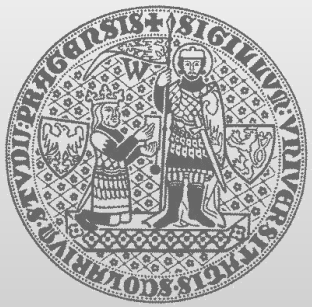
T.J. Watson Libraries for Analysis (WALA)

http://d3s.mff.cuni.cz

Department of Distributed and Dependable Systems

Pavel Parízek



CHARLES UNIVERSITY IN PRAGUE

faculty of mathematics and physics

WALA

- Static analysis library
  - Open source (SVN, Git, Wiki)
- Languages
  - Java (source, bytecode), JavaScript
- Home page
  - http://wala.sourceforge.net/
  - https://wala.github.io/

Department of Distributed and Dependable Systems D3S

How to get WALA

- Download and unpack somewhere
  - http://d3s.mff.cuni.cz/teaching/program_analysis_verification/files/wala-examples.zip
- Content
  - Subdirectory `wala`
    - binary (jar file), source code, configuration, exclusion file (ignored libraries)
  - Subdirectory `src`
    - two example analyses, two simple programs
  - Ant build script (`build.xml`)

Department of Distributed and Dependable Systems D3S

Example: static analyses

- Directory `src/analysis`
  - Merge operator: bit vector intersection
  - Analysis facts: `FieldID`, `ProgramPoint`
  - Several utility methods: `WALAUtils`

- Two static analyses
  - Locked local variables
  - Future field accesses

Department of Distributed and Dependable Systems D3S

SSA IR

- SSA: static single assignment form
- IR: intermediate representation

- Characteristics
  - Exactly one assignment to each local variable
  - Multiple assignments in source code → multiple incarnations in the SSA IR
  - Unique value numbers for variables (expressions)
  - Value numbers: `this`, parameters, local variables

Department of Distributed and Dependable Systems D3S

Example: locked local variables

- Intra-procedural forward analysis
  - For each method reachable in the call graph
- Flow-sensitive context-insensitive
- Analysis facts: SSA value numbers
- Merge operator: set intersection

- Relevant instructions (SSA)
  - Monitor enter
  - Monitor exit

Department of Distributed and Dependable Systems D3S

Task 1

- Run the *locked variables* analysis
  - Ant: `run.example.lockedvars`
  - Input program: `example/SeparateMethods`
- Fix your local configuration
  - File `wala/config/wala.properties`
    - Set path to JRE/JDK 8 (variable `java_runtime_dir`)
- Look into the source code of the analysis
- Check source code of the target program
- Check the static analysis output (results)

Department of Distributed and Dependable Systems D3S

Inter-procedural CFG

- Caller method
  - Nodes: call, return
- Callee method
  - Nodes: entry, exit

- ICFG edges
  - call → entry
  - exit → return
  - call → return

Example: future field accesses

- Inter-procedural backward analysis
  - Over the whole program inter-procedural CFG
- Flow-sensitive context-insensitive
- Merge operator: set union (“may”)

- Bytecode instructions
  - Field access (read, write)

- Running
  - Ant: `run.example.fieldaccess`
  - Input: `example.WholeProgram`

Department of Distributed and Dependable Systems D3S

Documentation

- Tutorial
  - http://wala.sourceforge.net/wiki/index.php/Tutorials
- User guide
  - WALA core technical overview
  - http://wala.sourceforge.net/wiki/index.php/Wala.core_technical_overview
- API docs
  - List of SSA instructions (`com.ibm.wala.ssa`)
  - http://wala.sourceforge.net/javadocs/trunk/

Department of Distributed and Dependable Systems

Task 2

- Implement static analysis
  - Live variables (values)
  - Available expressions
  - Your own ideas welcome
- Ask questions about WALA