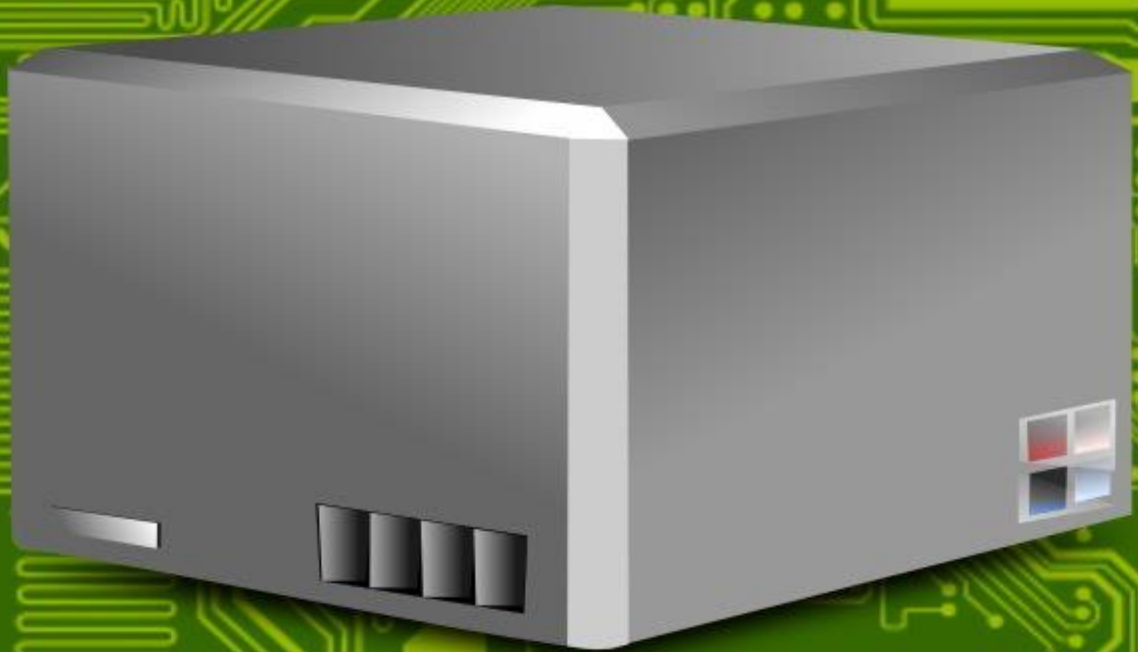
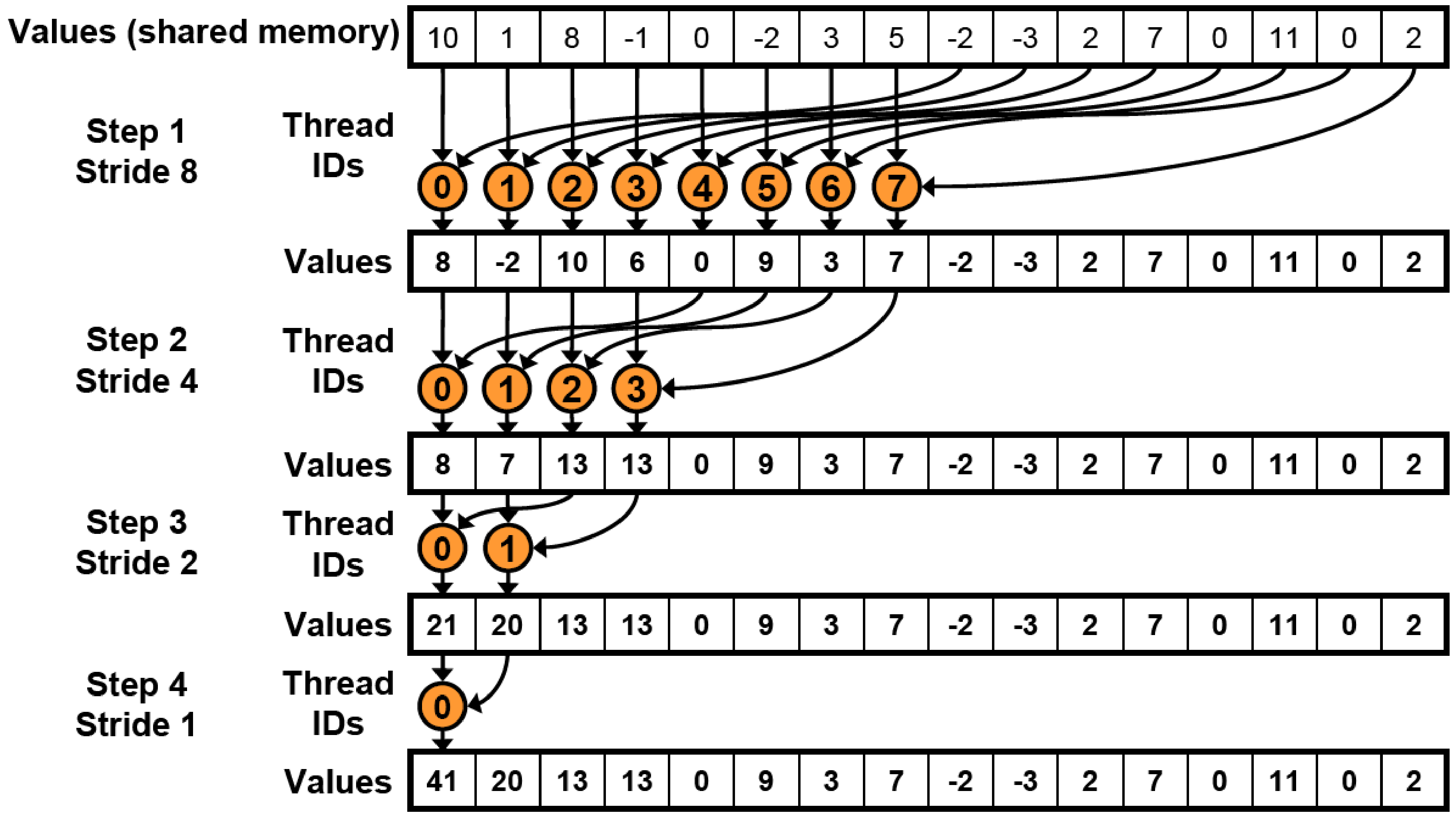
GPU Computing



