ELLIIT

LINKÖPING UNIVERSITY · LUNDS UNIVERSITET · BTH · HÖGSKOLAN I HALMSTAD

Excellence Center at Linköping – Lund in Information Technology

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

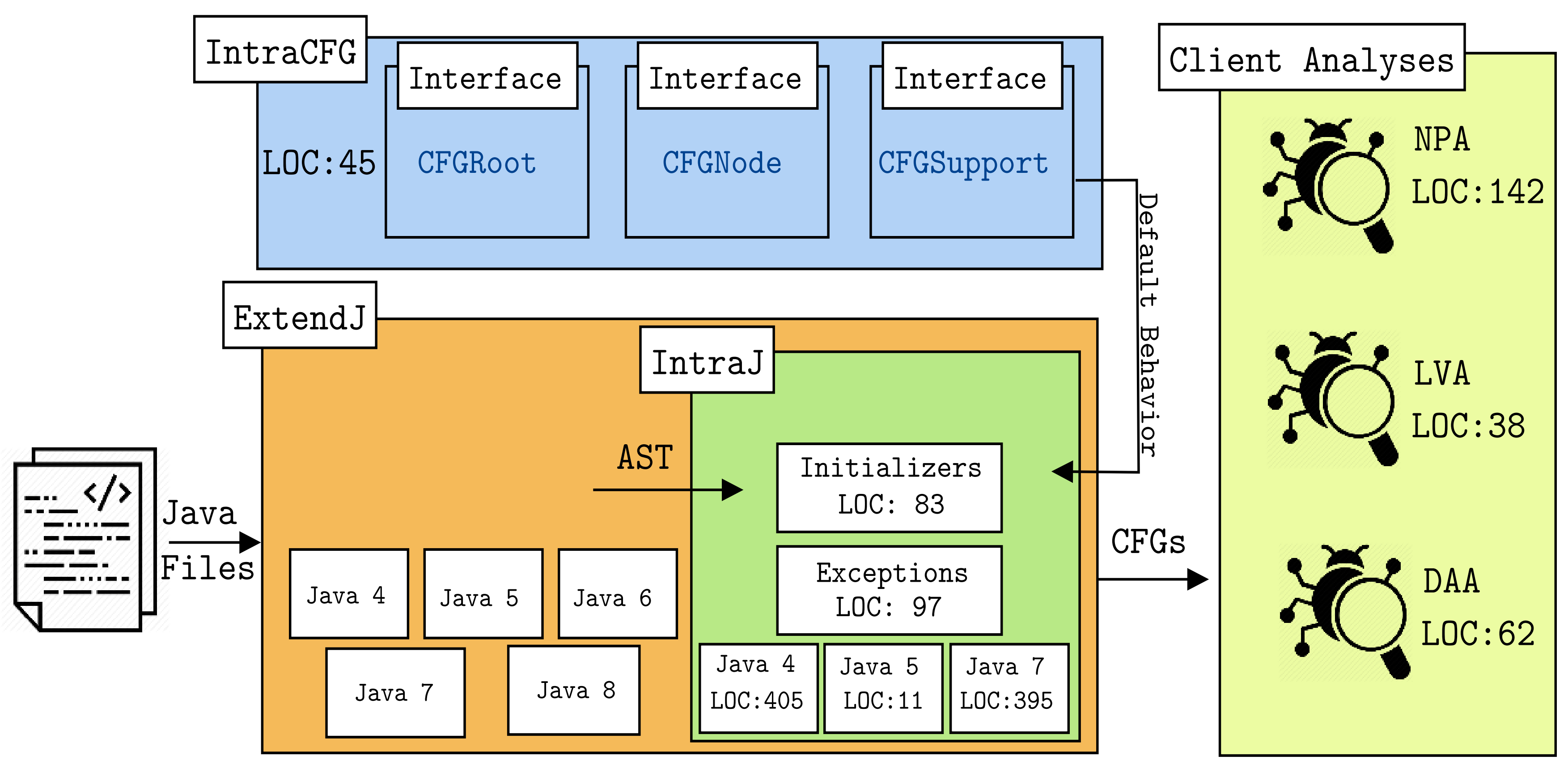
Static program analysis plays a fundamental role in software development and may help developers detect subtle bugs such as null pointer exceptions or security vulnerabilities. We present **IntraCFG**, a language-independent framework for constructing precise intraprocedural control-flow graphs (CFGs) 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 (IR).

