Theory Assignment 4: Early Stopping Consensus Algorithm for Crash Failures

SAI HARSHA KOTTAPALLI

CS17BTECH11036

(global constants)

integer: f ; // maximum number of crash failures tolerated(local variables)

integer: x ← local value;

integer curr_count, prev_count ← 0;

(1) Process Pi (1 ≤ i ≤ n) executes the consensus algorithm for up to f crash failures:

(1a) for round from 1 to f + 1 do

(1b) curr_count = 0

(1c) if the current value of x has not been broadcast then

(1d) broadcast(x);

(1e) yj ← value (if any) received from process j in this round;

(1f) if timeout then

(1g) curr_count += 1

(1h) x ← min ∀ j (x, yj) ;

(1i) if curr_count == prev_count then

(1j) break

(1k) output x as the consensus value.

Proof of correctness

- **Termination** condition is satisfied because the system is synchronous and every round has to end after a fixed time.
- **Agreement** is satisfied.

In unmodified algo, in f +1 rounds, there must be at least one round in which no process failed. The proof for the above statement is already explained in the textbook. Now, in the above algorithm when there is a round where the number of crashed processes are same as the round before it, then in this particular round, **code:** (1c)-(1e) and (1h) will make sure the minimum value is broadcasted to everyone and the processes which did not crash yet have received as well as processed the value and updated it. After which program exits, clearly the algorithm did not run for f+1 rounds but f’ + 1 rounds where f’ <= f. The actual number of crash failures being curr_count.

- **Validity** is satisfied.

The validity condition is satisfied because processes do not send fictitious values in this failure model. For all i, if the initial value is identical, then the only value sent by any process is the value that has been agreed upon as per the agreement condition.