These are the cases in which we timed. Even simpsh was ran with the time command, because otherwise it would be difficult to compare between them – since time runs the same way, it is more reliable for us to use it as a benchmark rather than rely on getrusage inside simpsh.

#Test 1

In simpsh as
./simpsh –verbose –rdonly a –pipe –creat –trunc –wronly c –creat –wronly d –command 0 2 4 sort –command 1 3 4 tr A-Z a-z --wait

real 0m0.599s
user 0m0.410s
sys 0m0.040s

In bash as
time ((sort a | tr A-Z a-z >c) 2> d)

real 0m0.585s
user 0m0.420s
sys 0m0.025s

In execline as
pipeline {

sort a

}

redirfd -w 1 c tr a-z A-Z

real 0m0.658s
user 0m0.008s
sys 0m0.024s

#Test 2

In simpsh
./simpsh --verbose –rdonly a –pipe –creat –trunc –wronly c –creat –wronly d –command 0 2 4 sort –command 1 3 4 tr A-Z a-z --append --command 0 2 4 grep 1234 –wait

real 0m0.605s
user 0m0.421s
sys 0m0.035s

In bash
time (({ sort a ; grep 1234 a ; } | tr A-Z a-z >c) 2>d)

real 0m0.582s
user 0m0.405s
sys 0m0.037s

In execline

real 0m0.670s
user 0m0.011s
sys 0m0.035s

Conclusions

We note that bash is very similar to simpsh, spending a majority of its time in user time. However, bash seems to be a bit faster. Execline is by far the slowest, and as we ran multiple iterations of the test, it was consistently slower. The user time was by and far the lowest while the system time was still fairly close to bash and simpsh's system time.