Комплексное исследование эффективности методов программной специализации для графических процессоров

Курсовая работа

Автор: Балашов Илья Вадимович, группа 17.Б10-мм
Научный руководитель: к.ф.-м.н., доц. С.В. Григорьев
Консультант: программист ООО “ИнтеллиДжей Лабс”, к.ф.-м.н Д.А. Березун

Санкт-Петербургский государственный университет
Кафедра системного программирования

Специализация (частичные вычисления)

- Агрессивная оптимизация с использованием статических данных
- $[\![p]\!]\ [in1, in2] = [\![\ \underbrace{[\![mix]\!]\ [p, in1]}_{\text{специализированная программа}}\ ]\!]\ in2$
  - *mix* - специализатор
- Особо эффективна при многократном исполнении программы

Актуальность и похожие работы

- Специализация широко применима
  - Оптимизация запросов к (графовым) БД
  - Обработка изображений
  - Машинное обучение
- Во многих прикладных задачах экспериментально подтверждена эффективность
  - Трассировка лучей (1996)
  - Оптимизация запросов к PostgreSQL (2017)
- Активно распространяются вычисления на видеокартах
  - Хочется использовать специализацию вместе с GPGPU
  - А.Тюрин (4й курс): специализация GPGPU с фреймворком AnyDSL — ускорение в несколько раз в отдельных задачах

Постановка задачи

Цель работы: исследовать применимость техники специализации к различным задачам, решаемых на графических процессорах

Задачи

1. Выполнить обзор существующих специализаторов с поддержкой CUDA
2. Выделить алгоритмы, теоретически поддающиеся специализации, а также распараллеливанию на CUDA
3. Провести экспериментальное исследование специализации на CPU
4. Исследовать применимость и эффективность специализации на видеокартах с выбранным специализатором

Существующие инструменты

- LLPE
  - \+ Работает над биткодом LLVM, языконезависим
  - \+ Хорошая документация
  - — Не стабилен
  - — Не предполагается работа с GPU
- AnyDSL
  - \+ Хорошая документация
  - \+ Заявленная поддержка GPGPU
  - — Отдельный язык для управления специализацией
- LLVM.mix
  - \+ Возможна специализации в несколько стадий
  - \+ Автором заявлена предполагаемая возможность работы с GPU
  - — Слабая документация
  - — Нет информации о применении на "реальных" задачах

