

Program type: CPU bound

Execute the test program test2**. How much memory does the program use? How many blocks are written out by the program? How does it correspond to your answer in Home Assignment 3?**

Memory usage: 1GB

Number of blocks written out: ca 5 million blocks

Memory usage was the same as the prediction from home assignment 3.

What is the cpu utilization when executing the test2 **program? Which type of program can we consider** test2 **to be?**

CPU utilization: ca 70%

Program type: I/O bound

Task 2

Study the two scripts run1 **(Listing 3) and** run2 **(Listing 4). What is the difference between them in terms of how they execute the test programs?**

The first script ”*run1*” executes, sequentially, *test1* three times, then *test2* two times. The second script ”*run2*” tries to execute test1 three times while executing *test2* two times at the same time.

Execute the script run1 **and measure the execution time. Study the cpu utilization using** top **during the execution. How long time did it take to execute the script and how did the cpu utilization vary?**

Time: 1 minute and 27,265 seconds

The CPU utilization was around 100% during the runtime of *test1*. During *test2* the CPU utilization was around 60%.

Execute the script run2 **and measure the execution time. Study the cpu utilization using** top **during the execution. How long time did it take to execute the script and how did the cpu utilization vary?**

Time: 14,805 seconds

The CPU utilization was around 100% during run2.

Which of the two cases executed fastest?

run2 executed faster.

In both cases, the same mount of work was done. In which case was the system best utilized and why?

run2 utilized the system best, because the system utilized the resources that otherwise would have been unused like in run1.

Task 4

Use the program that you developed in Task 3 to fill in the number of pages faults in Table 1 and Table 2. To your help,we have filled in some of the numbers in the tables (so you can use them to check that you program generate the correct number of page faults).

Table 1: Number of page faults for mp3d.mem when using FIFO as page replacement policy.

Number of Pages/ Page Size	1	2	4	8	16	32	64	128
128	55421	22741	13606	6810	3121	1503	1097	877
256	54357	20395	11940	4845	1645	939	669	478
512	52577	16188	9458	2372	999	629	417	239
1024	51804	15393	8362	1330	687	409	193	99

Table 2: Number of page faults for mult.mem when using FIFO as page replacement policy.

Number of Pages/ Page Size	1	2	4	8	16	32	64	128
128	45790	22303	18034	1603	970	249	67	67
256	45725	22260	18012	1529	900	223	61	61
512	38246	16858	2900	1130	489	210	59	59
1024	38245	16855	2890	1124	479	204	57	57

Task 5

What is happening when we keep the number of pages constant and increase the page size? Explain why!

The number of page faults decreases because the bigger the pages are the more memory references they can contain.

What is happening when we keep the page size constant and increase the number of pages? Explain why!

The number of page faults decreases because the more pages there are the more memory references can be stored.

If we double the page size and halve the number of pages, the number of page faults sometimes decrease and sometimes increase. What can be the reason for that?

The number of frames is directly proportional to the size of the pages. In effect, the bigger the pages are the less frames you can have. Because of a decrease in replacement choices, the chances of a page fault occurring will be increased greatly.

Focus now on the results in Table 2 (matmul)**. At some point decreases the number page faults very drastically. What memory size does that correspond to? Why does the number of page faults decrease so drastically at that point?**

Memory size change: 256->512.

There are alot more memory references with a value difference of 512 and 256.

At some point the number of page faults does not decrease anymore when we increase the number of pages. When and why do you think that happens?
Memory size change: 64->128.

The number of page faults are too few relative to the number of pages to have an effect in increasing the number of pages.

Task 7

Use the program that you developed in Task 6 to fill in the number of pages faults in Table 3. To your help, we have filled in some of the numbers in the table (so you can use them to check that you program generate the correct number of page faults).

Table 3: Number of page faults for mp3d.mem when using LRU as replacement policy.

Number of Pages/ Page Size	1	2	4	8	16	32	64	128
128	55421	16973	11000	6536	1907	995	905	796
256	54357	14947	9218	3811	794	684	577	417
512	52577	11432	6828	1617	603	503	362	206
1024	51804	10448	5605	758	472	351	167	99

Task 8

Which of the page replacement policies FIFO and LRU seems to give the lowest number of page faults? Explain why!

LRU seems to give the lowest number of page faults. Because LRU prioritizes the most used there will be more hits and as consequence less page faults.

In some of the cases, the number of page faults are the same for both FIFO and LRU. Which are these cases? Why is the number of page faults equal for FIFO and LRU in those cases? Explain why!

When number of pages = 1.
When number of pages = 128 and page size = 1024.

Because when number of pages = 1 it will always swap the same page. In the other case if the page size and number of pages are large enough there will be no reason to swap, because all of the memory references fit the pages.

Task 10

Use the program that you developed in Task 9 to fill in the number of pages faults in Table 4. To your help, we have filled in some of the numbers in the table (so you can use them to check that you program generate the correct number of page faults).

Table 4: Number of page faults for mp3d.mem when using Optimal (Bélády's algorithm) as replacement policy.

Number of Pages/ Page Size	1	2	4	8	16	32	64	128
128	32907	15856	8417	3656	1092	824	692	558
256	31492	14168	6431	1919	652	517	395	295
512	28821	11322	4191	920	470	340	228	173
1024	27955	10389	3367	496	339	213	107	99

Task 11

Based on the values in Table 1, Table 3, and Table 4, answer the following questions. As expected, the Optimal policy gives the lowest number of page faults. Explain why!

Because the algorithm checks if upcoming memory references match with current stored memory references, giving replacement priority to the last match. This will lead to a decrease in page faults as the algorithm increases the chances of getting hits.

Optimal is considered to be impossible to use in practice. Explain why!

Because to be able to implement it the algorithm must first know the future (in this case upcoming memory references).

Does FIFO and/or LRU have the same number of page faults as Optimal for some combination(s) of page size and number of pages? If so, for which combination(s) and why?

When number of pages = 128 and page size = 1024, because no page replacement ever happens.