High-Performance Scientific Computing
Lecture 5: More OpenCL, MPI

MATH-GA 2011 / CSCI-GA 2945 · October 3, 2012

Today

Tool of the day: Git

OpenCL: Device Language

OpenCL: Synchronization

Intro to MPI

Bits and pieces

- HW1 grades sent
- HW2 graded soon
- HW3 due
- HW4 out tomorrow
- Cuda cluster accounts
- Mailing list messages

Outline

Tool of the day: Git

OpenCL: Device Language

OpenCL: Synchronization

Intro to MPI

Git

Demo time

Outline

Tool of the day: Git

OpenCL: Device Language

OpenCL: Synchronization

Intro to MPI

Moar OpenCL!

Demo time

OpenCL Device Language

OpenCL device language is C99, with these differences:

- ⊕ Index getters
- ⊕ Memory space qualifiers
- ⊕ Vector data types
- ⊕ Many generic ('overloaded') math functions
- ⊕ Synchronization
- ⊖ Recursion
- ⊖ Fine-grained `malloc()`
- ⊖ Function pointers

Address Space Qualifiers

Type	Per	"Speed"
private*)	work item	super-fast
local	group	fast
global	grid	kinda slow

*) default, so optional

Address Space Qualifiers

Type	**Per**	**“Speed”**
private*)	work item	super-fast
local	group	fast
global	grid	kinda slow

*) default, so optional

Should really discuss “speed” in terms of latency/bandwidth.

Both decrease with distance from the point of execution.

Outline

Tool of the day: Git

OpenCL: Device Language

OpenCL: Synchronization

Intro to MPI

Concurrency and Synchronization

GPUs have layers of concurrency.

Each layer has its synchronization primitives.

Concurrency and Synchronization

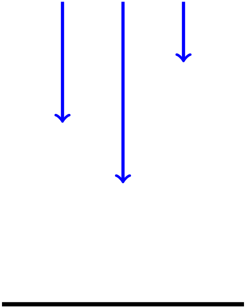
GPUs have layers of concurrency.

Each layer has its synchronization primitives.

- Intra-group:
  `barrier(...)`,
  `mem_fence(...)`
  `... =`
  `CLK_{LOCAL,GLOBAL}_MEM_FENCE`
- Inter-group:
  Kernel launch
- CPU-GPU:
  Command queues

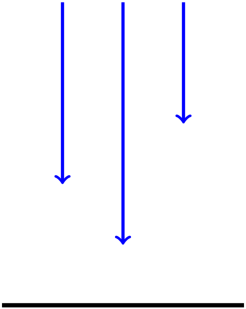
Synchronization

What is a Barrier?



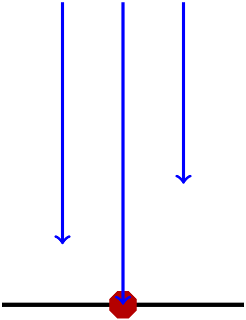
Synchronization

What is a Barrier?



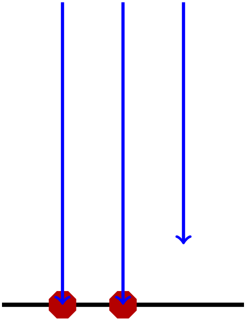
Synchronization

What is a Barrier?



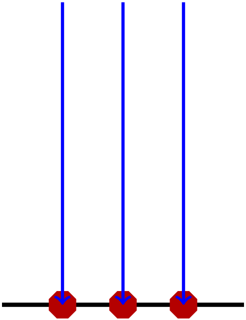
Synchronization

What is a Barrier?



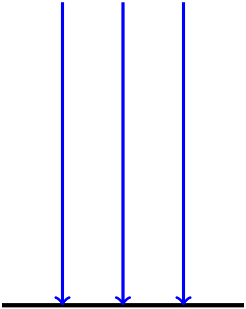
Synchronization

What is a Barrier?



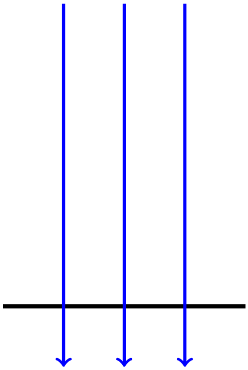
Synchronization

What is a Barrier?