LLVM.mix и взамодействие с ним - 1

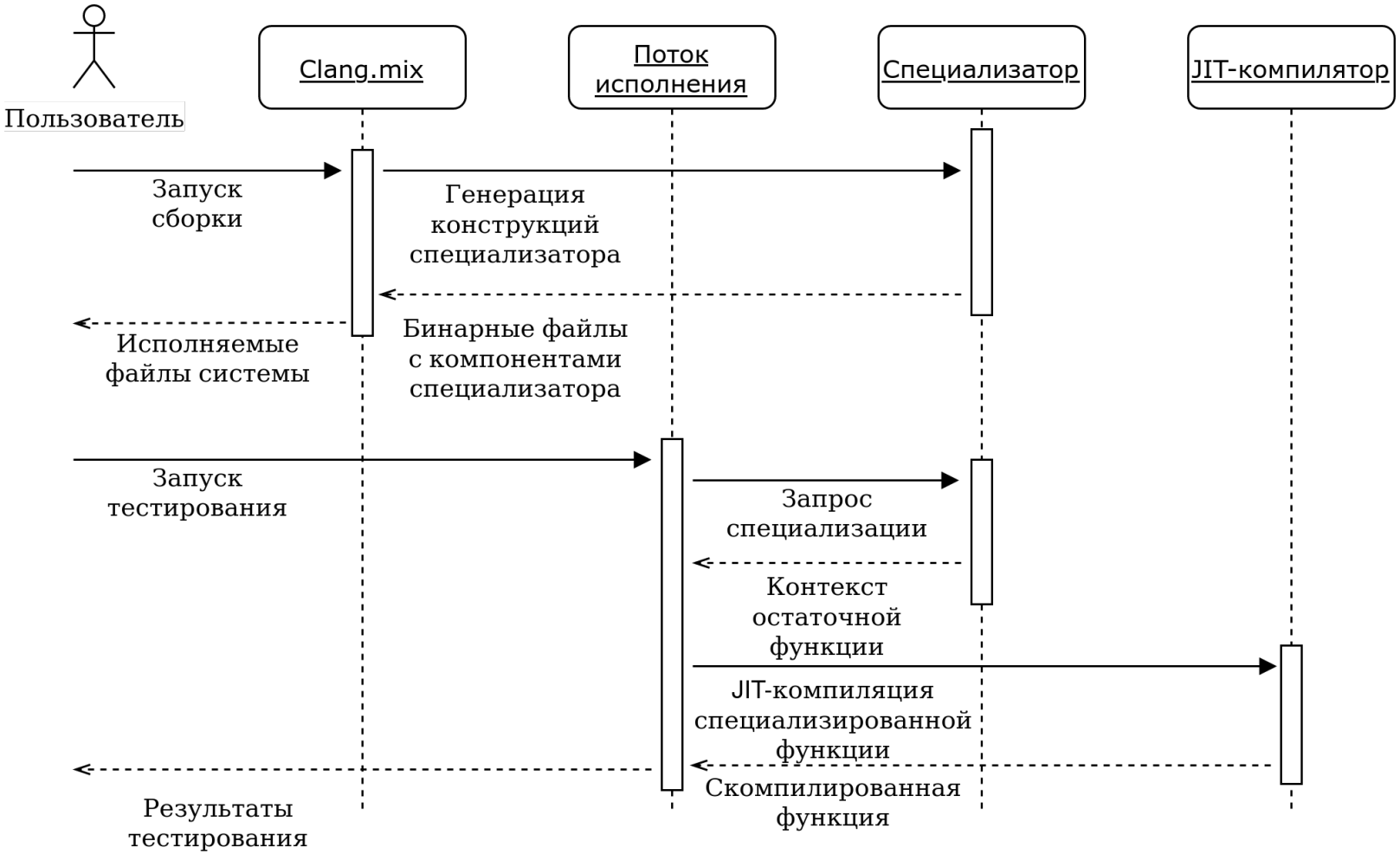


Диаграмма последовательности взаимодействия со специализатором

LLVM.mix и взамодействие с ним - 2

- Разработана система, осуществляющая взаимодействие
- https://github.com/ibalashov24/mix-benchmarks

