Alternative GPU friendly assignment algorithms

Paul Richmond and Peter Heywood
Department of Computer Science
The University of Sheffield

Graphics Processing Units (GPUs)

The University Of Sheffield.

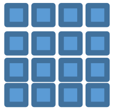
Context: GPU Performance

Serial Computing

~40
GigaFLOPS

1 core

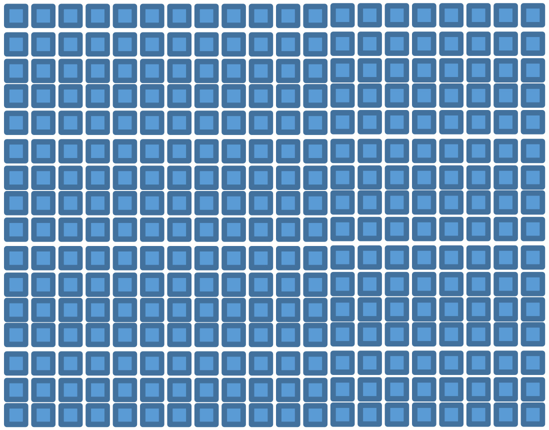
Parallel Computing



384
GigaFLOPS

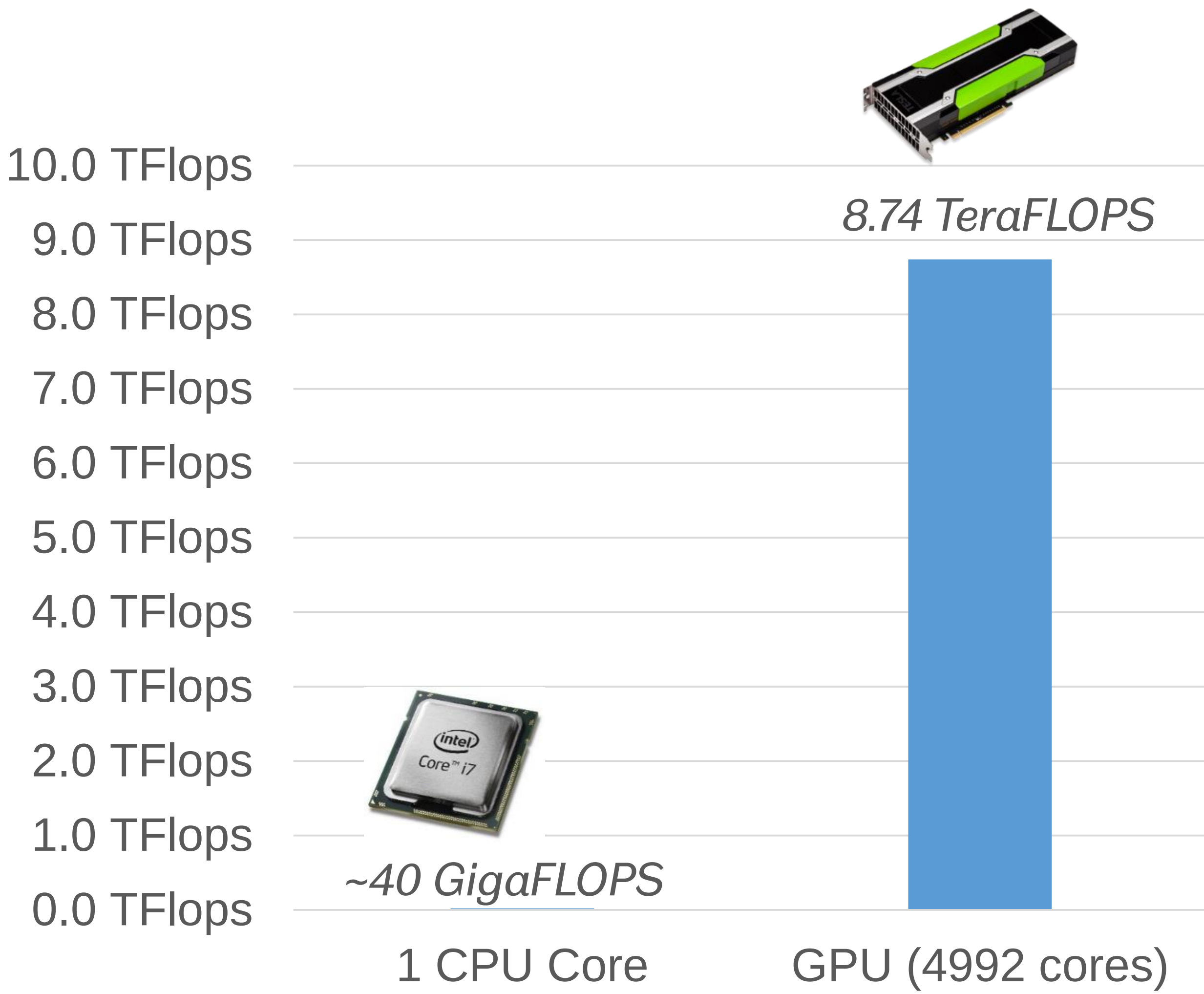
16
cores

Accelerated Computing



8.74
TeraFLOPS

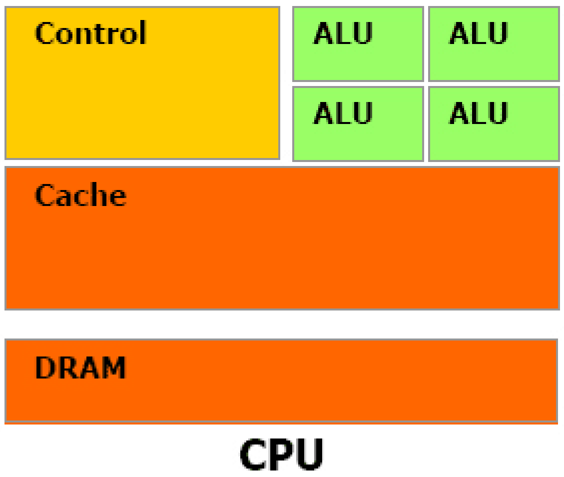
4992 cores

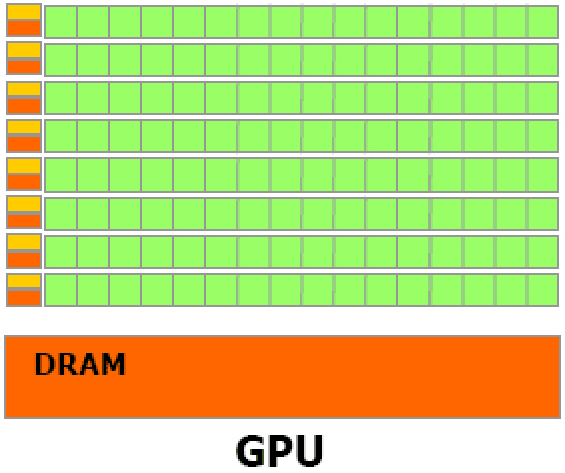


6 hours *CPU* time
vs.
1 minute *GPU* time

Accelerators

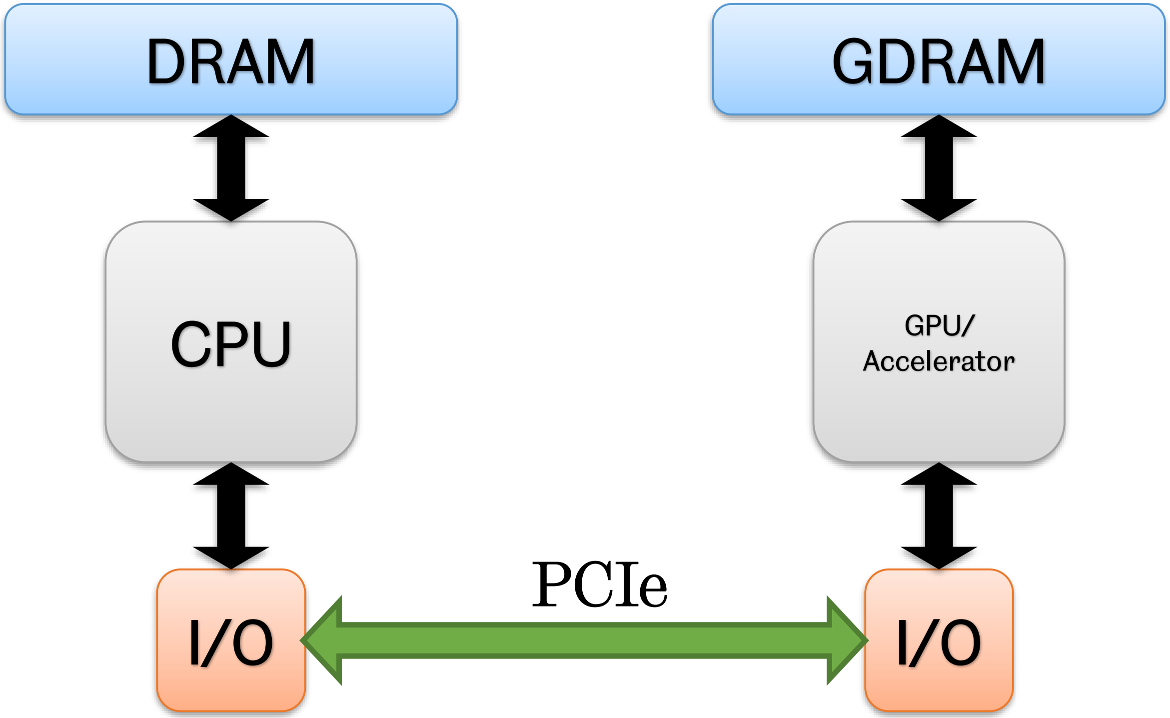
- Much of the functionality of CPUs is unused for HPC
  - Complex Pipelines, Branch prediction, out of order execution, etc.
- Ideally for HPC we want: **Simple**, **Low Power** and **Highly Parallel** cores





The University Of Sheffield.

