A Precise Framework for Source-Level Control-Flow Analysis

Idriss Riouak, Christoph Reichenbach*, Görel Hedin*, and Niklas Fors**

*idriss.riouak, christoph.reichenbach, gorel.hedin, and niklas.fors (@cs.lth.se)

Department of Computer Science, Lund University, Sweden

DESCRIPTION

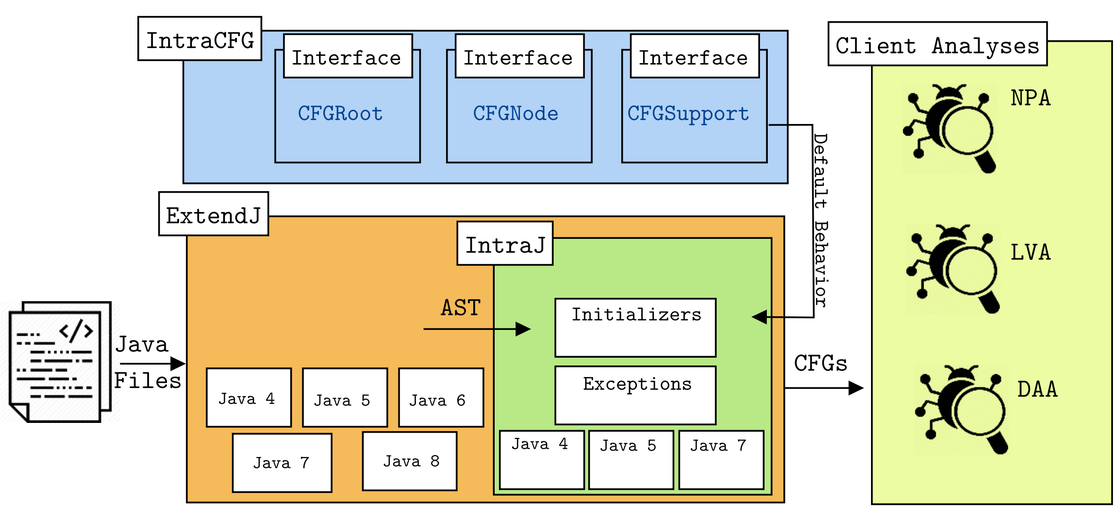
Static program analysis plays a fundamental role in software development and may help developers detect subtle bugs such as null pointer exceptions or security vulnerabilities. We present **IntraCFG**, a language-independent framework for constructing precise intraprocedural control-flow graphs (CFGs) superimposed on the Abstract Syntax Tree (AST). Source-level dataflow analysis permits easier integration with the IDEs and Cloud tools since the reports can be directly linked to the source code and do not require producing the Intermediate Representation.

OUR APPROACH

We build the CFGs on top of the AST using Reference Attribute Grammars (RAGs). Our approach consists on:

- Handling implicit control flow
- Fully declarative specification using JastAdd2
- Overcome the limitations of an earlier framework **JastAddJ–Intraflow,** which included *misplaced* and *redundant* nodes in the constructed CFGs.

FRAMEWORK



The language-independent framework **IntraCFG** defines three interfaces:

INTERFACE	ASTNODE
`CFGRoot`	`MethodDecl, ConstructorDecl, …`
`CFGSupport`	`WhileStmt, IfStmt, …`
`CFGNode`	All the `ASTNodes` that might appear in the CFGs.

These interfaces provide APIs like the successor and predecessor relations, used by client analyses with default behaviour. We implemented **IntraJ** as an extension of the ExtendJ Extensible Java Compiler for the Java7. **IntraJ** is an application of the **IntraCFG** framework.

