Does access pattern affect latency?

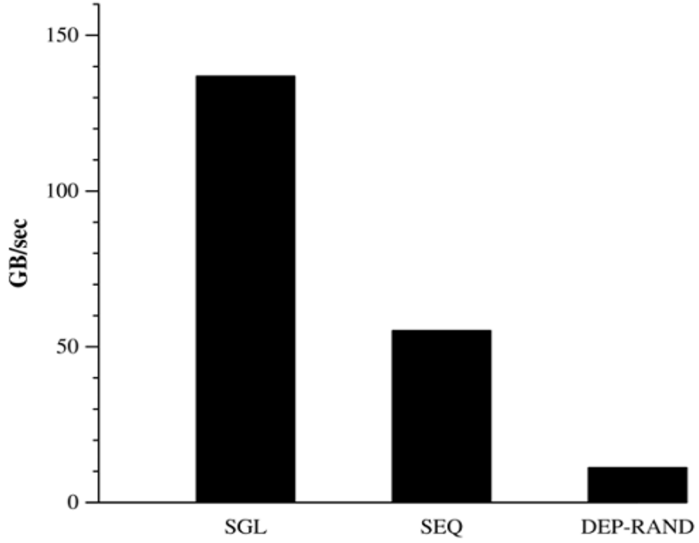
- This is the **most important** question.

- A benchmarking study done by Stanford University
  - Try different texture fetch
    - Cache – every fetch to the same texel
    - Sequential – every fetch increments address by 1
    - Random – dependent lookup with random texture

References: SIGRAPH 2007 Courses on GPGPU. http://www.gpgpu.org/s2007/

Results

- Random is **Bad**, Coherent is **Good**
- Just like a CPU!
  - **out of cache**
    - 147GB/s
  - **sequential**
    - 50GB/s
  - **random**
    - terrible



NVIDIA 8800GTX

References: SIGRAPH 2007 Courses on GPGPU. http://www.gpgpu.org/s2007/

Off-board bandwidth

- How fast can we get data on the board (download)?
- How fast can we get data off the board (readback)?
  - PCI express has a **theoretical limit** of 4GB/s
  - In practice, **GL** is about 1 GB/s, **CUDA** can do over 2.7GB/s (about 85%).
- GPU ←→ HOST = SLOW

