BenchPAL

Systematic Benchmarking Test Case Generation

Motivation

- Proof assistants are critical tools in formal verification used to ensure the correctness of mathematical theorems, software, and hardware systems.
- Lean, Idris, Agda, and Rocq are functional programming languages commonly used as proof assistants.
- Current user experience suggests that there are many low-level deficiencies.
- BenchPAL systematically tests these proof assistant languages (PALs) to highlight areas for improving their performance.

System Design

1. **CI Pipeline**
   Allows users to run the system utilizing Docker and Vercel.
2. **Test generator**
   Uses MHPG to dynamically generate scalable test modules based on size input.
3. **Translators**
   Translates the generated tests into each PAL.
4. **Tracking**
   Tests are type-checked, and both time and memory usage are recorded.
5. **Calculations**
   Startup times subtracted from runtimes, and log values are calculated if log interval selected.
6. **Visualization**
   Graphs are plotted using Matplotlib from JSON file and served in Flask backend.
7. **Future development**
   Test suite can be expanded by adding tests to Tests module (Tests.hs).

Structure of BenchPAL

1 → 2 → 3

1 — CI [GitHub Actions]

The user interacts with our project through a CI implemented in GitHub Actions or locally.

2 — Automated Code Generator

Automated code generator creates tests of increasing size and translates them to the four proof assistant languages.

3 — Webpage Generation

Test complexities visualized using Python's matplotlib in a Flask backend and hosted on Vercel to display results.

Example: Let Bindings Test Case

Agda

```
module LetExample where
open import Agda.Builtin.Nat
n : Nat
n = let
    x1 = 1
 in
    let
    x2 = x1
 in
    let
    x3 = x2
 in
    x3
```

Idris

```
module Main
n : Nat
n = let
    x1 = 1 in
    let
    x2 = x1 in
    let
    x3 = x2 in
    x3
main : IO()
main = putStrLn ""
```

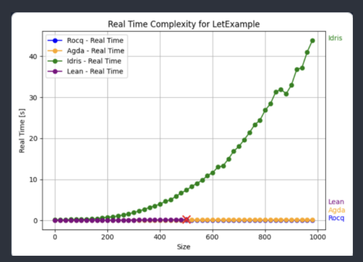
Lean

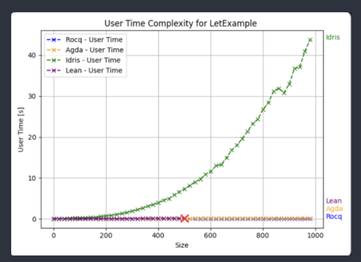
```
def n : Nat := let x1 := 1
let x2 := x1
let x3 := x2
x3
```

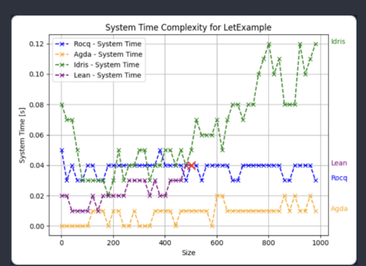
Rocq

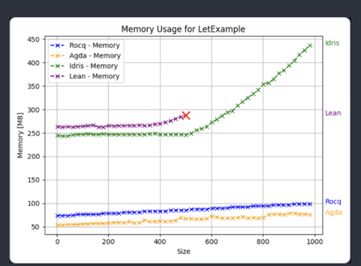
```
Module LetExample.
Definition n : nat := let x1 := 1 in
    let x2 := x1 in
    let x3 := x2 in
    x3.
End LetExample.
```

Results

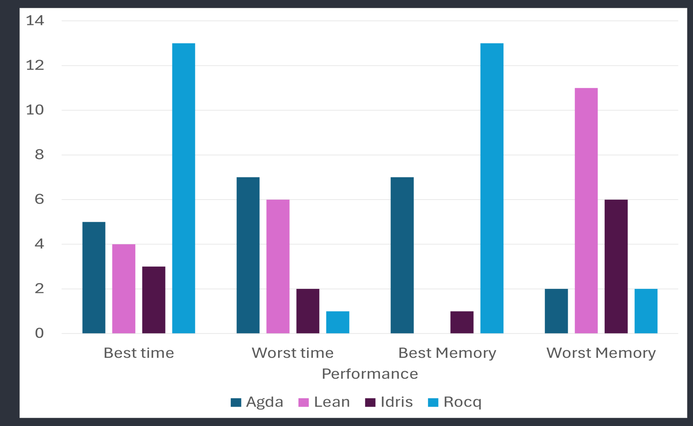








Conclusions

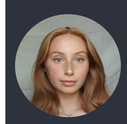


Best Time: Rocq
Best Memory: Rocq
Worst Time: Agda
Worst Memory: Lean

Technology Stack

Flask · Vercel

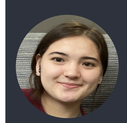
Team



Emma Willson

Esha Pisharody

Grace Croome



Marie Hollington

Proyetei Akanda

Zainab Abdulsada

Supervisors

Dr. Jacques Carette

Reed Mullanix

1011 011 01 1011001 10 11011 011 01 110110 110111 1101 1011 011 01 1011001 10 11011 011 01 110110 110111 1101 1011 011 01 1011001 10 11011 011 01 110110 110