

ОПТИМИЗАЦИЯ КОДА JIT-КОМПИЛЯТОРОМ

В работе проведено сравнение производительности компиляторов LLVM Clang(c++) и Hotspot-JIT C2(Java).

ВВЕДЕНИЕ

Существование компромисса между компиляторами и интерпретаторами привело к возникновению JIT-компиляторов, которые в свою очередь задумывались для повышения производительности интерпретируемых языков.

I. КРАТКАЯ СПРАВКА

Компилятор преобразует исходный код программы в машинный код и только после этапа компоновки программа становится исполняемой. Интерпретатор выполняет команды построчно. Отсюда и следуют их главные минусы: компилятор-долгая сборка проекта и отсутствие кроссплатформенности, интерпретатор-проблемы с производительностью конечного приложения. JIT-компилятор считается динамическим компилятором, т.к. обладает свойствами интерпретатора (построчное выполнение) и компилятора (выборочная компиляция). JIT-компилятору необходимы данные о работе программы, основываясь на которых он делает разнообразные оптимизации. JIT реализован в Java, .NET Framework, Javascript, Python(PyPy), Ruby.

II. НЕКОТОРЫЕ ОПТИМИЗАЦИИ, ПРОВОДИМЫЕ JIT

- Развёртка циклов
- Устранение общих подвыражений
- Устранение мёртвого кода
- Сворачивание констант
- Подмена операций со стеком на операции с регистрами
- Инлайнинг функций
- Скаляризация

Волощик Алексей Викторович, студент гр.961401, frscfi@gmail.com.

Научный руководитель: Гриневич Яна Григорьевна, ассистент кафедры вычислительных методов и программирования Белорусского государственного университета информатики и радиоэлектроники, ja.grinevich@bsuir.by.

III. РЕЗУЛЬТАТЫ ТЕСТИРОВАНИЯ

Таблица 1 – Сравнение времени выполнения задач между JIT и Clang

Тест	Условия	JIT мкс/оп	Clang мкс/оп
Сумма N элементов массива (N заранее неизвестно)	Размер массива - 10^4	3.2	0.8
Сложение элементов двух массивов в третий	Размер массива - 10^9	61.9	62.4
Сложение константы и элементов массива в i элемент исходного	Размер массива - 10^9	30.0	31.6
Сложение двух полей массива структур	Размер массива - 2^{11}	1.32	1.18
Суммирование при условии заблокированных потоков	-	2.8	190

IV. ВЫВОД

Традиционно C++ считается достаточно быстрым языком программирования, однако, по результатам тестирования, можно увидеть, что Java с использованием технологий JIT-компиляции в некоторых случаях выступает на равных, а в некоторых даже обходит его по скорости выполнения операций. Технология JIT-компиляции активно развивается различными технологическими гигантами и поэтому вполне возможно, что в ближайшем будущем нас ждёт универсальный и производительный инструмент, позволяющий отбросить ограничения многих интерпретируемых языков и сделать разработку ещё более удобной для человека.

Список литературы

1. <https://hacks.mozilla.org/2017/02/a-crash-course-in-just-in-time-jit-compilers/>
2. <https://www.telerik.com/blogs/understanding-net-just-in-time-compilation>
3. <https://ionutbalosin.com/2019/04/jvm-jit-compilers-benchmarks-report-19-04/>
4. <https://www.infoq.com/articles/Graal-Java-JIT-Compiler/>