EXPERIMENTS

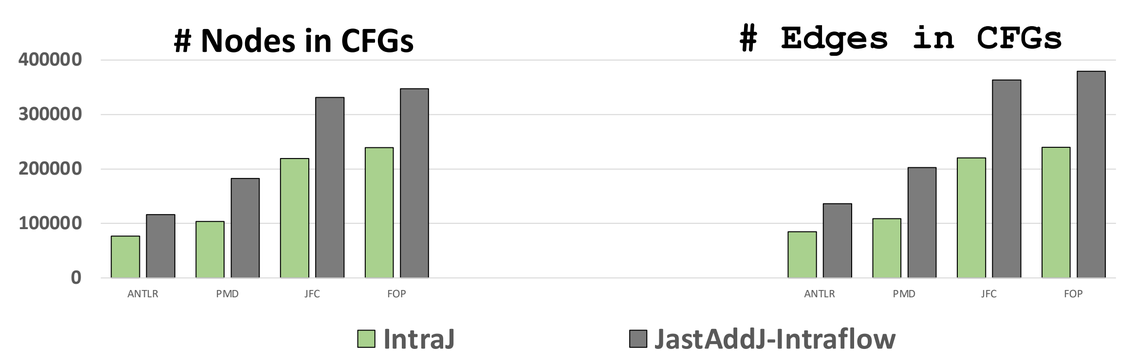
We compared the results of **IntraJ** with:

- **JastAddJ–Intraflow (JJI)**: a RAG based framework
- **SonarQube**: a highly tuned static analyser

We used as benchmarks:

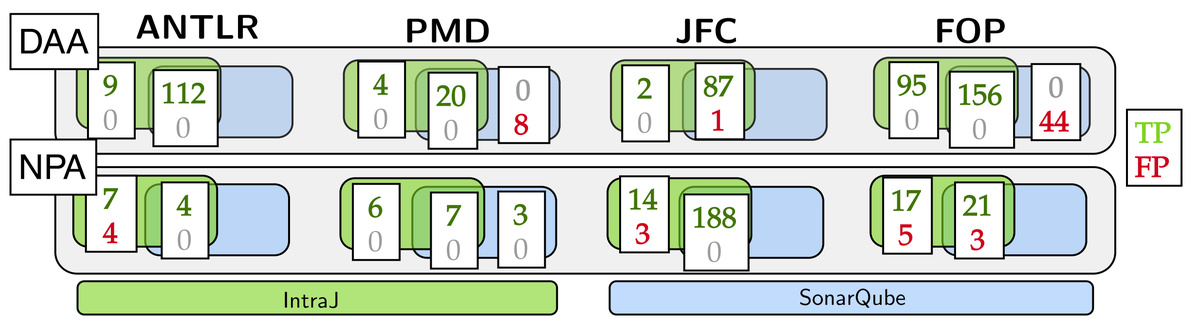
33K LOC | 49K LOC | 95K LOC | 97K LOC

IntraJ w.r.t. **JJI** reduces the CFGs size by 30-40%

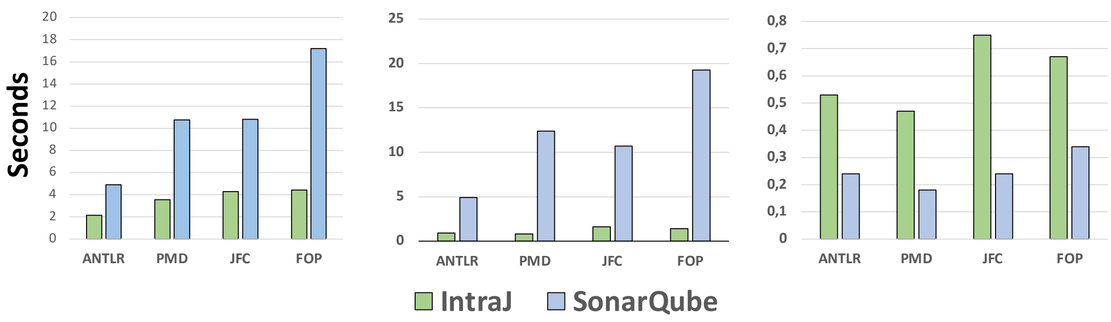


We compared the precision and the performance of **IntraJ** against **SonarQube** by implementing two dataflow analyses:

- Dead Assignment Analysis *[DAA]*
- Null Pointer Analysis *[NPA]*



Baseline | **Null Pointer Analysis** | **Dead Assignment Analysis**



CONCLUSIONS & FUTURE WORKS

IntraCFG is a language-independent RAGs framework that overcomes the limitation of the earlier approaches:

- High-Precision
- ≥30% fewer nodes
- Concise specification of **IntraJ**
- Competitive to **SonarQube**

We plan to:

- extend the support of **IntraJ** to Java 8
- extend **IntraCFG** to construct inter-procedural CFGs

WASP PhD project with relation to the ELLIIT project Cloud Tooling for Large-Scale Cyber-Physical System Model-Based Development