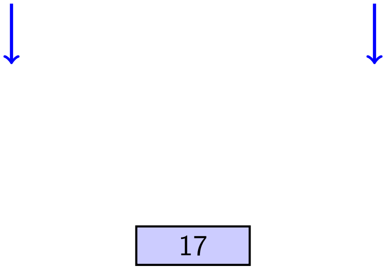
Synchronization

What is a Barrier?



Synchronization

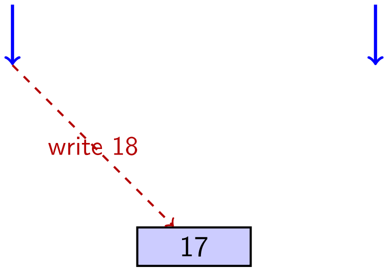
What is a Memory Fence?



17

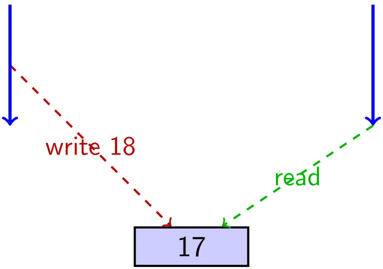
Synchronization

What is a Memory Fence?



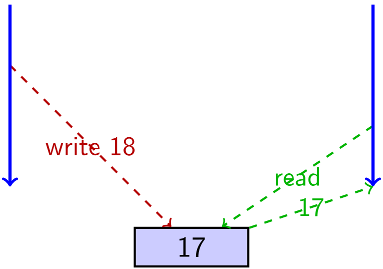
Synchronization

What is a Memory Fence?



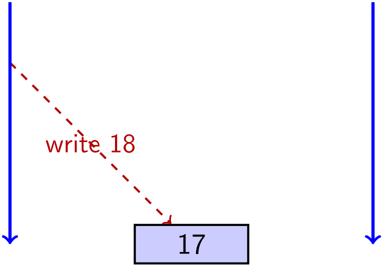
Synchronization

What is a Memory Fence?



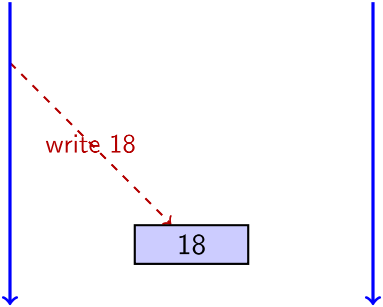
Synchronization

What is a Memory Fence?



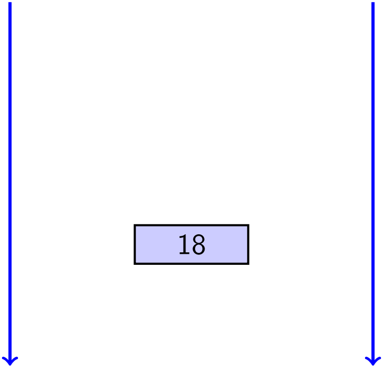
Synchronization

What is a Memory Fence?



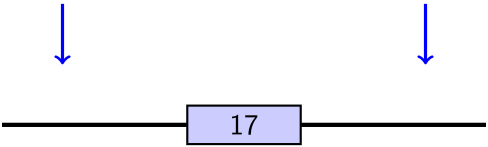
Synchronization

What is a Memory Fence?



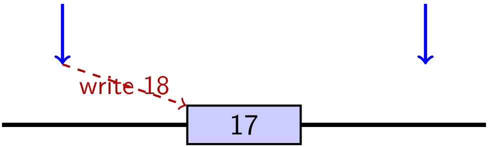
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



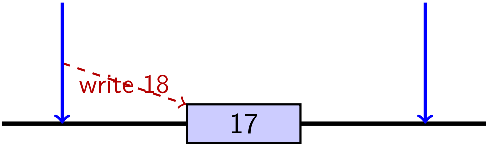
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



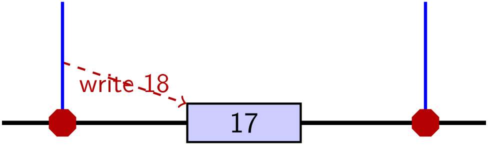
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



