Parallel Computing with GPUs

Shared Memory
Part 3 - Boundary Conditions

The University Of Sheffield.

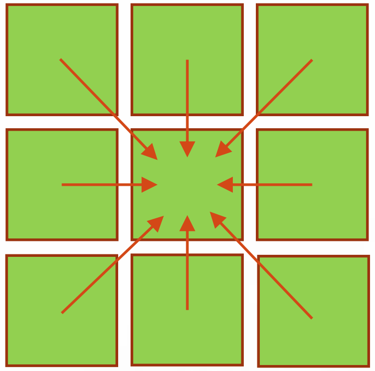
Dr Paul Richmond
http://paulrichmond.shef.ac.uk/teaching/COM4521/

This Lecture (learning objectives)

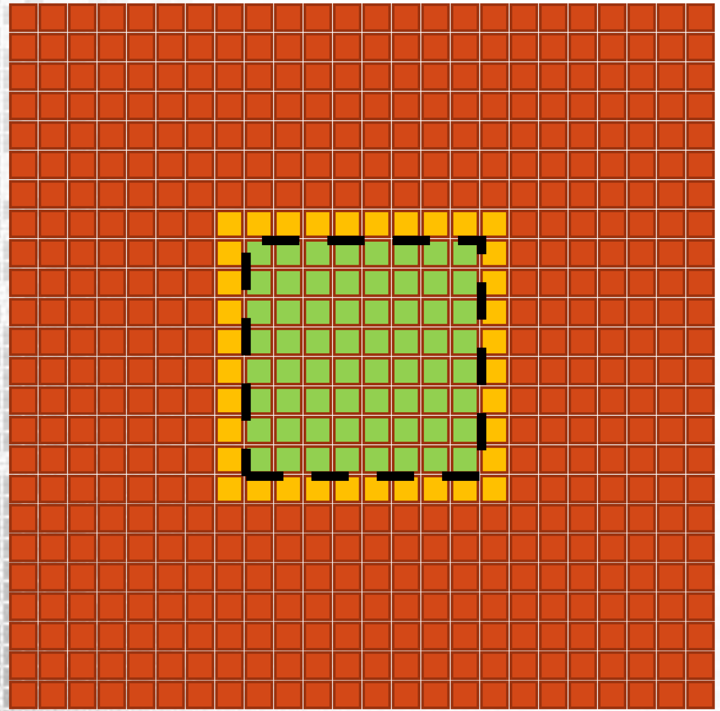
- ❑Boundary Conditions
  - ❑Demonstrate the impact of boundary conditions for 2D gather problems
  - ❑Compare and contrast different solutions to solving boundary problems

Boundary Conditions & Shared Memory Tiling

- ❑Consider a 2D problem where data is gathered from neighbouring cells
  - ❑Each cell reads 8 values (gather pattern)
  - ❑Sounds like a good candidate for shared memory
    - ❑We can tile data into memory



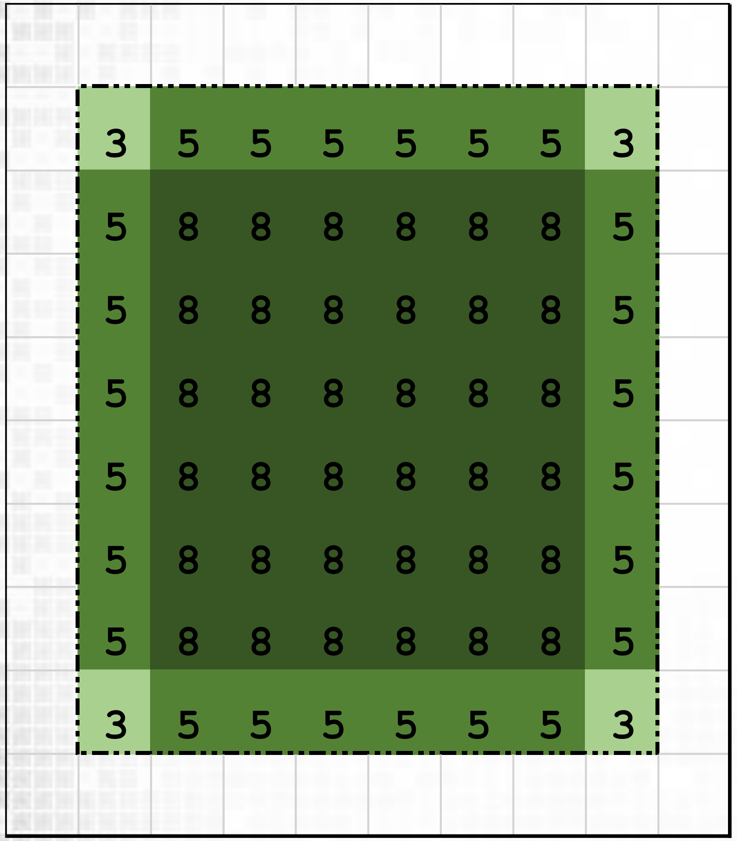
Gather pattern



Thread Block size is 8x8

🟩 Data tiled into shared memory
🟨 Data not tiled into shared memory

Problem with our tiling approach



3	5	5	5	5	5	5	3
5	8	8	8	8	8	8	5
5	8	8	8	8	8	8	5
5	8	8	8	8	8	8	5
5	8	8	8	8	8	8	5
5	8	8	8	8	8	8	5
5	8	8	8	8	8	8	5
3	5	5	5	5	5	5	3

