Colloquium de Shriram Krishnamurthi

4 avril 2018
http://www.lip6.fr/colloquium/

Programme

Master classes de 14:00 à 16:00 (Salle 24-25/405)

14:00–14:30 Sreeja Nair (DELYS)
   Designing geo-distributed applications with minimal coordination
14:30–15:00 Abdelraouf Ouadjaout (APR)
   Static value analysis of Python programs by abstract interpretation
15:00–15:30 Redha Gouicem (WHISPER)
   Process scheduling in modern multicore systems
15:30–16:00 Lucas Serrano (WHISPER)
   Automatic inference of software transformation rules for automatically porting legacy infrastructure software

Cocktail à 17:15 (à côté de l’Amphi 45A)

Colloquium de Shriram Krishnamurthi à 18:00 (Amphi 45A)
Curriculum design as an engineering problem
Master classes – Résumés

- **Designing geo-distributed applications with minimal coordination**
  Sreeja Nair (équipe DELYS)

  We present tools for helping the developer choose the just right amount of consistency required for preserving the invariants of a geo-distributed application. The tools take a sequential specification of an application and suggests synchronisation tokens which are needed to preserve the application invariant in a geo-replicated setting. One tool can be used for applications using operation-based update propagation and the other for applications using state-based updated propagation. Using these tools, we are planning to co-design a tunable and highly available geo-replicated file system.

  - Alexey Gotsman, Hongseok Yang, Carla Ferreira, Mahsa Najafzadeh, and Marc Shapiro. 2016. 'Cause I'm strong enough: Reasoning about consistency choices in distributed systems. SIGPLAN Not. 51, 1 (January 2016), 371-384. DOI: https://doi.org/10.1145/2914770.2837625

- **Static value analysis of Python programs by abstract interpretation**
  Abdelraouf Ouadjaout (équipe APR)

  In this talk, I present a static analyzer of Python programs based on the theory of abstract interpretation. It can soundly ensure complete coverage of all possible executions in finite time. One fundamental feature of Python programs is dynamic typing, which is handled by non-relational value abstraction that maintains a set of typed abstract values (such as intervals for integers and powersets for strings) for each variable and ensures sound type coercion among these values dynamically. In addition, the analyzer features a relational numeric domain to infer linear invariants among numeric variables, and supports complex non-local control flows such as exceptions and generators. Finally, I discuss some analysis experiments performed on real Python programs.

- **Process scheduling in modern multicore systems**
  Redha Gouicem (équipe WHISPER)

  Process scheduling is a key component of modern operating systems to achieve the best possible application performance. Due to the various application behaviors, the scheduler is a complex piece of software. In Linux, the Completely Fair Scheduler tries to be a generic
scheduler that handles well all application behaviors. As of last year, CFS’s codebase has reached more than 20k lines of code, which makes it difficult to understand and to keep it bug free. In recent years, multiple bugs were discovered in CFS, as well as other OS schedulers like FreeBSD. We present Ipanema, a domain-specific language designed to write safe scheduling policies, with assisted proving of scheduling properties.

- **Automatic inference of software transformation rules for automatically porting legacy infrastructure software**

Lucas Serrano (équipe WHISPER)

Large, real-world software must continually change, to keep up with evolving requirements, fix bugs, and improve performance, maintainability, and security. For example, a new release of the Linux kernel appears every 2-3 months, with each release resulting from 10-13K patches. This rate of change can pose difficulties for clients, whose code cannot always evolve at the same rate. Here we present approaches for solving this problem automatically by inferring transformation rules from a collection of relevant change examples.