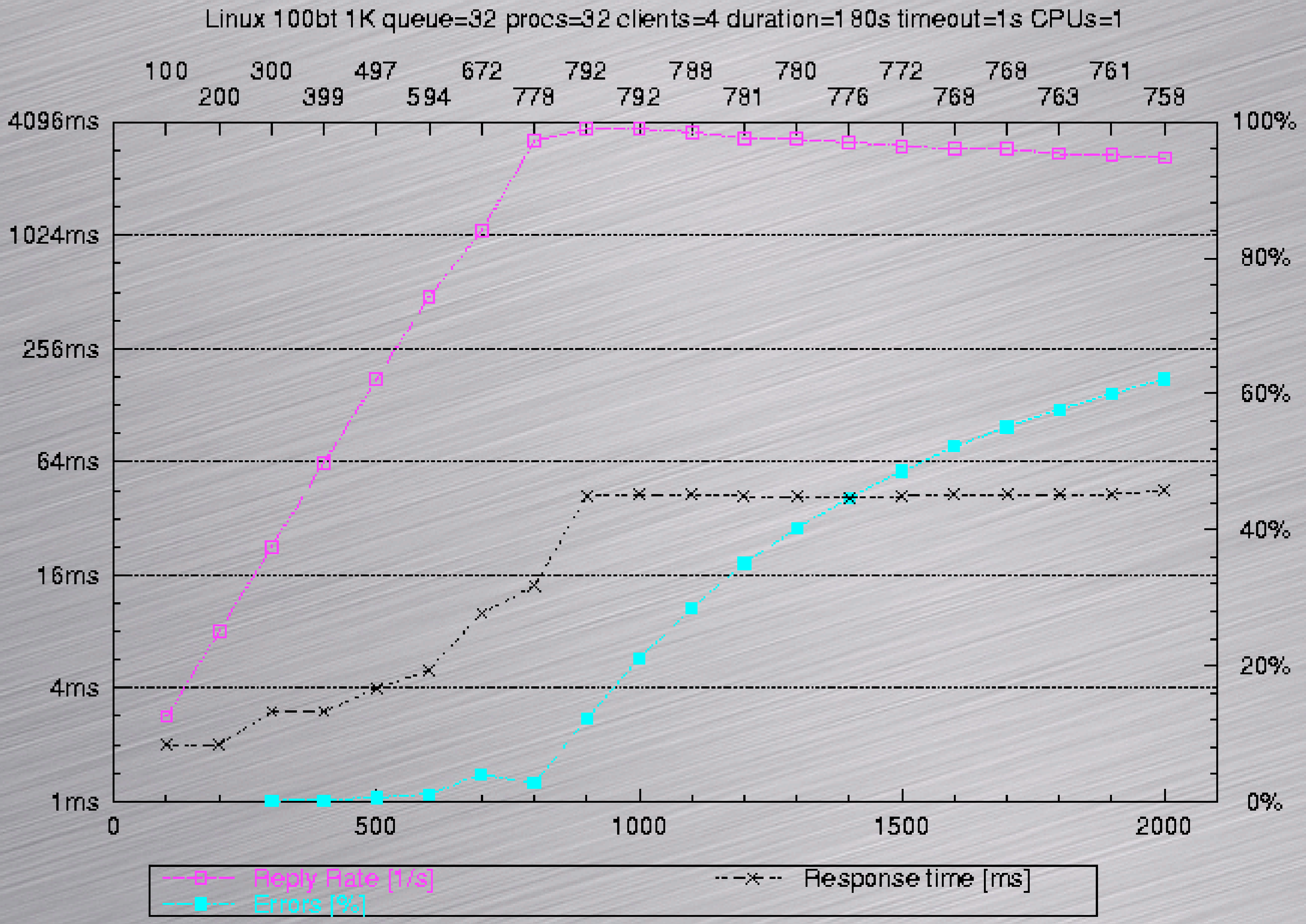
Threading by Appointment

Christoph Kirsch
University of Salzburg

Joint work with Harald Röck

Benchmarks!

