Paxos – Sample Input 1

Assume the following simulation:

$$(n_P = 1, n_A = 3, t_{\text{max}} = 15, E)$$

Where E contains a single event:

$$(t = 0, F = \emptyset, R = \emptyset, \pi_c = p_1, \pi_v = 42)$$

In other words, we have a system with one Proposer and three Acceptors, none of which fail during the simulation, and where the Proposer must propose value 42 at tick 0.

The simulation progresses as follows:

Tick 0

Since there is an event for this tick, we carry out the actions specified in the event. There are no computers to fail or recover, but there is a value to propose. This results in a `PROPOSE` message being delivered to p_1 (bypassing the network). According to the Paxos algorithm, the Proposer must send a `PREPARE` message to all the Acceptors. Enqueueing all three messages can be done in this single tick, but only one message will be delivered during each subsequent tick.

So, at the end of this tick, the contents of N are the following (the first row is the head of the queue, and the last row is the tail of the queue)

m.type	m.src	m.dst	Other attributes
`PREPARE`	p_1	a_1	proposal_id$=1$
`PREPARE`	p_1	a_2	proposal_id$=1$
`PREPARE`	p_1	a_3	proposal_id$=1$

Tick 1

There are no events for this tick, but there are messages pending delivery. We look at the first message in N, and see that both p_1 and a_1 are currently active (not failed), so we deliver the `PREPARE` message to a_1. According to the Paxos algorithm, since a_1 hasn't promised to participate in any other proposal, it can promise to participate in proposal 1. So, it will send a `PROMISE` message to p_1.

Contents of N at the end of tick 1

m.type	m.src	m.dst	Other attributes
PREPARE	p_1	a_2	proposal_id=1
PREPARE	p_1	a_3	proposal_id=1
PROMISE	a_1	p_1	proposal_id=1, prior_proposal=∅

Tick 2

Same as tick 1, but with Acceptor a_2.

Contents of N at the end of tick 2

m.type	m.src	m.dst	Other attributes
PREPARE	p_1	a_3	proposal_id=1
PROMISE	a_1	p_1	proposal_id=1, prior_proposal=∅
PROMISE	a_2	p_1	proposal_id=1, prior_proposal=∅

Tick 3

Same as tick 2, but with Acceptor a_3.

Contents of N at the end of tick 3

m.type	m.src	m.dst	Other attributes
PROMISE	a_1	p_1	proposal_id=1, prior_proposal=∅
PROMISE	a_2	p_1	proposal_id=1, prior_proposal=∅
PROMISE	a_3	p_1	proposal_id=1, prior_proposal=∅

Tick 4

In this tick, the `PROMISE` message from a_1 is delivered to p_1. p_1 makes a note of this, but doesn't send any messages, because a majority of Acceptors hasn't replied yet.

Contents of N at the end of tick 4

m.type	m.src	m.dst	Other attributes
PROMISE	a_2	p_1	proposal_id=1, prior_proposal=∅
PROMISE	a_3	p_1	proposal_id=1, prior_proposal=∅

Tick 5

In this tick, the `PROMISE` message from a_2 is delivered to p_1. This means a majority of Acceptors has promised to participate in proposal 1 (or, more accurately, promised to *not* participate in any proposals with a number lower than 1). This means p_1 can send `ACCEPT` messages to the Acceptors. Since the majority of Acceptors who replied with a `PROMISE` hadn't accepted any value in a prior proposal, the Proposer is free to choose any value (and it will use the number 42 specified in the first `PROPOSE` message).

Contents of N at the end of tick 5

m.type	m.src	m.dst	Other attributes
PROMISE	a_3	p_1	proposal_id=1, prior_proposal=∅
ACCEPT	p_1	a_1	proposal_id=1, value=42
ACCEPT	p_1	a_2	proposal_id=1, value=42
ACCEPT	p_1	a_3	proposal_id=1, value=42

Tick 6

In this tick, the remaining `PROMISE` message is delivered to p_1. However, since a majority was already reached in the previous tick, no new messages are generated.

Contents of N at the end of tick 6

m.type	m.src	m.dst	Other attributes
ACCEPT	p_1	a_1	proposal_id=1, value=42
ACCEPT	p_1	a_2	proposal_id=1, value=42
ACCEPT	p_1	a_3	proposal_id=1, value=42

Tick 7

In this tick, an `ACCEPT` message is delivered to a_1. According to the Paxos algorithm, a_1 must make a note of the proposal number and the accepted value, and it replies with an `ACCEPTED` message.

Contents of N at the end of tick 7

m.type	m.src	m.dst	Other attributes
ACCEPT	p_1	a_2	proposal_id=1, value=42
ACCEPT	p_1	a_3	proposal_id=1, value=42
ACCEPTED	a_1	p_1	proposal_id=1, value=42

Tick 8

Same as tick 7, but with Acceptor a_2.

Contents of N at the end of tick 8

m.type	m.src	m.dst	Other attributes
ACCEPT	p_1	a_3	proposal_id=1, value=42
ACCEPTED	a_1	p_1	proposal_id=1, value=42
ACCEPTED	a_2	p_1	proposal_id=1, value=42

Tick 9

Same as tick 8, but with Acceptor a_3.

Contents of N at the end of tick 9

m.type	m.src	m.dst	Other attributes
ACCEPTED	a_1	p_1	proposal_id=1, value=42
ACCEPTED	a_2	p_1	proposal_id=1, value=42
ACCEPTED	a_3	p_1	proposal_id=1, value=42

Tick 10

In this tick, the `ACCEPTED` message from a_1 is delivered to p_1. p_1 makes a note of this, but doesn't do anything, because a majority of Acceptors hasn't accepted the value yet.

Contents of N at the end of tick 10

m.type	m.src	m.dst	Other attributes
ACCEPTED	a_2	p_1	proposal_id=1, value=42
ACCEPTED	a_3	p_1	proposal_id=1, value=42

Tick 11

In this tick, the `ACCEPTED` message from a_2 is delivered to p_1. Since a majority of Acceptors have accepted value 42, p_1 determines that consensus has been achieved around that value. In a real Paxos system, this value would be broadcast to the Learners (or, alternatively, the `ACCEPTED` messages themselves would be broadcast to the Proposers and the Learners, and the Learners would learn about the consensus independently by noting that a majority of Acceptors had accepted a value).

Contents of N at the end of tick 11

m.type	m.src	m.dst	Other attributes
ACCEPTED	a_3	p_1	proposal_id=1, value=42

Tick 12

In this tick, the remaining `ACCEPTED` message is delivered. Since consensus had already been achieved, this message has no effect.

At this point, both N and E are empty, so the simulation can end (even though $t_{\max}$ is 15).