CS 603 (Park) **Assignment VI** **Due: Mar. 21 (Fri.), 1997**

PROBLEM 1

Read Sections 4.1–4.5 from Tanenbaum.

PROBLEM 2

A `barrier` is a synchronization primitive used in concurrent systems for achieving lock-step progress in computation/communication. A centralized algorithm consists of a *barrier server* which collects "ready" messages from the participating group members (invoked when calling `barrier`), waiting for all ready messages to arrive, at which time, a "go-ahead" message is broadcast to the group members which *returns* the blocking `barrier` call. Devise a distributed algorithm which uses sequence numbers to achieve decentralized synchronization. I.e., each `barrier` call multicasts a "ready" messages to all other group members where the message carries a sequence number identifying the iteration count. Use UDP with time-outs to achieve reliable transmission. Test your algorithm on 6 Xinu machines each running a driver program of the form

repeat 20 times
generate a random number x between 0–999
multicast x
compute and print the average of the numbers received
`barrier()`

Here *multicast* is an unreliable primitive. Group membership is ascertained by an initialization phase which reads the IP addresses and relevant port numbers from a data file.

PROBLEM 3

(a) The *group membership problem* consists of maintaining a consistent view of the non-faulty, i.e., *up*-nodes in a distributed system. Assuming bounded delay, synchrony, and fail-silent faults, devise a distributed algorithm which achieves consensus. I.e., if the delay bound is given by D, the synchrony lag given by T, and $[t_1, t_2]$ is any time interval such that $t_2 - t_1 > D + T$, then group membership consensus is reached and maintained among the *up*-nodes throughout $[t_1', t_2]$ where $t_1' = t_1 + D + T$. For simplicity, assume group membership changes are detected via a periodic reporting scheme where each *up*-member multicasts a "I am alive" (IAA) message which contains the sender's group view *among other things*. The period may be assumed to be miniscule (compared to D and T) so as to be ignorable in the analysis.

(b) Implement and test your algorithm on 8 Xinu machines where, after initial synchronization to an absolute starting time (recall the trick used in the *spy problem*), the first machine terminates itself after 10 seconds, the second and third after 20 seconds, and the fourth after 30 seconds. Set the reporting period to be 500ms. Use UDP for communication (without further reliability provision), set $D = 1$sec, and there is no need to implement T explicitly.

PROBLEM 4

Give a formal proof of the impossibilty result for consensus in the *two-army problem* (i.e., "Alice-Bob lunch appointment problem" discussed in class). Make sure to state the exact assumptions under which the impossibility result is true. That is, the first task is to *state* the theorem in its formal, correct form, followed by a rigorous proof.