Exercise 1

Consider to deploy the “Eager Reliable Broadcast” algorithm for the Regular Reliable Broadcast in a system consisting of 10 processes.

Assume that the computation delay on the processes is negligible and equal to 0, and that each process is able to send/receive (P2P) 10 msgs/sec.

If **each process execute a rb-broadcast every 20 seconds on average**, what is the expected time to rb-deliver to deliver a message in the system assuming all correct processes?

Algorithm 3.3: Eager Reliable Broadcast

Implements:
    ReliableBroadcast, **instance** *rb*.

Uses:
    BestEffortBroadcast, **instance** *beb*.

upon event ⟨ *rb*, *Init* ⟩ **do**
    delivered := ∅;

upon event ⟨ *rb*, *Broadcast* | m ⟩ **do**
    trigger ⟨ *beb*, *Broadcast* | [DATA, *self*, m] ⟩;

upon event ⟨ *beb*, *Deliver* | p, [DATA, s, m] ⟩ **do**
    if $m \notin$ *delivered* **then**
        delivered := *delivered* ∪ {m};
        trigger ⟨ *rb*, *Deliver* | s, m ⟩;
        trigger ⟨ *beb*, *Broadcast* | [DATA, s, m] ⟩;

message rate = 10 msgs/sec   service rate

rb-broadcast = 20 s on average   arrival time

10 processes

In the eager RB we assume all correct processes and the algo retrasmit every msg that received (broadcast | deliver).

Having 10 processes that broadcasts every 20 s on average, we can calculate the time that it is required to perform the operation for each process:

$\frac{1}{20} = 0.05$ msgs/sec   execution rate $= 10 \cdot 0.05$

Response time $= \dfrac{1}{10 \cdot 10 - 0.05} = 0{,}01$ s

arrival rate $\lambda = 10 \cdot 0{,}05 \cdot 10 = 5$ req/s

$$R = \frac{1}{\mu - \lambda} = \frac{1}{5} = 0{,}2 \text{ s}$$

Exercise 2

Consider the two distributed protocols presented in the following pseudo codes: *getValuesBiring* and *getValuesUniring.*

Suppose that the system is composed by **10 processes**, each of which is equipped with a local perfect failure detector that can provide the identity of a new process when a current one fails.

1) If a **process crashes every 200 minutes on average**, and the local failure detector takes **50 minutes to identify a faulty process** and to provide the identity of a new neighbor, evaluate the availability of the system
2) If each process receives **1 request** (i.e., a call to <G, get>) **per second**, what is the expected response time, i.e., the expected time between <G, get> and <G, return>, assuming that the computation delay on the processes is negligible and equal to 0, that no crashes occurs, and that each process is able to send/receive **10 msgs/sec**?

Exercise 2

Algorithm 7 getValuesUniring $\mathcal{G}$

procedure Init
  // v : local value that changes over time
  // next : neighbor's ID in the ring
  // id : local identifier

upon event $\langle \mathcal{G}, get \rangle$ **do**
  trigger $\langle PP2PL, send \mid next, \langle id, \emptyset \rangle \rangle$

upon event $\langle PP2PL, deliver \mid q, \langle source, values \rangle \rangle$ **do**
  locValues := values $\cup$ {v}
  if $source == id$ **then**
    trigger $\langle \mathcal{G}, return \mid locValues \rangle$
  else
    trigger $\langle PP2PL, send \mid next, \langle id, locValues \rangle \rangle$

upon event $\langle \mathcal{P}, newNext \mid newNext \rangle$ **do**
  next := newNext

Algorithm 8 getValuesBiring $\mathcal{G}$

procedure Init
  // v : local value that changes over time
  // left : neighbor's ID in the ring
  // right : neighbor's ID in the ring
  // id : local identifier

upon event $\langle \mathcal{G}, get \rangle$ **do**
  trigger $\langle PP2PL, send \mid left, \langle id, \emptyset \rangle \rangle$
  trigger $\langle PP2PL, send \mid right, \langle id, \emptyset \rangle \rangle$

upon event $\langle PP2PL, deliver \mid q, \langle source, values \rangle \rangle$ **do**
  locValues := values $\cup$ {v}
  if $source == id$ **then**
    trigger $\langle \mathcal{G}, return \mid locValues \rangle$
  else if $q == right$ **then**
    trigger $\langle PP2PL, send \mid left, \langle id, locValues \rangle \rangle$
  else
    trigger $\langle PP2PL, send \mid right, \langle id, locValues \rangle \rangle$

upon event $\langle \mathcal{P}, newRight \mid newRight \rangle$ **do**
  right := newRight

upon event $\langle \mathcal{P}, newLeft \mid newLeft \rangle$ **do**
  next := newLeft

10 processes ring

1) MTBF = 200 m
MTTR = 50 m

$$A = \frac{MTBF}{MTBF + MTTR} = 0,8 \qquad A = (0,8)^{10} = 0,107$$

2) arrival rate = 1 req/s (λ)

rate at which msgs arriving at a process

service rate = 10 req/s (μ)

a rate at which a system can handle, process, send, receive messages

$$R = \frac{1}{10 \cdot 10 - 1} = 0,01\ s$$

1 mex

unning $\lambda = 1 \cdot 10$

bining $\lambda = 2 \cdot 10$

2 mex

$\mu = 10$

EACH PROCESS RECEIVES 1 REQUEST

$\downarrow$

1 · PROCESSES

10 NODES in a RING $\rightarrow$ R · 10

Exercise 3

Consider to deploy the “Exclude on Timeout” algorithm for the perfect failure detector in a system consisting of **30 processes**.

Assume that the computation delay on the processes is negligible and equal to 0, and that each process is able to send/receive **10 msgs/sec**.

What is the expected **minimum value for the timeout Δ** that can be set?

Algorithm 2.5: Exclude on Timeout

```
Implements:
    PerfectFailureDetector, instance 𝒫.

Uses:
    PerfectPointToPointLinks, instance pl.

upon event ⟨ 𝒫, Init ⟩ do
    alive := Π;
    detected := ∅;
    starttimer(Δ);

upon event ⟨ Timeout ⟩ do
    forall p ∈ Π do
        if (p ∉ alive) ∧ (p ∉ detected) then
            detected := detected ∪ {p};
            trigger ⟨ 𝒫, Crash | p ⟩;
        trigger ⟨ pl, Send | p, [HEARTBEATREQUEST] ⟩;
    alive := ∅;
    starttimer(Δ);

upon event ⟨ pl, Deliver | q, [HEARTBEATREQUEST] ⟩ do
    trigger ⟨ pl, Send | q, [HEARTBEATREPLY] ⟩;

upon event ⟨ pl, Deliver | p, [HEARTBEATREPLY] ⟩ do
    alive := alive ∪ {p};
```

30 processes

service rate = 10 msgs/s

Time to get from the souce to the destination
is equal $\frac{1}{10} = 0,1$ s

Time for send and receive $2 \cdot 0.1 = 0.2$ s

$\Delta_{min} = 0,2$ s

$$\frac{\Delta}{2} = 1\left(10 - 30 \cdot \frac{\Delta}{2}\right)$$

Since every process are performing the failure detector, each process must be able in $\frac{\Delta}{2}$ to receive as many msgs as there are processes. $\Delta = 0,64$ s