VeriTest: Automatic Verification of Compiler Optimizations

VeriTest aims to provide analysis to GraalVM developers on their proposed optimizations and integrate it into their development workflow. This would make it easy for the developers to use VeriTest as they go, without being a "proof expert".

Achmad Afriza Wibawa
achmad.afriza@gmail.com

School of Electrical Engineering and Computer Science,
The University of Queensland

Supervisors

- Associate Professor Mark Utting
- Emeritus Professor Ian J. Hayes
- Brae J. Webb

01 Introduction

We introduce VeriTest: a software interface for automatic verification of GraalVM's [1] expression optimizations. VeriTest is inspired by the work done on LLVM by Alive [2].

The ideal workflow is:

1. GraalVM developers write a proposed optimization rule in Veriopt's Domain Specific Language (DSL) [3].
2. The rule would be passed into VeriTest to:
   a. Determine if it is verifiable — by leveraging Isabelle [4].
   b. If VeriTest can't determine, rules would be passed to "proof experts".

02 Isabelle

Isabelle is an interactive theorem prover.

It transforms the expressions into conjunctions that can be automatically reasoned with [4].

We focus on:

- Sledgehammer [4, Sec. 3] — to automatically verify rules.
- Nitpick [4, Sec. 4] and Quickcheck [4, Sec. 5] — to find counterexamples.

03 Veriopt's DSL

Veriopt's DSL [3] enables developers to express optimizations as rewriting rules in a familiar Java syntax within Isabelle.

```
optimization IncorrectRule: x + y ↦ x - y
```

- This optimization rule is false, we see that:

```
Let x = 1 and y = 2,
1 + 2 ≠ 1 - 2
```

- Finding counterexamples works in the same way.
- This works by determining if the transformation is satisfiable or not through Isabelle.

04 Methodology

VeriTest utilizes Isabelle Client-Server [6] **to interface** with Isabelle tools.

Isabelle Server acts as the core Isabelle process that allows theorems (i.e., Veriopt's theory base) to be loaded up and **processed in parallel** [6].

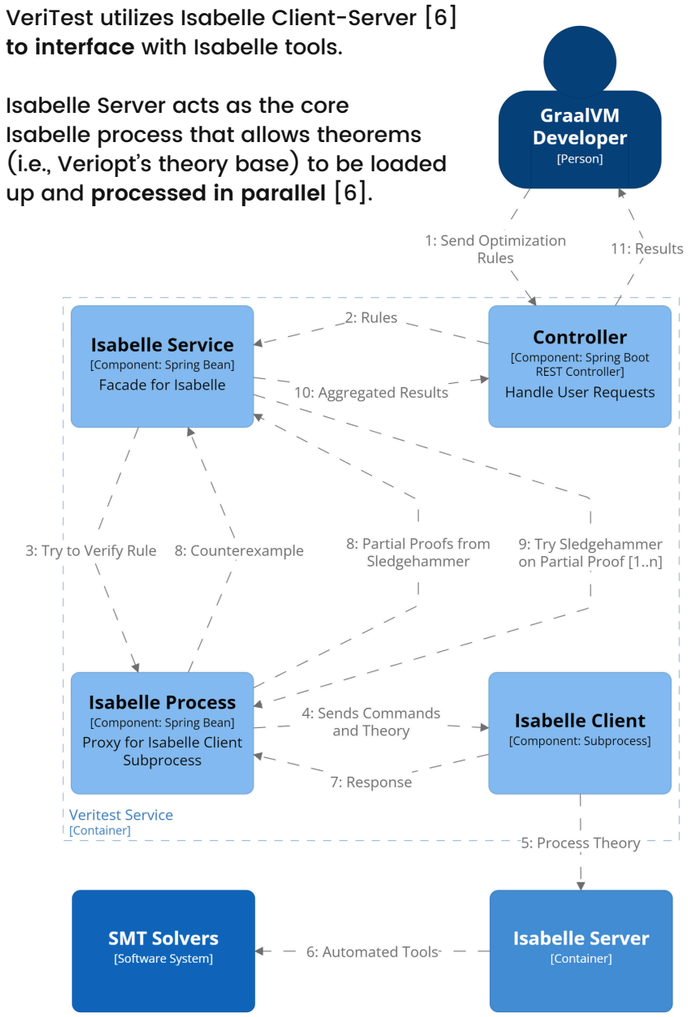

Figure 1. Sequence Diagram on Verifying Optimization Rules

05 Verifying InverseLeftSub

Request:

```
{
  "request_id": "IncorrectRule",
  "theory": "x + y ↦ x - y"
}
```

Response:

```
{
  "request_id": "IncorrectRule",
  "status": "FOUND_COUNTEREXAMPLE",
  "counterexample":
"Nitpick found a counterexample:
    Free variables:
     x::IRExpr = ConstantExpr V1
     y::IRExpr = ConstantExpr V2"
}
```

06 Results

Result	# Rules	Mean ± SD
Failed	59	87.06 ± 49.42
Found Auto Proof	7	37.73 ± 3.92
Found Proof	32	82.91 ± 29.43
Found Counterexample	1	40.79 ± 4.02
Malformed	13	37.96 ± 4.02
No Subgoal	2	37.93 ± 3.38

Figure 2. Evaluation of each existing Veriopt optimization rules based on runtime (in seconds)

07 Summary

- VeriTest yields a **comprehensive analysis** of rules by utilizing Isabelle.
- VeriTest provides fast feedback by **concurrently processing** rules.
- VeriTest demonstrates the capability of finding proofs by **integrating existing lemmas** inside Veriopt's theory base.
- The percentage of rules that can be automatically proven is expected to **improve** as the collection of theories grow.

References

[1] Oracle, "GraalVM: Run programs faster anywhere," 2020. [Online]. Available: https://github.com/oracle/graal
[2] J. Lee, C.-K. Hur, and N. P. Lopes, "AliveInLean: A verified LLVM peephole optimization verifier," in Computer Aided Verification, I. Dillig and S. Tasiran, Eds., 2019, pp. 445–455.
[3] B. J. Webb, I. J. Hayes, and M. Utting, "Verifying term graph optimizations using Isabelle/HOL," in Proceedings of the 12th ACM SIGPLAN International Conference on Certified Programs and Proofs, 2023, p. 320333.
[4] J. C. Blanchette, L. Bulwahn, and T. Nipkow, "Automatic Proof and Disproof in Isabelle/HOL," in Frontiers of Combining Systems, ser. LNCS, C. Tinelli and V. Sofronie-Stokkermans, Eds., 2011, vol. 6989, pp. 12–27.
[5] B. J. Webb, M. Utting, and I. J. Hayes, "A formal semantics of the GraalVM intermediate representation," in Automated Technology for Verification and Analysis, ser. LNCS, Z. Hou and V. Ganesh, Eds., vol. 12971, Oct. 2021, pp. 111–126.
[6] M. Wenzel, "The Isabelle System Manual." [Online]. Available: https://isabelle.in.tum.de/dist/Isabelle2023/doc/system.pdf