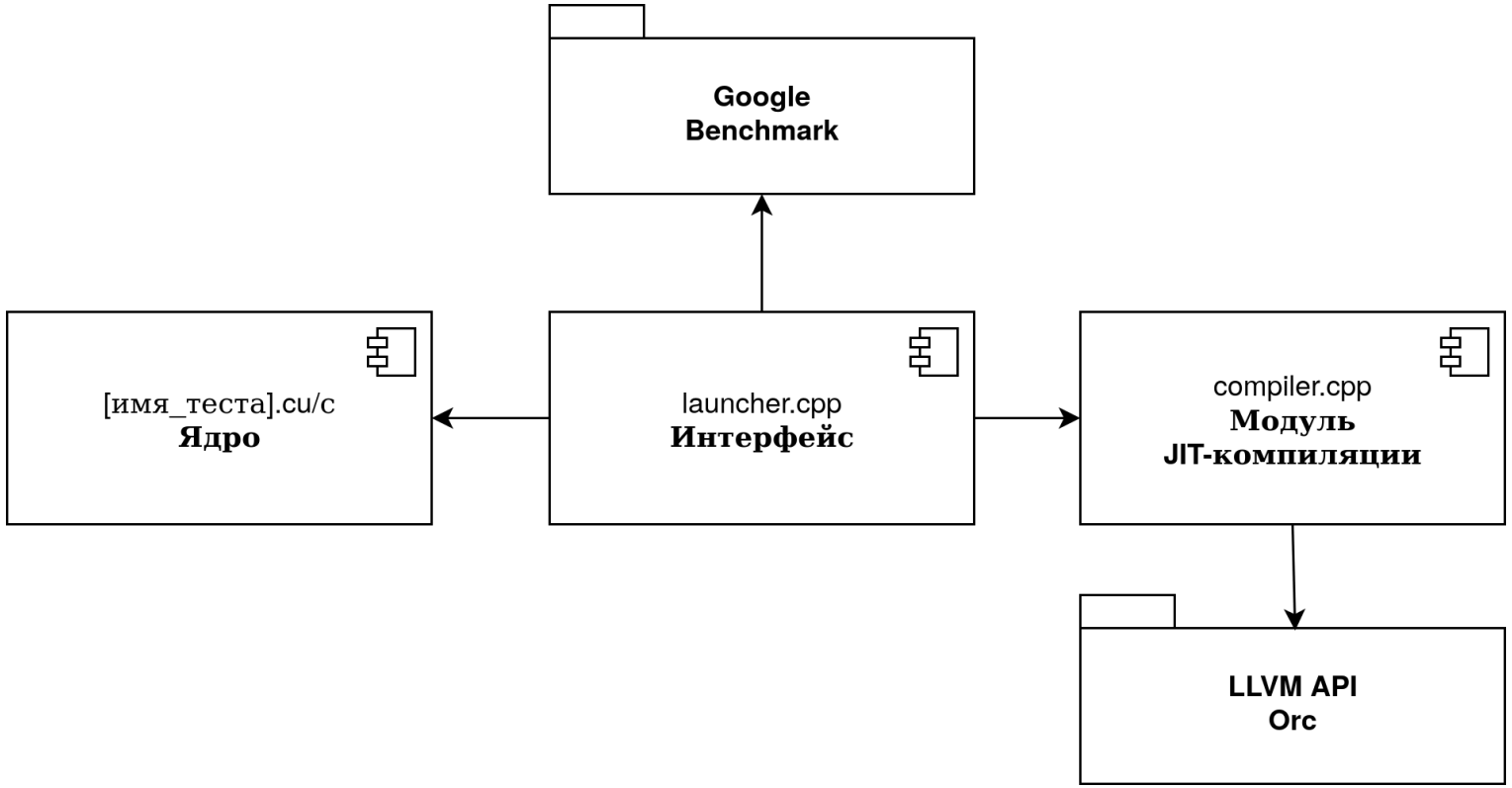


Диаграмма компонентов системы для тестирования

Объекты тестирования

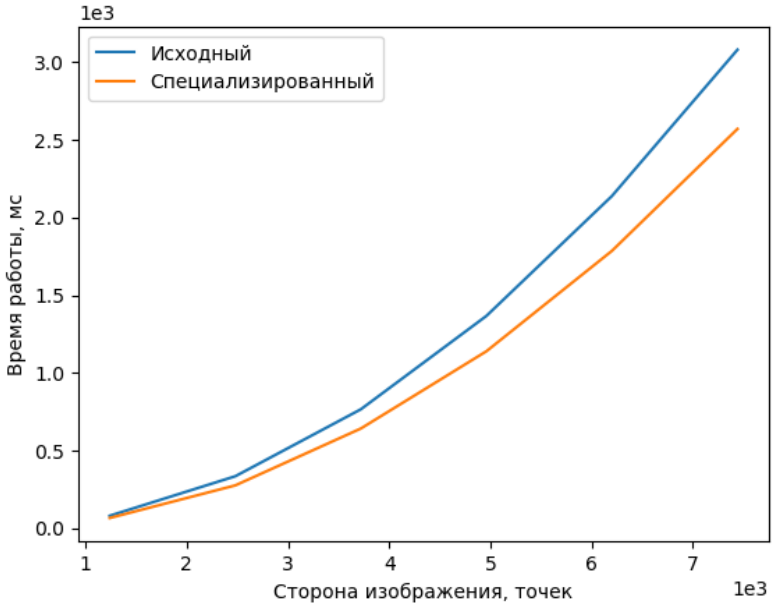
Выбраны алгоритмы из перспективных областей

- (Графовые) базы данных
  - Тензорное произведение с разреженной матрицей
  - Сопоставление множественных шаблонов
  - Сопоставление с регулярным выражением
- Обработка изображений
  - Свёртка изображений с заданным ядром

