Massachusetts Institute of Technology
Department of Electrical Engineering and Computer Science
6.826 Principles of Computer Systems

PROBLEM SET 4 SOLUTIONS

Problem 1. Timed Spec

a)

```
PROC TimedP() =
  VAR t: Time := now + minLatency |
  P();
  DO now < t => SKIP OD
```

b) Initialization:

```
<< deadlines := { now } + deadlines >>
```

Delay implementation:

```
PROC delay(k: Int) =
  VAR t:Time := now |
  << deadlines := deadlines - { t } + { t + k }; >>
  DO (now < t+k) => SKIP OD
```

c) In the implementation from *c*) time passes if *any* of the user threads is in its `delay` action, as opposed to time passing only when *all* user threads are in their `delay` actions.

Problem 2. Optimizing for the Uncommon Case

Average running time after the transformation A is:

$$\frac{(1-x)9+1}{10} = 1 - 0.9x$$

Average running time after the transformation B is:

$$\frac{7+(1-y)3}{10} = 1 - 0.3y$$

a) A is better iff $y < 3x$, B is better iff $y > 3x$ and they are equally bad iff $y = 3x$.

b) When $3x \geq 1$ i.e. $x \geq 1/3$ then B cannot be better than A. There is no value of B that guarantees that A can never be better.

Problem 3. Web Server

CPU utilization is $u_c = cn$. The request needs to be served first by CPU, which takes $c/(1-nc)$. There is no queue on the disk because all requests must first pass through CPU. So the total average response time is:

$$\frac{c}{1-nc} + d$$

Problem 4. Widgets

Each transaction requires

$$6ms + 2KB/(50MB/s) = 6.04ms$$

of disk time. Each transaction requires:

$$200K/(800M/s) = 0.25ms$$

of processor time.

a)

$$6.04ms/(0.25ms) = 24.16$$

so we need 25 disks.

b) Now the latency is averaged over a batch of 10 transactions, so latency becomes $0.6ms$ per transaction and total disk time per transaction is $0.64ms$.

$$0.64ms/(0.25ms) = 2.56$$

so we need 3 disks.

c) Now the CPU is the bottleneck so it determines the bottleneck. The rate is the inverse of time spent by the processor per request.

$$n = 1/(0.25ms) = 4(ms)^{-1}$$

So the rate is 4 transactions per millisecond.