# HALIDE: НОВЫЙ ЯЗЫК И КОМПИЛЯТОР ДЛЯ ЭФФЕКТИВНОГО ПАРАЛЛЕЛИЗМА В ОБРАБОТКЕ ИЗОБРАЖЕНИЙ

*Белорусский государственный университет информатики и радиоэлектроники*
*г. Минск, Республика Беларусь*

*Дараган Е. Н.*

Современные алгоритмы обработки изображений весьма требовательны к реализации на современных архитектурах: так, разница в скорости между оптимизированной и неоптимизированной версией может достигать целого порядка. Однако, такие оптимизации в основном достигаются ценой страданий программиста и сложности кода, т.к. вычисления необходимо переупорядочивать для достижения эффективного использования памяти и параллелизма.

В цифровой обработке изображений важной является организация как процесса выполнения программы, так и хранения данных. При этом в алгоритмах часто содержатся стадии с высокой степенью параллелизма, которые обычно упираются в пропускную способность памяти. Поэтому прирост скорости от многопоточности зависит не только от оптимизации внутренних циклов обработки, но и от стратегии распределения данных между потоками и последующего объединения результатов. При этом лучший выбор такой стратегии зависит от платформы и будет сильно отличаться, например, для x86 и современного GPU.

Язык *Halide* был впервые представлен на конференции *SIGGRAPH 2012* и предлагает серьезно пересмотреть подход к реализации алгоритмов цифровой обработки изображений, разделяя сам *алгоритм* и *расписание* его работы. Под *расписанием* подразумеваются решения *о хранении и разделении данных*, а также о *порядке* исполнения частей алгоритма. Такое разделение ведет к упрощению описания алгоритма: так, сами изображения и промежуточные буферы становятся функциями на целочисленном пространстве, без явного указания самих данных или их границ. Алгоритм становится комбинацией таких функций. Программист при этом отдельно указывает стратегии разделения данных для разных функций в составе алгоритма, позволяя использовать особенности конкретной архитектуры. Функциональное описание алгоритма позволяет компилятору эффективнее использовать зависимости по данным в цепочках «производитель-потребитель», что в сумме с широкими возможностями по заданию различных расписаний позволяет находить комбинации, дающие максимальную производительность на целевой платформе, превосходя в некоторых случаях даже высокооптимизированные ассемблерные реализации.

В качестве примера эффективности такого подхода можно привести сравнение размеров исходных кодов программ, осуществляющих размытие изображения квадратным ядром размера 3x3 (“box blur”) и их эффективностей на четырехядерном процессоре (таблица 1):

| **Язык и оптимизации** | **Строк кода** | **Скорость работы, миллисекунд / мегапик-сель** |
|---|---|---|
| C++, неоптимизированная версия | 8 | 9,94 |
| C++, оптимизированная версия (x86 SIMD + OpenMP) | 28 | 0,9 |
| Halide | 8 (4 – алгоритм, 4 – распи-сание) | 0,9 |

Таблица 1. Сравнение различных реализаций алгоритма размытия «box blur» размером 3x3

*Halide* является функциональной надстройкой над C++, которая не несет в себе новых синтаксических конструкций, а работает на уровне типов. После того, как программист функционально описал алгоритм и задал расписание, компилятор автоматически совмещает их в императивный код, все еще состоящий из встроенных примитивов языка. Во время исполнения программы скомпилированный код транслируется в низкоуровневый байт-код LLVM, подвергаясь первичным оптимизациям. *Halide* также предусматривает возможность трансляции в C-подобный код для подробного изучения внутренних процессов, происходящих в программе. Байт-код LLVM переводится в машинный код для целевой платформы, который может быть непосредственно исполнен или записан в объектный файл для включения в другой проект. В качестве целей для генерации кода в Halide предусмотрены x86 (SSE, AVX), ARM (NEON), x86/ARM NaCl, CUDA, OpenCL, GLSL, что обеспечивает превосходную переносимость алгоритмов.

Halide распространяется с открытым исходным кодом под коммерчески разрешительной лицензией MIT, что позволяет и приветствует его использование в любых проектах.

Список использованных источников:

1. J. Ragan-Kelley, C. Barnes, A. Adams, S. Paris, F. Durand, S. Amarasinghe.Halide: A Language and Compiler for Optimizing Parallelism, Locality, and Recomputation in Image Processing Pipelines – PLDI 2013
2. J. Ragan-Kelley, A. Adams, S. Paris, M. Levoy, S. Amarasinghe, F. Durand Decoupling Algorithms from Schedules for Easy Optimization of Image Processing Pipelines - SIGGRAPH 2012