OUR APPROACH

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

- Handles implicit control flow
- Fully declarative specification using JastAdd2
- Overcomes the limitations of an earlier RAG framework, eliminating *misplaced* and *redundant* nodes in the constructed CFGs.

FRAMEWORK



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

The **IntraCFG** interfaces provide client APIs for the successor and predecessor relations, and default behaviour that simplifies constructing CFGs for a specific language. We used **IntraCFG** to construct high-precision CFGs for Java 7, extending the ExtendJ Java compiler.

EXPERIMENTS

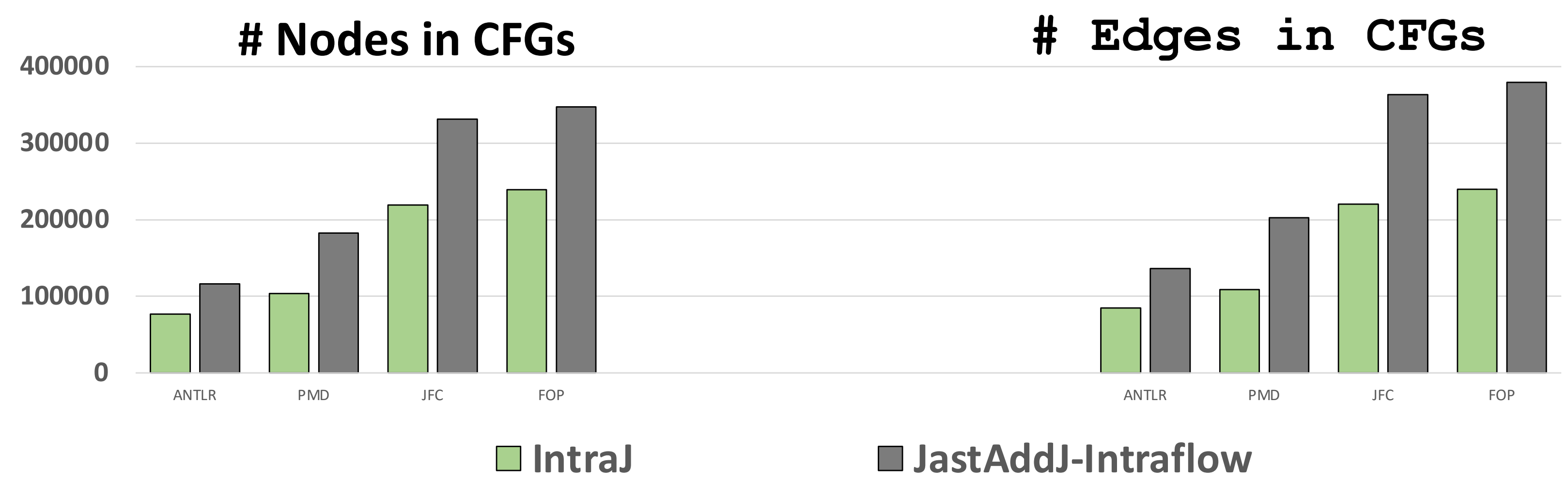
We compared the results of **IntraJ** with:

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

We used as benchmarks:

