

## Assignment #7: Consensus and Self-Stabilization

**DUE:** Friday, June 1st.

---

### 1. Asynchronous Consensus (10 points)

Consider an asynchronous system with 6 processes, at most *one* of which can fail. A failure consists of a process crash (*i.e.*, the process stops executing actions). Channels are reliable so every message that is sent is guaranteed to arrive after some finite (though unbounded) delay.

Each process begins with an initial value consisting of a single bit (ie 0 or 1). Consider the following algorithm for solving consensus:

Every process sends its initial bit to all processes (including itself) and waits for 5 messages to arrive. Every process decides on the majority of the five bits received.

Is this algorithm correct? If so, prove it is correct. If not, construct a counter-example. That is, construct a scenario in which the algorithm does not satisfy all three of the required properties of consensus.

### 2. Synchronous Consensus (10 points)

**Part I.** Consensus in the presence of Byzantine faults is guaranteed only if  $N > 3f$ . Give an example of an execution of the Byzantine Agreement algorithm (without authentication) with 6 processes, 2 of which are faulty, in which consensus is *not* achieved.

**Part II.** Multiple rounds are required to achieve consensus in the presence of multiple Byzantine faults, even if authenticated messages are available. Consider modifying the Byzantine agreement algorithm (with authentication) so that it uses only 2 rounds. Give an example of an execution of this modified algorithm in which consensus is *not* achieved, despite  $N > 3f$ .

### 3. Self-stabilization (10 points)

Compare the speed of convergence of Dijkstra's  $k$ -state self-stabilizing token ring and 4-state token ring. (Calculate the number of algorithmic steps required, in the worst case, to return to a safe state.)