22C:166 Distributed Systems and Algorithms
Homework 4
Total points = 50
Assigned 11/2/11, due 11/09/11 11:59 PM
You can work in groups of two for this assignment

Background. In a 1983 paper, Ben-Or [B83] showed how to overcome the FLP impossibility result on asynchronous consensus using probabilistic actions. Ben-Or’s solution is described below.

Let \( n \) be the total number of processes, of which at most \( t \) processes may crash. The proposed consensus algorithm progresses in several asynchronous rounds, each round consists of several steps. Based on the response received in a particular round, actions in the next round are determined. Only binary decision values (0 or 1) are considered. Each message sent out by a process has the following four fields:

- A step number \( s \) that indicates the current step in a round;
- A round number \( r \) that indicates the current round;
- A binary value \( b \) which is either 0 or 1;
- A flag \( u \) or \( d \) indicating two different stages (undecided or decided) in decision-making

<table>
<thead>
<tr>
<th>Step</th>
<th>{Program for process i}</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>(step 0: initialization) ( b := ) initial value of process ( i ); ( r := 0 ) do true ( \rightarrow )</td>
</tr>
<tr>
<td>1</td>
<td>{step 1} broadcast ( (1, r, b, u) )</td>
</tr>
</tbody>
</table>
| 2    | {step 2} receive at least \( n-t \) messages of type \( (1, r, \cdot, \cdot) \); \{Let \( m \) be the maximum number of processes that sent the same value \( v \)\}  
| 2.1  | if \( m > n/2 \) \( \rightarrow \) broadcast \( (2, r, v, d) \)  
| 2.2  | \( m \leq n/2 \) \( \rightarrow \) broadcast \( (2, r, b, u) \)  |
|      | fi |
| 3    | {step 3} receive at least \( n-t \) messages of type \( (2, r, \cdot, \cdot) \); \{Let \( p \) be the max \# of processes that sent \( (2, r, v, d) \) messages\}  
| 3.1  | if \( 0 < p < t + 1 \) \( \rightarrow \) \( b := v \);  
| 3.2  | \( p \geq t + 1 \) \( \rightarrow \) \( b := v \); decide \( v \); \{this is the final decision\}  
| 3.3  | \( p = 0 \) \( \rightarrow \) \( b := \) random \( \{0,1\} \);  |
|      | fi |
| 4    | {step 4} \( r := r+1 \) od |
What you have to do

Study how the solution works. Ben-Or claimed that the above algorithm solves the asynchronous consensus problem when \( n > 2t \). You have to prove the following three lemmas and answer the last question:

**Lemma 1.** If every process has the same initial value \( v \), then every process decides \( v \) within one round.

**Lemma 2.** Two non-faulty processes cannot decide different values.

*Hint.* Two different non-faulty processes may not reach agreement when they set their b-values differently using the action in line 3.1 or 3.2. To prove agreement, first show that in any round \( r \), it is impossible for one process \( i \) to receive a \( (2, r, v, d) \) message, and another process \( j \) to receive a \( (2, r, w, d) \) message, \((v \neq w)\).

**Lemma 3.** Show that at least one process eventually decides.

**Question.** If at least one process finally decides \( v \) in round \( r \), then in which round will every process finally decide \( v \)?

Observe that Step 2 requires "more than \( n/2 \) out of the \( n-t \) messages received" to have the same value \( v \) in order that a process changes its b-value to \( v \). This is not guaranteed unless \( n-t > n/2 \).

**Reference**


(Feel free to look at the original paper. The author did not present any proof of his algorithm there)