CUDA

CUDA Streams and Constant Memory
Convolution

\[(f \ast g)(x) = \int f(x-t)g(t)\,dt\]

- Modify every point of \(f()\) based on weighted average of its neighbors.
- Weights in \(g()\).
- \(g()\) is known as filter function or convolution kernel.
Convolution

\[ f \ast g \]

\[
\begin{array}{cccccccccccc}
3 & 4 & 3 & 10 & 9 & 11 & 10 & 3 & 2 & 4 & 3 & 3 \\
\end{array}
\]

\[
\begin{array}{cccc}
-1 & 0 & 1 \\
\end{array}
\]

\[
\begin{array}{cccccccccccc}
4 & 0 & 6 & 6 & 1 & 1 & -8 & -8 & 1 & 1 & -1 & -3 \\
\end{array}
\]

(Technically cross-correlation)
__global__ void convolution_1D_basic(float *N, float *M, float *P,
           int Mask_Width, int Width) {

    int i = blockIdx.x*blockDim.x + threadIdx.x;
    float Pvalue = 0;
    int N_start_point = i - (Mask_Width/2);
    for (int j = 0; j < Mask_Width; j++) {
        if (N_start_point + j >=0 && N_start_point + j < Width) {
            Pvalue += N[N_start_point + j]*M[j];
        }
    }
    P[i] = Pvalue;
}

Convolution – basic kernel
Constant Memory

- Typically small amount (64K) – dependent on card

- Similar to global GPU memory
  __global__ - can be read / written by kernels
  __constant__ - can be read by kernels, but NOT written

-
Constant Memory

- Not dynamically allocated.
- Aggressively cached.
- Declared as global variables, e.g.

```c
const int FILTER_SIZE = 11;
__constant__ float filter[FILTER_SIZE];

cudaMemcpyToSymbol(dest, src, size);
```
CUDA Cache

- L1 cache in each SM
- L2 cache on each card. Size depends on card implementation (~768K on Gaea)
Convolution — \texttt{__constant\_} kernel

\texttt{__constant\_} float M[MASK\_WIDTH];
\texttt{__global\_} void convolution_1D\_basic(float *N, float *P,

 int Mask\_Width, int Width) {

 int i = blockIdx.x*blockDim.x + threadIdx.x;

 float Pvalue = 0;
 int N\_start\_point = i - (Mask\_Width/2);
 for (int j = 0; j < Mask\_Width; j++) {

  if (N\_start\_point + j >=0 && N\_start\_point + j < Width) {

   Pvalue += N[N\_start\_point + j]*M[j];
  
  }


 P[i] = Pvalue;


 } } // Only difference in kernel is not passing M – it's constant

CUDA Streams

cudaError_t cudaStreamCreate(cudaStream_t *)
cudaStreamDestroy(cudaStream_t)

- Kernel calls -
  - func <<< Dg, Db, Ns, S >>>> ( args )
    - Dg – grid size
    - Db – block size
    - Ns – dynamically allocated shared memory (default 0)
    - S – stream (default 0)
CUDA Streams – memory

Page-locked (pinned) host memory

- not subject to swapping in host's virtual memory system.
- allows concurrent I/O and kernel execution

- `cudaHostAlloc(void ** dest, int size, flags)`
- `cudaFreeHost(void * ptr)`
- `cudaMemcpyAsync(dest, src, size, direction, stream_t)`
CUDA Streams – memory

- Mapped memory (> 1.0)
  - Same memory visible on host and device
  - Pass `cudaHostAllocMapped` to `cudaHostAlloc()`
  - Use `cudaHostGetDevicePointer(void ** device, void * host, flags)` in kernel to get device pointer to this memory block.
  - No copy needed between device and host
- Unified addressing
  - same address on both host and device
CUDA Streams - synchronization

- `cudaDeviceSynchronize()`
  - wait for all streams
- `cudaStreamSynchronize()`
  - wait for specific stream to finish
- `cudaStreamQuery()`
  - is a specific stream finished?
Compiler options revisited

- `-ptxas-options=-v`
  - compiler reports register, local, shared, and constant memory usages for kernels
Memory

- Page locked host memory
- Global memory accessed in 32, 64, or 128 byte transfers
- Global memory allocation (variables or dynamic) on 256 byte boundaries
- Padding 2D arrays
Miscellaneous

- Branch predication for small branches