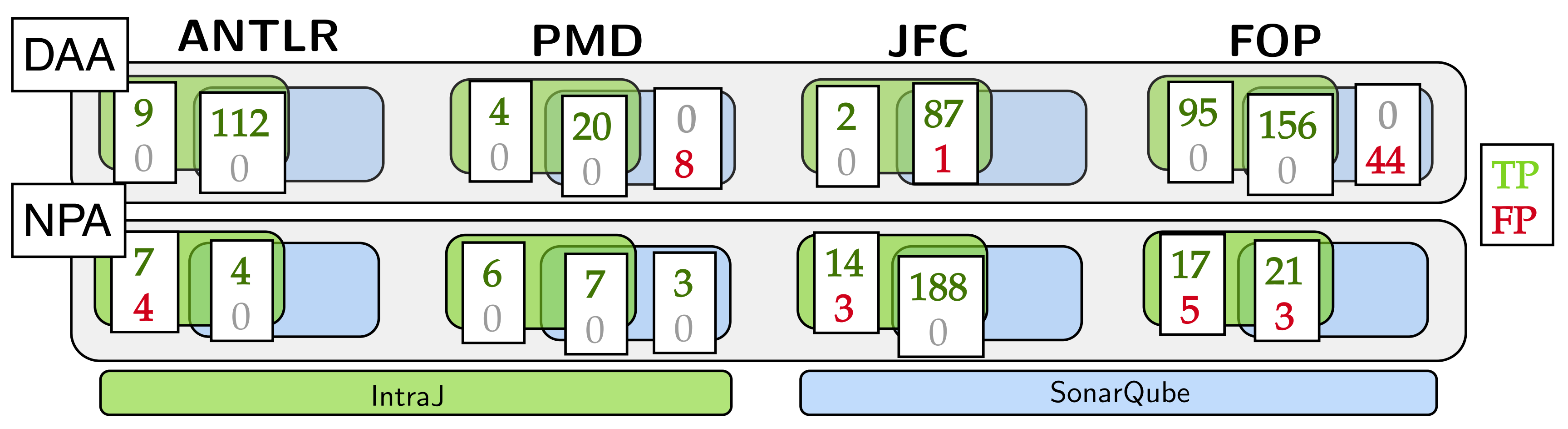
ANTLR	PMD	JFreeChart	FOP
33K LOC	49K LOC	95K LOC	97K LOC

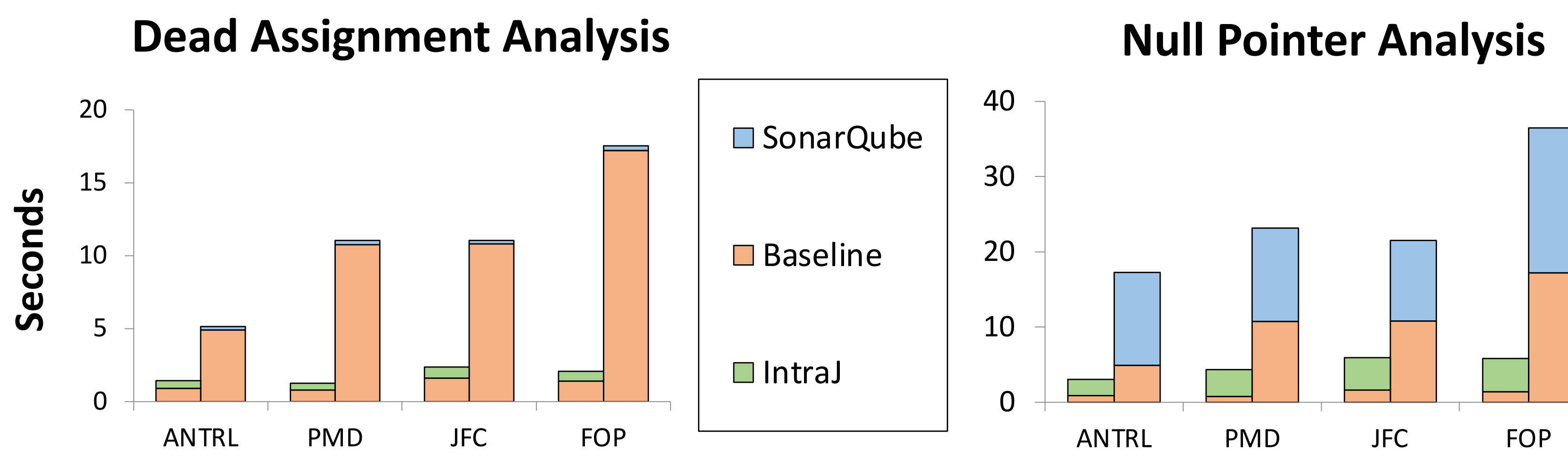
- CFG size reduced by 30-40%



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

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





- Higher precision and better overall performance

CONCLUSIONS & FUTURE WORK

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

- High-Precision
- ≥30% fewer nodes
- Concise CFG specification
- 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