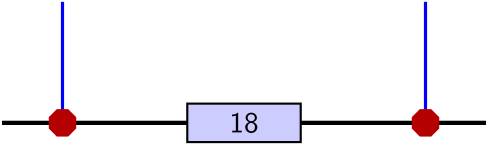
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



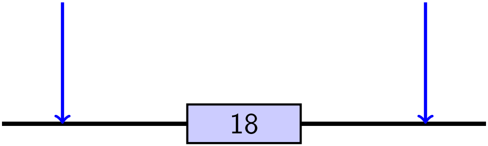
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



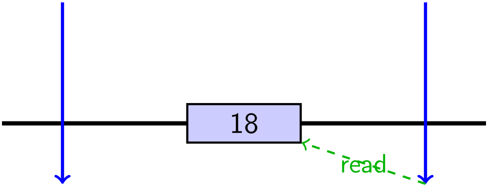
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



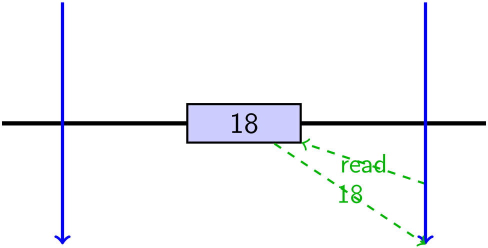
Synchronization

What is a Memory Fence? An ordering restriction for memory access.



Synchronization

What is a Memory Fence? An ordering restriction for memory access.



Synchronization between Groups

Golden Rule:

Results of the algorithm must be independent of the order in which work groups are executed.

Synchronization between Groups

Golden Rule:

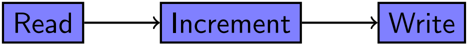
Results of the algorithm must be independent of the order in which work groups are executed.

Consequences:

- Work groups may read the same information from global memory.
- But: Two work groups may not validly write different things to the same global memory.
- Kernel launch serves as
  - Global barrier
  - Global memory fence

Atomic Operations

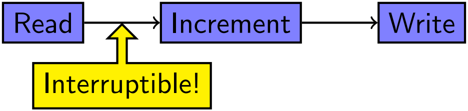
Collaborative (inter-block) Global Memory Update:



Read → Increment → Write

Atomic Operations

Collaborative (inter-block) Global Memory Update:



Read

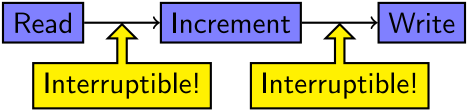
Increment

Write

Interruptible!

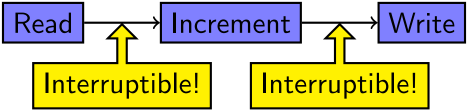
Atomic Operations

Collaborative (inter-block) Global Memory Update:

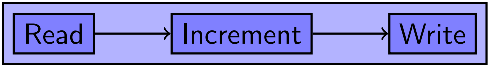


Atomic Operations

Collaborative (inter-block) Global Memory Update:

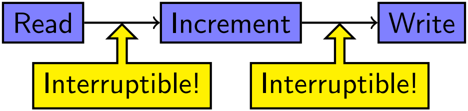


Atomic Global Memory Update:

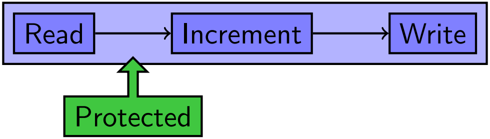


Atomic Operations

Collaborative (inter-block) Global Memory Update:

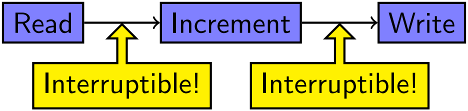


Atomic Global Memory Update:

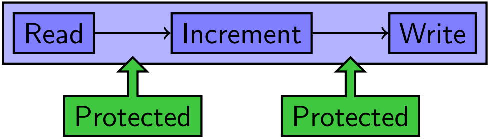


Atomic Operations

Collaborative (inter-block) Global Memory Update:

