Program Analysis and Compilers

Capstone Project Spring 2023
Justice Howley
Jason Weeks
Ian Yelle

AUGUSTA UNIVERSITY

INTRODUCTION

- Studying Quasi-Invariance
- Researching LLVM and Compilers
- Finding Code Examples and Checking Examples for QI
- Creating a Benchmark and Benchmarking Examples

QUASI-INVARIANCE

- What is Quasi-Invariance?
  - Why does it matter?
- How to remove it?
  - Loop Invariant Code Movement (LICM) for invariance of 1
  - Invariance > 1? Loop **Quasi Invariant** Code Movement (LQICM)

QUASI-INVARIANCE

- Peeling: Executing the body of a loop once (before the loop itself)
- Hoisting: Removing invariant code
- Idea to peel loop and hoist invariants until loop has no invariance/QI

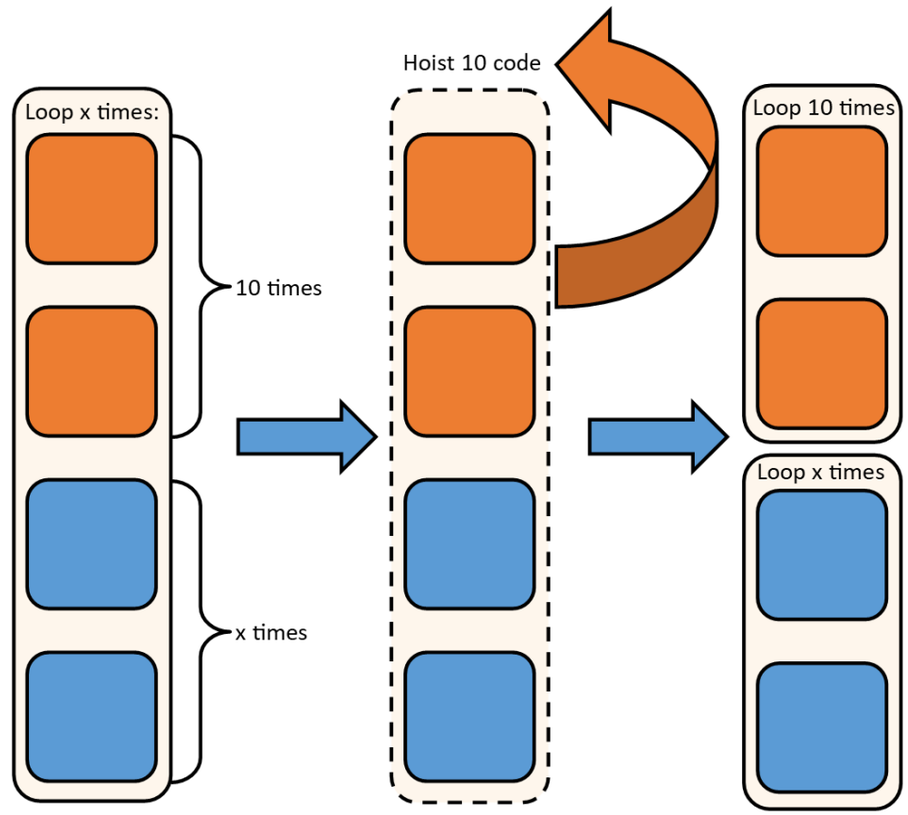


Fig. 1: Example of LQICM in action

EXAMPLES

- Found code examples in source code and GitHub Repositories
- Created our own examples based on the examples we find
- We determined if the examples had quasi-invariance given the following criteria:
  - Loops that has conditionals and reassignments
  - Loops within loops, especially those with above criteria

EXAMPLES

```c
#include <stdio.h>

int main(){ //VDG Example from Paper
    int j, s;
    int i = 0;
    int x = 42;
    int y = 5;
    int a = 12;

    while(i < 1000000){
        j = 0;
        s = 1;
        while(j<y){
            s = s * j;
            j = j + 1;
        }
        if(x>100) y = x + a;
        if (x<=100) y = x + 100;

        j = i - 1;
        a = 1;
        i = j + 2;
    }

    return 0;
}
```

