ASSIGNMENT REPORT 1: PROCESS AND THREAD IMPLEMENTATION

CENG2034, OPERATING SYSTEMS

Emre Ertürk
emreerturk3@posta.mu.edu.tr
https://github.com/Emre81

Sunday 7th June, 2020

Abstract

This homework's goal is to understand child and parent relationships and using multiprocessing method. I used some essential libraries in my python script. I used them for download images, compare image's hash codes, using thread and multiprocessing methods. And I realized python is very suitable language for doing efficient projects.

1 Introduction

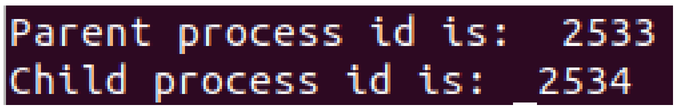
This homework's purpose create a child process and execute some functions in parent process and in child process. In first part, i created a new child process. And I printed the PID of child process's. In the child process i downloaded the images in given array. For the orphan sitution i called a syscall function. I checked images duplicate or not, with using multiprocessing techniques.

2 Assignments

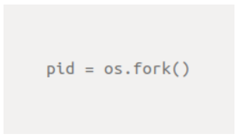
I had 4 tasks to do in this homework. I write my script in python language. I used Linux Ubuntu in my virtual machine.

2.1 Create a child process (os.fork())

In the tasks one i created a child process and i used os.fork() syscall for it. I printed its pid using os.getpid() method. As seen in figure 1 parent and child processes have different process ids.



```
Parent process id is:  2533
Child process id is:  2534
```

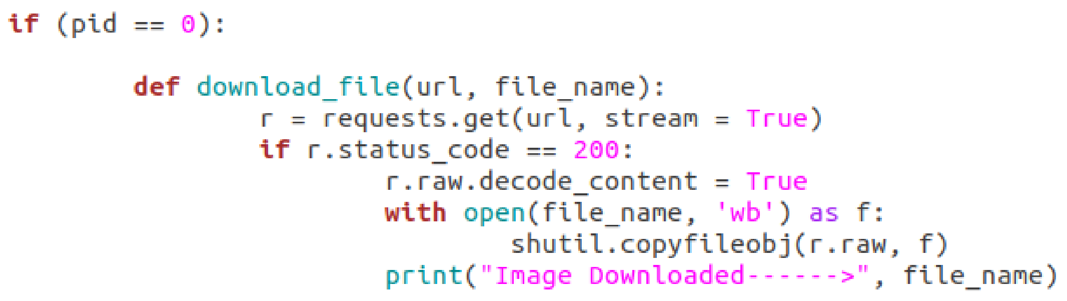


```
pid = os.fork()
```

Figure 1: Left: Child and Process id. Right: usage of os.fork()

2.2 Making operations with child process (if (pid == 0))

In the task two, i downloaded images from the given array under the child process. As seen in Figure 1's right side. I assigned os.fork() method to the pid variable. If pid variable is equals to 0 that means you are in the child process. If pid is greater than 0 that means you are in parent process. I wrote a image downloader function using shutil and requests libraries.The Function gets file name and url as parameters. You can see the image download function in the child process below.

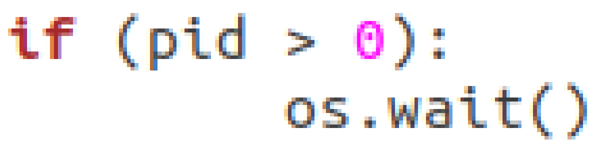


```
if (pid == 0):

        def download_file(url, file_name):
                r = requests.get(url, stream = True)
                if r.status_code == 200:
                        r.raw.decode_content = True
                        with open(file_name, 'wb') as f:
                                shutil.copyfileobj(r.raw, f)
                        print("Image Downloaded------>", file_name)
```

2.3 Orphan process (os.wait())

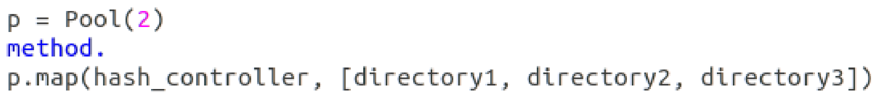
Orphan process is the situation when the parent finishes before child. I used os.wait() method in the parent process. Parent process waits until child process done.



```
if (pid > 0):
        os.wait()
```

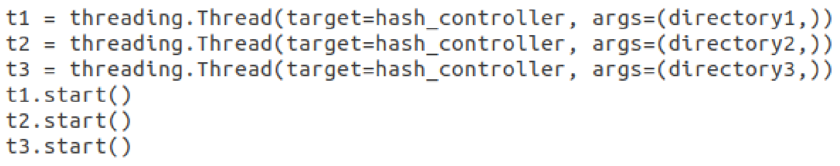
2.4 Multiprocessing and duplicate file finder (import multiprocessing)

I searched for a directory and got hash codes of files. I added hash codes to an array and in the for loops, i looked hash codes are duplicate or not. My duplicate finder function gets directory as parameter. So i used globl library and assigned directories to variables. At the last part i called function using multiprocessing, thread and normal technique.