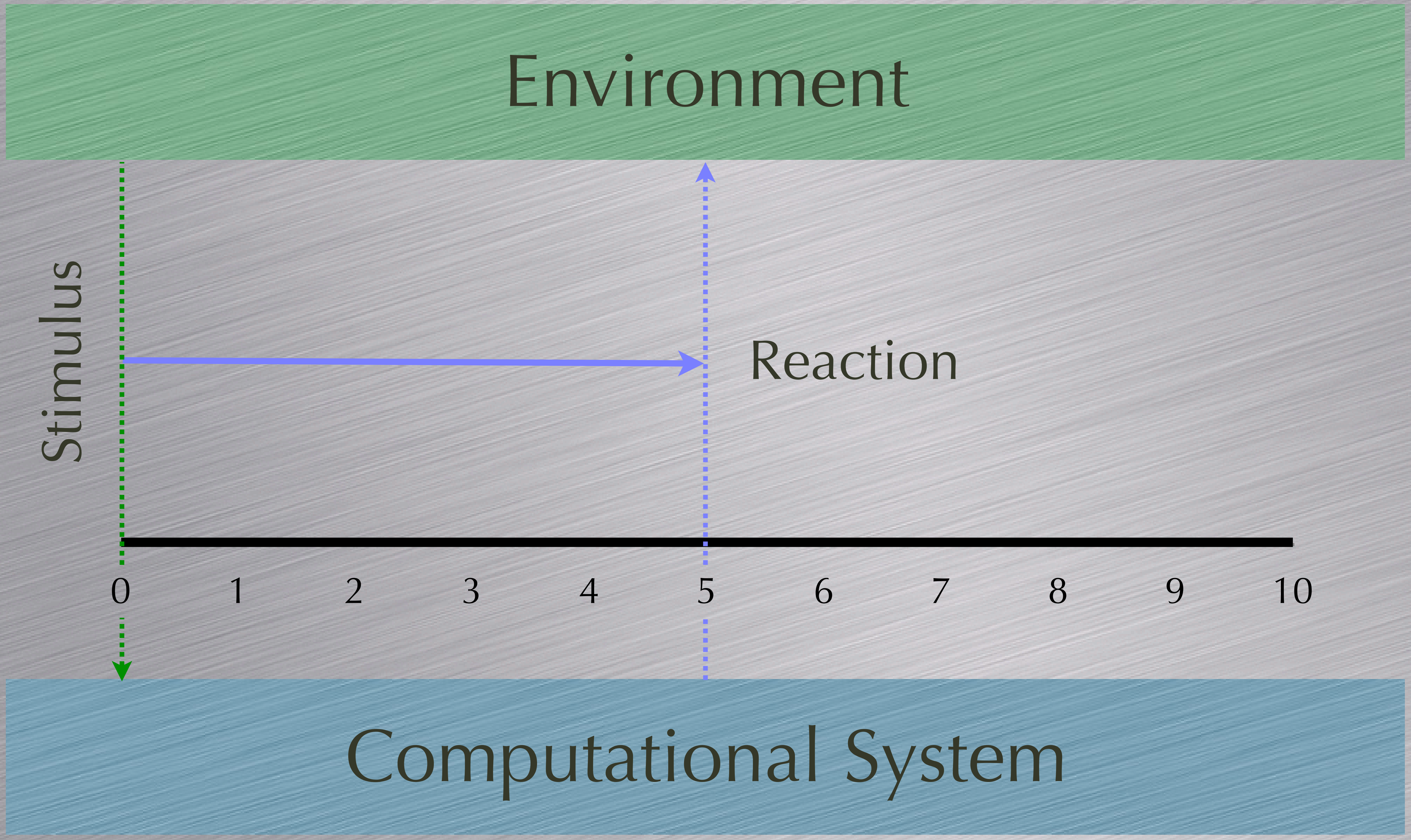


[httperf benchmarks]

The C10k Problem

- C10k: servers should be able to handle >10000 clients/sec
- Given 20000 clients and a 1GHz CPU with 2GB RAM & 1GBit/sec Ethernet
- We have 50KHz/client, 100KB/client, and 50KBit/sec/client
- Is this enough to grab 4KB from disk and send it to the network once a second for each of the 20000 clients?