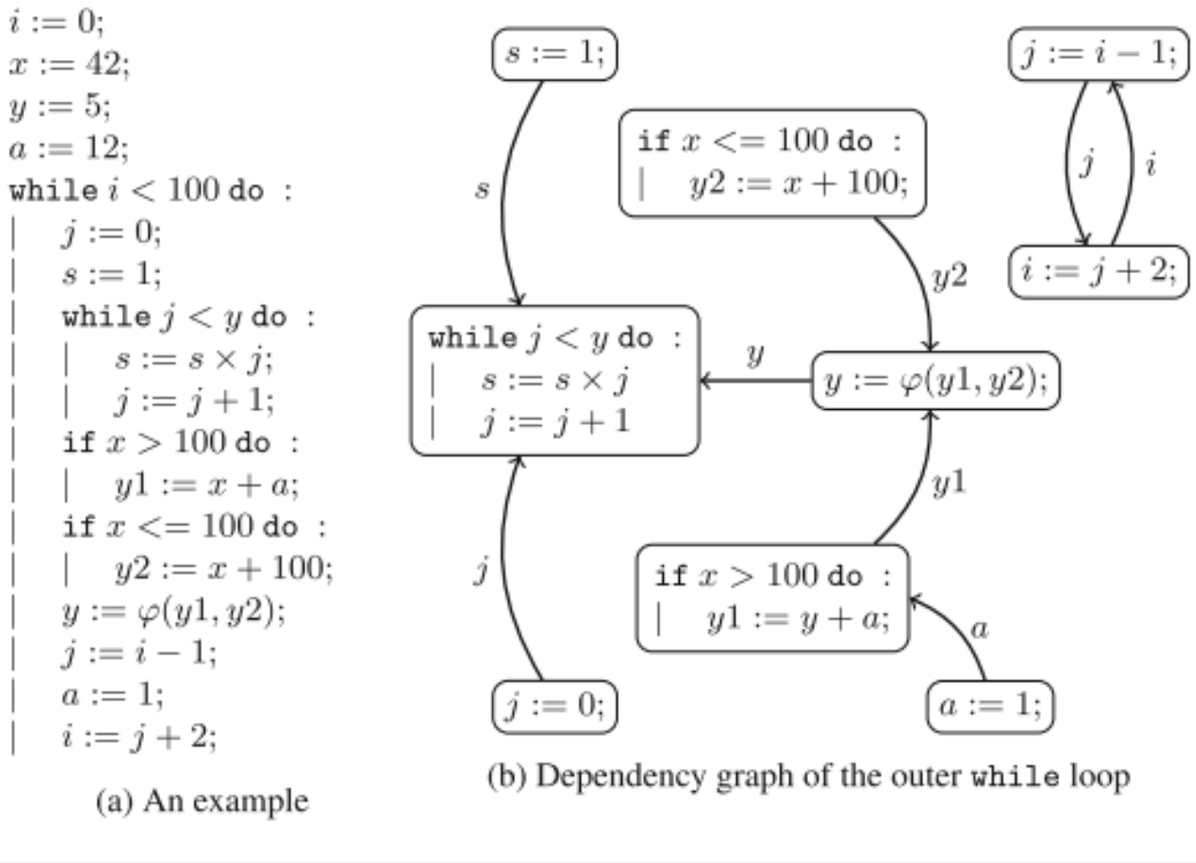
Fig. 2: VDGExample.c before peeling

```c
#include <stdio.h>

int main(){ //VDG Example from Paper
    int j, s;
    int i = 0;
    int x = 42;
    int y = 5;
    int a = 12;

    j = 0;
    s = 1;
    s = s * j;

    while(j<y){
        j = j + 1;
    }
    if(x>100) y = x + a;
    if (x<=100) y = x + 100;
    a = 1;
    while(i < 1000000){
        j = i - 1;
        i = j + 2;
    }

    return 0;
}
```

Fig. 3: VDGExample.c after peeling

DEPENDENCY GRAPHS

- With examples, we used the dependency graphs outlined within the paper to detect QI
- This technique was good for manual removal of QI, but had problems being automated



