**Unit Testing for Concurrent Programs**

### Schedule-Based Unit Testing
- Execution using a fixed schedule
  - Deterministic and machine-independent execution
- Execution using all different schedules
  - Success of unit tests implies correct concurrent behavior
- Unit testing not performance critical
  - Overhead more acceptable in unit testing than in acceptance testing

### Schedule Suite Validation
- Schedules depend on code base
  - May need to be regenerated if code changes
- Detect whether a code change affects synchronization behavior
  - Only regenerates schedules if necessary
- Detect whether a code change affects a unit test
  - Only execute unit test if necessary

### Testing Distributed Programs
- Extend framework to distributed Java programs using RMC (Remote Method Invocation)
- Data cannot be shared except using RMI operations
  - Treat RMI operations as additional synchronization points
- Challenges:
  - Make all machines observe the dependencies in the replayed schedule

### Schedule Generation
- Atomic blocks
  - Defined by synchronization points (or lock on the open right)
  - Instructions in an atomic block cannot directly affect another thread if shared variables are guarded
- Generation Technique 1: Random delays
  - Delays inserted between atomic blocks to change scheduling behavior
  - Large number of schedules modifies this way should ensure good coverage
- Generation Technique 2: Exhaustive search
  - Enumeration of all arrangements of atomic blocks covers all possible program behaviors
  - Fewer arrangements of atomic blocks than of instructions

### Monitored Replay
- Custom class loader
  - Analyzes and modifies class files before they are loaded by the Java VM
- Modified class files will run on any Java VM
  - Portable and open-source
- Bytestreams can be dynamically compiled by an embedded JIT compiler
  - Faster than interpretation

### Detection of Unguarded Variable Access
- Unguarded access to shared variables leads to non-deterministic behavior
  - Schedule generation relies on guarded variable access
- Custom class loader dynamically rewrites bytecode to monitor access to all shared variables

### Schedule Suite Validation
- Schedules depend on code base
  - May need to be regenerated if code changes
- Detect whether a code change affects synchronization behavior
  - Only regenerates schedules if necessary
- Detect whether a code change affects a unit test
  - Only execute unit test if necessary

### Schedule Suite Validation
- Our Experience with DrJava
  - 60% of code paths are tested
  - 30% of code lines are unit tests

### Schedule Suite Validation
- Detection of Unguarded Variable Access
  - Eraser algorithm detects if each variable
  - Each variable should be guarded by at least one lock during all accesses
  - Systematic Schedule Suite Validation
    - Enumerates all possible arrangements of atomic blocks
    - Faster than interpretation

### Schedule Suite Validation
- Schedules depend on code base
  - May need to be regenerated if code changes
- Detect whether a code change affects synchronization behavior
  - Only regenerates schedules if necessary
- Detect whether a code change affects a unit test
  - Only execute unit test if necessary

### Schedule Suite Validation
- Schedules depend on code base
  - May need to be regenerated if code changes
- Detect whether a code change affects synchronization behavior
  - Only regenerates schedules if necessary
- Detect whether a code change affects a unit test
  - Only execute unit test if necessary

### Schedule Suite Validation
- Schedules depend on code base
  - May need to be regenerated if code changes
- Detect whether a code change affects synchronization behavior
  - Only regenerates schedules if necessary
- Detect whether a code change affects a unit test
  - Only execute unit test if necessary

### Schedule Suite Validation
- Schedules depend on code base
  - May need to be regenerated if code changes
- Detect whether a code change affects synchronization behavior
  - Only regenerates schedules if necessary
- Detect whether a code change affects a unit test
  - Only execute unit test if necessary