Эксперименты на CPU — 1

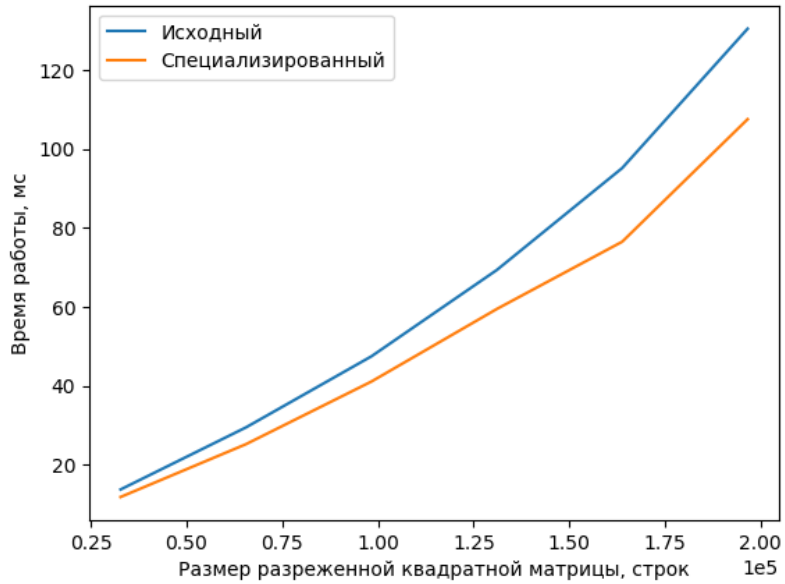
- Ubuntu 18.04, Intel® Core™ i7-6700 (4x4GHz), 64GB RAM

Свёртка изображения



- Ускорение около 20%
- $\Delta < 0.001\%$

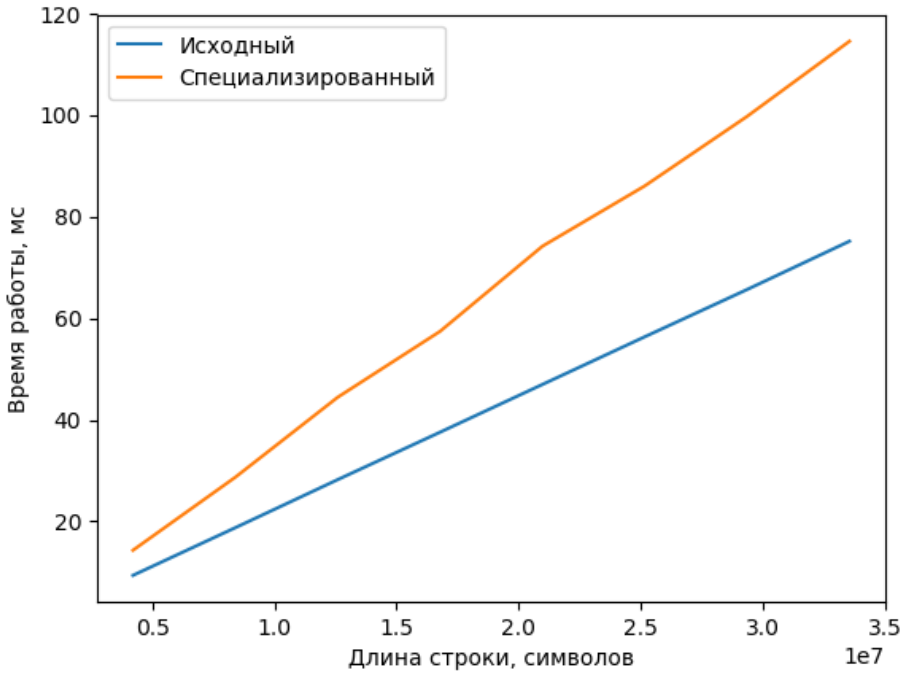
Тензорное произведение



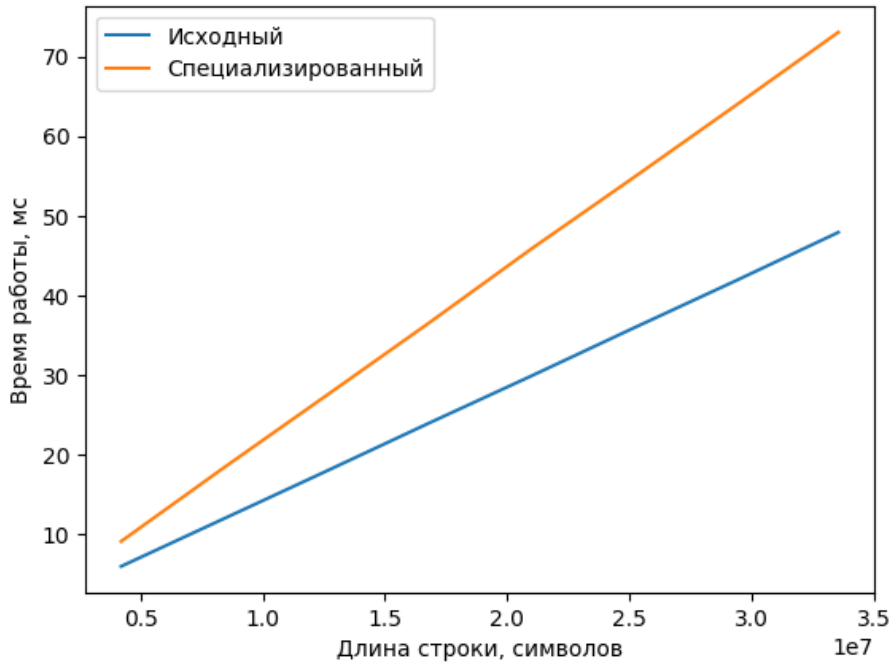
- Ускорение около 18%
- $\Delta < 0.001\%$