(a) An example

(b) Dependency graph of the outer `while` loop

Fig. 4: Dependency Graph from Paper

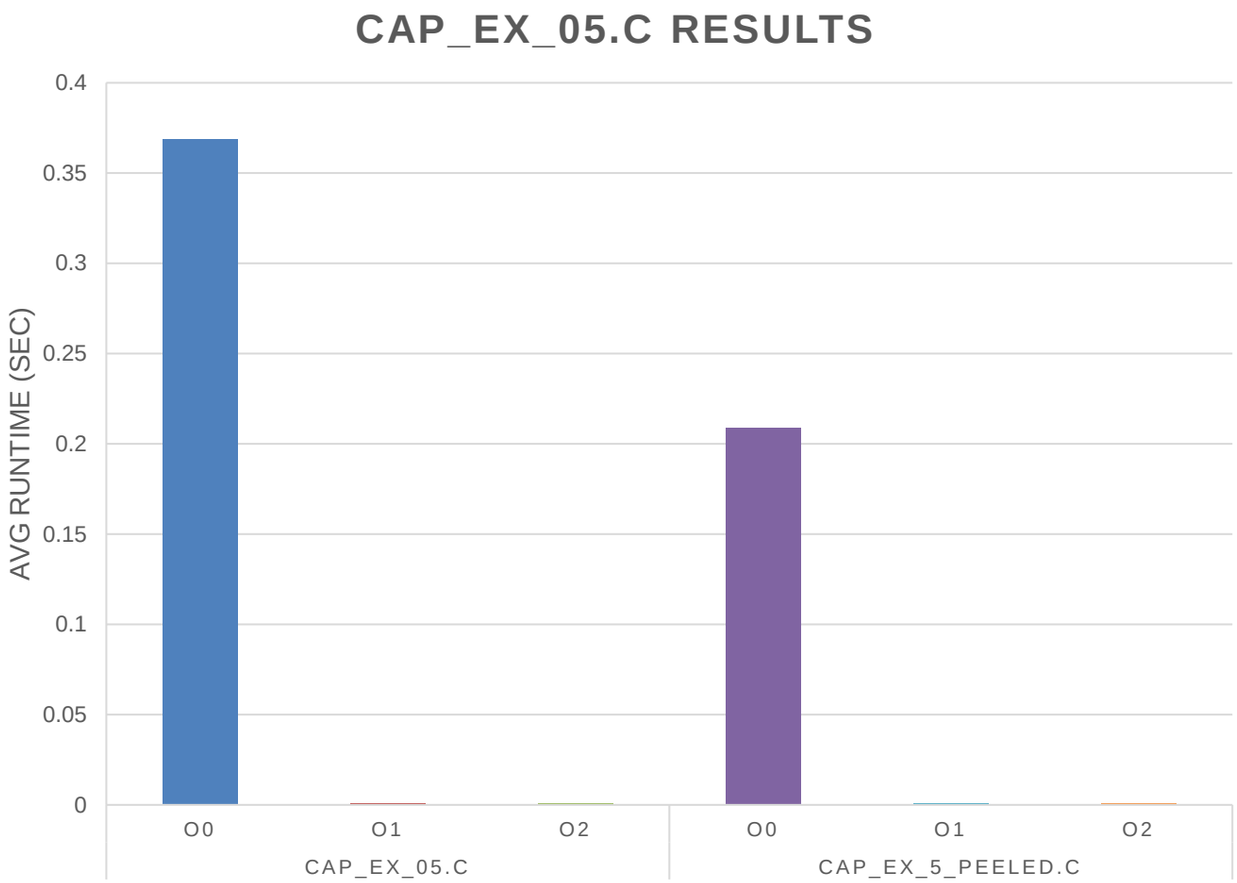
LLVM

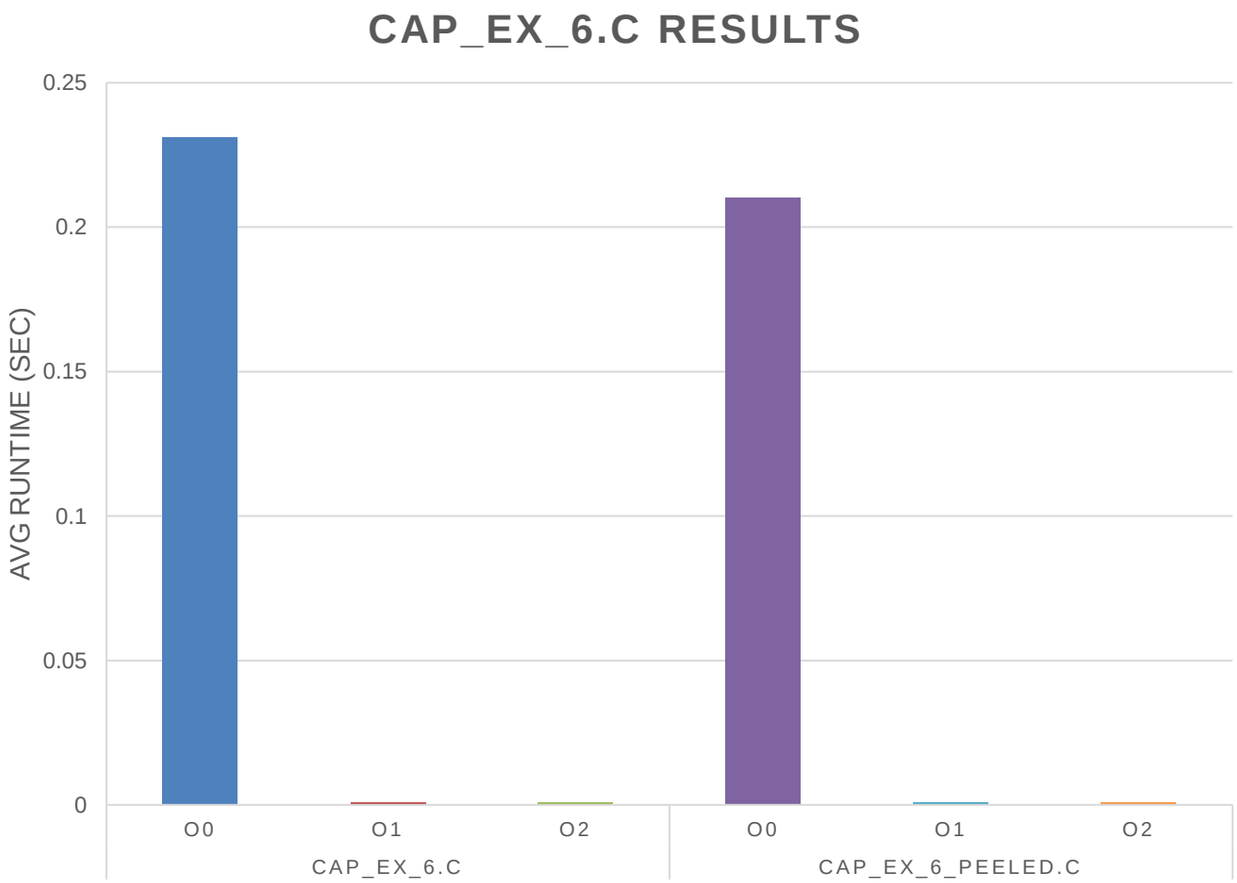
- Worked with LLVM to get familiar with its passes like LICM
- We want to eventually implement an LLVM pass that implements LQICM.
- We tried using the LLVM dependency graph tool to determine QI in our examples, but it proved harder than mapping the dependencies ourselves.

