Algorithm – Atomic Register

Algorithm 1 Read-Impose Write-Majority

Implements:

(1, N)-AtomicRegister, instance onar.

Uses:

BestEffortBroadcast, instance beb.
PerfectPointToPointLinks, instance pp2p.

1: upon event (Init) do
2: (ts, val) := (0, ⊥)
3: wts := 0
4: acks := 0
5: rid := 0
6: ∀p∈Π readlist[p] := ⊥
7: readval := ⊥
8: reading := FALSE
9: upon event (onar, Read) do
10: rid := rid + 1
11: acks := 0
12: ∀p∈Π readlist[p] := ⊥
13: reading := TRUE
14: trigger (beb, Broadcast | [READ, rid])
15: upon event (beb, Deliver | p, [READ, r]) do
16: trigger (pp2p, Send | p, [Value, r, ts, val])
17: upon event (pp2p, Deliver | p, [Value, r, ts', v']) do
18: if r = rid then
19: readlist[p] := (ts', v')
20: if |readlist| > \frac{N}{2} then ▷ Where N = |Π|.
21: (maxts, readval) := HIGHEST(readlist)
22: ∀q∈Π readlist[q] := ⊥
23: trigger (beb, Broadcast | [WRITE, rid, maxts, readval])
24: upon event ( onar, Write | v ) do
25:     rid := rid + 1
26:     wts := wts + 1
27:     acks := 0
29: upon event ( beb, Deliver | p, [Write, r, ts', v'] ) do
30: if ts' > ts then
31:     (ts, val) := (ts', v')
32:     trigger ( pp2p, Send | p, [Ack, r] )
33: upon event ( pp2p, Deliver | p, [Ack, r] ) do
34: if r = rid then
35:     acks := acks + 1
36: if acks > \frac{N}{2} then
37:     acks := 0
38: if reading then
39:     reading := FALSE
40:     trigger ( onar, ReadReturn | readval )
41: else
42:     trigger ( onar, WriteReturn | )
Algorithm 2 Read-Impose Write-Consult-Majority

Implements:
- $(N, N)$-AtomicRegister, instance $nnar$.

Uses:
- BestEffortBroadcast, instance $beb$.
- PerfectPointToPointLinks, instance $pp2p$.

1: upon event $\langle Init \rangle$ do
2: $(ts, wr, val) := (0, 0, \bot)$
3: $acks := 0$
4: $writeval := \bot$
5: $rid := 0$
6: $\forall p \in \Pi \ readlist[p] := \bot$
7: $readval := \bot$
8: $reading := \text{FALSE}$
9: upon event $\langle nnar, Read \rangle$ do
10: $rid := rid + 1$
11: $acks := 0$
12: $\forall p \in \Pi \ readlist[p] := \bot$
13: $reading := \text{TRUE}$
14: trigger $\langle beh, Broadcast \mid [\text{READ, rid}] \rangle$
15: upon event $\langle beh, Deliver \mid p, [\text{READ, r}] \rangle$ do
16: trigger $\langle pp2p, Send \mid p, [\text{VALUE, r, ts, wr, val}] \rangle$
17: upon event $\langle pp2p, Deliver \mid p, [\text{VALUE, r, ts', wr', v'}] \rangle$ do
18: if $r = rid$ then
19: $readlist[p] := (ts', wr', v')$
20: if $|\text{readlist}| \geq \frac{N}{2}$ then $\triangleright$ Where $N = |\Pi|$.
21: $(maxts, rr, readval) := \text{HIGHEST}(\text{readlist})$
22: $\forall q \in \Pi \ readlist[q] := \bot$
23: if $\text{reading}$ then
24: $bcastval := readval$
25: else
26: $rr := \text{RANK}(\text{self})$
27: $maxts := maxts + 1$
28: $bcastval := writeval$
29: trigger $\langle beh, Broadcast \mid [\text{WRITE, rid, maxts, rr, bcastval}] \rangle$
upon event ⟨nnar, Write | v⟩ do
31:   rid := rid + 1
32:   writeval := v
33:   acks := 0
34:   ∀p∈Π readlist[p] := ⊥
35:   trigger ⟨beb, Broadcast | [Read, rid]⟩
upon event ⟨beb, Deliver | p, [Write, r, ts′, wr′, v′]⟩ do
36:   if (ts′, wr′) > (ts, wr) then ▷ Tuple comparison.
37:       (ts, wr, val) := (ts′, wr′, v′)
38:   trigger ⟨pp2p, Send | p, [Ack, r]⟩
upon event ⟨pp2p, Deliver | p, [Ack, r]⟩ do
39:   if r = rid then
40:       acks := acks + 1
41:       if acks > N/2 then
42:           acks := 0
43:       if reading then
44:           reading := False
45:           trigger ⟨nnar, ReadReturn | readval⟩
46:       else
47:           trigger ⟨nnar, WriteReturn | ⟩