The I/O Problem



Environment

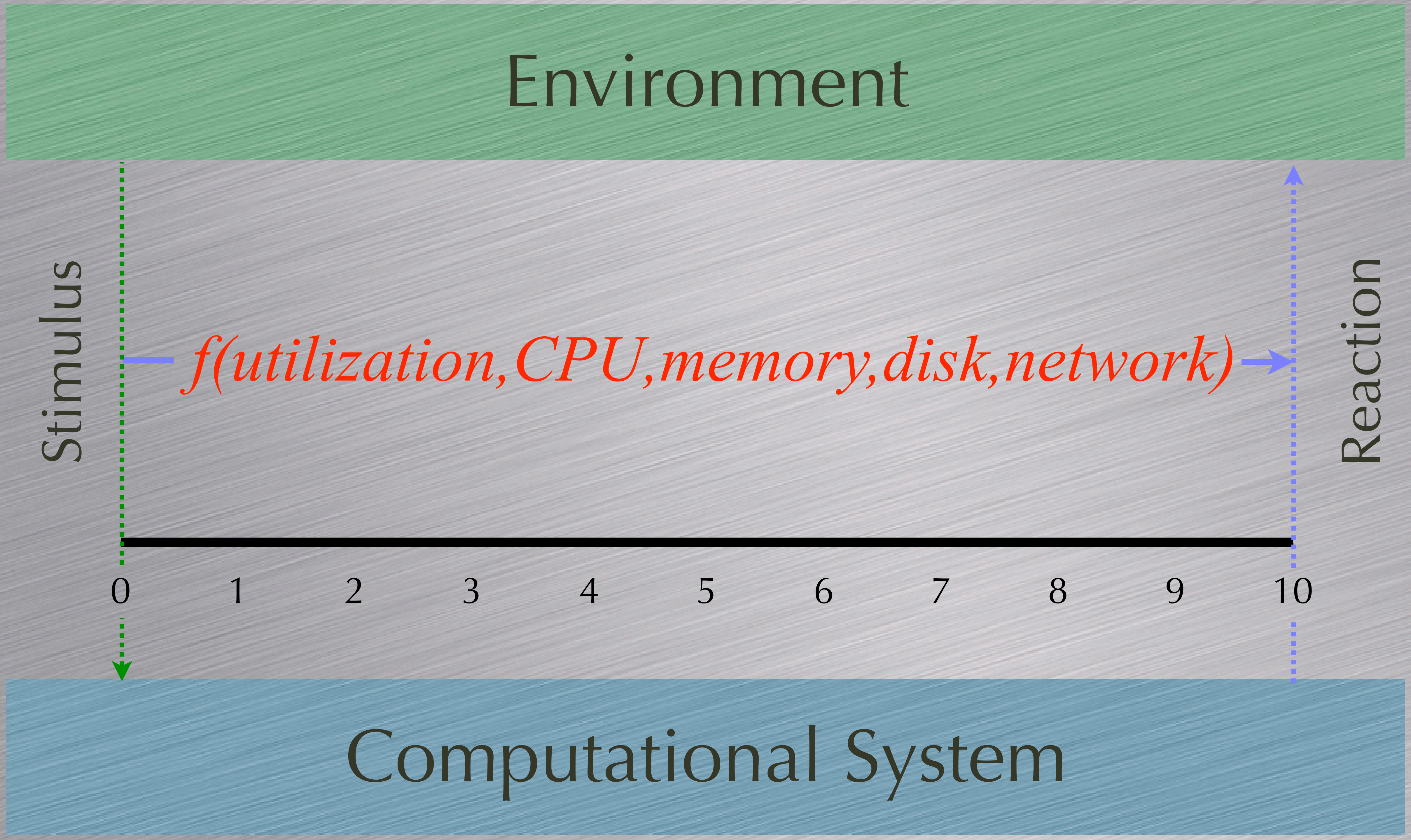
Stimulus

Reaction

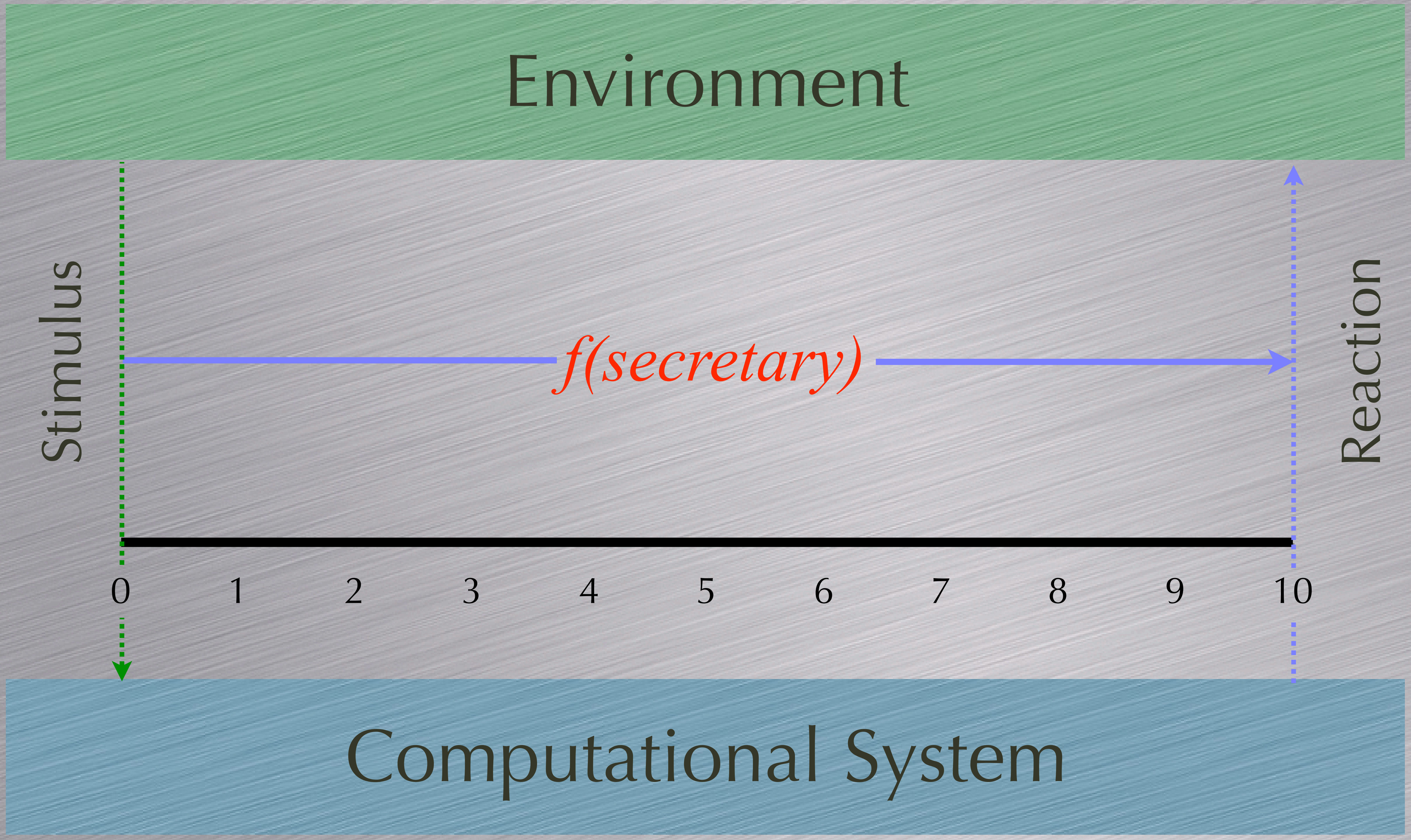
0 1 2 3 4 5 6 7 8 9 10