BENCHMARKING

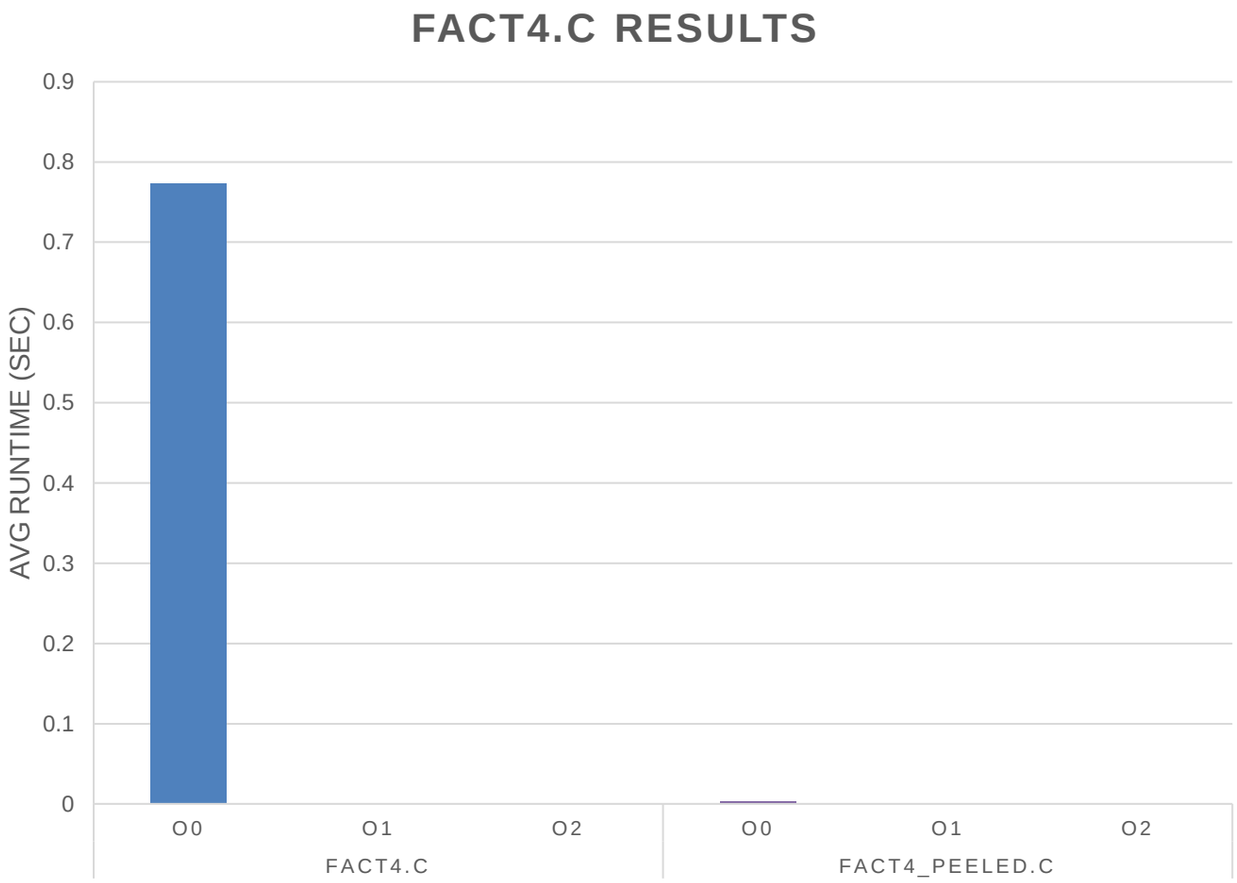
- Learning how to benchmark software
- Optimizations
- Ease-of-Use changes
- On GitHub (https://github.com/jweeks2023/LQICM-Benchmark)

