# О подходах к общим вычислениям на графических процессорах

Адинец А. В., Сахарных Н. А.

adinetz@gmail.com

# План

- Особенности графических процессоров
- Существующие подходы программирования
- Система C$
  - Описание подхода и языка
  - Трансляция
- Результаты и будущие направления

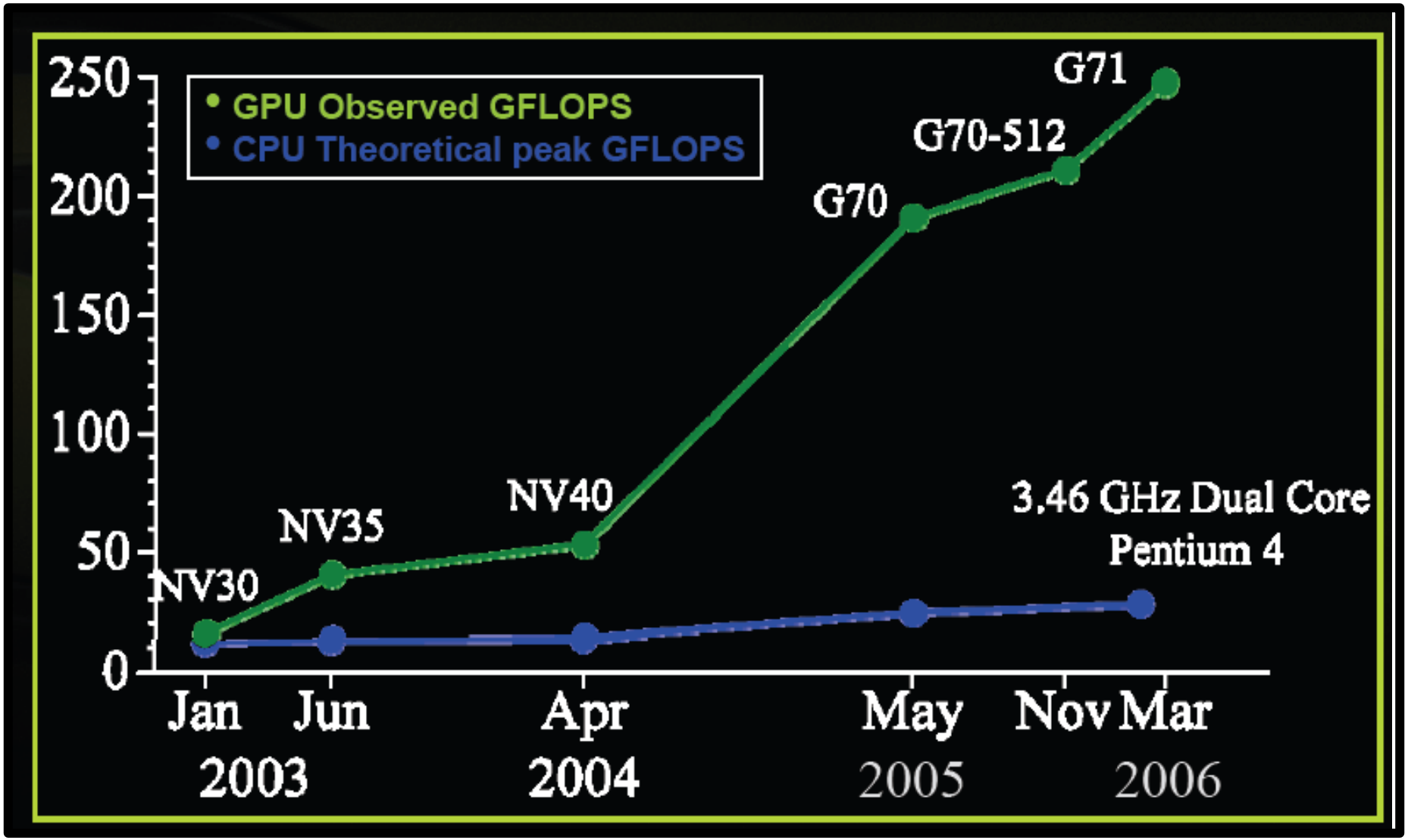
# План

- **Особенности графических процессоров**
- Существующие подходы программирования
- Система C$
  - Описание подхода и языка
  - Трансляция
- Результаты и будущие направления