An accelerated system



- Co-processor not a CPU replacement

Thinking Parallel

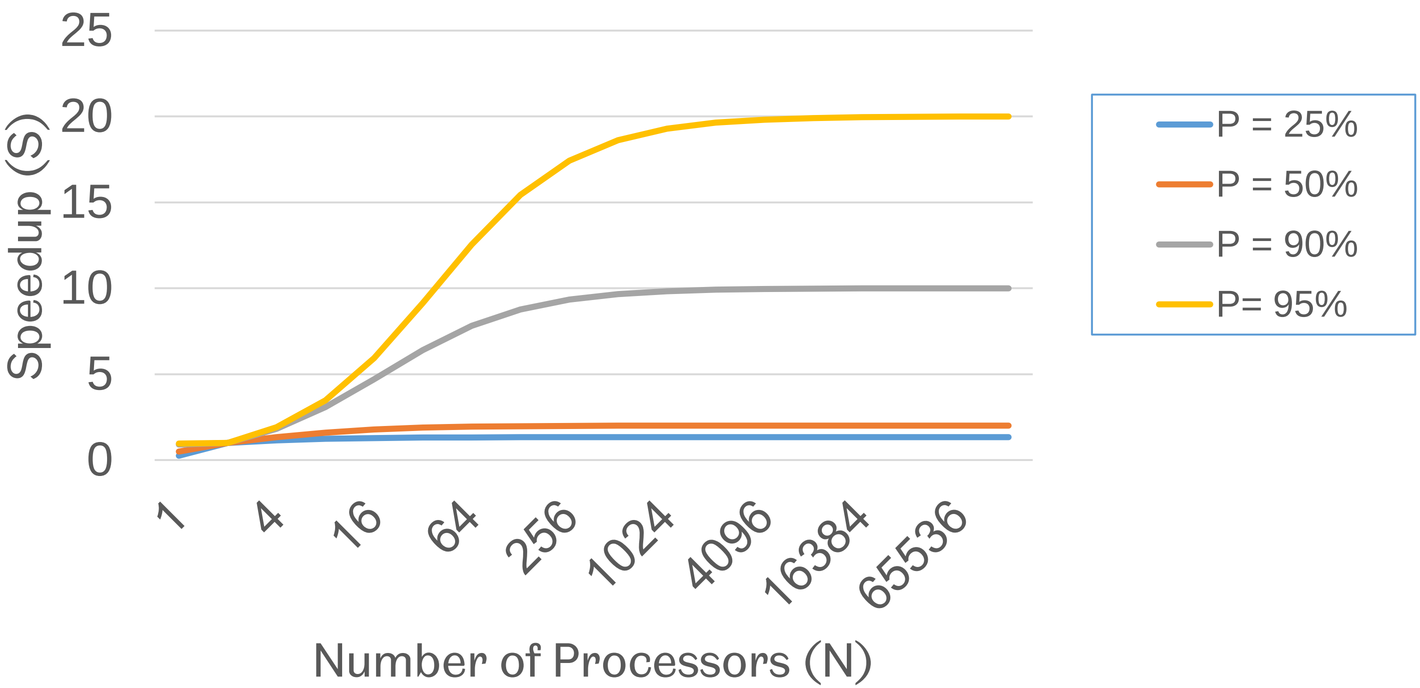
- Hardware considerations
  - High Memory Latency (PCI-e)
  - Huge Numbers of processing cores
- Algorithmic changes required
  - High level of parallelism required
  - Data parallel != task parallel

"If your problem is not parallel then think again"

The University Of Sheffield.

Amdahl's Law

$$Speedup\ (S) = \frac{1}{\frac{P}{N} - (1 - P)}$$



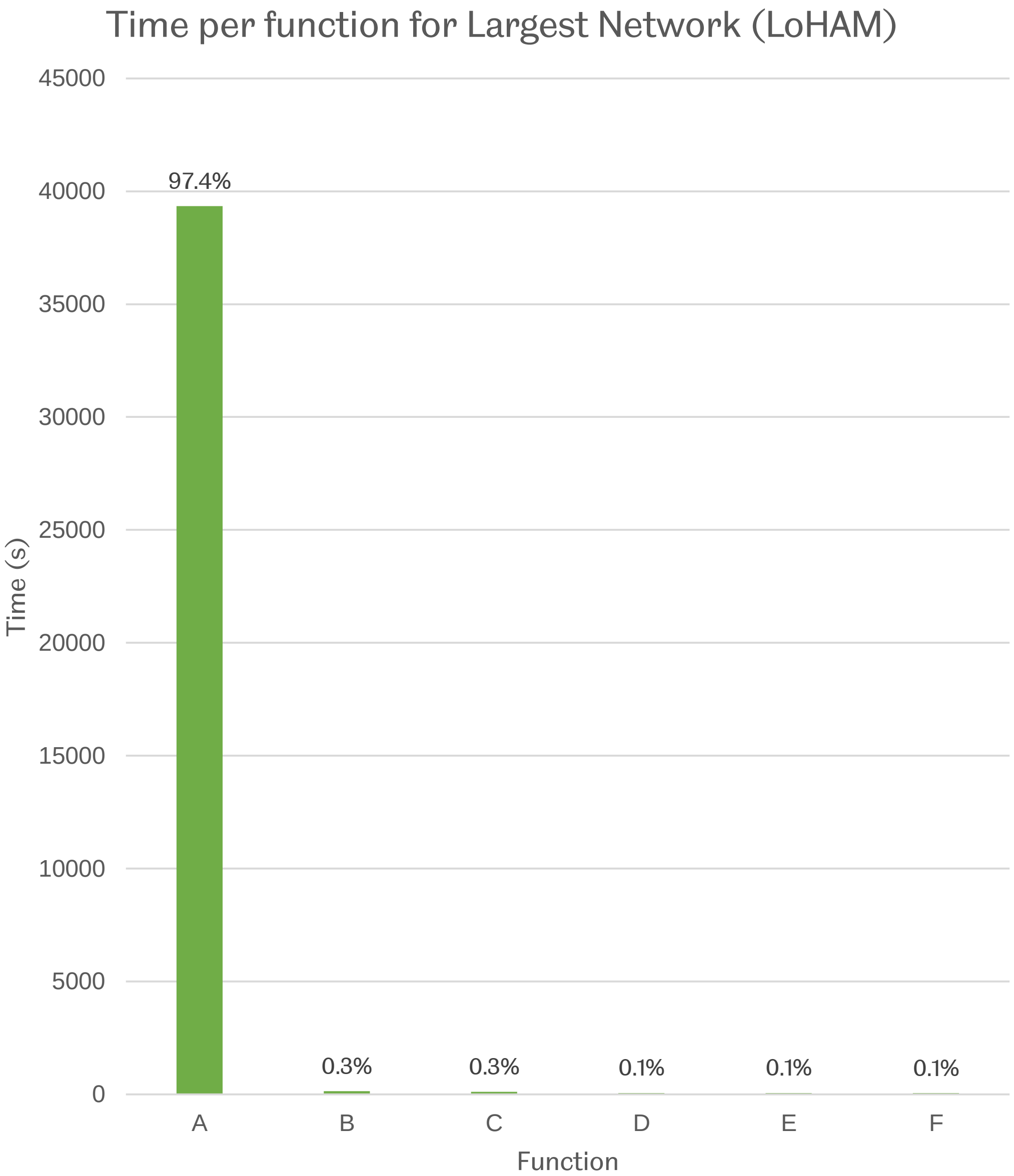
- Speedup of a program is limited by the proportion than can be parallelised
- Addition of processing cores gives diminishing returns

SATALL Optimisation

The University Of Sheffield.