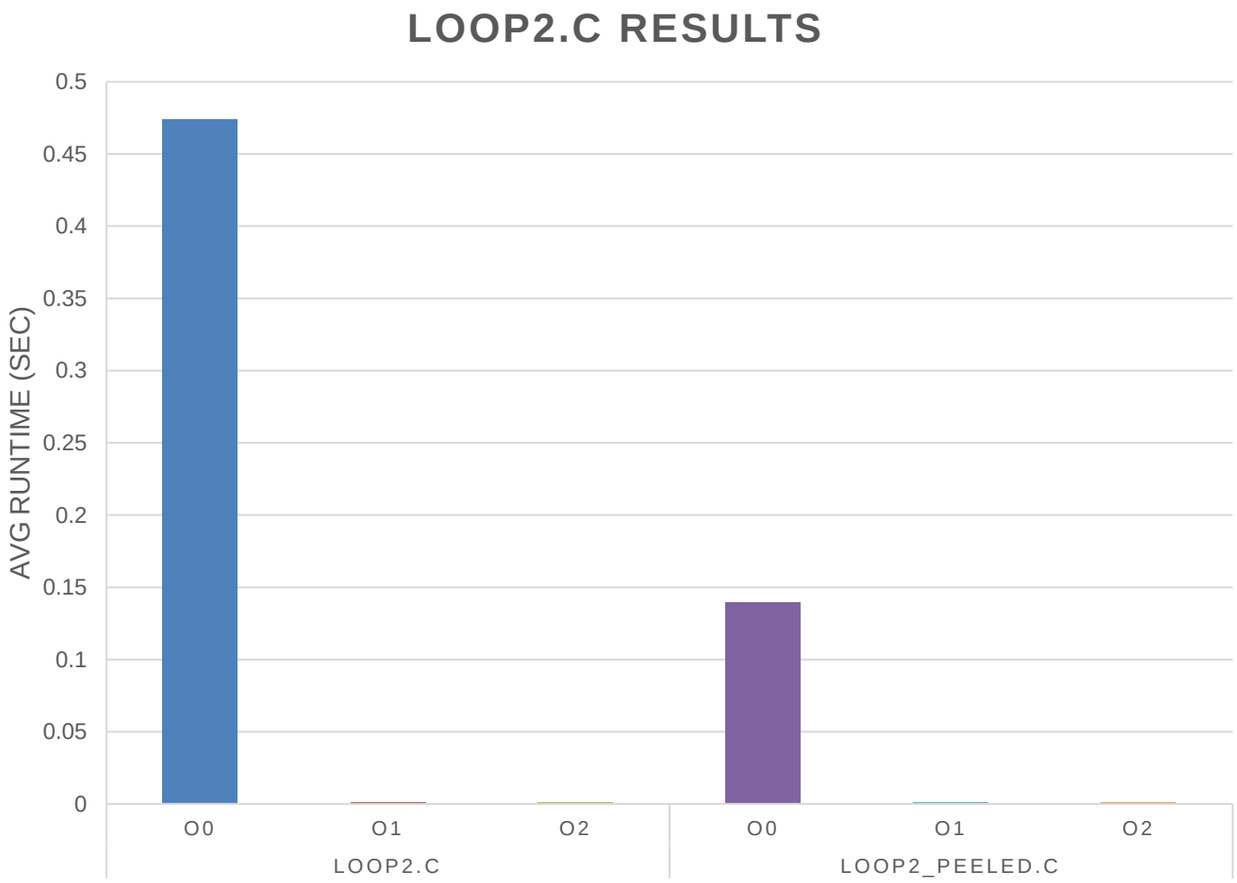
RESULTS


CAP_EX_05.C RESULTS
AVG RUNTIME (SEC)
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
O0
O1
O2
CAP_EX_05.C
O0
O1
O2
CAP_EX_5_PEELED.C

