LLVM and JitBuilder: Measuring and Comparing the Overhead of two API-based JIT Frameworks

Eric Coffin
Faculty of Computer Science
University of New Brunswick
Fredericton, NB, Canada
eric.coffin@unb.ca

ABSTRACT

Just-in-Time compilation has allowed for significant performance gains during the run-time of applications. Two popular open-source JIT frameworks are LLVM MCJIT and OMR JitBuilder, both of which can be embedded within applications, offering interaces to define and generate native code at run-time. LLVM is a collection of modular compiler and toolchain components, while MCJIT is a framework based on these components to provide JIT compilation. Similarly, JitBuilder is a JIT framework based on the OMR compiler and runtime components. In this report we discuss the different approaches these two frameworks employ. In addition, we measure the overhead of each framework while compiling and then executing a small handful of functions. We found that while LLVM required a larger memory footprint, in certain cases it was able to generate code more quickly. Furthermore, the code LLVM generated typically offered high throughput. JitBuilder, a relatively young project compared to LLVM, does not currently expose the underlying configuration of TRJIT. Instead, it locks the compilation level to the *warm* setting, thus limiting the oppourtunities for optimizations.

CCS CONCEPTS

• **Software and its engineering → Compilers**.

KEYWORDS

compilers, just-in-time, optimization

1 INTRODUCTION

2 BACKGROUND

2.1 LLVM

LLVM.

2.2 JITBuilder

JITBuilder.

3 METHODOLOGY

Methodology

3.1 Results

4 RELATED WORK

Related work.

5 FUTURE WORK

Future Work

6 SUMMARY

Summary

7 ACKNOWLEDGEMENTS

REFERENCES