Profile the Application

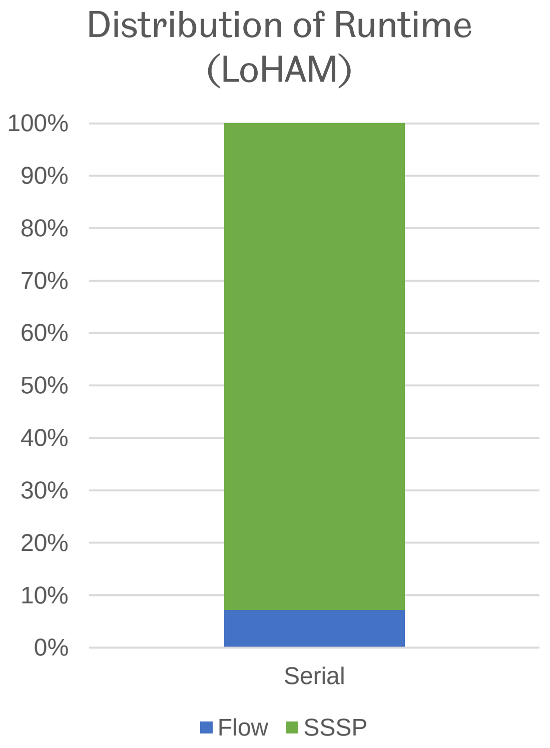
- 11 hour runtime
- Function A
  - 97.4% runtime
  - 2000 calls
- Hardware
  - Intel Core i7-4770k 3.50GHz
  - 16GB DDR3
  - Nvidia GeForce Titan X



The University Of Sheffield.

Function A

- Input
  - Network (directed weighted graph)
  - Origin-Destination Matrix
- Output
  - Traffic flow per edge
- 2 Distinct Steps
  1. **Single Source Shortest Path (SSSP)**
     All-or-Nothing Path
     For each origin in the O-D matrix
  2. **Flow Accumulation**
     Apply the OD value for each trip to each link on the route

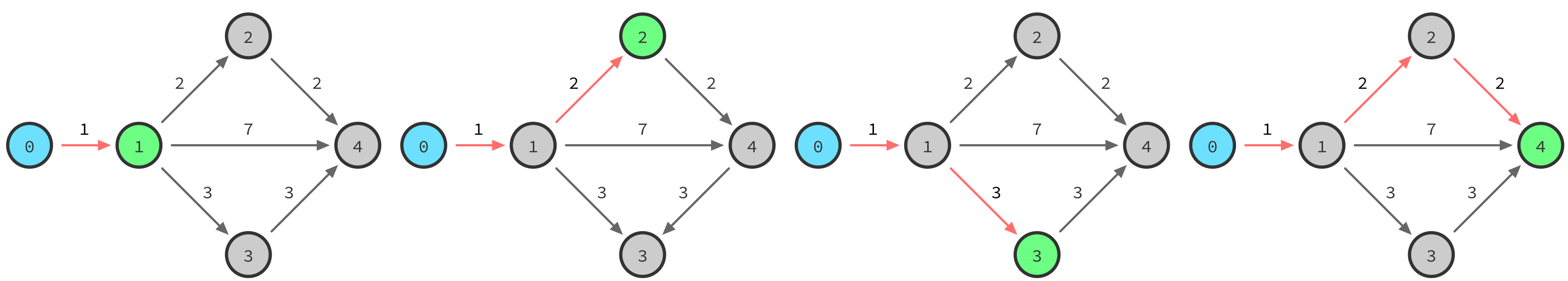


Single Source Shortest Path

For a single **Origin Vertex** (Centroid)

Find the route to each **Destination Vertex**

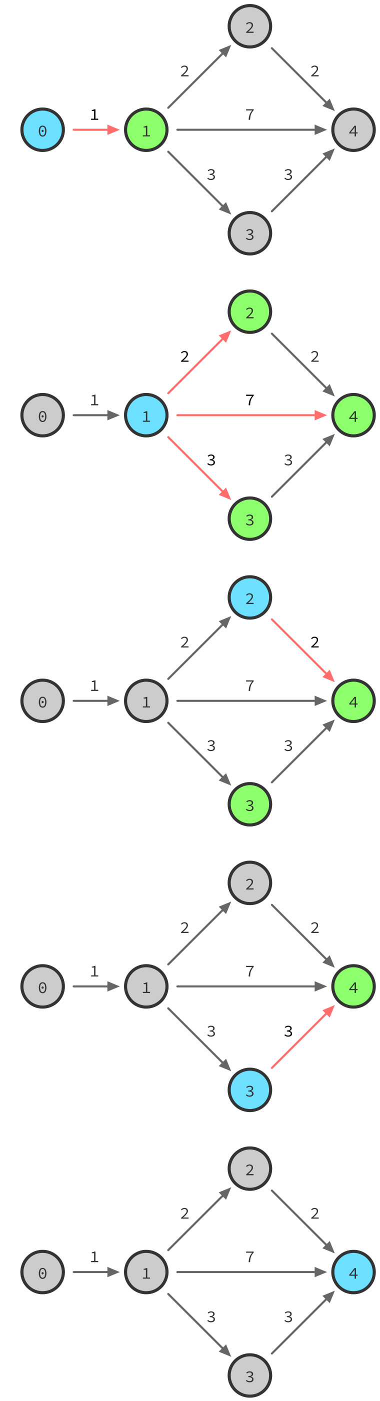
With the **Lowest Cumulative Weight** (Cost)



The University Of Sheffield.

Serial SSSP Algorithm

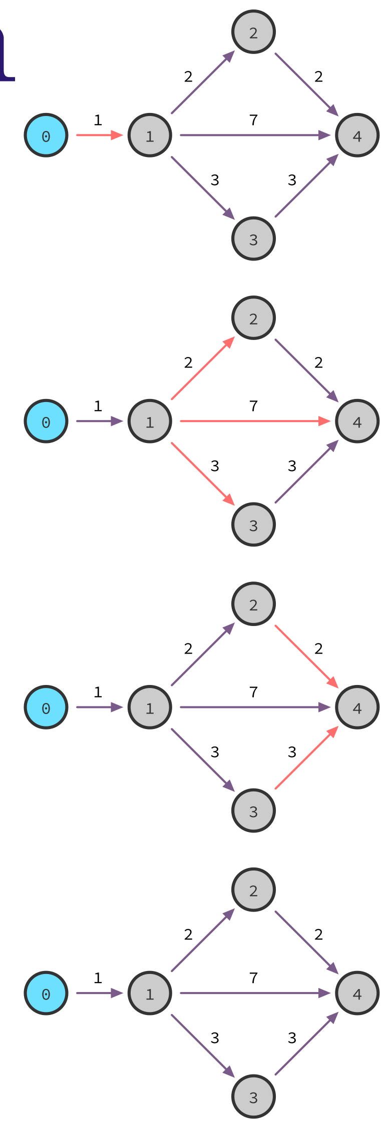
- D'Esopo-Pape (1974)
  - Maintains a **priority queue** of vertices to explore
    Highly Serial
  - Not a **Data-Parallel** Algorithm
- We must change algorithm to match the hardware



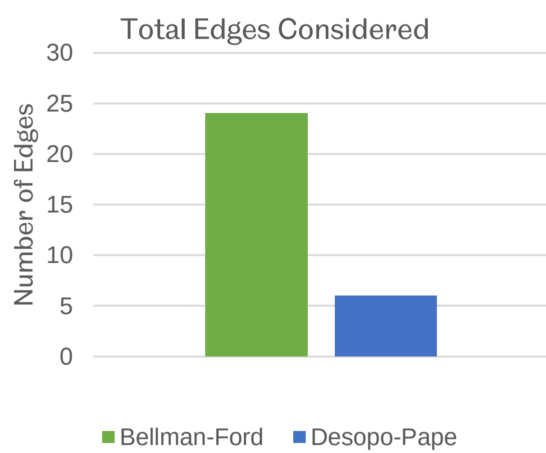
Pape, U. "Implementation and efficiency of Moore-algorithms for the shortest route problem." Mathematical Programming 7.1 (1974): 212-222.

The University Of Sheffield.

