



Brahma.FSharp как средство “прозрачного” использования GPGPU в программах на F#

Семён Григорьев

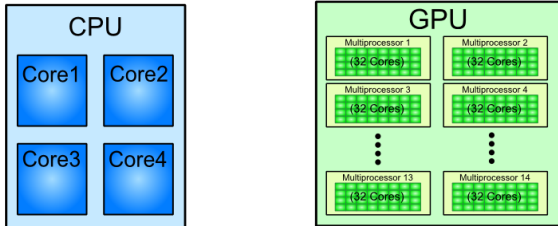
JetBrains Research, лаборатория языковых инструментов
Санкт-Петербургский государственный университет

30.11.2017

Введение: GPGPU

- GPGPU — General-purpose computing for graphics processing units
 - ▶ Реализация SIMD — одна инструкция применяется ко многим данным
 - ▶ Массовый параллелизм общего назначения

CPU/GPU Architecture Comparison



1

¹http://blog.goldenhelix.com/wp-content/uploads/2010/10/cpu_vs_gpu.png

Введение: Техники реализации GPGPU

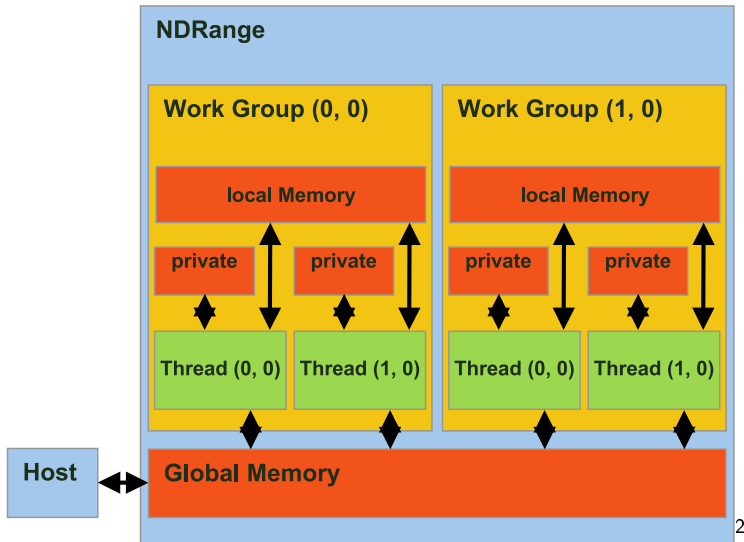
- CUDA
- **OpenCL**
- OpenACC
- C++ AMP
- ...

- GPGPU в F#: но зачем?
 - ▶ Есть ли будущее у таких решений?
 - ▶ Где их можно применять?
- Почему именно так, а не иначе?
 - ▶ Уместен ли такой подход?
 - ▶ Может можно проще?
- Что под капотом у `Brahma.FSharp`?

Зачем и почему GPGPU?

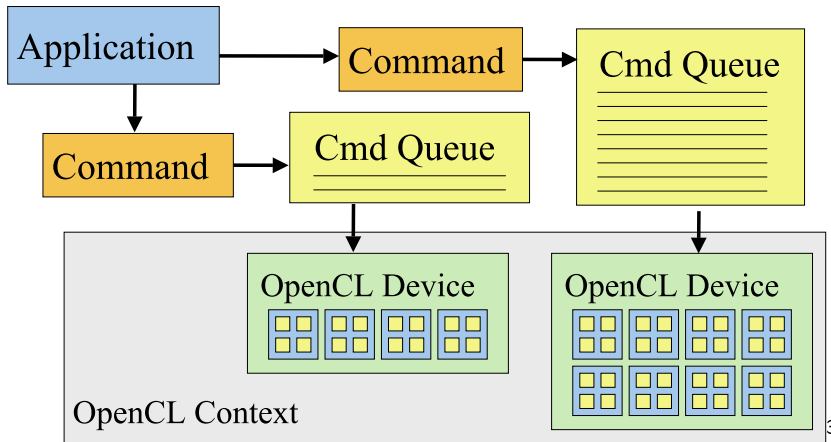
- Обработка больших объёмов данных “регулярным” способом
 - ▶ Аналитика
 - ▶ Моделирование
 - ▶ ...
 - ▶ BigData
- Вычислительные возможности растут
 - ▶ Тысячи ядер
 - ▶ Гигабайты оперативной памяти
 - ▶ Динамический параллелизм

Модель мира OpenCL



²Wen-mei Hwu and John Stone, "The OpenCL Programming Model"

Модель мира OpenCL



³Wen-mei Hwu and John Stone, "The OpenCL Programming Model"

- Средства программирования видеопроцессоров в .NET
 - ▶ Alea GPU (CUDA)
 - ▶ Brahma.FSharp (OpenCL)
 - ▶ FSCL (OpenCL)
- Средства запуска CUDA-кода из ЯВУ:
 - ▶ CUSP
 - ▶ ManagedCuda
- “Низкоуровневые драйвера”
 - ▶ OpenCL.NET
 - ▶ CUDA.NET

Почему именно так, а не иначе?

- “Надёжность” — использование системы типов F# и других особенностей языка и компилятора
- “Прозрачность”/гомогенность разработки для гетерогенных систем

- Функция построения типа по пользовательскому контексту
- Преимущества перед кодогенерацией
 - ▶ Интеграция с пользовательским контекстом
 - ▶ Статическая типизация
 - ▶ Вспомогательная информация доступна в процессе разработки (работает автодополнение и т.д.)
- Недостатки
 - ▶ Высокая сложность тестирования
 - ▶ Высокая сложность отладки
- Помощь трудящимся: FSharp.TypeProviders.SDK
(<https://github.com/fsprojects/FSharp.TypeProviders.SDK>)

Цитирование кода (Code quotation)

- `https://docs.microsoft.com/en-us/dotnet/fsharp/language-reference/code-quotations`
- Предоставление доступа к дереву разбора F#-кода во время выполнения

Что дальше?

- Расширение возможностей транслятора
- Улучшение управления памятью
- Смешанные вычисления
- Использование возможностей F# для параллельного/асинхронного программирования
 - ▶ MailboxProcessor
 - ▶ Норас
 - ▶ ...

- Есть ли будущее у такого подхода?
 - ▶ Какие альтернативы?
 - ▶ Нужна ли гомогенность?
 - ▶ ...
- Какие потенциальные области применения?

- Почта: `semen.grigorev@jetbrains.com`
- Проект на GitHub:
`https://github.com/YaccConstructor/Brahma.FSharp`