Computational System

State-of-the-Art



Threading by Appointment


Environment

Stimulus

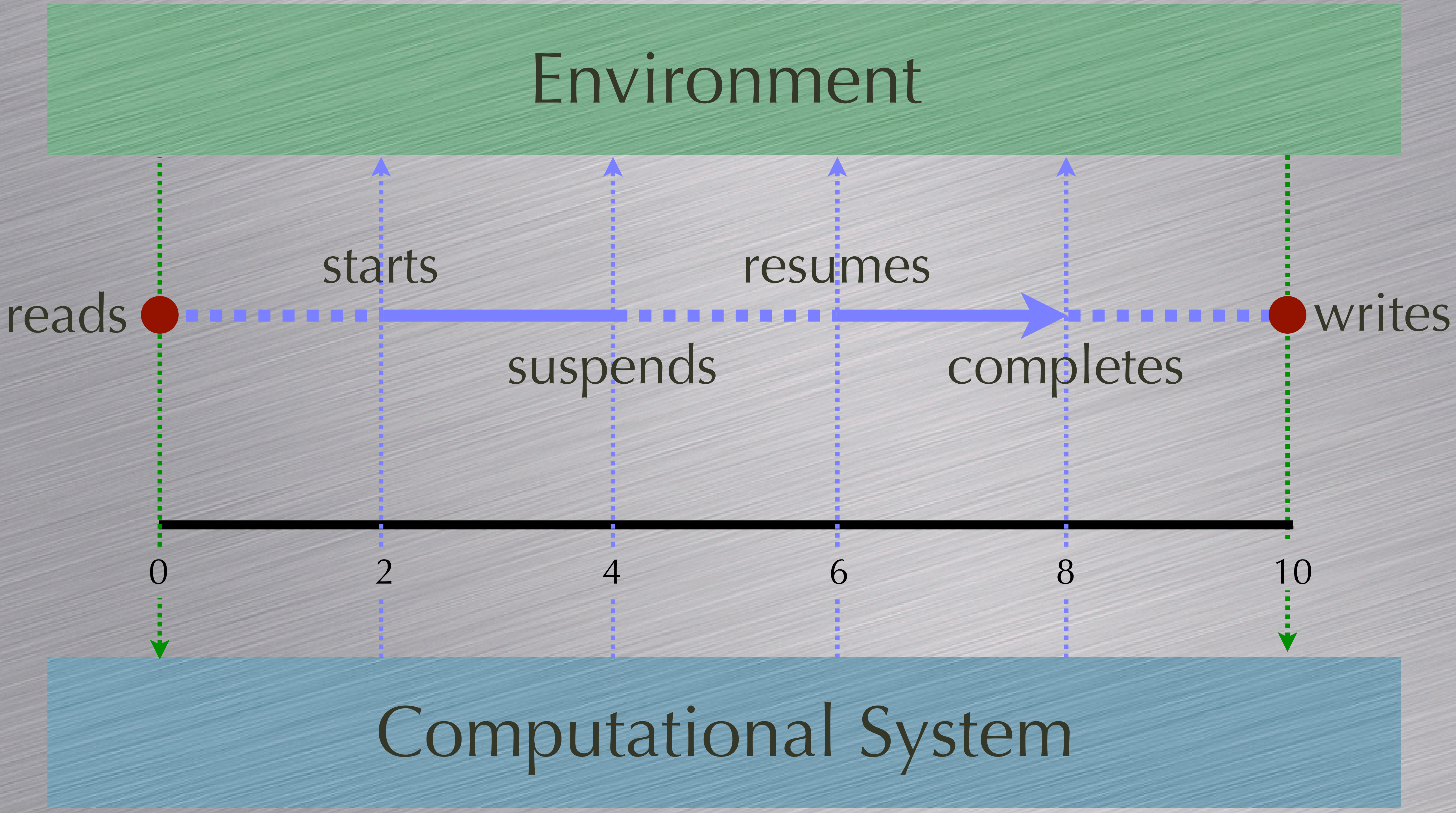
f(secretary)