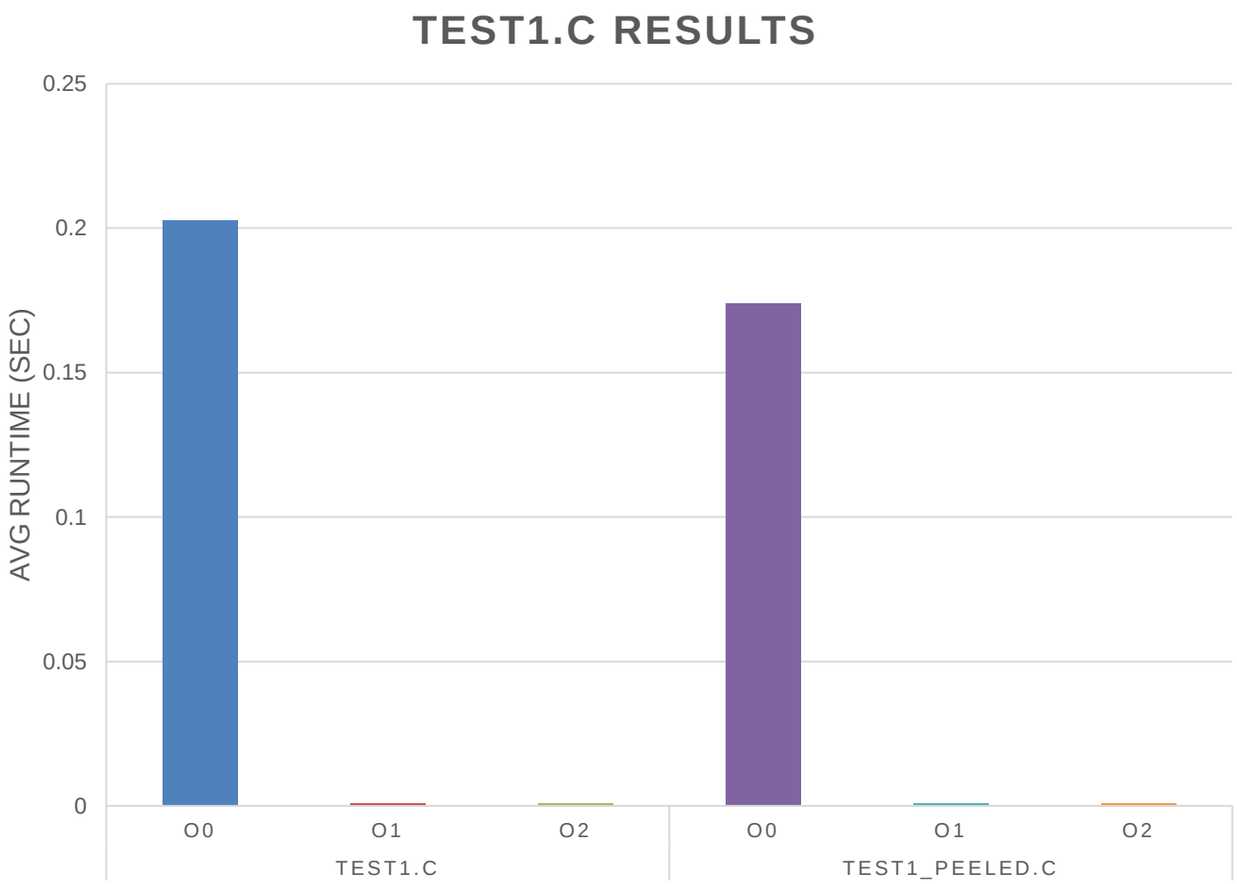

CAP_EX_6.C RESULTS
AVG RUNTIME (SEC)
0.25
0.2
0.15
0.1
0.05
0
O0
O1
O2
CAP_EX_6.C
O0
O1
O2
CAP_EX_6_PEELED.C

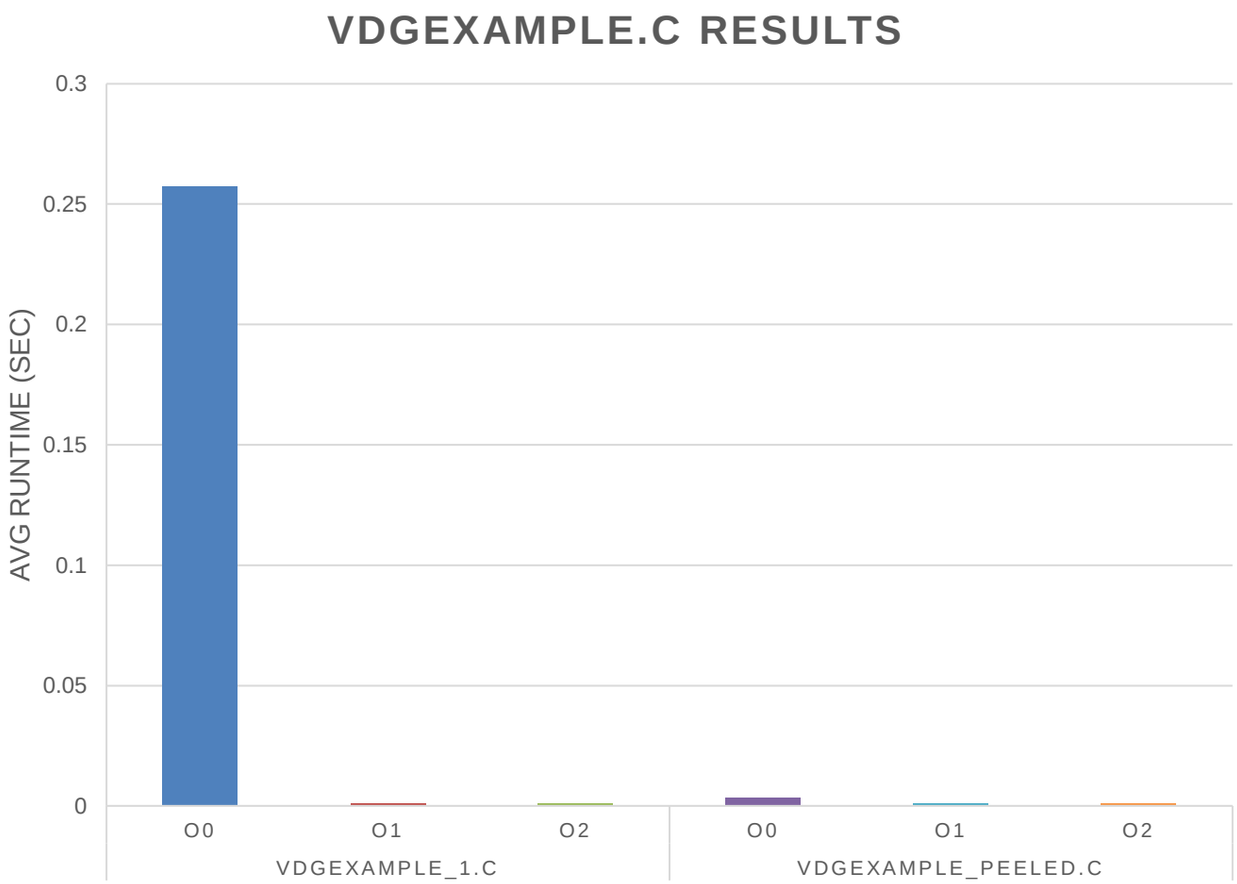
RESULTS


FACT4.C RESULTS
AVG RUNTIME (SEC)
0.9
0.8
0.7
0.6
0.5
0.4
0.3
0.2
0.1
0
O0
O1
O2
O0
O1
O2
FACT4.C
FACT4_PEELED.C


LOOP2.C RESULTS
AVG RUNTIME (SEC)
0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
O0
O1
O2
O0
O1
O2
LOOP2.C
LOOP2_PEELED.C

RESULTS


TEST1.C RESULTS
AVG RUNTIME (SEC)
0.25
0.2
0.15
0.1
0.05
0
O0
O1
O2
TEST1.C
O0
O1
O2
TEST1_PEELED.C


VDGEXAMPLE.C RESULTS
AVG RUNTIME (SEC)
0.3
0.25
0.2
0.15
0.1
0.05
0
O0
O1
O2
VDGEXAMPLE_1.C
O0
O1
O2
VDGEXAMPLE_PEELED.C

ANALYSIS

- With no compiler optimization, LQICM shows significant runtime improvements
- Added compiler optimization along with LQICM reduce the amount of runtime improvements
- Further testing needed with more examples to fully gauge efficacy

CHALLENGES

- Finding code examples
- LLVM installation/use
- Slight miscommunication with client
- Trying to use pre-made benchmarking software
- Benchmark influenced by hardware

MOVING FORWARD

- Summer Scholars Program
  - Research more in LLVM and LLVM passes
  - Develop an LLVM pass that handles LQICM
- Benchmark
  - Maintain normal benchmark as open-source C benchmark
  - Possibly fork this to automatically measure C code before and after going through LQICM pass

CONCLUSION

- Thank you!
- Questions?