Parallel and Distributed Systems Group
Department of Software and Computer Technology
Faculty of Electrical Engineering, Mathematics, and Computer Science
Delft University of Technology

Examination Distributed Algorithms (IN4150)

10 april 2012, 9-12 AM

Notes:

1. The number of exercises is 4, and the number of pages is 2.
2. The solutions to the exercises can be either in Dutch or in English.
3. Try to give **short, concise, and precise answers**.
4. The maximum number of points to be obtained for each part of each exercise is indicated between parentheses. The final grade is computed as 12 plus the total number of points obtained, divided by 10 and rounded to the nearest integer.

1. (a) (4) Give the definition of causal ordering of point-to-point messages.
   (b) (6) Give an example of a system with three processes in which causal ordering of point-to-point messages is violated.
   (c) (7) Give in words or in pseudo-code the algorithm of Schiper-Eggli-Sandoz for causal ordering of point-to-point messages.
   (d) (5) Show how the algorithm of (c) would have enforced causal message ordering in your example of (b).

2. (a) (6) Give in words or in pseudo-code Peterson's algorithm for election in a unidirectional ring.
   (b) (4) What is the worst-case message complexity of this algorithm? Explain your answer!
   (c) (6) Consider rings of 8 and 16 processors. Give arrangements of the numbers 0,1,...,7 and 0,1,...,15 along these rings as the ids with which the processors start the election such that it takes three or four rounds until only one processor remains active, respectively.
   (d) (6) Describe a general procedure for arranging the numbers $0, 1, \ldots, N-1$ with $N = 2^n$ for any value of n along a ring as the ids with which the processors start the election such that it takes n rounds until only one processor remains active.

3. (a) (4) Formulate the Byzantine agreement problem. In particular, state the conditions for *agreement* and *validity*.

   (b) (8) Give in words or in pseudocode the algorithm for randomized Byzantine agreement.

   Given a system with 11 processes, two of which are faulty, five of which are correct and have an initial value of 0, and four of which are correct and have an initial value of 1.

   (c) (5) Is it possible that all correct processes decide 0 in the first round? If yes, show an execution of the algorithm in which this happens (and in which it is clear that the faulty processes are indeed faulty). If no, argue why not.

   (d) (5) Is it possible that any correct process decides 1 in the first round? If yes, show an execution of the algorithm in which this happens (and in which it is clear that the faulty processes are indeed faulty). If no, argue why not.

4. (a) (5) Explain the concepts of "infective," "susceptible," and "removed" nodes in epidemic protocols.

   (b) (5) Explain why in a (simple) epidemic protocol, using only the pull model gives faster convergence to the state in which every node is infective than using only the push model.

   (c) (6) Explain the `fingertable` data structure in Chord, and show how searching can be optimized with the help of this data structure.

   (d) (6) Explain the structure of the routing table used by Pastry nodes for searching for a file.

Parallel and Distributed Systems Group
Department Software and Computer Technology
Faculty EEMCS
DELFT UNIVERSITY OF TECHNOLOGY

DISTRIBUTED ALGORITHMS (IN4150)

List of algorithms

D.H.J. Epema

April 2012

Below is a list of the algorithms in the order of treatment in the course, with the names of the original authors, if applicable.

Chapter 3: Synchronization

1. Alpha-, beta-, and gamma-synchronizer: Awerbuch
2. Causal message ordering (broadcast): Birman-Schiper-Stephenson
3. Causal message ordering (point-to-point): Schiper-Eggli-Sandoz
4. Total message ordering
5. Determining global states: Chandy-Lamport
6. Termination detection in a unidirectional ring
7. Termination detection in a general network
8. Deadlock detection for AND requests: Chandy-Misra-Haas
9. Deadlock detection for OR requests: Chandy-Misra-Haas
10. Deadlock detection for M-out-of-N requests (with/without instantaneous communication): Bracha-Toueg

Chapter 4: Coordination

1. Assertion-based mutual exclusion: Lamport
2. Assertion-based mutual exclusion: Ricart-Agrawala
3. Assertion-based mutual exclusion: Maekawa
4. Generalized assertion-based mutual exclusion

5. Token-based mutual exclusion: Suzuki-Kasami
6. Token-based mutual exclusion: Singhal
7. Detection of loss and regeneration of a token
8. Election in a synchronous unidirectional ring (non-comparison-based)
9. Election in a bidirectional ring: Hirschberg-Sinclair
10. Election in a bidirectional ring (enhanced version)
11. Election in a unidirectional ring: Chang-Roberts
12. Election in a unidirectional ring: Peterson
13. Election in a synchronous complete network: Afek-Gafni
14. Election in an asynchronous complete network: Afek-Gafni
15. Minimum-weight spanning trees: Gallager-Humblet-Spira

Chapter 5: Consensus

1. Agreement with stopping failures
2. Byzantine agreement with oral messages: Lamport-Pease-Shostak
3. Byzantine agreement with authentication: Lamport-Pease-Shostak
4. Randomized Byzantine agreement
5. Stabilizing mutual exclusion: Dijkstra
6. Stabilizing stop-and-wait datalink algorithm
7. Stabilizing sliding-window datalink algorithm

Chapter 6: Peer-to-peer Systems

1. Unstructured: Freenet and Gnutella
2. DHT: Chord, Pastry
3. BitTorrent
4. Epidemic protocols: anti-entropy and rumor mongering