- ❑ Diagram shows number of cached reads
- ❑ Memory access pattern is good for threads at centre of the block
  - ❑ 6x6x8=288 cached reads
- ❑ Memory access for threads at the boundary of the block is poor
  - ❑ 132 cached reads
  - ❑ **92 un-cached reads**

Boundary Conditions Solutions (Easy)

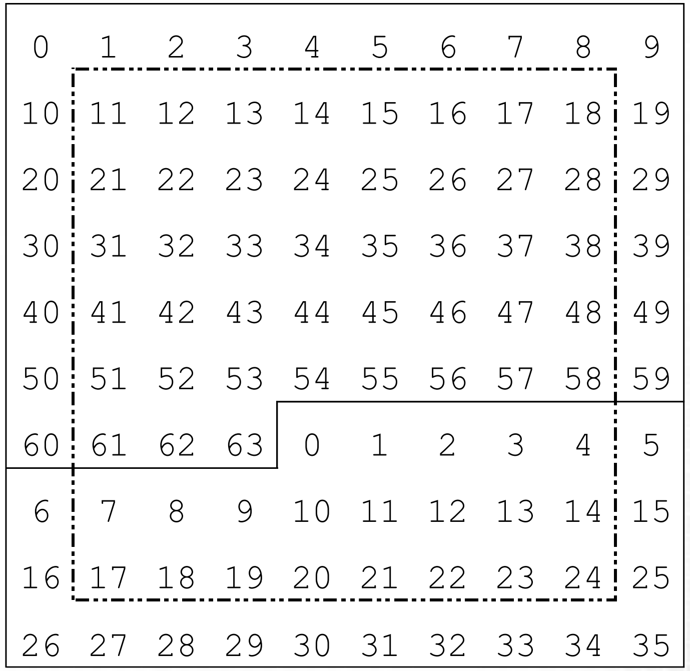
$$Utilisation = \frac{DIM^2}{(DIM + 2)^2}$$

DIM	Utilisation
8	64%
12	73%
16	79%
20	83%
24	85%
28	87%
32	89%
36	90%
40	91%
44	91%
48	92%

- ❑Launch more threads
  - ❑Launch thread block of `DIM+2 × DIM+2`
  - ❑Allocate one element of space per thread in SM
  - ❑Every thread loads one value
  - ❑Only threads in inner DIM x DIM compute values
    - ❑Causes under utilisation
- ❑Use more shared memory per thread
  - ❑Launch same `DIM × DIM` threads
  - ❑Allocate `DIM+2 × DIM+2` elements of space in SM
  - ❑Threads on boundary load multiple elements
    - ❑Causes unbalanced loads
  - ❑All threads perform compute values

Boundary Conditions Solution (Harder)

- Use more shared memory per thread
  - Launch same `DIM × DIM` threads
  - Allocate `DIM+2 × DIM+2` elements of space in SM
  - Distribute the loading of SM evenly between threads
    - Thread position in the block must be translated to a position in SM for each load
    - Only last warp will have imbalance of at worse one load
- 100 loads
- 512/512 cached reads



0	1	2	3	4	5	6	7	8	9
10	11	12	13	14	15	16	17	18	19
20	21	22	23	24	25	26	27	28	29
30	31	32	33	34	35	36	37	38	39
40	41	42	43	44	45	46	47	48	49
50	51	52	53	54	55	56	57	58	59
60	61	62	63	0	1	2	3	4	5
6	7	8	9	10	11	12	13	14	15
16	17	18	19	20	21	22	23	24	25
26	27	28	29	30	31	32	33	34	35

Acknowledgements and Further Reading

- Overview of Shared Memory Bank Conflicts
  - http://cuda-programming.blogspot.co.uk/2013/02/bank-conflicts-in-shared-memory-in-cuda.html
- Architecture Specific Guidance
  - http://acceleware.com/blog/maximizing-shared-memory-bandwidth-nvidia-kepler-gpus
  - https://on-demand.gputechconf.com/gtc/2018/presentation/s81006-volta-architecture-and-performance-optimization.pdf

Summary

- ❑Boundary Conditions
    - ❑Demonstrate the impact of boundary conditions for 2D gather problems
    - ❑Compare and contrast different solutions to solving boundary problems