Chalmers University of Technology and Gothenburg University

Operating Systems
EDA092, DIT 400

Exam 2017-08-15

Date, Time, Place: Tuesday 2017/08/15, 14.00-18.00, Maskin-salar

Course Responsible: Vincenzo Gulisano (031 772 61 47), Marina Papatriantafilou (031 772 54 13)

Auxiliary material: You may have with you
- An English-Swedish, Swedish-English dictionary.
- No other books, notes, calculators, PDA's etc.

Grade-scale ("Betygsgränser"):
CTH: 3:a 30-39 p, 4:a 40-49 p, 5:a 50-60 p
GU: Godkänd 30-49p, Väl godkänd 50-60 p

Exam review ("Granskningstid"):
Will be announced after the exam.

Instructions

- Do not forget to write your personal number, if you are a GU or CTH student and at which program ("linje").
- Start answering each assignment on a new page; number the pages and use only one side of each sheet of paper.
- Write in a **clear manner** and **motivate** (explain, justify) your answers. If it is not clear what is written, your answer will be considered wrong. If it is not explained/justified, even a correct answer will get **significantly** lower (possibly zero) marking.
- If you make **any assumptions** in answering any item, do not forget to clearly state what you assume.
- The exam is organized in groups of questions. The credit for each group of questions is mentioned in the beginning of the respective group. Unless otherwise stated, all questions in a group have equal weight.
- Answer questions in English, if possible. If you have large difficulty with that and you think that your grade can be affected, feel free to write in Swedish.

Good luck !!!!

1. (12 p)

   (a) (4 p) Please describe in detail what happens when the following code is executed:

```
int main() {
   pid_t pid1, pid2, pid3;
   pid1 = getpid();
   pid2 = fork();
   pid3 = getpid();
   if (pid3==pid1) {
      printf("A");
   }
}
```

   HINT: After fork (assuming it works correctly) pid1 and pid3 are the same for the parent but not for the child. Only the parent prints A.

   (b) (4 p) Describe the similarities and the differences between a context switch between two processes and between a process and an interrupt handler.

   HINT: Both will require storing of the registers and other information for the running process before it is switched. Answering the interrupt and running a different process, nevertheless, have not the same priority and are not done with the same efficiency. The interrupt can be associated with an handler routing and the switching to it is optimized (based on the interrupt number). The new running process, on the hand hand, could be decided after running a scheduler and is not as optimized as the interrupt handler.

   (c) (4 p) The manual for the fork call says *"Under Linux, fork() is implemented using copy-on-write pages, so the only penalty that it incurs is the time and memory required to duplicate the parent's page tables."* What else could be duplicated (but is not)? Why is the page table copied instead of shared between the processes?

   HINT: The pages themselves. Multiple processes can be spawn with COW and have different modified pages, hence they need different page table copies.

2. (12 p)

   (a) (4 p) The processes A, B and C access their pages in the following order:

   A : 1 2 3 5 2 3 7 6 1 2 3 7 8 1 4
   B : 1 2 3 4 9 1 8 6 9 2 3 1 2 5 2
   C : 1 9 1 8 2 3 7 6 2 2 4 9 8 4 3

   Assuming a working set window of 14 page references, compute the total demand of frames of the processes (for the given window size) and check if trashing is occurring given that 22 frames are available.

   HINT: number of pages accessed by each process over the last 14 accesses: 7 and 8 for A, 8 and 8 for B and 8 and 8 for C. Sums = 23 and 24, so trashing is occurring.

   (b) (4 p) How does demand paging work and why is it beneficial?

   HINT: Load page only when requested. Less pages loaded –¿ more space / more processes loaded, faster loading of processes.

(c) (4 p) Suppose a process has size of 200 bytes and the frame size (for paging) is set to 2^4 bytes. Compute the size in bits of the page table if the frame addressing requires 1 byte and dirty bits are not used.

HINT: pages needed: 13. page table = 13 * (8 + 1) = 117 (+1 because of valid/invalid) .