规约算法

Hui Liu
Email: hui.sc.liu@gmail.com

Reduction #3: Sequential Accesses



- Eliminates bank conflicts

Reduction #3: Code Changes

- Replace stride indexing in the inner loop:

```
// do reduction in shared mem
for (unsigned int s=1; s < blockDim.x; s *= 2) {
        int index  = 2 * s * tid;

        if (index < blockDim.x == 0) {
                sdata[index] += sdata[index + s];
        }
        __syncthreads();
}
```

- With reversed loop and threadID-based indexing:

```
// do reduction in shared mem
for (unsigned int s = blockDim.x/2; s > 0; s /= 2) {

        if (tid < s) {
                sdata[tid] += sdata[tid + s];
        }
        __syncthreads();
}
```

Performance for 4M element reduction

	Time (2^{22} ints)	Bandwidth	Step Speedup	Cumulative Speedup
Kernel 1: interleaved addressing with divergent branching	**8.054 ms**	**2.083 GB/s**		
Kernel 2: interleaved addressing non-divergent branching	**3.456 ms**	**4.854 GB/s**	**2.33x**	**2.33x**
Kernel 3: sequential addressing	**1.722 ms**	**9.741 GB/s**	**2.01x**	**4.68x**

Reduction #3: Bad resource utilization

- All threads read one element
- First step: half of the threads are idle
- Next step: another half becomes idle

```
// do reduction in shared mem
for (unsigned int s = blockDim.x/2; s > 0; s /= 2) {

        if (tid < s) {
                sdata[tid] += sdata[tid + s];
        }
        __syncthreads();
}
```

THANK YOU