Detecting Concurrency Errors with JPF

http://d3s.mff.cuni.cz

Pavel Parízek
Questions about JPF ?
Configurations

- Default JPF: exhaustive search (DFS)
  - Threads scheduled in the order given by their IDs

- Random search order
  $+cg.randomize_choices=VAR\_SEED$

- Preemption bounding
  $+vm.scheduler.sync.class=gov.nasa.jpf.vm.schedule.ContextBoundingSyncPolicy$
  $+vm.scheduler.sharedness.class=gov.nasa.jpf.vm.schedule.ContextBoundingSharednessPolicy$
  $+contextbound.max\_number\_of\_preemptions=<N>$
Breadth first search (BFS)
+search.class=gov.nasa.jpf.search.heuristic.BFSHeuristic
+search.heuristic.queue_limit=-1

Maximize thread preemption
+search.class=gov.nasa.jpf.search.heuristic.Interleaving

Minimize preemption
+search.class=gov.nasa.jpf.search.heuristic.MinimizePreemption

Maximize blocked threads
+search.class=gov.nasa.jpf.search.heuristic.MostBlocked
What to do now

- Finish remaining tasks from the last seminar
  - Writing reasonable environment for LinkedList and Semaphore (try different workloads)

- Play with different configurations aiming at efficient detection of concurrency errors
  - Use additional benchmark programs (examples)
    - [http://d3s.mff.cuni.cz/teaching/program_analysis_verification/files/concur_bench.zip](http://d3s.mff.cuni.cz/teaching/program_analysis_verification/files/concur_bench.zip)