3. (12 p)

(a) (4 p) What is a free space bitmap? Suppose the block size for a disk of 1GB is 4KB. How many blocks would be used to maintain the bitmap for the entire disk?

HINT: a bitmap in which bit i specifies whether block i is free or not. number of blocks = 262144. Bytes to store the bitmap = 32768. Blocks to store the bitmap = 8.

(b) (4 p) Consider a file system that uses i-nodes to represent files. Disk blocks are 8-KB in size and a pointer to a disk block requires 4 bytes. This file system has 12 direct disk blocks plus single, double, and triple indirect disk blocks. Suppose that, when possible, files are stored entirely in the disk block pointers of the i-node rather than the disk blocks. What would be the maximum size of a file that can be entirely stored in a i-node?

HINT: (12+1+1+1)*4 bytes = 60 bytes. (To be really precise, you should also use some space to indicate whether the content of the pointers are real pointers or data, but that's OK).

(c) (4 p) What is the relocation register? Does it need to be changed during a context switch? Discuss why.

HINT: Base to be summed to address to get real address. Yes, because each process has its own space.

4. (12 p)

(a) (6p) Consider a system where you know that the offered load consists of periodic real-time tasks and interactive processes. As a system designer you are able to decide on the scheduling policy to use. Discuss the design of two scheduling policy alternatives suitable for such a system. (i.e., discuss how you would think in order to decide on a policy to use, the advantages and problems of the alternatives you are considering, how these methods could be implemented, whether you could make use of additional information).

HINT: one possibility: use EDF, with dedlines for interactive jobs = time of issue + maximum reasonable response time for the user (if available)
other: maintain 2 queues, one for RT, one for interactive, serve the RT first with EDF or RM, when empty move to interactive. If use EDF in the RT and the offered RT load (sum[(exec-time)/period] is less than 1, there will be time left for interactive ones, else, there may be starvation to interactive jobs.
alternative: insert the interactive jobs with fixed priorities in the same queue as the RT tasks. fixing those priorities would need knowledge of how critical it would be to miss an interactive process (i.e. to let it starve, or to eliminate it from the queue in case of congestion) and what are the tolerated response-time margings for these processes.

(b) (6 p) An issue in multiprocessor scheduling is how to decide about the sharing of the ready queue(s). Describe two common approaches and argue about their advantages and disadvantages.

HINT:shared versus per-processor; can facilitate loab lanacing but can become hotspot + no-processor-affinity can result in large migration costs

5. (12 p)

(a) (6p) Design a solution to the mutual exclusion problem for arbitrary number of processes/threads in a system with SPARC processors, where the following *atomic instruction*, called Compare-and-Swap, is available by the hardware. Discuss carefully the properties of your solution.

```
int CAS(int *addr, int old, int new)
   if (*addr == old) { *addr = new; return(SUCCESS) }
   else return(FAILURE)
```

Answer sketch: One can use CAS to immitate the behaviour of test-and-set or exchange instructions and design a solution similar to the ones we discussed in that context. the solution ensures mutual exclusion, no deadlock, but may suffer from starvation. A solution free from starvation employs an idea as in the n-process mutex-algo by Peterson. Answers along the first direction that discuss the starvation possibility, are also accepted as fully correct.

(b) (4p) Consider the following suggested solution to the readers-writers problem. Does it correctly solve the problem? Argue why or why not.

```
int readcount; // (initial value = 0)
semaphore w; // ( initial value = 1 )

//READER
readcount++;
if (readcount == 1)
   wait(w);
// reading is performed
readcount--;
if (readcount == 0)
    signal(w);

//WRITER
wait(w);
// writing is performed
signal(w);
```

Answer sketch: readcount is not protected when ++ or −, hence a race consition can cause the if statement to be false even through there can be eg. 2 readers and a writer and thus cause violation of mutual exclusion between the readers and the writer.

(c) (2p)What is the meaning of the term "busy-waiting"? What other kinds of waiting can there be in an operating system?

Answer sketch: spinning + block