Reaction

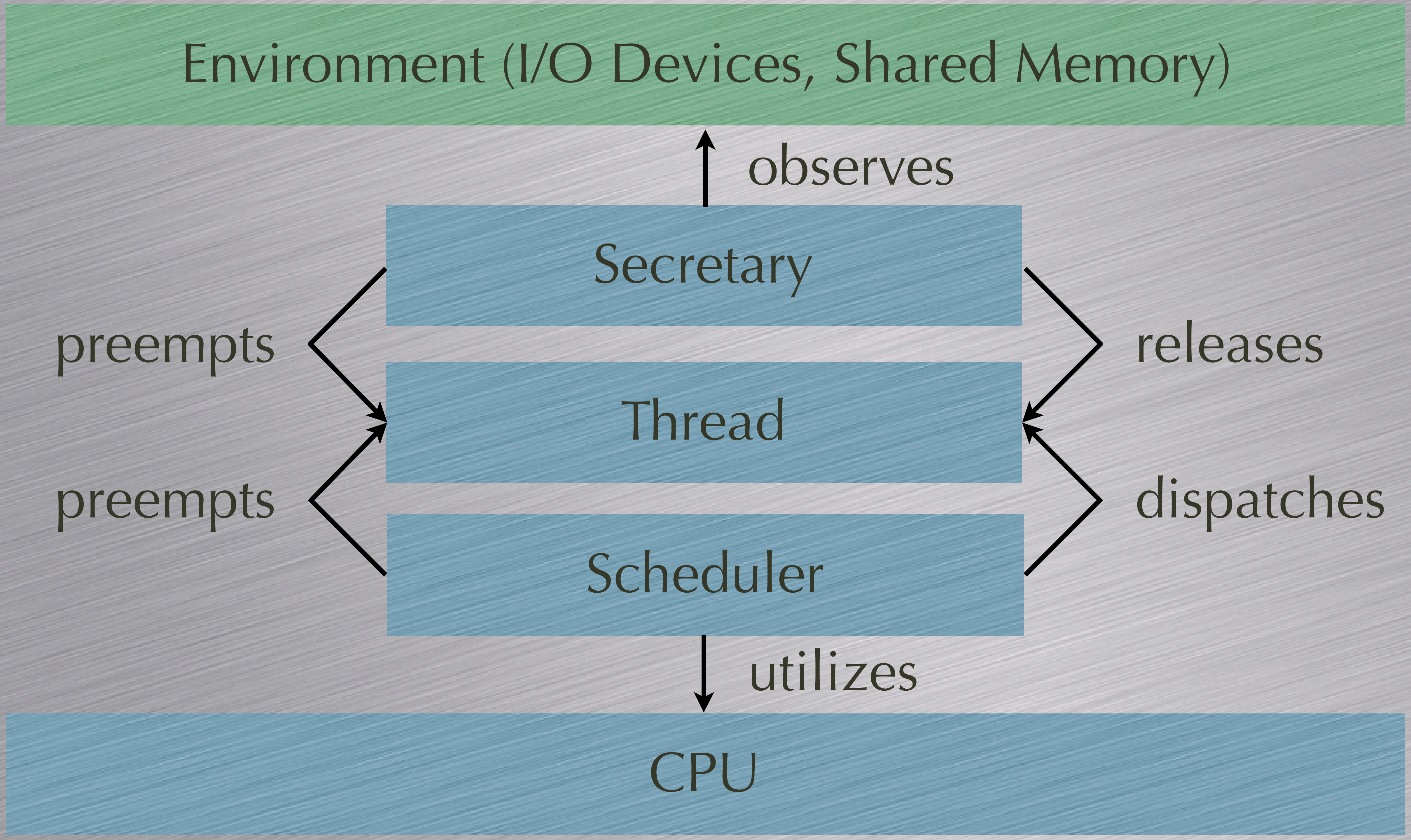
0 1 2 3 4 5 6 7 8 9 10

Computational System

Logical Execution Time



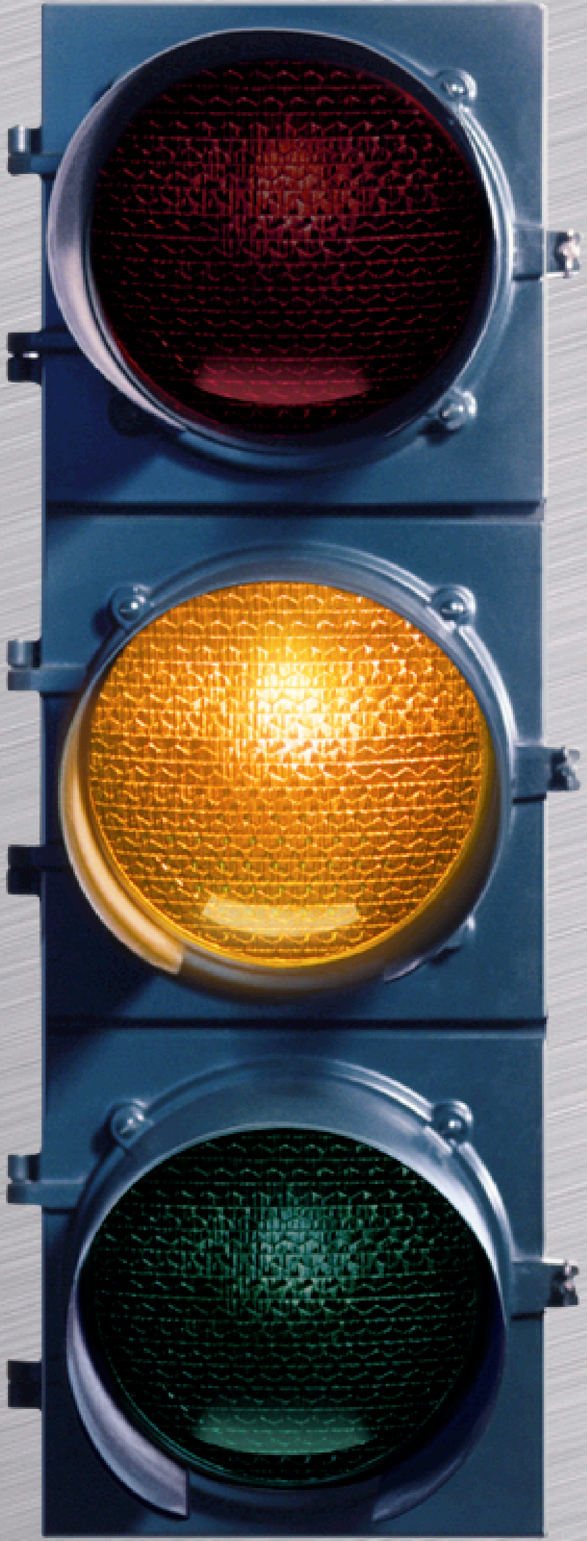
System Structures