Эксперименты на CPU — 2

Сопоставление с автоматом



Сопоставление шаблонов

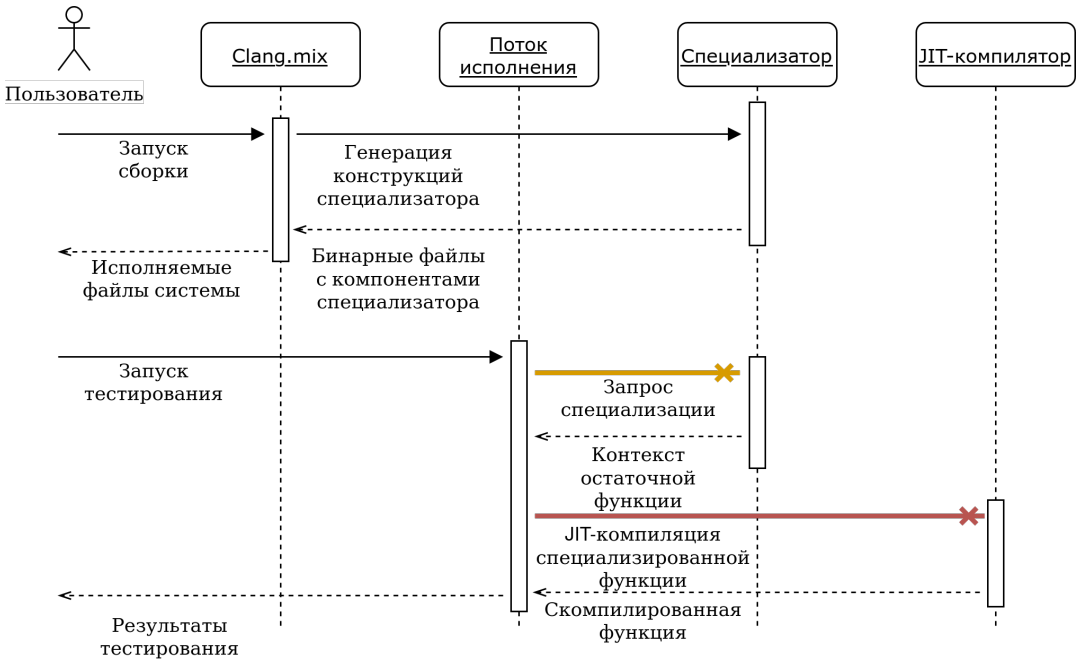


Отрицательный эффект

- Много динамического кода $\Longrightarrow$ нет ускорения
- Накладные расходы специализатора $\Longrightarrow$ замедление

Переход на GPU

- Обнаружена неприменимость специализатора для GPGPU
- Выявлены причины
  - Некорректная передача функций с конструкциями одновременно LLVM.mix и CUDA между объектными модулями
  - Некорректная обработка JIT–компилятором функций на CUDA C
- Требуется существенная модификация инструмента



- **Решение:** использование Clang.JIT (появился в 2020)

Результаты

1. Выполнен обзор специализаторов кода на CUDA C
2. Выделены алгоритмы, теоретически поддающиеся специализации, а также распараллеливанию на CUDA C
3. Проведено экспериментальное исследование специализации на CPU со специализатором LLVM.mix
   - Тензорное произведение — ускорение $18 \pm 0.001\%$
   - Свёртка изображений — ускорение $20 \pm 0.001\%$
4. Исследована применимость и эффективность специализации на GPU с LLVM.mix
   - Выявлена неприменимость специализатора без радикальных модификаций
   - Предложено возможное решение — Clang.JIT