# Графические процессоры

- Высокая производительность
  - 200+ Гфлопс на 32-разрядных вычислениях
- Высокая доступность
  - В каждом компьютере
  - Не задействованы
  - Низкий $/Гфлопс (1 – 2 $/Гфлопс)

# ГПУ и ЦПУ

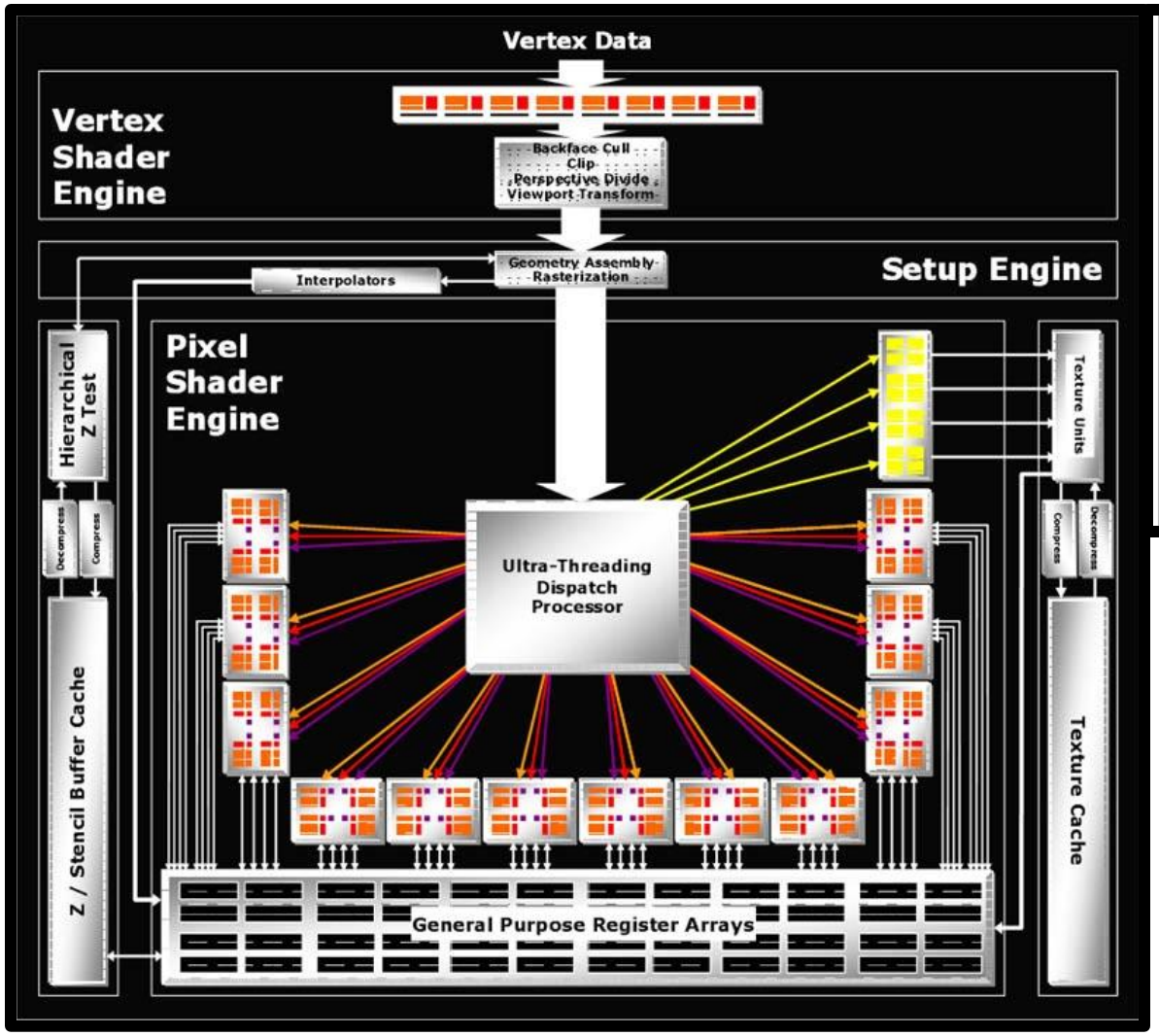

250
200
150
100
50
0
GPU Observed GFLOPS
CPU Theoretical peak GFLOPS
G71
G70-512
G70
NV40
NV35
NV30
3.46 GHz Dual Core
Pentium 4
Jan
Jun
Apr
May
Nov
Mar
2003
2004
2005
2006

