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

Examination Distributed Algorithms (IN4150)

23 June 2015, 6:30-9:30 PM

Notes

1. The **number of exercises** in this exam is 3, and the **number of pages** of this exam is 2.
2. The **maximum number of points** to be obtained for each part of each exercise is indicated between parentheses.
3. The **final grade** for the exam is computed as 12 plus the total number of points obtained for the three exercises and the paper summary divided by 10 and rounded to the nearest integer or integer+0.5.
4. The solutions to the exercises can be either **in Dutch or in English**.
5. Give **short, concise, and precise answers**.

1. (a) (6) Explain the concept and goal of *synchronizers* in an asynchronous distributed system.
   (b) (4) Explain how a node can determine that it has received all messages that were sent to it in a certain round (use the notion of a *safe node*).
   (c) (6) Explain the alpha-synchronizer and its time and message complexity.
   (d) (6) Explain the beta-synchronizer and its time and message complexity.

2. (a) (3) Explain the notion of *request sets* in Maekawa's algorithm for mutual exclusion.

   (b) (3) What is the order of the minimal size of these sets when there are N processes? Give an example where this order is attained.

   (c) (8) Explain **in words** Maekawa's algorithm for mutual exclusion, without the part dealing with deadlock prevention.

   (d) (8) Explain **in words** the deadlock-prevention part of Maekawa's algorithm for mutual exclusion.

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

   Assume in the rest of this exercise that no authentication is used, and that n is the total number of generals and f the number of traitors.

   (b) (6) Show that there is no solution to Byzantine agreement with $n = 3$ and $f = 1$.

   (c) (6) Explain **in words** the algorithm OM(f) for Byzantine agreement.

   (d) (6) Assume that $n = 7$ and $f = 2$, and that the commander does not exhibit failures (is loyal). List in a systematic way all the messages a loyal lieutenant receives in every round of the algorithm, and show in detail how he deduces his final decision from these messages.

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

Distributed Algorithms (IN4150)

List of algorithms

March 27, 2015

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

Chapter 5: Fault Tolerance

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