Parallel SSSP Algorithm



- Bellman-Ford Algorithm (1956)
    - Poor serial performance & time complexity
    - Performs significantly more work
    - **Highly Parallel**
    - Suitable for GPU acceleration

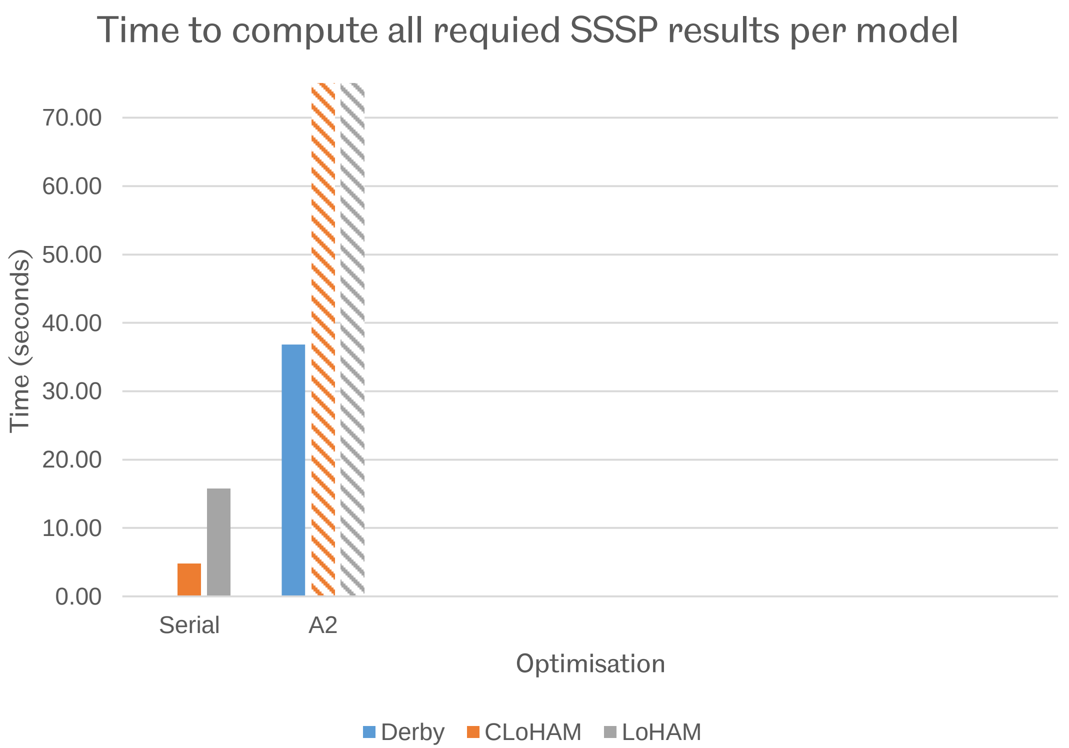


Bellman, Richard. On a routing problem. 1956.

Implementation

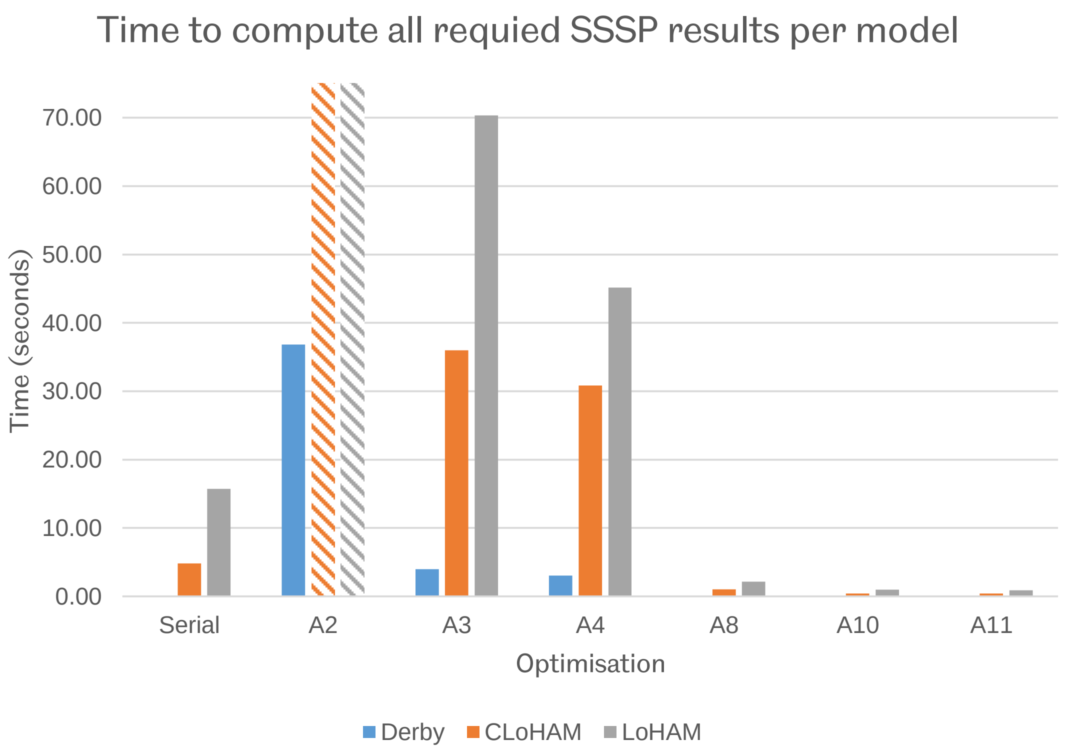
- **A2** - Naïve Bellman-Ford using Cuda
- Up to 369x slower
- Striped bars continue off the scale

Derby	36.5s
CLoHAM	1777.2s
LoHAM	5712.6s



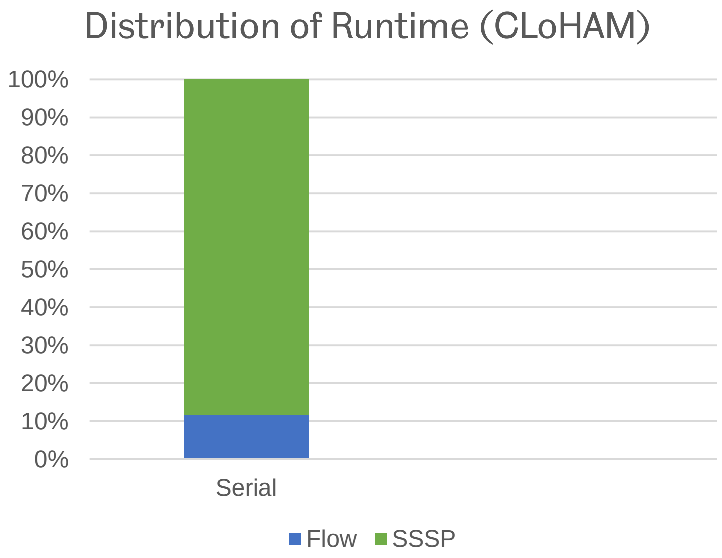
Implementation

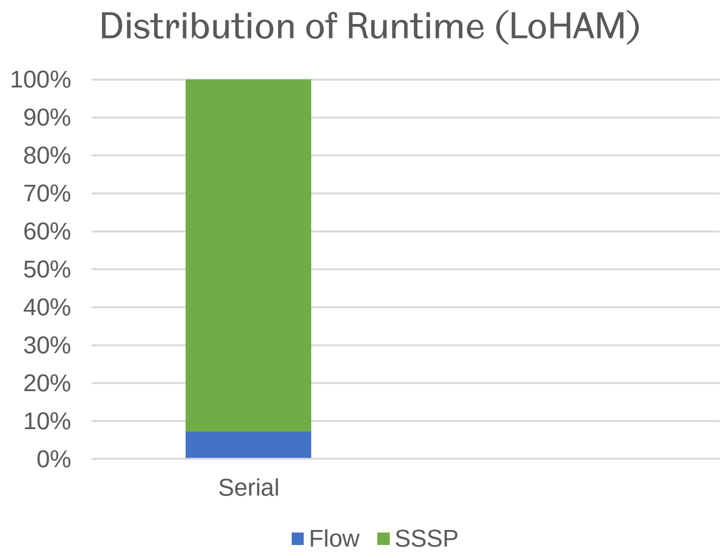
- Followed iterative cycle of performance optimisations
- A3 – Early Termination
- A4 – Node Frontier
- **A8 – Multiple origins Concurrently**
    - SSSP for each Origin in the OD matrix
- A10 – Improved load Balancing
    - Cooperative Thread Array
- A11 – Improved array access



The University Of Sheffield.