Running Thread

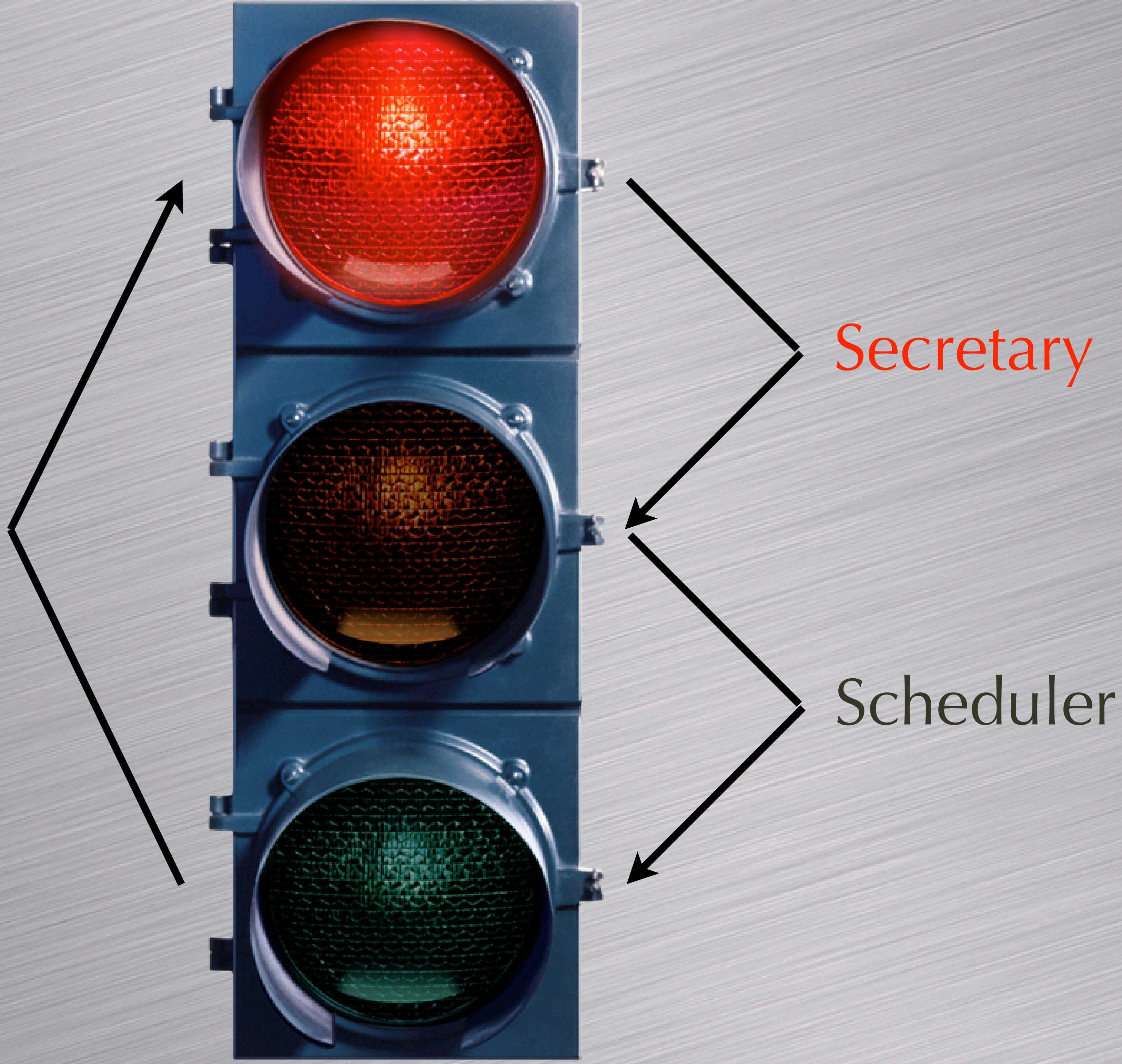
Blocked Thread

Released Thread

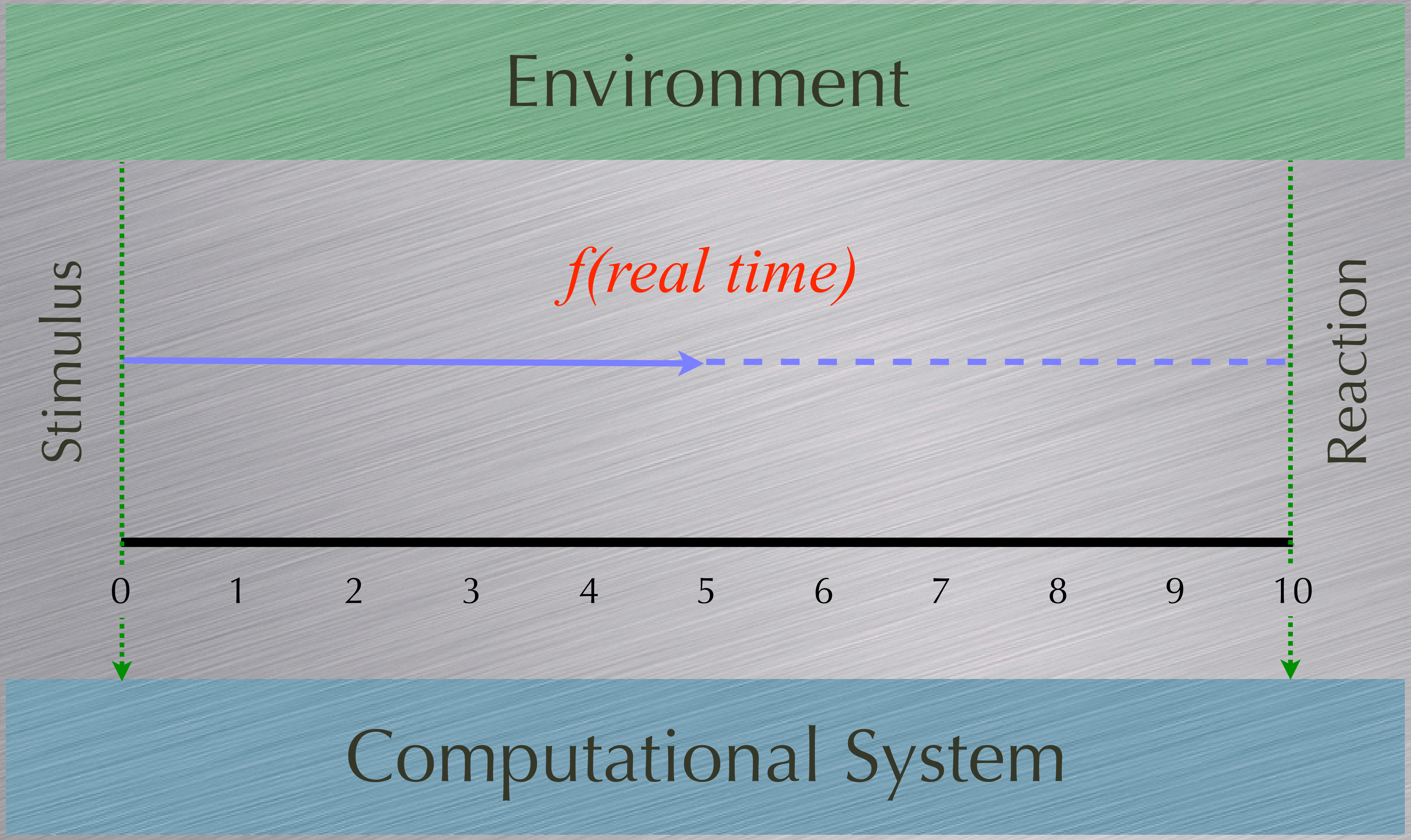


Running Thread

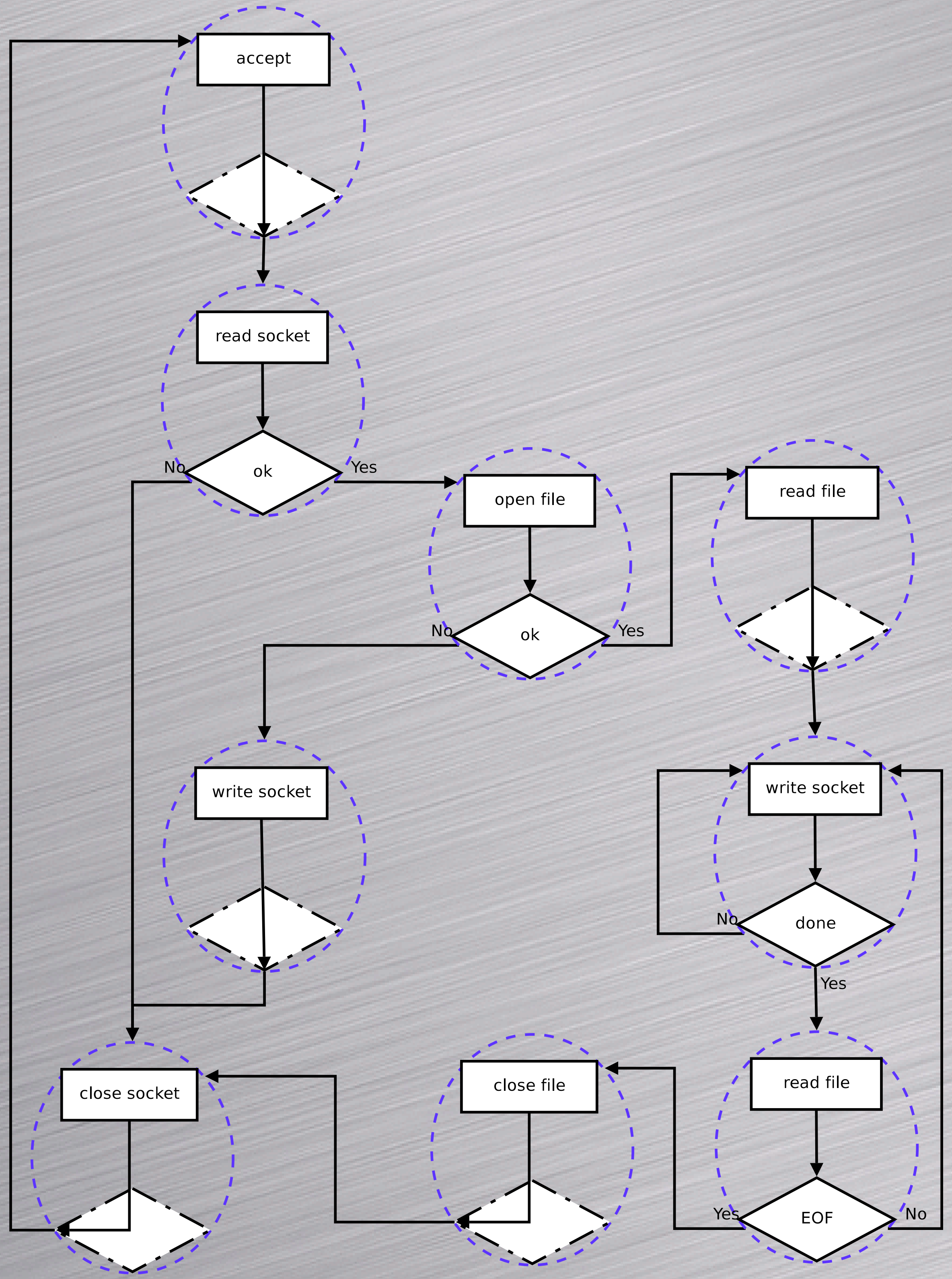
