Chris Banci
CS433
Professor Zhang
February 22, 2017

Homework 2

1) 3.12 Including the initial parent process, how many processes are created by the program shown in Figure 3.32? Explain your answer.

- The code in the figure is calling fork() four times and with this, we can calculate the number of process created using a formula like 2^N - 1 where N is how many times fork() is being called. In this case, including the initial parent process, 16 processes are created from calling fork() four times.

2) 3.14 Using the program in Figure 3.34, identify the values of pid at lines A, B, C, and D. (Assume that the actual pids of the parent and child are 2600 and 2603, respectively.)

- Given that the parent process is 2600 and the child process is 2603…
  - Line A = 0 // it is the value returned by forking the parent process.
  - Line B = 2603 // it is the actual pid of the child process.
  - Line C = 2603 // it is the value returned by child process.
  - Line D = 2600 // it is the actual pid of the parent process.

3) 3.17 Using the program shown in Figure 3.35, explain what the output will be at lines X and Y.

- Line X,the array is iterated to index 4 and each element is multiplied by -i.

```
-----------------------------------------------------
CHILD: 0        // nums[0] = 0      0 x 0 = 0
CHILD: -1       // nums[0] = 1       1 x -1 = -1
CHILD: -4       // nums[0] = 2       2 x -2 = -4
CHILD: -9       // nums[0] = 3       3 x -3 = -9
CHILD: -16      // nums[0] = 4       4 x -4 = -16
```

- Line Y, the array is iterated to index 4 and each index is printed.

```
-----------------------------------------------------
PARENT: 0       // nums[0] = 0
PARENT: 1       // nums[1] = 1
PARENT: 2       // nums[2] = 2
PARENT: 3       // nums[3] = 3
PARENT: 4       // nums[4] = 4
```

4) 3.18 What are the benefits and the disadvantages of each of the following? Consider both the system level and the programmer level.

(a) Synchronous and asynchronous communication

- Benefit of synchronous communication is having an instant response, its fast and simpler. However, its downside is blocking on both ends. The disadvantage of asynchronous communication is time delay waiting for a response and at the system level, asynchronous is more complicated.

(b) Automatic and explicit buffering

- In automatic buffering, the queue is infinite, the sender will never have to block and any amount of messages can wait in it. In explicit buffering, the queue size is specified and only a specified amount of messages can be stored in it. If it is not empty, it messages can be placed in it and sender can continue, however, it it is full, sender must block until space is available.

(c) Send by copy and send by reference

- Send by reference is better for big data structures because no copies are made, but at the programmer level, it is harder to implement. Send by copy increases overhead by making a copy of the parameter passed, but it is easier to code at the programmer level.

(d) Fixed-sized and variable-sized messages

- Fixed-sized messages are easier to implement at the system level, however it makes programming more difficult. Variable-sized messages on the other hand require a more complex system level implementation, but the programming task becomes simpler.

5) 4.15 Consider the following code segment:

(a) How many unique processes are created?

- Including the parent process, 6 unique processes are created.
- First unique process created by calling pid = fork(). Second unique process created in the if statement by calling fork(). After the if statement, there are three process and each call the last fork() function, totalling to 6 unique process.

(b) How many unique threads are created (not counting the main thread of each process)?

- Not including the main thread of each process, 2 unique threads are created in the child process statement block.
- First unique thread created in the child if block calling fork(). Second unique thread created in the if block thread after calling fork().

6) 4.17 The program shown in Figure 4.16 uses the Pthreads API. What would be the output from the program at LINE C and LINE P?

- Line C, a new thread is created for the child process and the value is set to 5 by runner function.
    - CHILD: value = 5

- Line P, the parent process waits until child process finishes, global value is still 0.
    - PARENT: value = 0

7) 4.18 Consider a multicore system and a multithreaded program written using the many-to-many threading model. Let the number of user-level threads in the program be greater than the number of processing cores in the system. Discuss the performance implications of the following scenarios.

(a) The number of kernel threads allocated to the program is less than the number of processing cores.

- When the amount of kernel threads < number of processing cores, most processor cores won't be utilized because the scheduler only allows each kernel thread to use only one processor. Therefore, there will be processors that will remain idle since there are more kernel threads.

(b) The number of kernel threads allocated to the program is equal to the number of processing cores.

- When the amount of kernel threads = number of processing cores, all kernel threads and processes will run concurrently. Therefore, each processor will be utilized.

(c) The number of kernel threads allocated to the program is greater than the number of processing cores but less than the number of user-level threads.

- When the amount of kernel threads > number of processing cores, processes can easily swap to other kernel threads if a kernel thread is busy. Therefore, this allows an increase in multitasking ability with all processors being utilized.