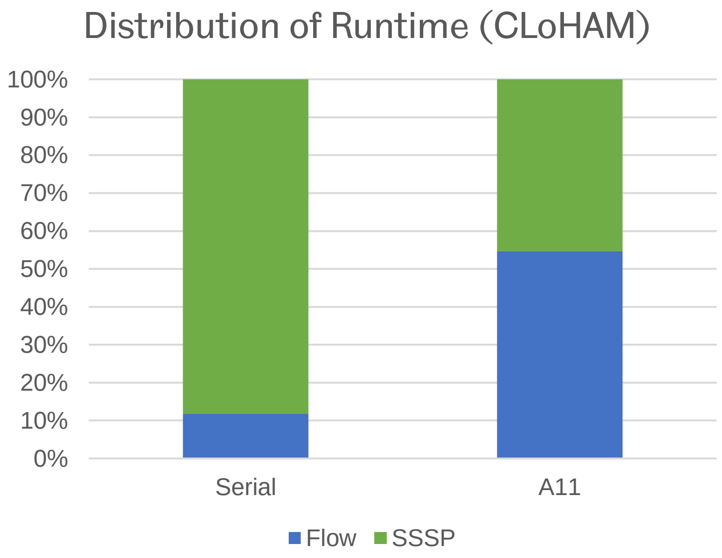
Limiting Factor (Function A)

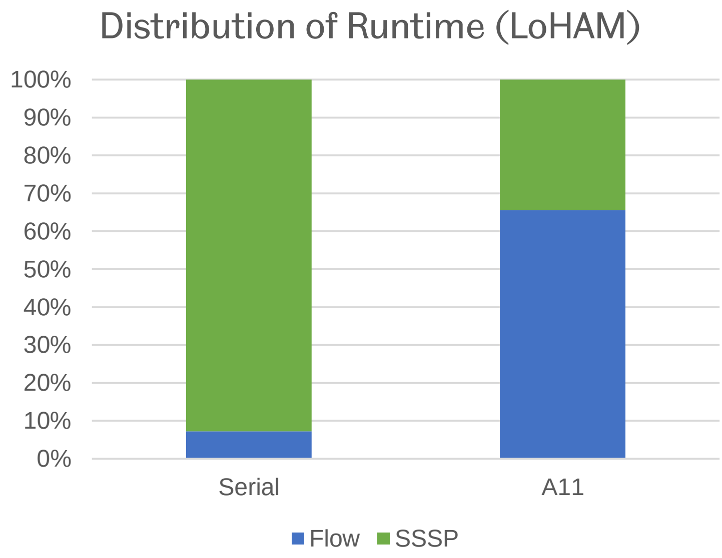




Limiting Factor (Function A)

- Limiting Factor has now changed
- Need to parallelise Flow Accumulation





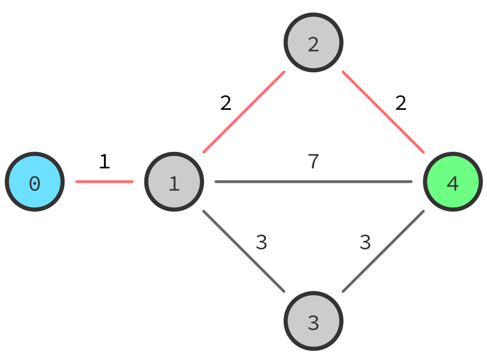
Flow Accumulation

- Shortest Path + OD = Flow-per-link

  For each origin-destination pair

    Trace the route from the destination to the origin increasing the flow value for each link visited

- Parallel problem
  - **But** requires synchronised access to shared data structure for all trips (atomic operations)



+

	0	1	2	...
0	0	2	3	...
1	1	0	4	...
2	2	5	0	...
...	...	...	...	

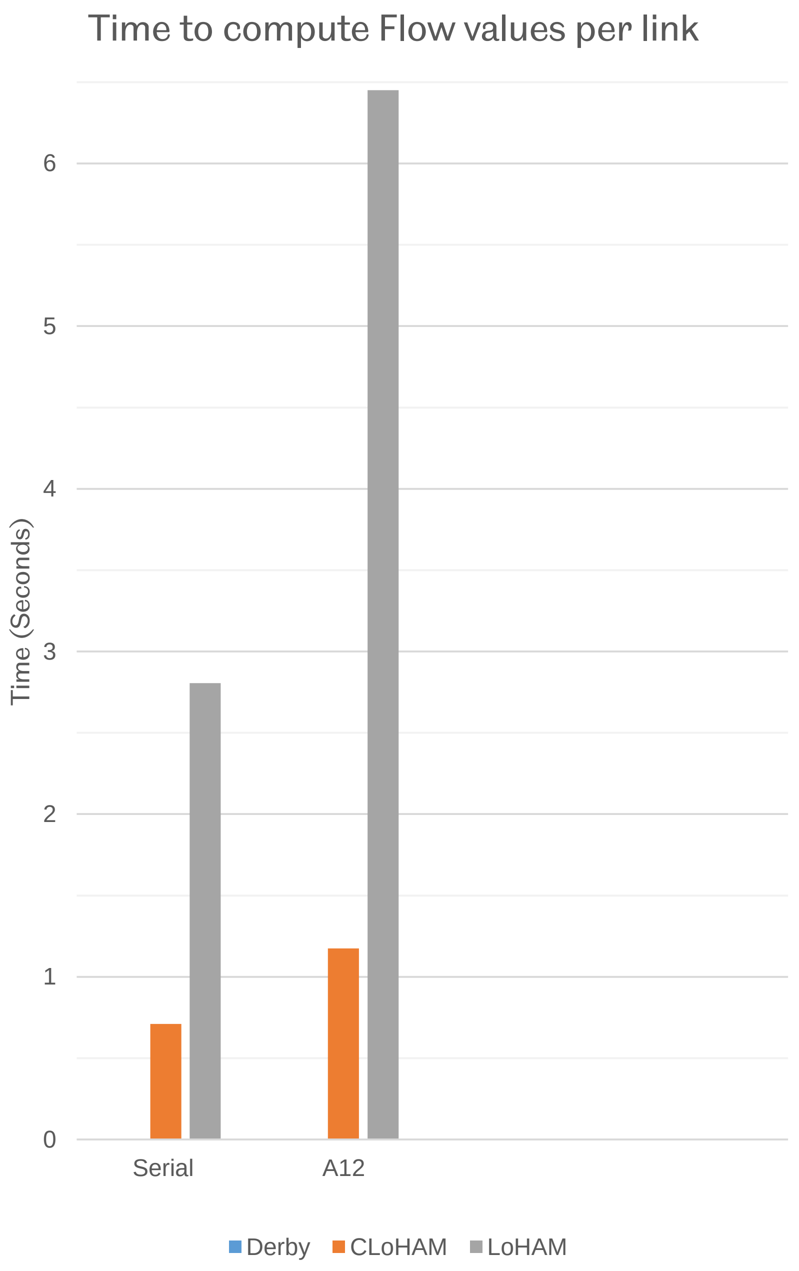
=

Link	0	1	2	3	...
Flow	9	8	6	9	...

The University Of Sheffield.

Flow Accumulation

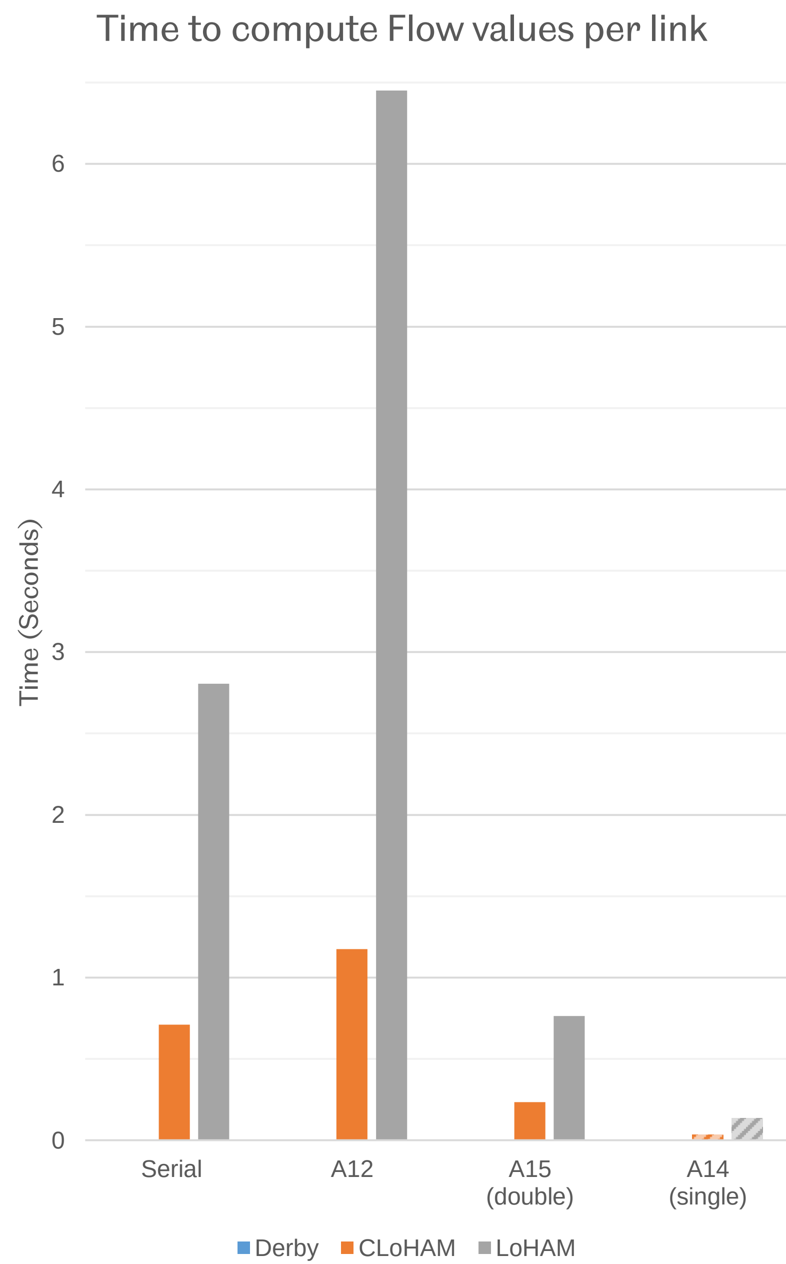
- **Problem:**
  - **A12** - lots of atomic operations serialise the execution