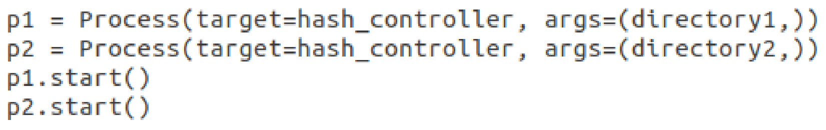
```
p = Pool(2)
method.
p.map(hash_controller, [directory1, directory2, directory3])
```

Figure 2: Multiprocessing pool method



```
t1 = threading.Thread(target=hash_controller, args=(directory1,))
t2 = threading.Thread(target=hash_controller, args=(directory2,))
t3 = threading.Thread(target=hash_controller, args=(directory3,))
t1.start()
t2.start()
t3.start()
```

Figure 3: Threading method



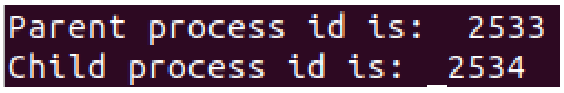
```
p1 = Process(target=hash_controller, args=(directory1,))
p2 = Process(target=hash_controller, args=(directory2,))
p1.start()
p2.start()
```

Figure 4: Process method

Figure 5: As you see 3 different method

3 Results

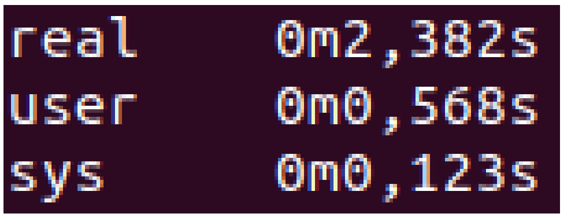
I created child process in task 1. Child process has different process id from parent process.



```
Parent process id is:  2533
Child process id is:  2534
```

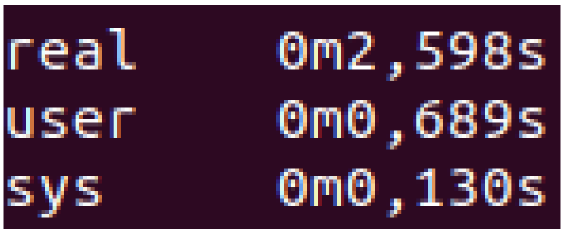
Figure 6: Parent process id and child process id are different.

I used 4 different method to run the duplicate finder function. The most logical method according to the results I got, that is pool method. It is the the most fastest method i tried for this question. Pool method get parameter in array and take elements from array and execute them.



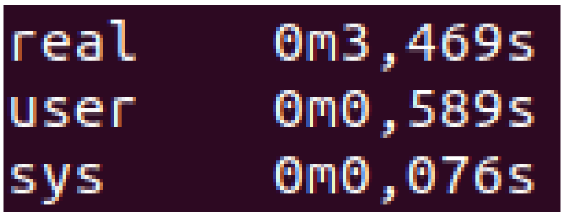
```
real    0m2,382s
user    0m0,568s
sys     0m0,123s
```

Figure 7: Multiprocessing pool method function time.



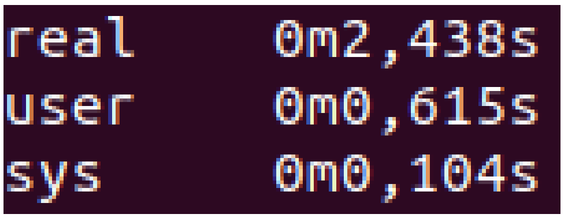
```
real    0m2,598s
user    0m0,689s
sys     0m0,130s
```

Figure 8: Multiprocessing process method function time.



```
real    0m3,469s
user    0m0,589s
sys     0m0,076s
```

Figure 9: Normal usage of function.



```
real    0m2,438s
user    0m0,615s
sys     0m0,104s
```

Figure 10: Threading method.

4 Conclusion

We can create child process using os.fork(). We can make different works in parent process and child process, it depends value of os.fork(). os.wait() prevent the orpan sitution. We can run the function with using different techniques. Multiprocessing pool technique was the most efficient method for my function.