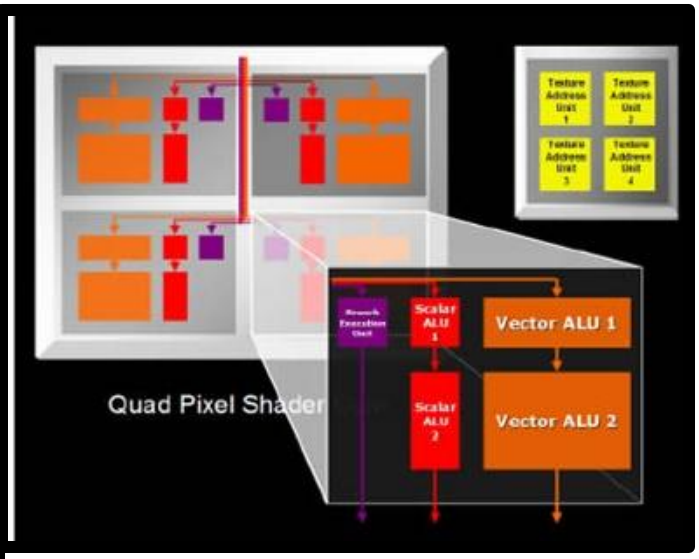
# Реальные задачи

| **Задача** | **Гфлопс** | **% от пика** |
|---|---|---|
| **Умножение матриц** | 120 | 20% |
| **БПФ (FFT)** | 52 | ~10% |

- Трассировка лучей
- Вычислительные задачи биологии
- Обработка изображений
- …

# Архитектура



# Архитектура (2)

- ГПУ
  - ОКМД-процессор
  - Подчиненный процессор
- Шейдер
  - Программа, обрабатывающая один элемент изображения
  - Работает для каждой точки 2-мерной сетки
- Выход
  - Один элемент
  - Нет произвольной записи

# План

- Особенности графических процессоров
- **Существующие подходы программирования**
- Система C$
  - Описание подхода и языка
  - Трансляция
- Результаты и будущие направления

# Подходы программирования

- Графические библиотеки
  - OpenGL, DirectX
- Библиотеки потокового программирования
  - Accelerator
  - PeakStream
  - RapidMind
- Языки
  - Brook GPU

# Низкоуровневые подходы

- NVIDIA
  - CUDA
  - Язык C + библиотека программирования на GPU
  - Подмножество C исполняется на GPU
- ATI/AMD
  - DPVM/CTM/CAL – ассемблер + небольшая библиотека
  - Предполагает построение стека инструментов

# План

- Особенности графических процессоров
- Существующие подходы программирования
- **Система C$**
  - **Описание подхода и языка**
  - Трансляция
- Результаты и будущие направления

# Вычисление на ГПУ…

- Это вычисление функции
  - Нет побочных эффектов
  - Вычисление на области (сетке)
  - Независимо вычисляется каждое значение
- Ленивые вычисления
  - Построение функции
  - Вычисление ее на области

# Язык и библиотека

- Библиотека
  - Работает на .NET
  - Исполнение на GPU
- Язык
  - Удобная нотация для вызовов библиотеки
  - C# + дополнительные конструкции

# Функции и массивы в C$

- Функция (без побочного эффекта)
  - Интерфейсный тип
  - Объект
  - Наследование от функции
- Массив
  - Частный случай функции
  - **float**(**int**, **int**) x = {{1, 2}, {3, 4}}

# Операции с функциями

- Суперпозиция

  float [] a, b;

  var c = a + b; // автовывод, суперпозиция

  float (int, float) x = a + sin; // более сложно

- Редукция

  float [] a = …;

  float x = + a; // sum(a)

# Связанные переменные

```
type matrix = float(int, int); // псевдоним типа
matrix mul(matrix a, matrix b) {
  var c(j, k) = + (a(j, l) * b(l, k));
  return c;
}
```

