ECE 526 (Spring 2013) Distributed Algorithms
Homework 2 Due: 9:30 a.m., February 12, 2013 (Tuesday)

Please type your answers (e.g., using Latex, or Word).

See the policy of 48-hour extension described in the course handout.

In questions below dealing with Byzantine consensus, f denotes the maximum number of processes that may be faulty.

1. Consider a synchronous system consisting of n processes, where n is an even number. The processes are connected by *undirected* links. Each process has links to d_i other processes, where $d_i > \frac{n}{2} + 1$. Let N_i denote the set of processes to whom process i has links. Thus, $d_i = |N_i|$. We do not include i in N_i, although process i can send messages to itself.

   Process i's initial state $v_i[0]$ is its input x_i (x_i is real-valued). Process i's state at the end of the t-th iteration is $v_i[t]$. The processes implement an iterative consensus algorithm.

   The above assumptions apply to both parts (a) and (b) below.

   - (a) Each process i performs the following state update in the t-th iteration.

   $$v_i[t] \;=\; \frac{1}{d_i+1} \sum_{j\in N_i\cup\{i\}}^{n} v_j[t-1]$$

   - (b) Each process i performs the following state update in the t-th iteration, where $\alpha = \frac{1}{2\ \max_i d_i}$.

   $$v_i[t] \;=\; (1-d_i\alpha)\, v_i[t-1] + \alpha \sum_{j\in N_i}^{n} v_j[t-1]$$

     See SUGGESTED EXERCISE for part (b).

   Each of the above algorithms can be expressed in a vector form as follows

   $$v[t] \;=\; M\ v[t-1]$$

   where $v[t]$ is a column vector consisting of the states of the n processes (with i-th element of $v[t]$ being $v_i[t]$), and M is an $n\times n$ *transition* matrix.

   For part (a) above, provide the following answers.

   (i) Determine matrix M.

(ii) Is matrix M row stochastic, column stochastic, or doubly stochastic?

(iii) Coefficient of ergodicity $\lambda(M)$ is defined in the paper by Wolfowitz (see link to the paper on the Lectures page). Assuming that $n > 8$, argue that $\lambda(M) < 1$ – if you believe that this is not true, provide an argument for that.

(iv) When the iterative computation is performed for a large number of iteration, does it lead to consensus, or average consensus ? Justify your answer.

2. Consider an *asynchronous* system. The communication network in the system is a complete graph. The processes in the system execute the following algorithm. Each process proceeds through "rounds" of the execution, with different processes possibly reaching different rounds at very different times (due to the asynchrony). Each process i maintains state v_i, with the initial state of the process being $v_i[0]$, and its state after its t-th round being $v_i[t]$. The input at each process is real-valued.

   Each process i performs the following steps in its t-th round:

   (a) Send message $< i, t, v_i[t-1] >$ to all the processes (including itself). The second field being t indicates that this is a round t message. The last field is said to be the "value" of this message.

   (b) Wait for round t messages from $n - f$ processes; define S to be the multiset of values contained in these $n - f$ messages.
   Eliminate the smallest f and the largest f values from S.
   $v_i[t]$:= average of the remaining $n - 3f$ values in S.

   The above algorithm needs to satisfy the following two conditions for "approximate agreement", where ϵ is a positive constant.

   - ϵ-Agreement: For a sufficiently large t, $|v_i[t+u] - v_j[t+u]| \leq \epsilon$, where $u \geq 0$.
   - *Validity:* $v_i[t]$ must be within the range of the inputs at the fault-free processes.

   The above algorithm is known to satisfy the above conditions when the number of processes $n \geq 5f + 1$.

   Prove that the algorithm will not always satisfy the above conditions when $n = 3f+1$. Hint: Try to prove this for $f = 1$ and $n = 4$.

SUGGESTED EXERCISES
(not for credit – you do not need to submit solutions for these)

- Answer question 1 above for part (b).

- Consider the algorithm for average consensus over lossy links in the paper by Vaidya, Hadjicostis and Domnguez-Garcia (from CDC 2012). The paper is linked from the Lectures page. The algorithm uses two parallel iterations, and the consensus value is obtained as a *ratio* of the state of the two parallel iterations.

  To implement the algorithm, the processes must transmit their state (or a function of the state) to their neighbors in each iteration. Will the correctness of the algorithm hold if the states of the two *parallel* iterations are transmitted via two *separate* messages? The two messages may be lost *independent* of each other. Explain your answer.

- Given a timeline for multiple processes communicating via message passing, you should be able to determine which events are concurrent, and whether there is a *happens before* relation between a pair of events. See such an example in the slides.

- Given a timeline for multiple processes communicating via message passing, and a cut, you should be able to determin the maximal consistent cut that is contained in that cut. See such an example (and the algorithm for computing this cut) in the slides.