The University Of Sheffield.

Flow Accumulation

- **Problem:**
  - **A12** - lots of atomic operations serialise the execution
- **Solutions:**
  - **A15** - Reduce number of atomic operations
    - Solve in batches using parallel reduction
  - **A14** - Use fast hardware-supported single precision atomics
    - Minimise loss of precision using multiple 32-bit summations
    - 0.000022% total error



Integrated Results

Assignment Speedup relative to Serial

Serial

- LoHAM – 12h 12m

Double precision

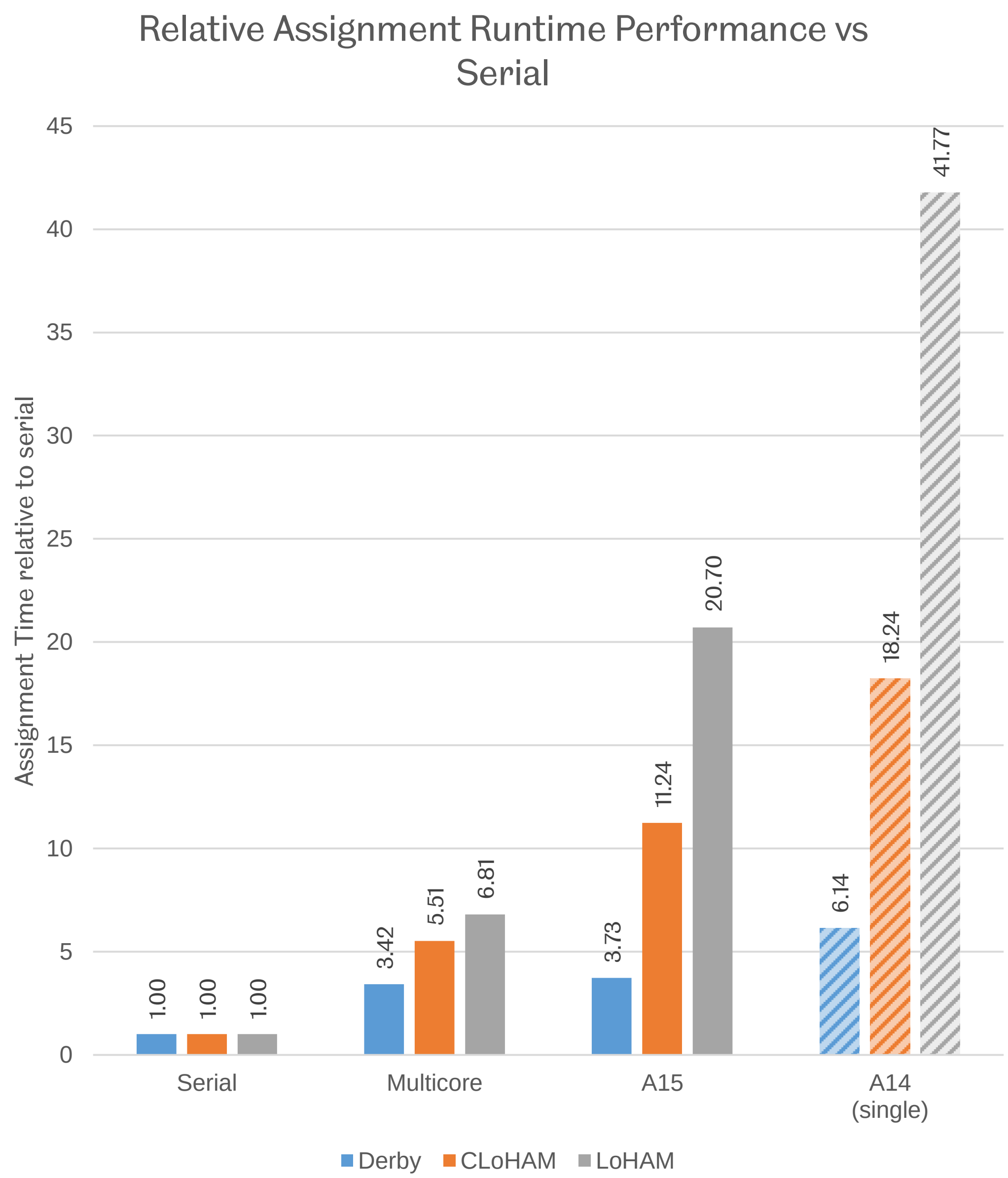
- LoHAM – 35m 22s

Single precision

- Reduced loss of precision
- LoHAM – 17m 32s

Hardware:

- Intel Core i7-4770K
- 16GB DDR3
- Nvidia GeForce Titan X



Assignment Speedup relative to Multicore

Multicore

- LoHAM – 1h 47m

Double precision

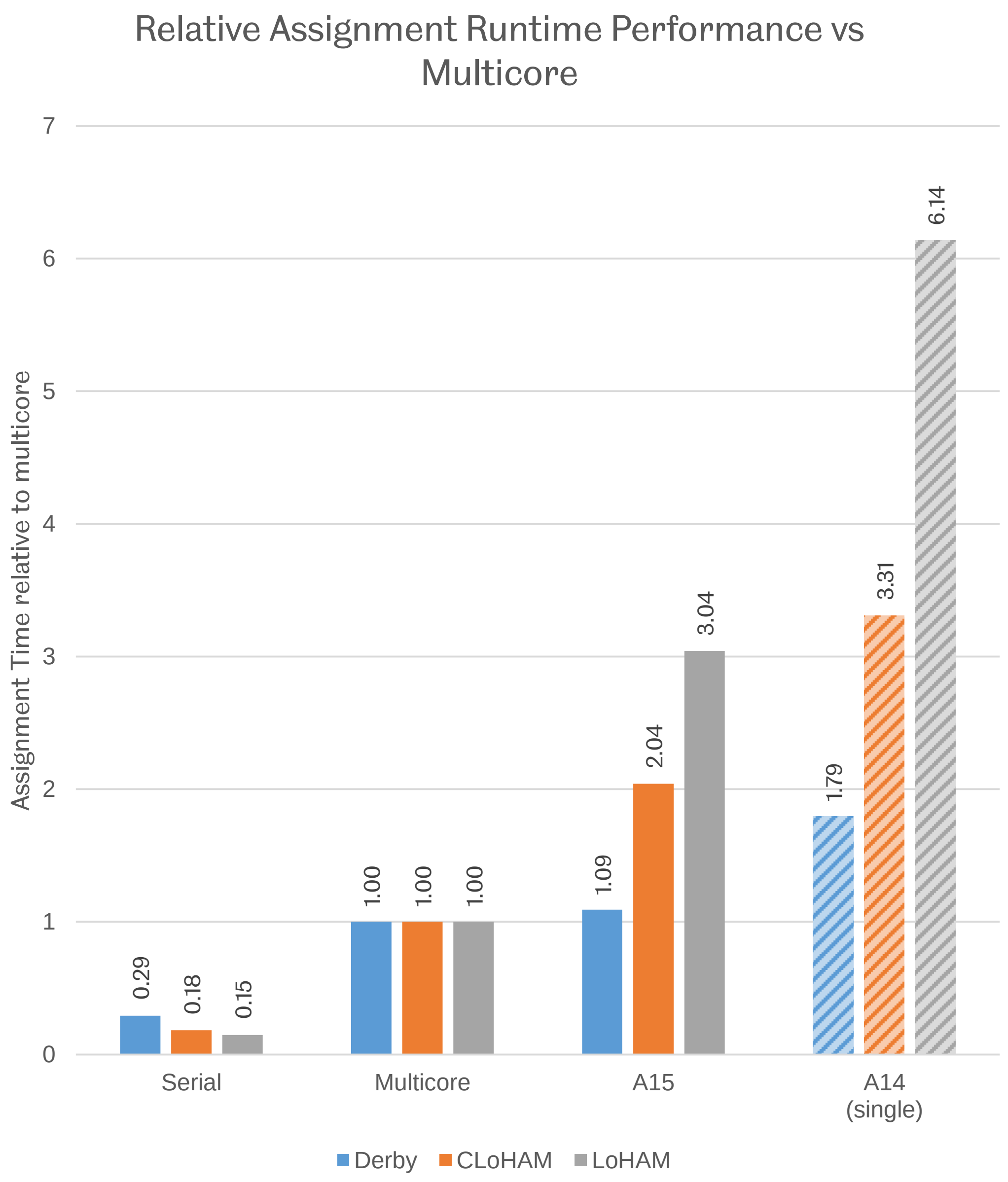
- LoHAM – 35m 22s

Single precision

- Reduced loss of precision
- LoHAM – 17m 32s

Hardware:

- Intel Core i7-4770K
- 16GB DDR3
- Nvidia GeForce Titan X

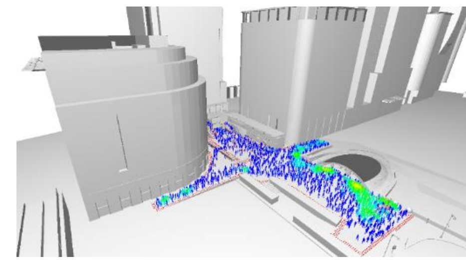


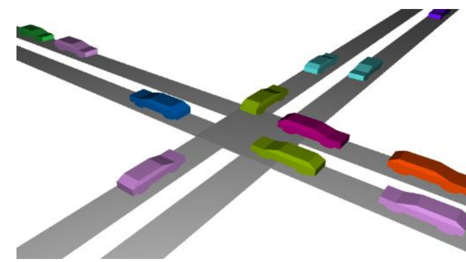
GPU Computing at UoS

The University Of Sheffield.

Expertise at Sheffield

- Specialists in GPU Computing and performance optimisation
- Complex Systems Simulations via FLAME and FLAME GPU
- Visual Simulation, Computer Graphics and Virtual Reality
- Training and Education for GPU Computing





Research Software Engineering Sheffield.

NVIDIA GPU EDUCATION CENTER

NVIDIA GPU RESEARCH CENTER

FLAME GPU

Thank You

Paul Richmond
p.richmond@sheffield.ac.uk
paulrichmond.shef.ac.uk

Peter Heywood
p.heywood@sheffield.ac.uk
ptheywood.uk

Largest Model (LoHAM) results

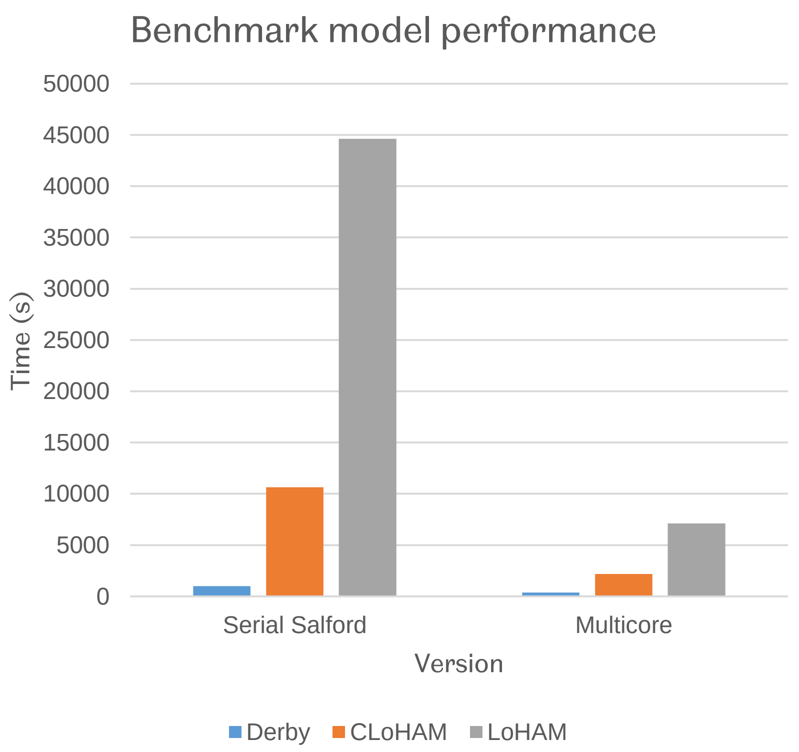
	Runtime	Speedup Serial	Speedup Multicore
Serial	12:12:24	1.00	0.15
Multicore	01:47:36	6.81	1.00
A15 (double precision)	00:35:22	20.70	3.04
A14 (single precision)	00:17:32	**41.77**	**6.14**

Backup Slides

Benchmark Models

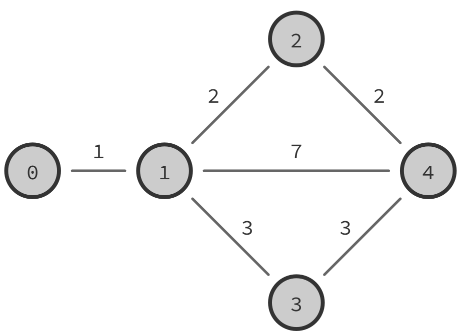
- 3 Benchmark networks
  - Range of sizes
  - Small to V. Large
  - Up to 12 hour runtime

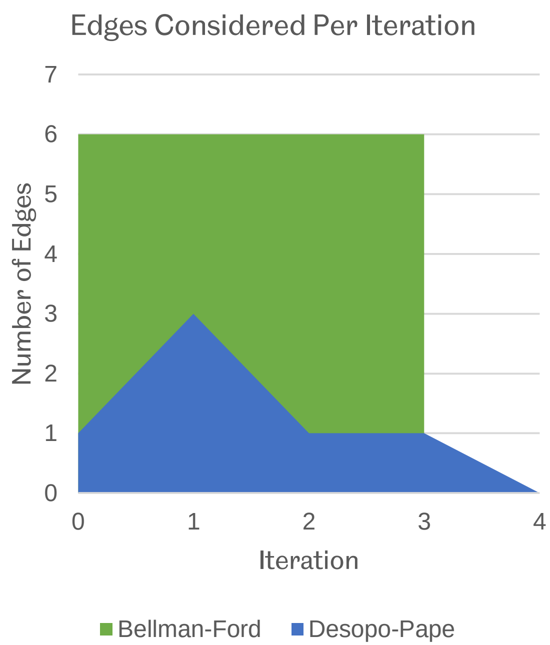
Model	Vertices (Nodes)	Edges (Links)	O-D trips
Derby	2700	25385	547^2
CLoHAM	15179	132600	2548^2
LoHAM	18427	192711	5194^2

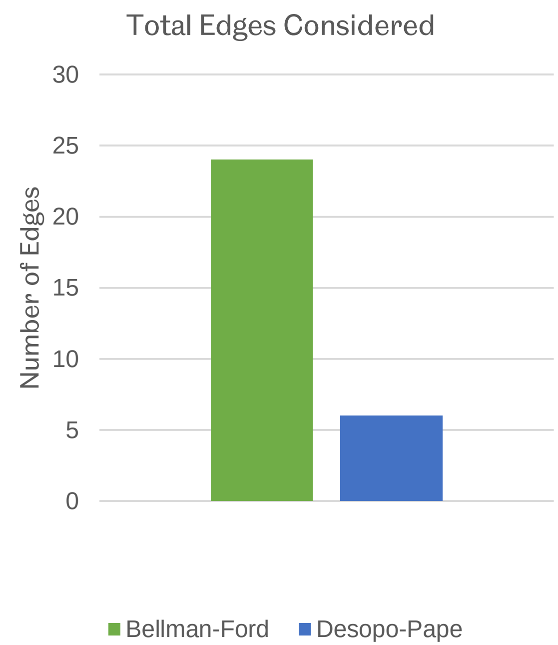


The University Of Sheffield.