State Transitions



Secretary's Strategy



TAP Web Server



© C. Kirsch 2005

Predicted Behavior

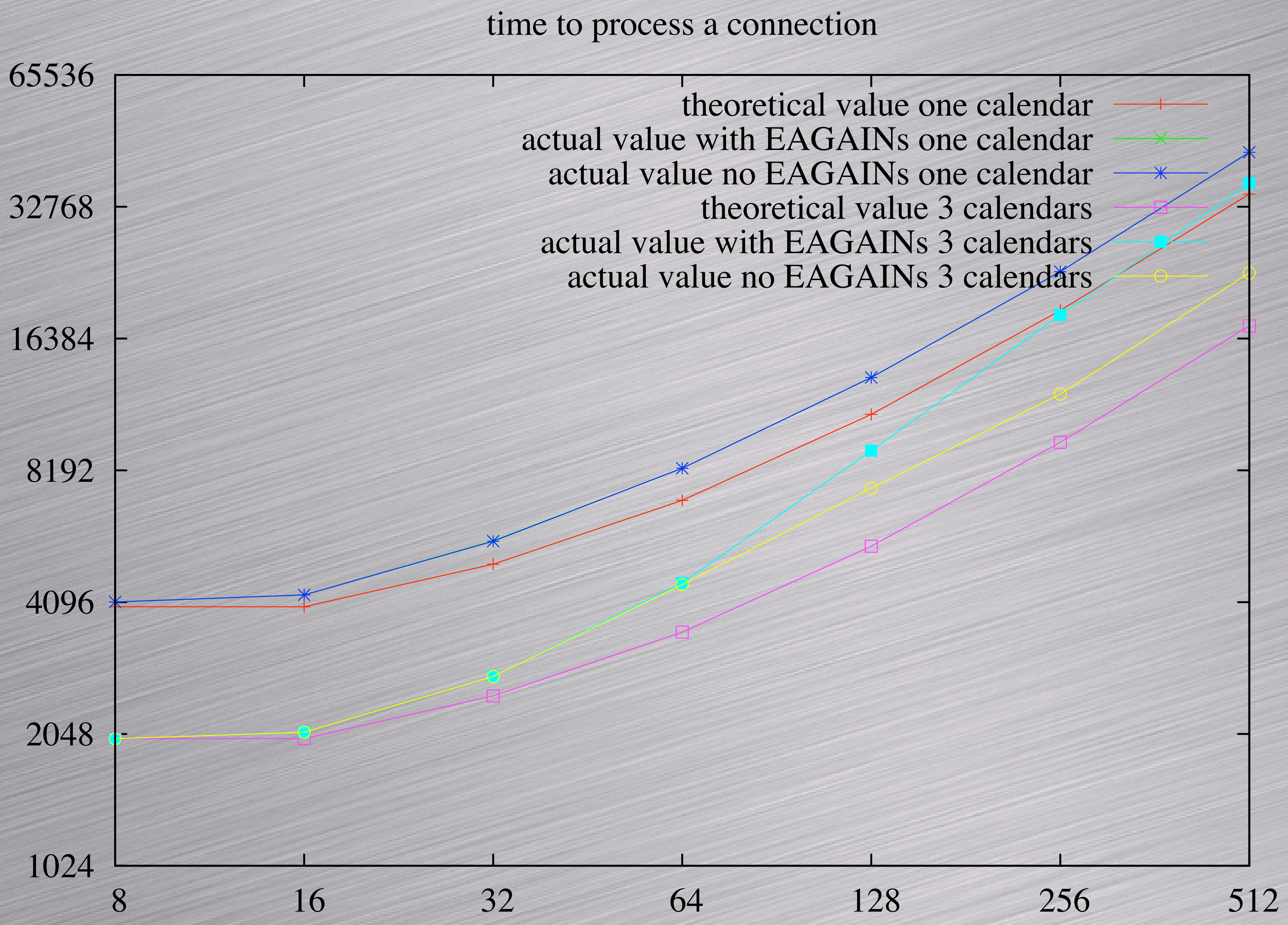
$$\text{Reply-Time} \leq N_A * f_T * f_C$$

