

Operating Systems Laboratory (CS39002)

Assignment 3

Group 26

Seemant G. Achari (19CS10055)

Rajas Bhatt (19CS30037)

Solution for Task 1 (b)

In general, there is a limit to the number of processes that can be created, so the `fork` call will fail if we demand the creation of a large number of child processes.

In our case, the `fork` call usually fails because of an `EAGAIN` error (Resource temporarily unavailable when `perfor` is used), which is due to the system imposed limit on the number of threads. According to the man page of the `fork` call, this occurs due to one of the following situations:

1. The limit on the number of processes. This is given by the `RLIMIT_NPROC` flag. This limit can be checked by typing in `ulimit -u`.
2. The system-wise limit on the number of processes and threads, given by `/proc/sys/kernel/threads-max`
3. The maximum number of PIDs was reached. This is given by `/proc/sys/kernel/pid_max`
4. The PID limit (`pids.max`) imposed by the process group was reached. This can be checked using `/sys/fs/cgroup/pids/user.slice/user-1000.slice/pids.max`

In our system, we find out that the values are as follows (in processes):

1. `ulimit -u: 127628`
2. `cat /proc/sys/kernel/threads-max: 255257`
3. `cat /proc/sys/kernel/pid_max: 4194304`
4. `cat /sys/fs/cgroup/pids/user.slice/user-1000.slice/pids.max: 84234`

It can be seen that the limiting value is given by the value of `pids.max` (84234). This is basically the limiting value for the maximum number of processes created in a group. Since `fork` assigns the same `pgid` to children, the maximum number of children that can be created are bounded by 84234.

We create $r1 * c2$ child processes in total, so the maximum value of $r1 * c2$ will be equal to this value (`pids.max`). In practice, after repeated attempts, it can be seen that the value of $r1 * c2$ less than 83600 works in practice. This may be due to some other process in the process group or due to some system constraints. Therefore, a value of $r1 = 289$ and $c2 = 289$ works (which makes $r1 * c2 = 83521$).

```
> ./part1
1 1 1 83550
-8 -16 -2 4 -16 18 -14 16 -18 2 12 -6 8 -12 -16 0 8 -16 -16 -4 0 -16 -18 -8 8 -14 -10 -12 -2 12 12 10 -16
16 2 2 -16 0 12 2 -16 -14 -14 -14 -2 16 8 2 -16 -16 -18 -6 -16 -6 14 -18 -16 -18 -6 14 0 -4 18 -10 6 6 16
8 -4 10 -10 -12 16 18 14 -6 -16 0 -16 4 -12 0 12 18 -12 -10 -2 18 -18 10 10 -16 -14 -10 -18 -12 18 -14 -2
8 -2 6 6 4 10 -8 -14 6 -14 -16 6 -12 14 -18 -2 -8 6 -8 18 10 -6 10 18 4 -12 -8 -14 -4 0 8 14 -14 -4 0 10
4 10 10 2 8 -14 2 4 -14 6 10 -18 18 12 16 16 -14 6 -14 4 -18 14 14 8 -4 6 6 -16 8 -8 -2 10 -10 -4 0 16 2
16 14 -8 -12 4 0 16 4 -4 4 6 -2 -14 12 -2 16 6 8 2 8 16 -18 0 6 -10 16 0 -8 10 -14 -10 -18 -18 8 -10 12 -1
16 -2 -2 12 8 -8 -10 -10 14 -10 -4 -6 2 6 4 6 -10 10 -8 -6 10 -18 0 -14 -2 -18 10 8 14 -14 18 -14 4 4 -14
14 -8 4 6 16 16 -14 18 0 12 18 0 -16 -2 -16 -18 8 -2 -4 8 -12 -16 2 16 -10 6 -6 6 2 -2 -16 4 16 8 -8 12 4
8 -4 -10 12 -12 12 12 6 -4 12 4 -16 2 10 12 12 -2 -16 -18 -2 -2 16 2 -10 0 -14 14 12 -10 2 10 -4 -14 18 -4
4 -10 -8 -18 -10 6 2 -10 16 8 10 -8 6 -12 -18 -4 4 12 4 -14 10 -4 -8 10 14 8 -2 -4 -12 18 -8 -18 -10 8 -10
-14 -4 0 14 12 -12 -8 10 -4 12 4 -12 10 -14 -4 10 -10 -12 2 -16 -12 -8 -6 4 18 6 12 -14 -4 -12 -2 8 4 18 4
2 -12 2 0 12 -14 18 14 0 14 16 4 16 16 12 8 0 -2 2 6 -12 10 -2 -2 8 -10 -4 -6 -12 16 -12 8 -14 16 -4 -16
-4 -2 -8 12 2 2 8 -8 -18 18 10 8 -14 -8 -12 12 -4 -12 -6 10 -8 -8 2 18 4 18 -12 -14 -8 4 2 14 16 2 4 4
16 14 -14 6 4 10 12 -16 14 -14 8 14 -4 4 -6 18 2 -10 10 12 -16 14 0 4 -14 14 -14 4 -6 -8 6 4 -8 18 -10 16
-2 10 -14 -2 -12 -16 -4 -4 -8 -12 18 -6 16 -16 -16 -14 -8 -6 0 -14 2 -16 -18 -6 -2 -12 -14 18 16 -8 16 2
```

But $r1 = 290$ and $c2 = 290$ (which makes $r1 * c2 = 84100$) doesn't.

```
> ./part1
290 290 290 290
fork: Resource temporarily unavailable
```