Edges considered per algorithm

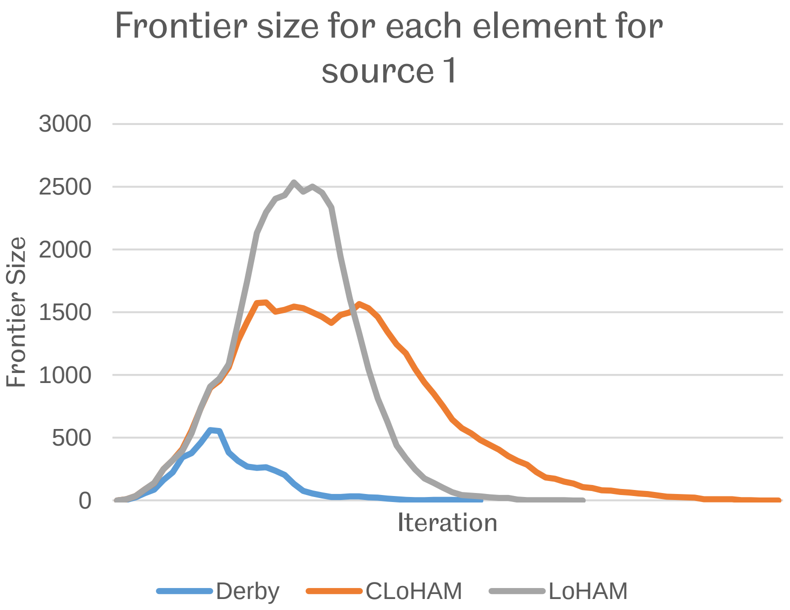


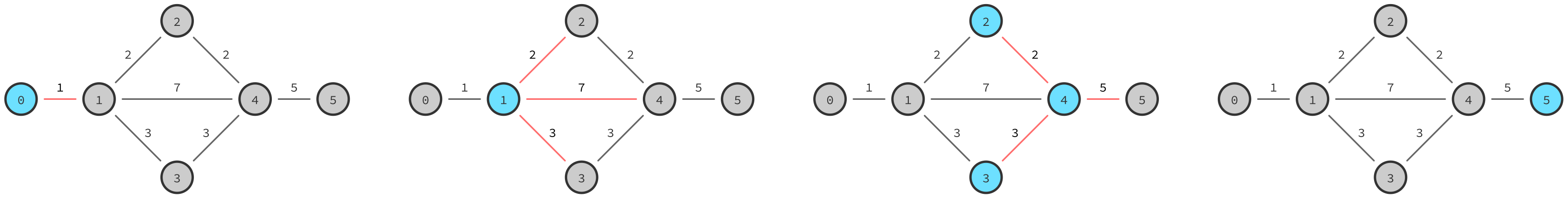




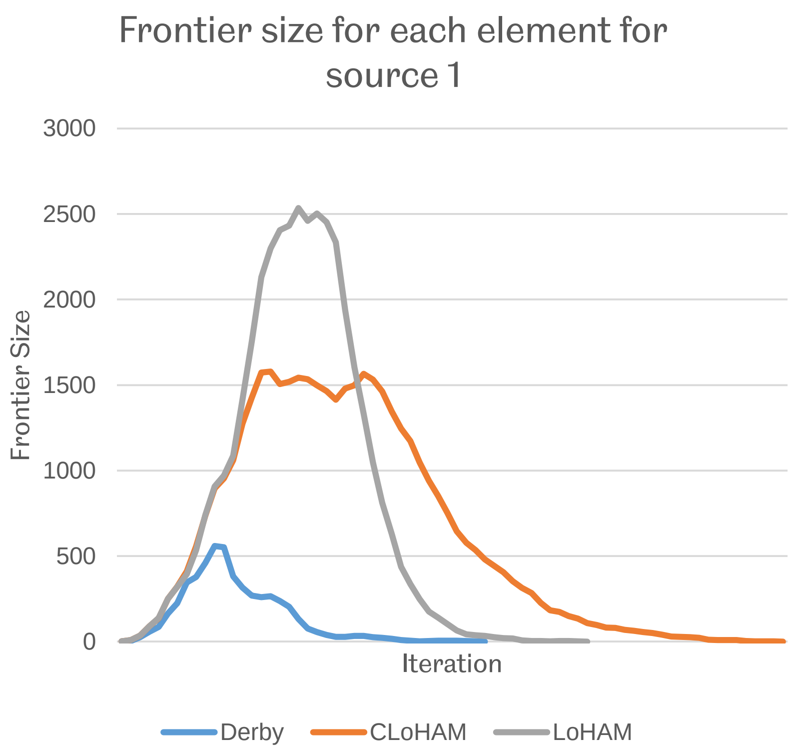
Vertex Frontier (A4)

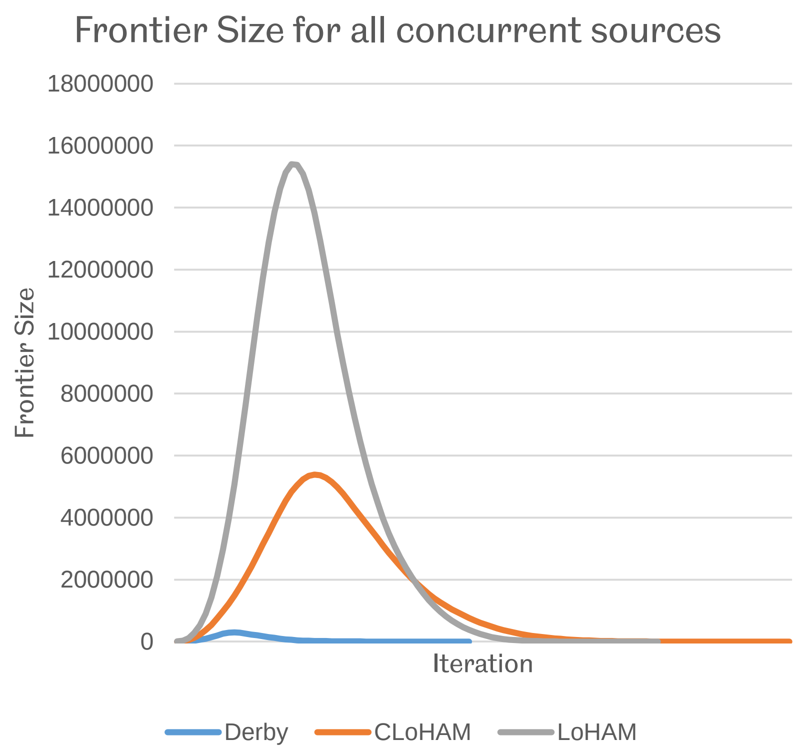
- Only Vertices which were updated in the previous iteration can result in an update
- Much fewer threads launched per iteration
  - Up to 2500 instead of 18427 per iteration





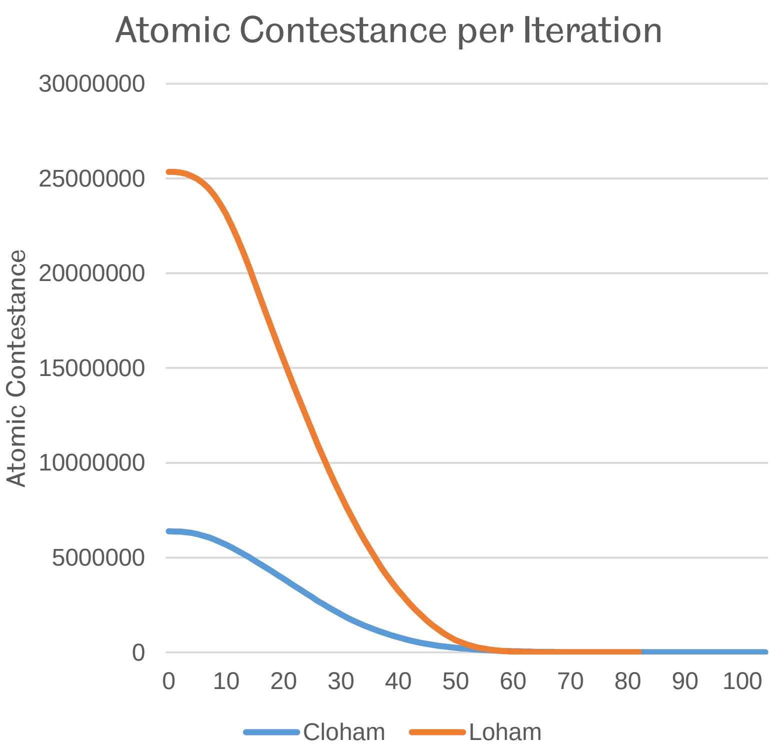
Multiple Concurrent Origins (A8)





Atomic Contention

- Atomic operations are guaranteed to occur
- Atomic Contention
  - multiple threads atomically modify same address
  - Serialised!
- `atomicAdd(double)` not implemented in hardware
  - Not yet
- Solutions
  1. Algorithmic change to minimise atomic contention
  2. Single precision



Raw Performance

Hardware:

- Intel Core i7-4770K
- 16GB DDR3
- Nvidia GeForce Titan X