S4599: An Adventure in Porting:

Adding GPU-Acceleration to Open-Source 3D Elastic Wave Modeling

Robin M. Weiss and Jeffery Shragge

March 26, 2014

GPU Technology Conference

THE UNIVERSITY OF CHICAGO | Research Computing Center

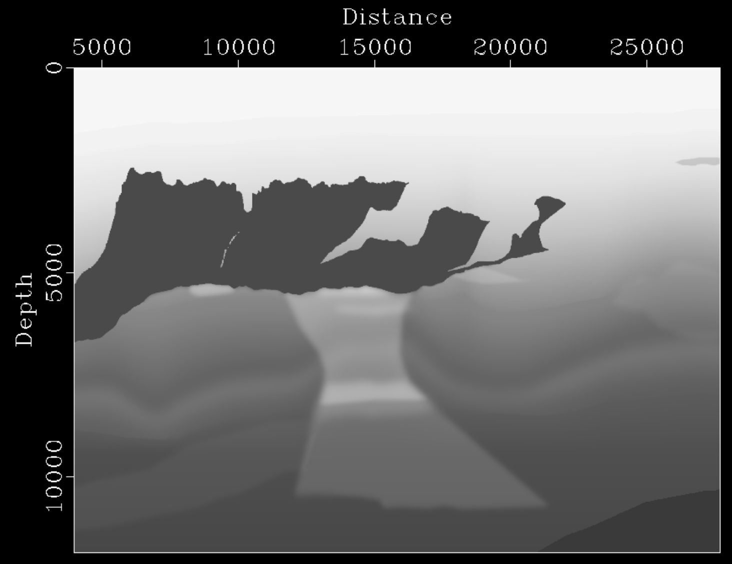
CPGCO2

Centre for Petroleum Geoscience & CO2 Sequestration

The University of Western Australia

Seismic Data Modeling

- Elastic modeling
  - Anisotropic and heterogeneous materials
- Seismic imaging with elastic models
  - Increasing interest in this area
  - Solved as an inverse problem
  - Requires accurate, fast forward solvers
- Computational issues
  - Large model sizes
  - Generally requires a ton of compute power



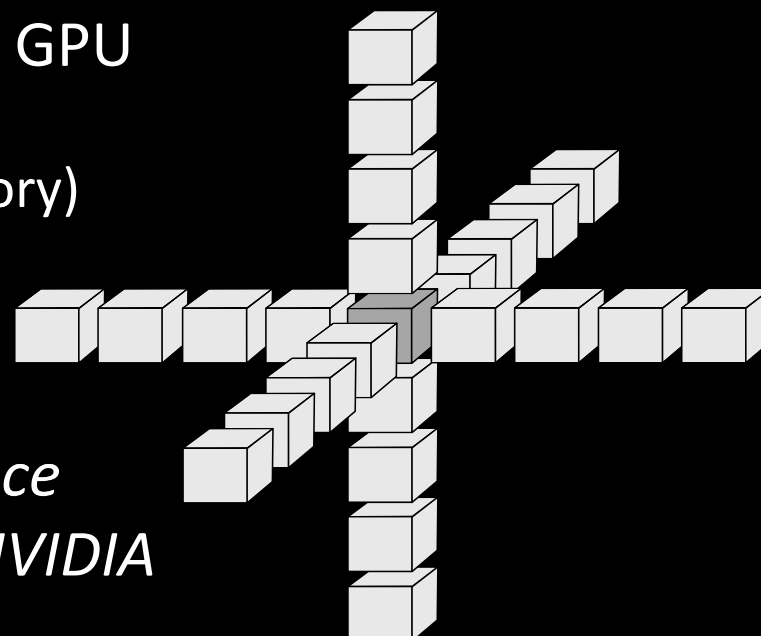
Open-Source 3D Elastic Modeling

- Open-source computational geophysics package
  - Madagascar – www.reproducibility.org (RSFSRC/user/rweiss)
  - Targets reproducibility, transparency, and clarity
- Madagascar's CPU-based (w/ OpenMP) EWE solver implementation:
  - Paul Sava, Colorado School of Mines
  - Anisotropic elastic wave equation
  - Stress-Stiffness formulation
  - 3D Finite-difference solver, centered derivatives, regular grid
    - 8^{th} order spatial, 2^{nd} order time