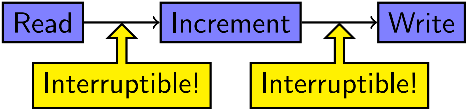


Atomic Global Memory Update:

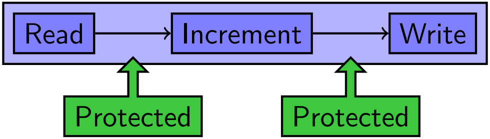


Atomic Operations

Collaborative (inter-block) Global Memory Update:



Atomic Global Memory Update:



How?

atomic_{add,inc,cmpxchg,...}(int *global, int value);

Atomic: Compare-and-swap

```
int atomic_cmpxchg (__global int *p, int cmp, int val)
int atomic_cmpxchg (__local int *p, int cmp, int val)
```

Does:

- Read the 32-bit value (referred to as old) stored at location pointed by p.
- Compute `(old == cmp) ? val : old`.
- Store result at location pointed by p.
- Returns old.

Atomic: Compare-and-swap

```
int atomic_cmpxchg (__global int *p, int cmp, int val)
int atomic_cmpxchg (__local int *p, int cmp, int val)
```

Does:

- Read the 32-bit value (referred to as old) stored at location pointed by p.
- Compute `(old == cmp) ? val : old`.
- Store result at location pointed by p.
- Returns old.

Implement atomic `float` add?