N_A: number of appointments required for transaction

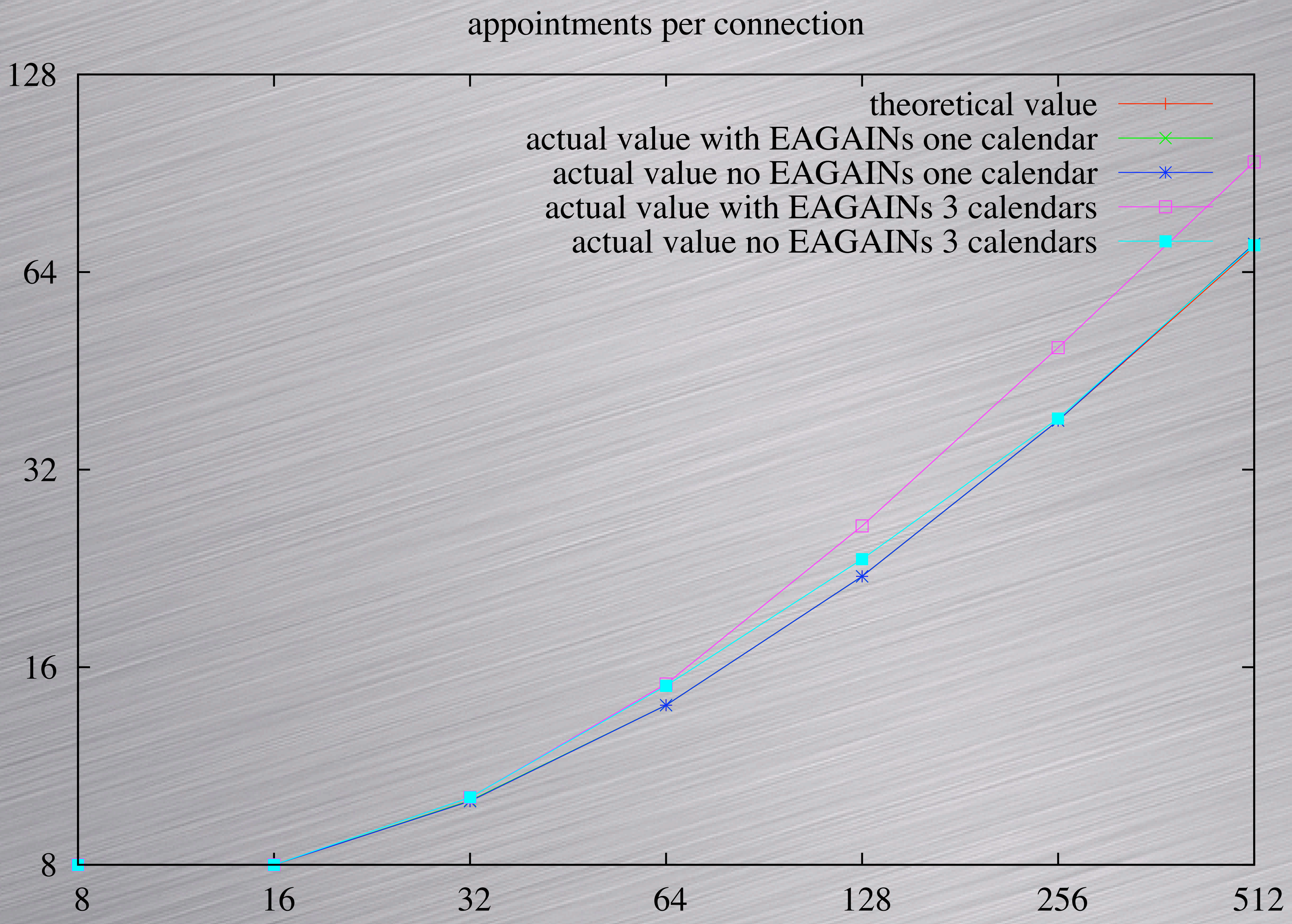
f_T : time between two appointments

f_C : number of other appointments between two transaction-related appointments

Experiments



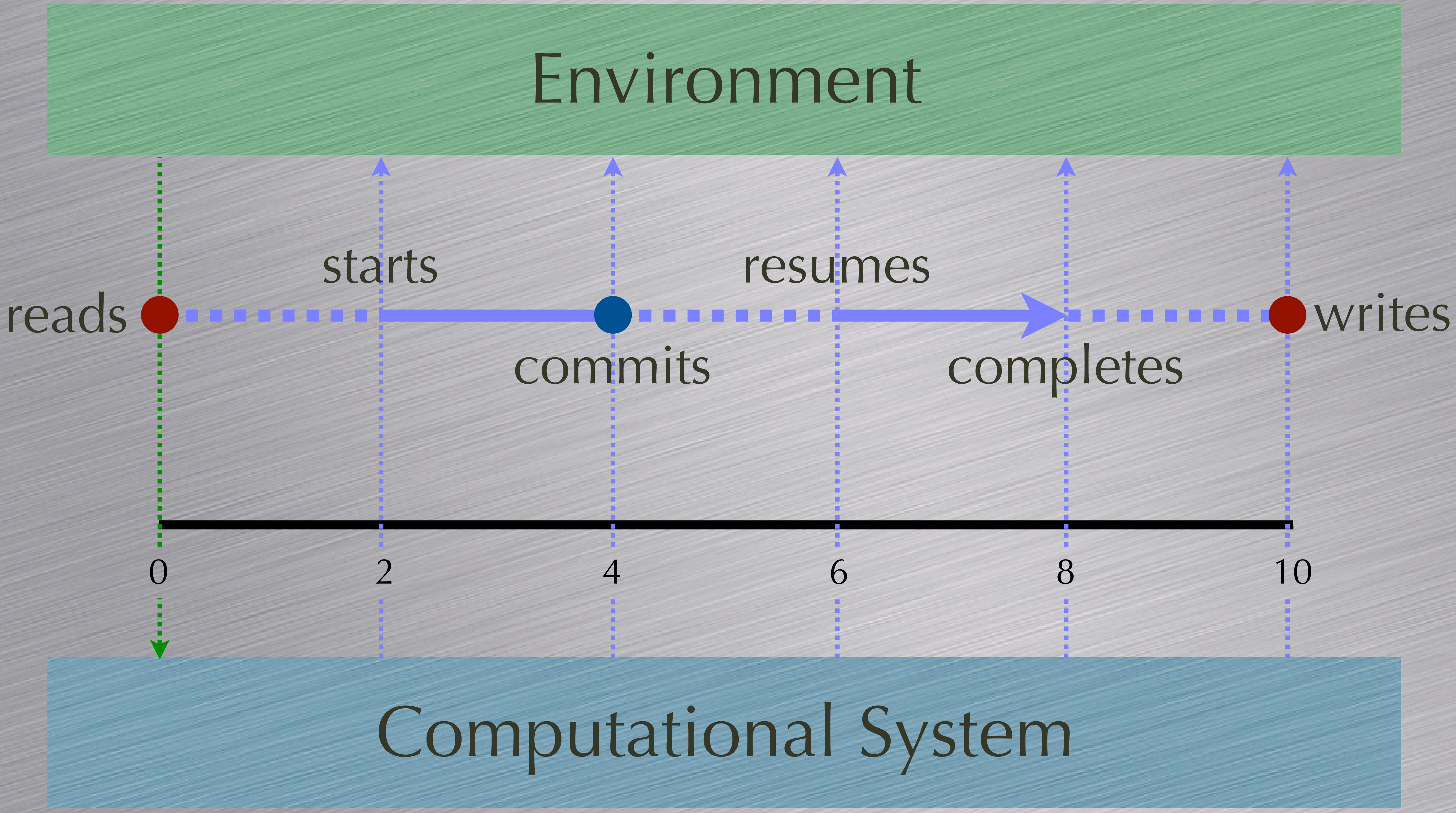
Experiments



setjmp/longjmp

- `int setjmp (jmp_buf env)`
  saves context in `env`
-
- `int longjmp(jmp_buf env, int val)`
  restores context from `env` previously saved by `setjmp`

Getting an Appointment



Thank you