References: SIGRAPH 2007 Courses on GPGPU. http://www.gpgpu.org/s2007/

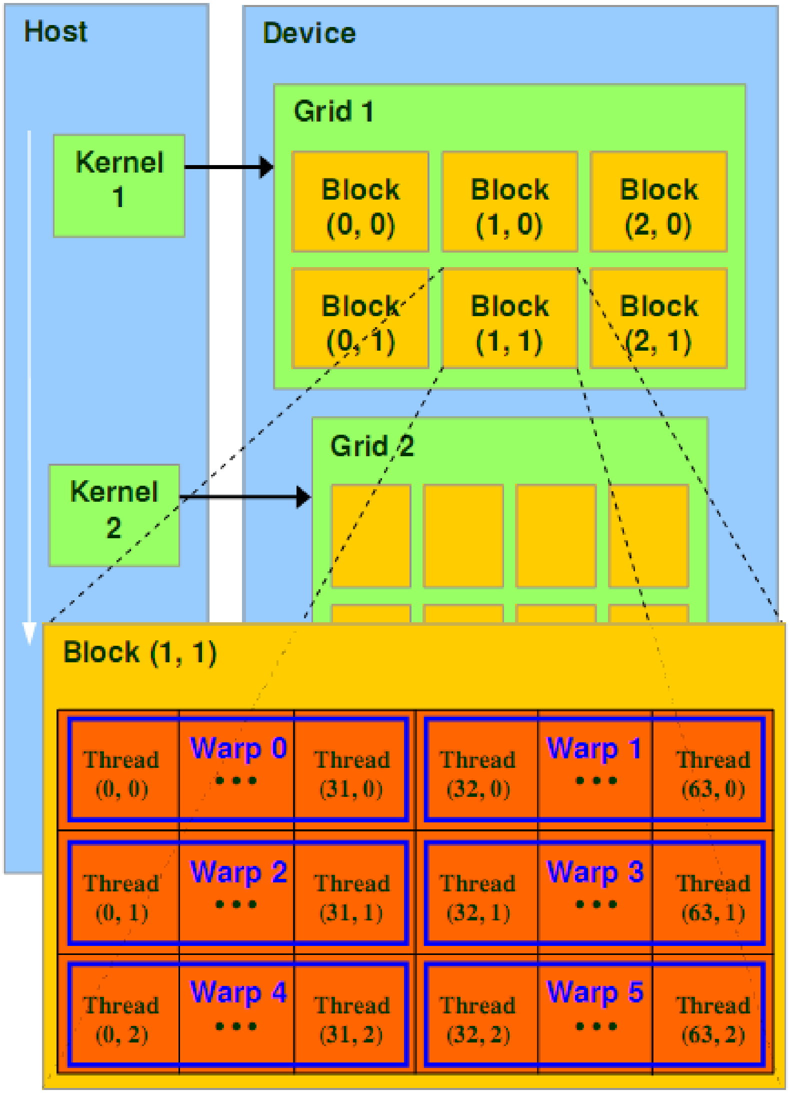
Programming on the GPU

- **Old Model**: use graphics API such as OpenGL and DirectX
  - Use programming tricks
  - Hard to do
  - Relatively slow
- **New Model**: Nvidia CUDA
  - Extension to C
  - Special Compiler - host code and kernal code
  - (Huge) speed up

References: SIGRAPH 2007 Courses on GPGPU. http://www.gpgpu.org/s2007/

8800GTX Architecture

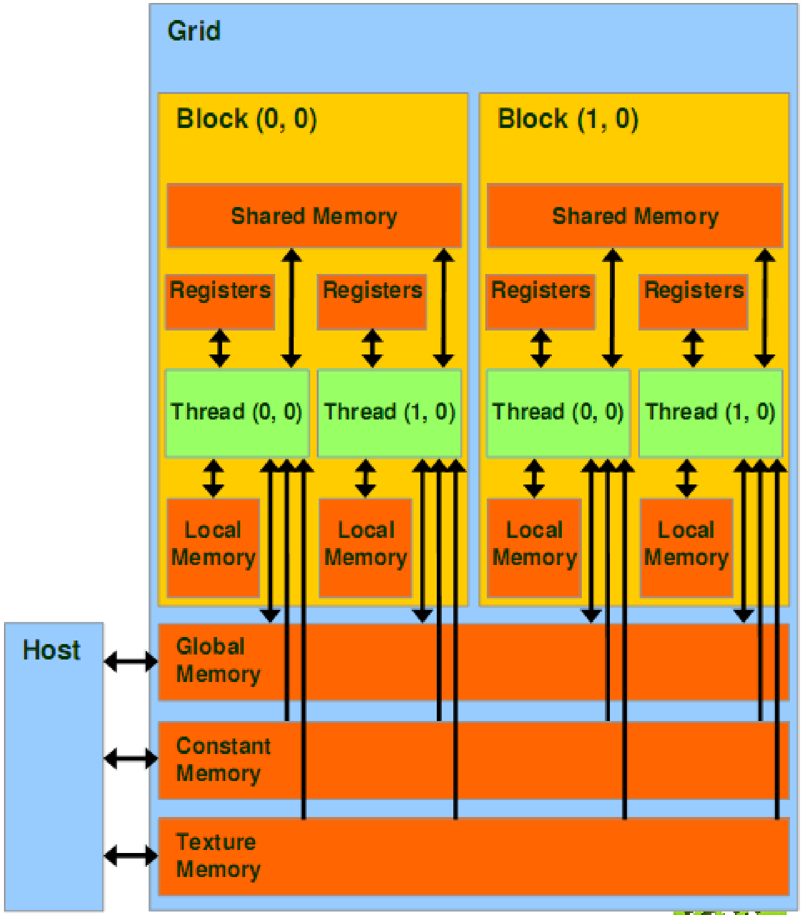
- **GPU** – CUDA device
- **Host** – CPU program
- **Thread** – unit of parallelism in CUDA
- **Warp** – a group of threads
- **Block** – a group of warp
- **Grid** – a group of blocks



References: SIGRAPH 2007 Courses on GPGPU. http://www.gpgpu.org/s2007/

Memory Architecture – key to good performance

- **Host memory**
  - Device ↔ host memory bandwidth is 4 GB/s peak (PCI-express x16)
- **Global/local device memory**
  - High latency, not cached
  - 80 GB/s peak, 1.5 GB (Quadro FX 5600)
- **Shared memory**
  - On-chip, low latency, very high bandwidth, 16 KB
  - Like a user-managed per-multiprocessor cache
- **Texture memory**
  - Read-only, high latency, cached
- **Constant memory**
  - Read-only, low latency, cached, 64 KB



GPGPU NVIDIA

References: SIGRAPH 2007 Courses on GPGPU. http://www.gpgpu.org/s2007/

Performance Strategies

- Maximize **parallelism**
  - Parallelism in algorithm
  - Concurrency of CPU and GPU
- Optimize **access pattern**
- Minimize CPU ←→GPU **data transfer**
- Group **data transfer**
- Maximize use of **shared memory**

References: SIGRAPH 2007 Courses on GPGPU. http://www.gpgpu.org/s2007/