Outline

Tool of the day: Git

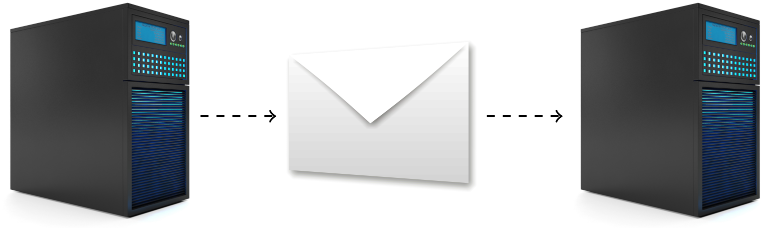
OpenCL: Device Language

OpenCL: Synchronization

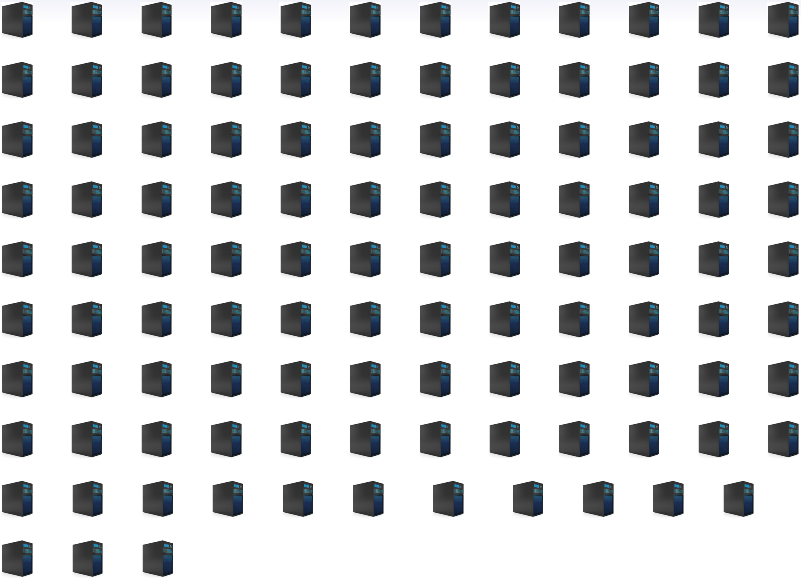
Intro to MPI

MPI

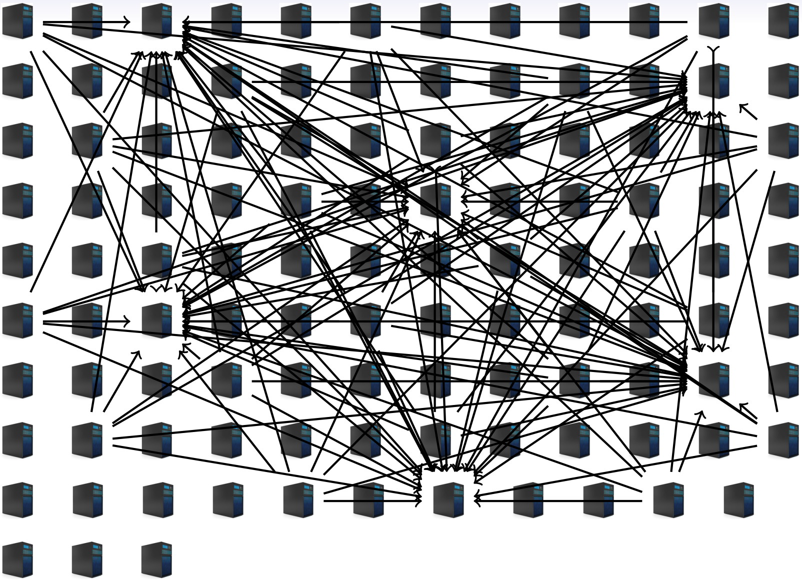
Message **P**assing **I**nterface:



MPI

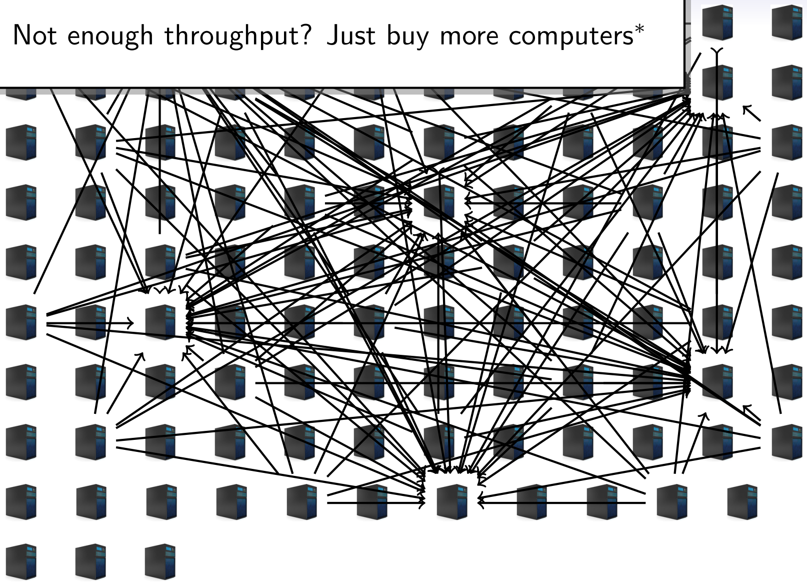


MPI

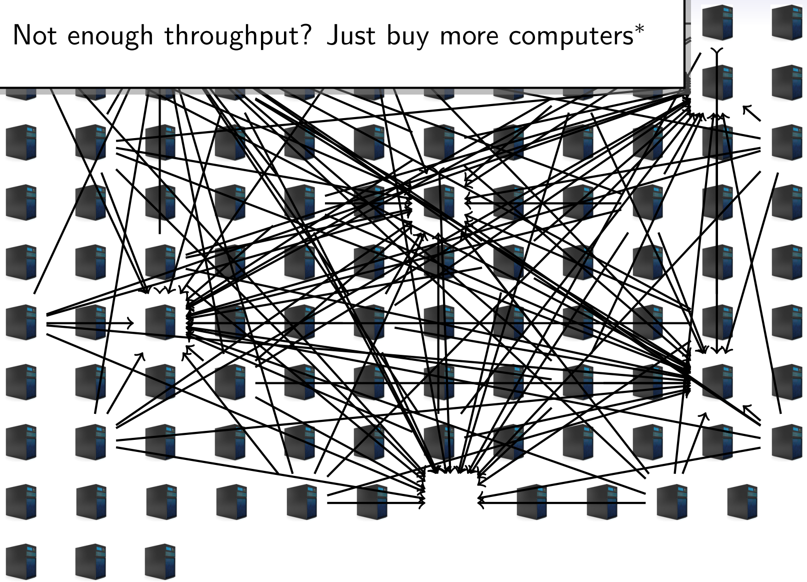


MPI

Not enough throughput? Just buy more computers*

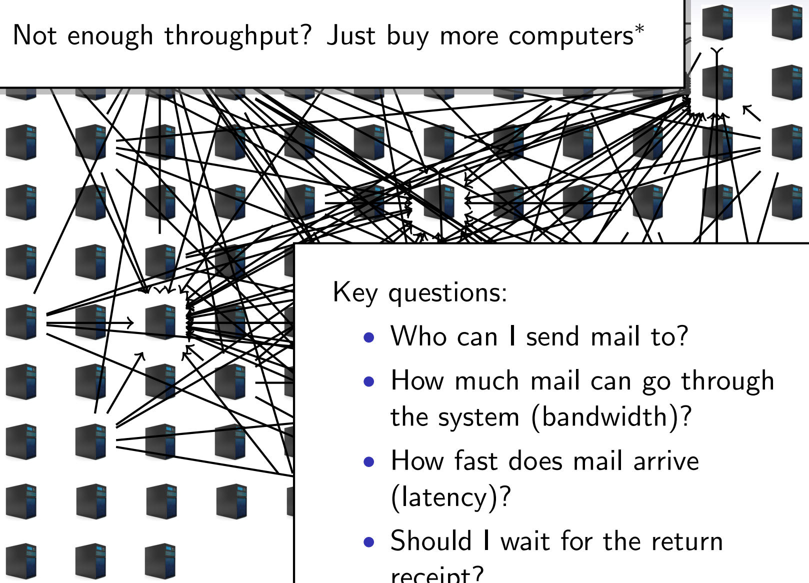


MPI



Not enough throughput? Just buy more computers*

MPI

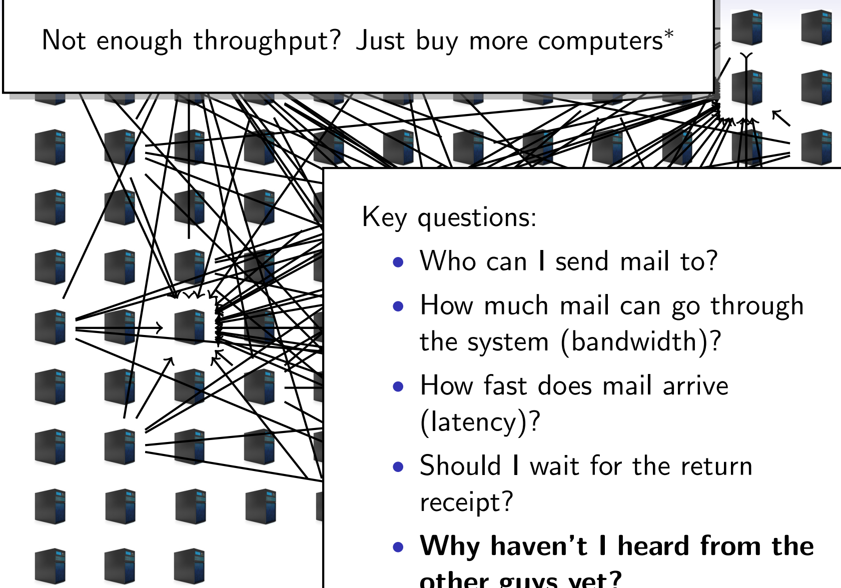


Not enough throughput? Just buy more computers*

Key questions:

- Who can I send mail to?
- How much mail can go through the system (bandwidth)?
- How fast does mail arrive (latency)?
- Should I wait for the return receipt?

MPI



Not enough throughput? Just buy more computers*

Key questions:

- Who can I send mail to?
- How much mail can go through the system (bandwidth)?
- How fast does mail arrive (latency)?
- Should I wait for the return receipt?
- **Why haven't I heard from the other guys yet?**

MPI

MPI 3.0

Born September 21, 2012

MPI 1.0: June 1994

MPI

Demo time

Questions?

?

Image Credits

- Onions: flickr.com/darwinbell 🅭
- Server: sxc.hu/Kolobsek
- Envelope: sxc.hu/ilco
- Gift box: sxc.hu/iprole