

T.J. Watson Libraries for Analysis (WALA)

<http://d3s.mff.cuni.cz>

Department of
Distributed and
Dependable
Systems



Pavel Parízek



FACULTY
OF MATHEMATICS
AND PHYSICS
Charles University

- Static analysis library
 - Open source (SVN, Git, Wiki)
- Languages
 - Java (source, bytecode), JavaScript
- Home page
 - <https://wala.github.io/>
 - <https://github.com/wala/WALA/> (wiki)

How to get WALA

- Download and unpack somewhere
 - <http://d3s.mff.cuni.cz/files/teaching/nswi132/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`)

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

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
- Literature
 - https://en.wikipedia.org/wiki/Static_single-assignment_form

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

Inter-procedural CFG

- Caller method
 - Nodes: call, return
- Callee method
 - Nodes: entry, exit
- ICFG edges
 - call \rightarrow entry
 - exit \rightarrow return
 - call \rightarrow 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`

Documentation

- Tutorial
 - <https://github.com/wala/WALA>
 - Section “Getting Started”
- User guide
 - <https://github.com/wala/WALA/wiki>
 - WALA core technical overview
- API docs
 - List of SSA instructions (`com.ibm.wala.ssa`)
 - <https://wala.github.io/javadoc/>