laSalle
UNIVERSITAT RAMON LLULL

Operating Systems

2021-2022

Session 2: Forks

30/09/21

Goals

The goal of this session is to introduce the student to multiprogramming, that is, the programming of applications with two or more processes running “at the same time”. For this, the *fork* tool will be used.

Motivation

More specifically, with this session, the students must exercise:

- Creation of new processes (*fork*).
- The distinction of parent/child processes (*PID*).
- Process management (*wait*, *getpid*).

Previous documentation

To complete this session, it is recommended to read the following references:

SALVADOR, J. (2014). *Programació en UNIX per a pràctiques de Sistemes Operatius.*

STEVENS, W. R. & RAGO S.A. (2008). *Advanced Programming in the UNIX Environment*, 2nd edition.

TANENBAUM, A. S. (2009). *Modern Operating Systems*, 3rd Edition.

Circle of life

The number π is a very interesting one for a lot of reasons, but this time we are looking at him for the infinite number of decimals it has and the everlasting search of all of them.

We are going to develop a program that calculates the number step by step, and for each iteration improves the result having a better accurate representation of π.

To improve the accuracy of the number, the results from one step are going to be inherited by the child process, which will continue with the next calculation iteration.

Each process will take between 1 and 5 seconds to perform the calculation. If it takes more than 3 seconds, we will consider that the process couldn't make the calculation successfully (timeout) and the next child will have to redo the calculation of that iteration again.

To approximate the number π we will use the **Nilakantha Infinite Series**:

$$\pi = 3 + \frac{4}{2 \times 3 \times 4} - \frac{4}{4 \times 5 \times 6} + \frac{4}{6 \times 7 \times 8} - \frac{4}{8 \times 9 \times 10} + \cdots$$

This cycle will be performed a certain number of times, entered by the user as a parameter.

Take into consideration that the matagalls and montserrat servers have a limit of 30 processes per user, so we will use 20 as a limit. If the limit is reached, the program will stop making the iterations. To test the session without the limit, you can use a virtual machine with the server image locally.

The program execution will follow this format:

s2 [number of iterations]

Execution example

The identifier numbers for each process are the PID.

```
> ./S2 5
Number of calculations iterations = 5

I'm the father with id: 14118.
Calculating PI...
PI Calculated: 3.16667.
I'm the son of 14118 with id: 14119.
Calculating PI...
PI Calculated: 3.13333.
I'm the son of 14119 with id: 14120.
Calculating PI...
PI Calculated: 3.14524.
I'm the son of 14120 with id: 14124.
Calculating PI...
PI Calculated: 3.13968.
I'm the son of 14124 with id: 14126.
Calculating PI...
Calculation timeout.
I'm the son of 14126 with id: 14128.
Calculating PI...
PI Calculated: 3.14271.
RIP 14128
RIP 14126
RIP 14124
RIP 14120
RIP 14119
RIP 14118
```

```
> ./S2 20
Number of calculations iterations= 20

I'm the father with id: 14118.
Calculating PI...
PI Calculated: 3.16667.
I'm the son of 14118 with id: 14119.
Calculating PI...
PI Calculated: 3.13333.
Something went wrong, fork failed: Resource temporarily unavailable
Something went wrong, fork failed: Resource temporarily unavailable
I'm the son of 14119 with id: 14120.
Calculating PI...
PI Calculated: 3.14524.
Something went wrong, fork failed: Resource temporarily unavailable
...
```

Operating Systems 2021-2022

Session 2: Forks 30/09/21

Considerations

- When the program ends, all the dynamic memory used must have been freed, and all file descriptors must be closed.
- The program *output* must be **completely identic** at the one shown in the execution examples.
- The use of global variables is not allowed, except the ones necessary for the control of the signals and forks.
- All inputs and outputs must be done using file descriptors. **The use of printf, scanf, FILE*, getchar or similar is NOT allowed.**
- **You must compile using the –Wall and –Wextra flags.**
- **Any deliverable containing warnings or errors will be directly discarded.**
- **At the start of the .c you must include a comment with your logins, names, and surnames.**
- **For submitting the session, you must hand over a file "S2.c", and delivered it through eStudy platform.**
- You must control that the number of arguments is correct.
- The user will only enter numbers as a parameter.
- **The use of the functions "system", "popen", or from the same family is NOT allowed**.
- You must ensure that, when the execution ends, all the processes have ended properly.
- **You must control if the fork() function works. In case of failure, an error message must appear.**
- The timings must be implemented using signals.
- The order of the messages may be different from the ones in the example.
- **You must control Ctrl+C.**
- In case of not killing the forks properly, the use of the following instruction is strongly advised: **pkill -u <login>.**