- Цикл без заголовка
- Аналог Comprehension

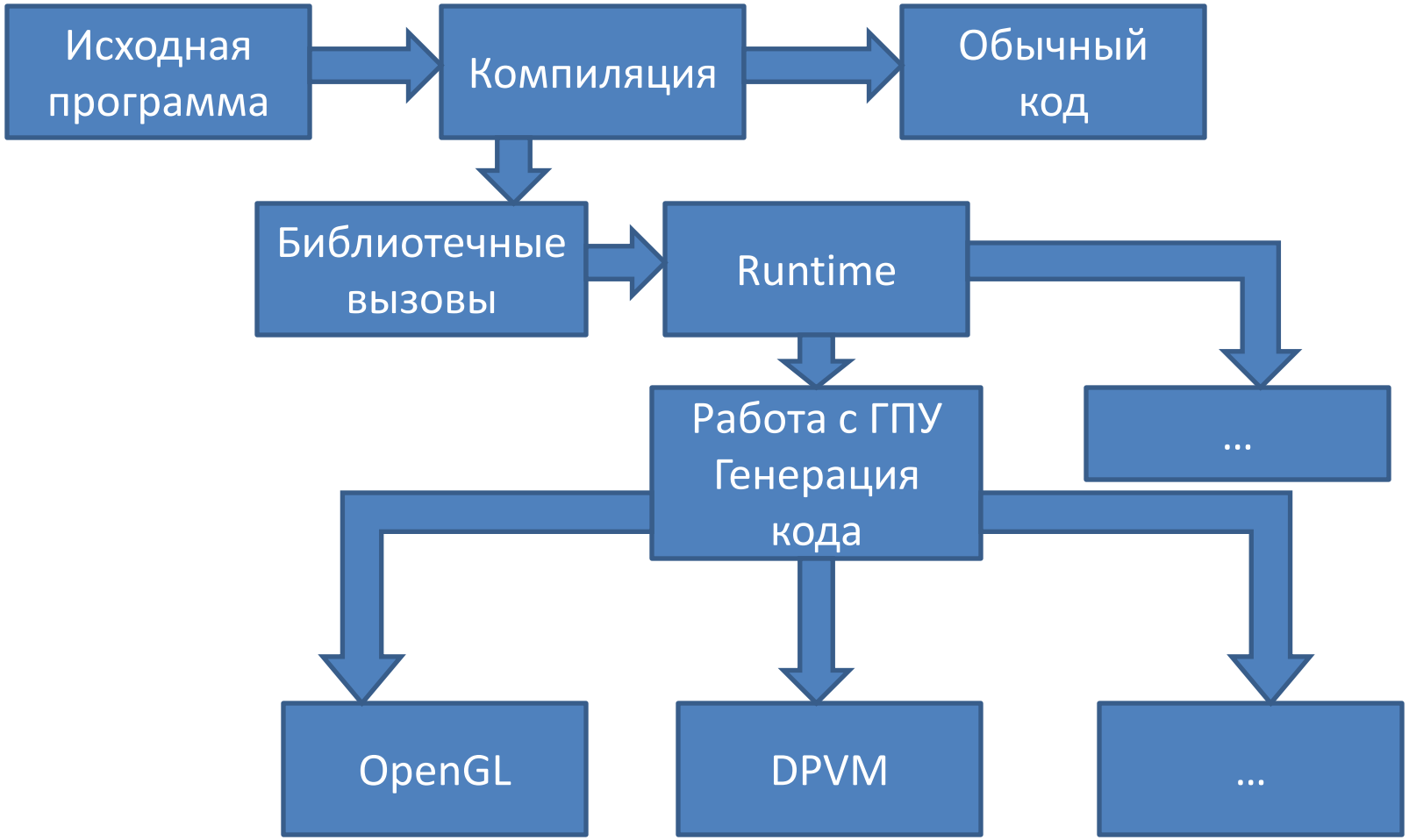
# Фрагмент программы

```
void main() {
  float[,] a = Utils.fromFile(“a.txt”),
  b = Utils.fromFile(“b.txt”);
  var c = mul(a, b); // ленивые вычисления
  Utils.toFile(“c.txt”, ([])c); // сохранение
}
```

# План

- Особенности графических процессоров
- Существующие подходы программирования
- **Система C$**
  - Описание подхода и языка
  - **Трансляция**
- Результаты и будущие направления

