Call for a Teaching Assistant

**Course title**: Foundation of Model Checking

**Instructor(s)**: Nicolas A. Mazzocchi

**Track(s):** Computer Science

**Semester:** Spring 1

**Course type:** Taught course

**Course description/course goals:**

Reactive systems pervaded our daily life, and in particular safety-critical applications. Yet, such systems exhibit characteristics (such as concurrency, resource limitations) making them notoriously difficult to be implemented correctly. Hence, to ensure a dependable design, researchers and industrials have advocated the use of so-called formal methods, that rely on mathematical models to express precisely and analyze the behaviors of reactive systems. Model Checking is one of the most popular verification techniques with successful applications (as a matter of fact three contributions have been acknowledged with a ACM Turing award). Intuitively, the approach consists to determine whether all behaviors feasible by the system (expressed in a trace-formalism suitable for algorithmic treatment) satisfy a specification (expressed in a logical language suitable for defining trace properties). In contrast to run-time monitoring or theorem-prover assistants, Model Checking is both exhaustive and automatic.

The content of this course is based on the book “Principles of Model Checking” by C. Baier and J.-P. Katoen. At the end of the course, the students should be able to design a classical model checker with efficient implementation techniques (bisimulation, antichains, complementation) and prove its correctness. They should appreciate the expressivity of the traditional logics of the field (LTL, CTL, PCTL) and understand related publications (such as quantitative, symbolic and timed extensions of Model Checking).

**Target audience:**

Students interested in formal methods are exactly the audience. In particular, this lecture contains notions that members of Chattergee’s and Henzinger’s groups are expected to be aware of.

Remark, Chatterjee’s lectures on Formal Methods will provide additional perspectives on algorithms and formalisms presented in this course and vice-versa. Yet both courses are totally independent.

**Prerequisites:** None

**Teaching format(s):** Lectures

**Evaluation:** Final Exam