Multiprocess parallelism (Concurrency)

concurrency — multiple tasks are active at once

parallelism — multiple tasks proceed simultaneously

Loriket — Falcon extension

(1) Syntax & (2) Semantics

```
<expr> ::= .....
         | sleep(<expr>)
         | parallel(<expr>)
         | send(<expr>, <expr>)
         | recieve(<expr>)
```

wait for some number of seconds

expects to receive a closure; (waiting for a channel arg) that closure is called in another process; evaluates to a "channel"

transmit on a channel (1st arg) some int or bool (2nd arg)

recieve from a channel some value

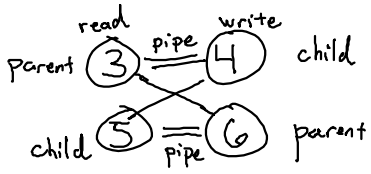
A channel is a pair of queues

4	5
5	4
7	7

pipe is (basically) an OS-buffered queue

```
def f chan =
  let n = receive(chan) in
  let _ = print(n) in
  let k = n+2 in
  send(chan, k)
end
let chan = parallel(f) in
let _ = send(chan, 5) in
let _ = print(4) in
print(receive(chan))
```

Each call to parallel creates two pipes



Channels end in 0000 0011

0x RRRRRRRWWWWWWW03

parent channel: 0x0000003 0000006 03

child channel: 0x0000005 0000004 03

Fork

fork syscall returns the PID of child or 0 if you are child process

Loriceet parallel call:

parent process gets parent channel

child process calls closure with child channel as arg

after call: exit w/ code 0

OS on fork must

- copy virtual memory tables — all pages are flagged COW (copy-on-write)
- account for open file descriptors
- manage processes
- ...

OS does not copy all memory