# Схема системы



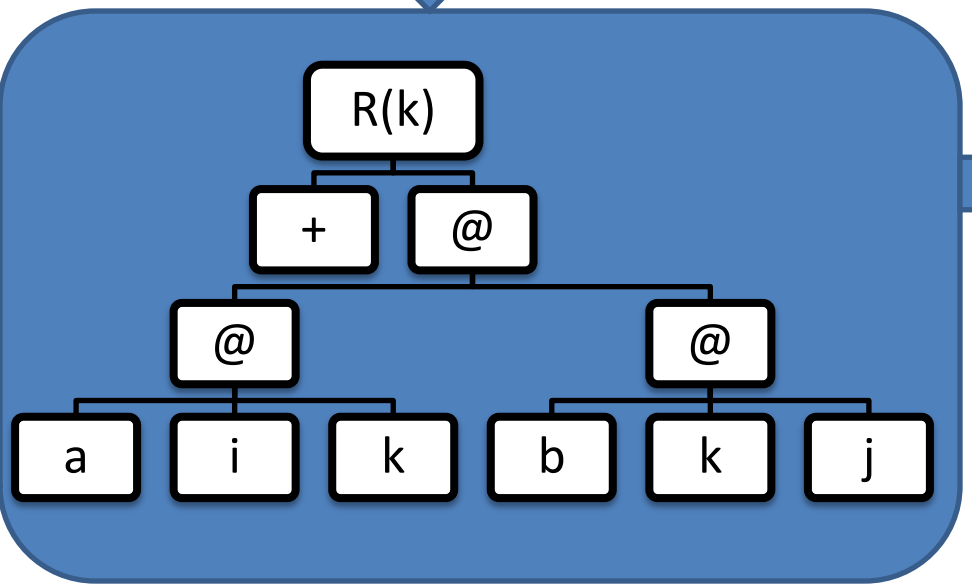
# Оптимизации для ГПУ

- Размещение в памяти
  - Размерности массивов
  - Форматы текстур
- Работа с float4
  - Ускорение арифметических операций
- Повышение вычислительной мощности задачи
- Развертка циклов

# Пример трансляции

**C$:**

```
float(int, int) a, b, c;
c(i, k) = sum(a(i, j) * b(j, k));
```




**Шейдер DirectX**

```
ps_3_0
  dcl vPos.xy
  dcl_2d s0
  dcl_2d s1
  mov r1.x, vPos.x
  mov r1.y, c1.x
  mov r2.x, c1.x
  mov r2.y, vPos.y
  mov r0, c0.zzzz
  rep i1
  rep i0
    texld r3, r1, s0
    texld r4, r2, s1
    add r1.y, r1.y, c0.y
    add r2.x, r2.x, c0.y
    mul r5, r3.xxzz, r4.xyxy
    mad r6, r3.yyww, r4.zwzw, r5
    add r0, r0, r6
  endrep
  endrep
  mov oC0, r0
```

# План

- Особенности графических процессоров
- Существующие подходы программирования
- Система C$
  - Описание подхода и языка
  - Трансляция
- **Результаты и будущие направления**

# Что сделано

- Proof-of-concept интерпретатор C$
  - С трансляцией в GPU
- Поддержка простых функций
  - Арифметические операции
- Суперпозиция, редукция, связанные переменные
- Прямоугольные массивы
  - Автовывод областей

# Тестовая система

- Графический процессор
  - ATI Radeon X1800 XL
  - 16 процессоров * float4
  - 70 ГФлопс
  - 256 МБ видеопамяти
- Прочее
  - Pentium 4 HT 3 ГГц
  - 1024 ГБ оперативной памяти

# Производительность (ГФлопс)

| | **128 x 128** | **256 x 256** | **512 x 512** | **1024 x 1024** |
|---|---|---|---|---|
| **float** | 3.94 | 5.00 | 5.16 | 4.61 |
| **float4** | 8.10 | 14.20 | 15.66 | 15.81 |
| Ручная оптимизация | - | - | ~ 30 | ~ 32 |

# Поддержка

- Проект поддержан грантом компании AMD/ATI для молодых ученых

# Дальнейшая работа

- Создание .NET-компилятора
- Оптимизация ГПУ-части
  - Более сложное размещение и исполнение
- Поддержка новых ГПУ
  - CUDA, CAL
- Сложные типы данных
- Непрямоугольные массивы

ВОПРОСЫ?