Масштабирование межмодульных оптимизаций в компиляторе на основе LLVM

Долгорукова Ксения Юрьевна

Аспирант

Институт системного программирования РАН, Москва, Россия

E-mail: unerkannt@ispras.ru

Большинство приложений имеют модульную структуру, но оптимизация таких программ при сборке по классической раздельной схеме “компиляция-связывание” ограничивается отдельным модулем. И хотя многие компиляторы поддерживает межмодульные оптимизации, в случае больших приложений их полноценное проведение зачастую неприемлемо ввиду значительных затрат времени и памяти [1].

Целью работы является реализация поддержки межмодульных оптимизаций на этапе связывания для больших программ, таких как Firefox, состоящего из приблизительно 9.5 тысяч модулей на языках C/C++, содержащих около 4 миллионов строк кода.

Инфраструктура LLVM (Low-Level Virtual Machine – низкоуровневая виртуальная машина) и все его компиляторные компоненты используют промежуточное представление – биткод – набор RISC-подобных трёхадресных инструкций [2].

Так как подсистема межмодульных оптимизаций в LLVM предназначена для оптимизации программ среднего и малого размера (до нескольких сотен тысяч строк кода), на фазе межпроцедурного анализа загружается вся программа в промежуточном представлении. Оптимизация и генерация машинного кода проводится над большим общим файлом в один поток. Возможности управлять количеством потребляемой памяти и временем исполнения больших программ на стадии кодогенерации нет. Таким образом, на текущий момент отсутсвует возможность проведения межмодульных оптимизаций во время связывания для программ, содержащих несколько миллионов строк кода. Кроме того, генерация кода и компоновка, происходящая в один поток, часто является узким местом системы. Разбиение генерации кода на несколько задач позволит существенно сократить общее время сборки.

Для решения вопроса масштабируемости по времени было предложено решение распараллелить процесс сборки. Для этого была реализована утилита, разбивающая посредством жадного алгоритма

файлы с биткодом на группы для работы оптимизирующего компоновщика в несколько потоков. За основу была взята разработанная в ИСП РАН система двухэтапной компиляции, предоставляющая возможность оптимизировать и связывать программы в промежуточном представлении LLVM для целевых архитектур ARM и x86 [3][4].

В результате тестирования максимальное ускорение процесса сборки на архитектуре Intel Core i5-2500 3,3GHz с 8 Гб ОЗУ для процессоров семейства ARM составило до 37% на тестах C-ray, Evas, Coremark, Lame, CLucene, lzma и smallpt при общем неухудшении производительности программ.

Литература

1. Xinliang David Li, Raksit Ashok, Robert Hundt. Lightweight Feedback-Directed Cross-Module Optimization. CGO'10, April 24–28, 2010, Toronto, Ontario, Canada. ACM 978-1-60558-635-9/10/04.

2. Chris Lattner and Vikram Adve. LLVM: A compilation framework for lifelong program analysis and transformation. In Proceedings of the 2004 International Symposium on Code Generation and Optimization (CGO04), March 2004.

3. С.С. Гайсарян, Ш.Ф. Курмангалеев, К.Ю. Долгорукова, В.В. Савченко, С.С. Саргсян. Применение метода двухфазной компиляции на основе LLVM для распространения приложений с использованием облачного хранилища. Труды Института системного программирования РАН, том 26, 2014 г. Выпуск 1. ISSN 2220-6426 (Online), ISSN 2079-8156 (Print), Стр. 315-326.

4. Ш.Ф. Курмангалеев. Методы оптимизации Си/Си++ - приложений распространяемых в биткоде LLVM с учетом специфики оборудования. Труды Института системного программирования РАН, том 24, 2013 г. ISSN 2220-6426 (Online), ISSN 2079-8156 (Print), Стр. 127-144.