ROYAL INSTITUTE
OF TECHNOLOGY

ID2203 - Distributed Systems, Advanced Course

Exam Preparation

Course leader: Paris Carbone

Assistant: Harald Ng, Max Meldrum

{hng, mmeldrum}@kth.se

ROYAL INSTITUTE OF TECHNOLOGY

Exam Structure

- 2 Sections, 50P total + quizzes, programming exercises, and project
  - 30P Multiple Choice Questions
  - 20P Reasoning Questions
- 2h total time
- One attempt on Canvas

MCQ Example

Let $v(e)$ denote the vector clock of event e, and $t(e)$ denote the Lamport logical clock of event e. Which of the following statements are true?

(a) $v(a) < v(b)$ implies that $t(a) < t(b)$

(b) $t(a) < t(b)$ implies that $v(a) < v(b)$

(c) $v(a) < v(b)$ implies that $\neg(t(b) < t(a))$

(d) $t(a) < t(b)$ implies that $v(a) \leq v(b)$

MCQ Example

Let $v(e)$ denote the vector clock of event e, and $t(e)$ denote the Lamport logical clock of event e. Which of the following statements are true?

(a) $v(a) < v(b)$ implies that $t(a) < t(b)$

(b) $t(a) < t(b)$ implies that $v(a) < v(b)$

(c) $v(a) < v(b)$ implies that $\neg(t(b) < t(a))$

(d) $t(a) < t(b)$ implies that $v(a) \leq v(b)$

ROYAL INSTITUTE OF TECHNOLOGY

MCQ Example

Let $v(e)$ denote the vector clock of event e, and $t(e)$ denote the Lamport logical clock of event e. Which of the following statements are true?

(a) $v(a) < v(b)$ implies that $t(a) < t(b)$

(b) $t(a) < t(b)$ implies that $v(a) < v(b)$

(c) $v(a) < v(b)$ implies that $\neg(t(b) < t(a))$

(d) $t(a) < t(b)$ implies that $v(a) \leq v(b)$

ROYAL INSTITUTE OF TECHNOLOGY

MCQ Example

Let $v(e)$ denote the vector clock of event e, and $t(e)$ denote the Lamport logical clock of event e. Which of the following statements are true?

(a) $v(a) < v(b)$ implies that $t(a) < t(b)$	+1/2P
(b) $t(a) < t(b)$ implies that $v(a) < v(b)$	-1/2P
(c) $v(a) < v(b)$ implies that $\neg(t(b) < t(a))$	-1/2P
(d) $t(a) < t(b)$ implies that $v(a) \leq v(b)$	+1/2P

ROYAL INSTITUTE OF TECHNOLOGY

MCQ Point Total

- The MCQ part of the exam is subdivided into multiple subsections (e.g., *Basic Abstractions* & *Failure Detector*)
- Each section has an associated point total (2P/Question)
- Section point total with will be `max(0, s)`, where `s` is simply the sum of all individual questions within the section
- This means negative points do not carry across sections!

What to learn?

- All of the formal definitions
- All of the system/failure models
- All of the abstractions (their properties)
- Relationships between the abstractions (reductions)
- The high level mechanisms that make the algorithms work, e.g.
  - Read-Impose mechanism
  - Paxos invariants
  - log reconciliation in Raft

What not to learn?

- Correctness Proofs for the algorithms
  - Though it might help you learn the mechanisms to read them again
- Pseudocode for the algorithms
  - You'll be given that in the exam, if required

Exercise 1

Does the following statement satisfy the synchronous-computation assumption?

On my server, no request ever takes more than 1 week to be processed.

ROYAL INSTITUTE
OF TECHNOLOGY

Exercise 2

In a fail-stop model, mark the following properties as safety or liveness.

1. every process that crashes is eventually detected
2. no process is detected before it crashes
3. no two processes decide differently
4. no two correct processes decide differently
5. every correct process decides before `t` time units
6. if some correct process decides then every correct process decides.

ROYAL INSTITUTE
OF TECHNOLOGY

Exercise 3

Why do we need partial synchrony in Paxos? Which property of Uniform Consensus cannot be achieved if Paxos is used in an asynchronous model?

Uniform Consensus Properties:

(1) Termination: Every correct process eventually decides on some value

(2) Validity: If a process decides v, then v was proposed by some process

(3) Integrity: No Process decides twice

(4) Uniform agreement: No two processes decide differently

Exercise 4

Can we devise a uniform reliable broadcast algorithm with an eventually perfect failure detector but without assuming a majority of correct processes?

Exercise 5a

Suppose an algorithm A *implements a distributed programming abstraction* M *using a failure detector* D *that is assumed to be eventually perfect. Can* A *violate a safety property of* M *if* D *is not eventually perfect, for example, when* D *permanently outputs the empty set?*

Exercise 5b

Suppose an algorithm A *implements a distributed programming abstraction* M *using a failure detector* D *that is assumed to be eventually perfect. Can* A *violate a safety property of* M *if* D *is not eventually perfect, for example, when* D *permanently outputs the empty set?*

Now, what about a liveness property?