“Concurrency in the 21st Century: It’s Weak!"

Gustavo Petri. IRIF -Paris Diderot - Paris 7. Francia.

Abstract:

Classical models of concurrent programming assume that in a multi-process program, the operations of each individual process execute one at a time, and the behavior of the overall program comprises all possible interleavings of the operations of different processes.

However, the inception of multi-core machines at the turn of the century evidenced that this model of concurrency was clearly insufficient. Hardware and software optimization artifacts such as caches, store buffers and out-of-order-execution (to name but a few), mean that the classical “interleaving semantics" is clearly inadequate to capture the behaviors of a concurrent program under a multicore machine. To explain these behaviors new semantical models had to be created. These are called relaxed (or weak) memory models, and are the subject of this course.

In this short course I will introduce relaxed memory models: why they are needed, what are their consequences in programming, and what tools can we bare to understand and operate with them. Whenever possible I will show hands-on experiments to illustrate the problems they induce. Hence, the focus of the course is on the semantics, the programming, and the verification of programs under relaxed memory models.

Another area of increasing relevance where the same kind of problems arise is distributed computing, where data objects and data bases that are replicated across the globe can suffer from inconsistencies due to network failures, message delays, etc. At the end of this course I will show some connexions between relaxed memory models, and weak consistency properties for distributed and replicated objects.

Programa:

1. Relaxed Memory Models: a Premier
   - 1.1. Classical Semantics of Concurrency
   - 1.2. Simple Hardware Models:
     - Preliminary Definitions
     - Anomalies
     - Semantics (axiomatic, operational)
     - Demo
   - 1.3. Realistic Hardware Models:
     - Modelling
     - Semantics
     - Testing
     - Demo
2. From Hardware to Software:
   - 2.1. Concurrency Semantics of High-level Programming Languages
     - Compiler Optimizations
     - Barriers and Locks

- The cases of Java and C11
- Demo

2.2. Taming Relaxed Memory Models:

- The Data Race Freedom Guarantee

2.3. Verification under Relaxed Memory Models:

- Complexity
- Program Logics
- Open Problems

3. From Software to the Cloud:
   3.1. Consistency for Replicated Objects and Data Bases
   3.2. Semantics of Weakly Consistent Replicated Data Objects
   3.3. Some approaches to the Verification of Distributed Replicated Objects