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
  
  \[ +vm.scheduler_factory.class=gov.nasa.jpf.vm.ContextBoundingSchedulerFactory \]
  \[ +cg.max_number_of_preemptions=<N> \]
Configurations

- **Breadth first search (BFS)**
  
  ```java
  +search.class=gov.nasa.jpf.search.
  .heuristic.BFSHeuristic
  +search.heuristic.queue_limit=-1
  ```

- **Maximize thread preemption**
  
  ```java
  +search.class=gov.nasa.jpf.search.
  .heuristic.Interleaving
  ```

- **Minimize preemption**
  
  ```java
  +search.class=gov.nasa.jpf.search.
  .heuristic.MinimizePreemption
  ```

- **Maximize blocked threads**
  
  ```java
  +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)