Summary of The Synod Algorithm (without the learning phase)

Process State (stored in stable storage)
Proposer:
maxPropNum = maximum proposal number that the proposer has seen so far

Acceptor:
maxPrepare = largest proposal that acceptor has sent a response to, initially 0
(accNum, accVal) = largest numbered proposal that acceptor has accepted, initially (null, null)

<table>
<thead>
<tr>
<th>Proposer</th>
<th>Acceptor</th>
</tr>
</thead>
<tbody>
<tr>
<td>1. Choose unique proposal number m that is greater than maxPropNum Set maxPropNum = m Send prepare(m) to all acceptors</td>
<td>2. if m &gt; maxPrepare maxPrepare = m send promise(accNum, accVal) to proposer else send nack(maxPrepare) to proposer</td>
</tr>
<tr>
<td>3. On receive of promise from majority set S of acceptors if all promises in S are (null, null) v = proposer’s value else v = accVal from promise with max. accNum send accept(m,v) to acceptors</td>
<td></td>
</tr>
<tr>
<td>4. if m ≥ maxPrepare maxPrepare = m (accNum, accVal) = (m,v) send ack to proposer else send nack(maxPrepare) to proposer</td>
<td></td>
</tr>
</tbody>
</table>

If proposer does not receive enough promises (after step 2) or acks (after step 4) by a “timeout”
Start step 1 again

If proposer receives nacks from a majority of acceptors (after step 2 or step 4)
Update maxPropNum based on nacks
Start step 1 again