Software Verification and Validation (VIMMD052)

Exam Topics in 2020

1.	• The notion of verification and validation. Overview of the typical verification and validation activities during software development. • Efficient verification of complex systems by symbolic model checking.	• L01 • L08
2.	• Basic formalisms for modelling behaviour: Kripke Structure, Kripke Transition System, Labelled Transition System, Timed Automata. • Formal relations for refinement checking: “may preorder” and “must preorder”, their relationship with testing.	• L04 • L19
3.	• Verification of the software requirement specification: criteria and techniques. • Verification of invariant properties by bounded model checking.	• L02 • L08
4.	• Verification of the software architecture design: criteria and techniques. • Formalization and checking of requirements using HML and linear temporal logics (LTL).	• L03 • L05, L06
5.	• Verification of the detailed design: criteria and techniques. Categorization of the typical techniques of formal verification. • Model based test case generation by model checking and bounded model checking.	• L04 • L18
6.	• The role of development standards in the verification and validation of critical systems. • Software model checking: The counterexample guided abstraction refinement (CEGAR) approach with predicate abstraction.	• L02 • L15
7.	• Verification of program source code: criteria and techniques. • Model checking of time dependent behaviour: basic modelling formalism (timed automata) and timed temporal logic.	• L13 • L10
8.	• Specification based testing of software modules: test design techniques. • Correctness criteria and basic strategies for proving program correctness.	• L16 • L14, L15

9.	• Structure based testing of software modules: test coverage criteria. • Formal relations for checking behavioural equivalence: Strong bisimulation and weak bisimulation (observational equivalence).	• L16 • L12
10.	• Model based test case generation techniques: graph based algorithms. • Model checking of stochastic properties: basic modelling formalism and temporal logic (Continuous Stochastic Logic).	• L18 • L11
11.	• Software integration testing techniques. • Formalization and checking of requirements using branching time temporal logics (CTL* and CTL).	• L17 • L07
12.	• Verification during software maintenance: criteria and techniques. • Source code based test input generation by symbolic execution.	• L20 • L17