IOLAB addendum

Jordi Fornés

15 June 2021

This document complements the wording of the ioLab assignment. It tries to answer some student questions appeared during the last week.

Figure 1 shows a sequence diagram of the program parFastq. Each vertical rectangle means one or more system calls of one kind. It's up to you to decide, for instance, how many calls to `read()` the parent process needs when it is accessing to the fastq file in order to get all the information it needs to send to its child.

Besides, sometimes you are dealing with strings, sometimes with bytes and sometimes with integers. You can use Python functions to interpret data accordently. For instance, to create an integer from a byte string you can use `int.from_bytes()` and viceversa `int.to_bytes()`. Note that the byte order is important here. See slides 9-12 of Unit *Computer Architecture*.

For example, how to pass the integer 49 to a byte string of two bytes. And from integer 49 to bytes; using little endian byte ordering:

```
>>> int.to_bytes(49,2,'little')
b'1\x00'
>>> int.from_bytes(b'1\x00','little')
49
```

For instance, how to pass from a string '23' to a byte string b'23':

```
>>> bytes(str(23),"utf-8")
b'23'
```

Recall good practices of programming. Each system call should be protected within a `try: except:`. Two typical errors in this program will be `OSError` and `TypeError`.

And last but not least, as this is concurrent programming, different executions could produce different results. Imagine two scenearios: (a) the parent process

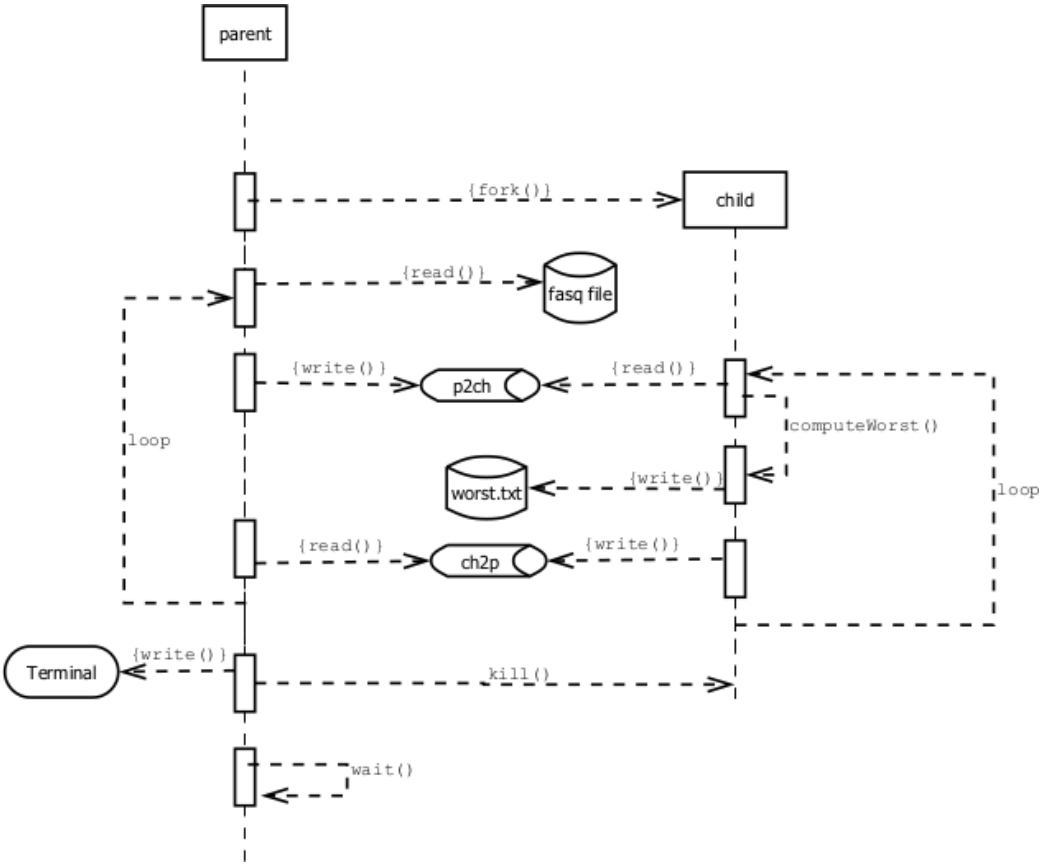

Figure 1: Sequence diagram of parFastqParser.

closes its pipe for writting (`p2ch`) and kills its child before the child process tries to read from that pipe. (b) the parent process closes its pipe for writting (`p2ch`) and its child process reads from the pipe before its parent kill it. In scenario (a) nobody reads from the pipe `p2ch`, in scenario (b), the child gets `0` bytes from the pipe. Do your programm finishes properly in both cases?