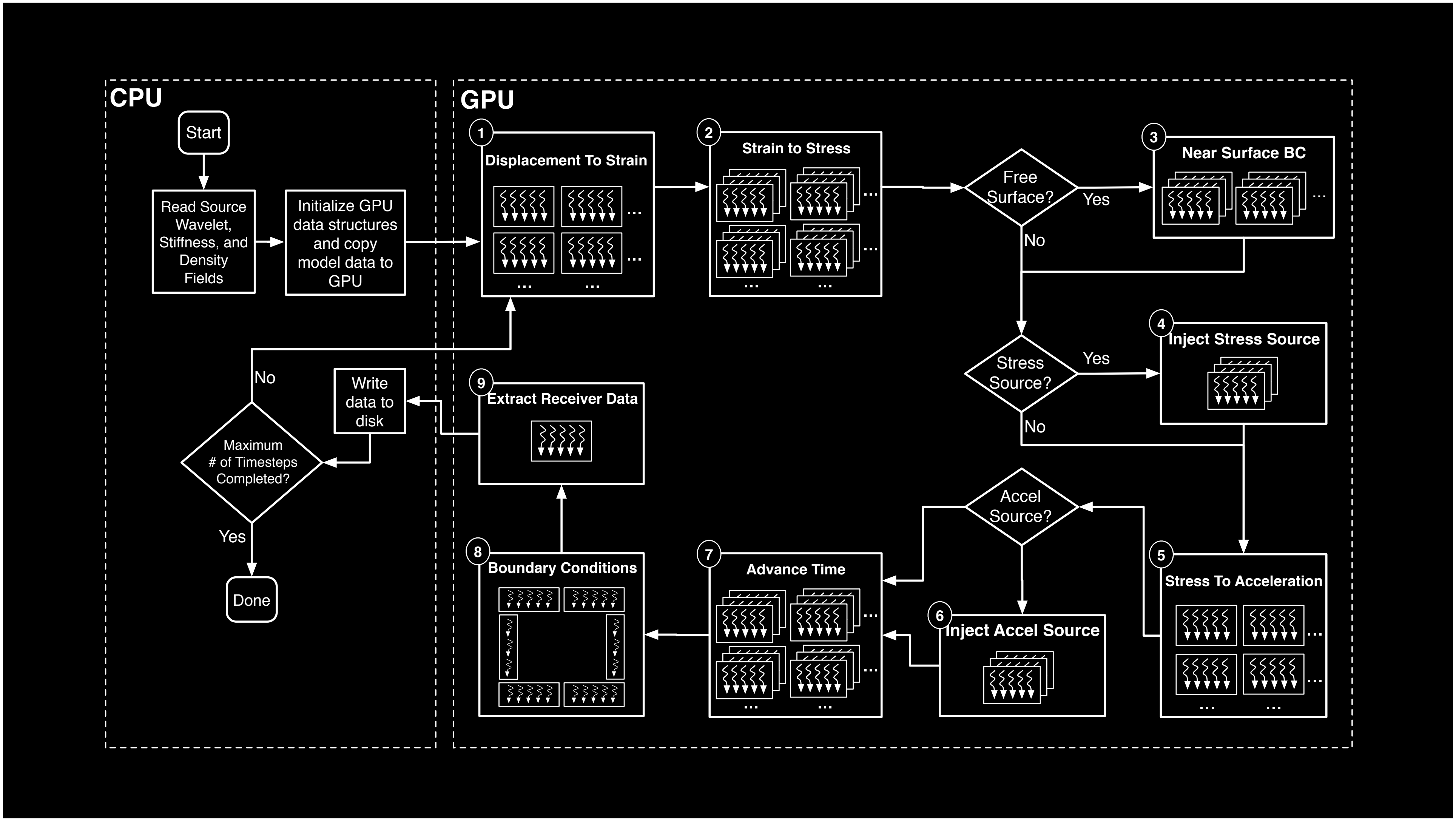
Weiss and Shragge, 2013, *Solving 3D anisotropic elastic wave equations on parallel GPU devices, Geophysics*

Finite-Difference and GPUs

- Represents the bulk of the compute work
- Compact FD stencils map nicely to GPU
  - Regular memory access pattern
  - Some data reuse (use shared memory)
  - But still memory bound
    - GTC 2013 - S3176



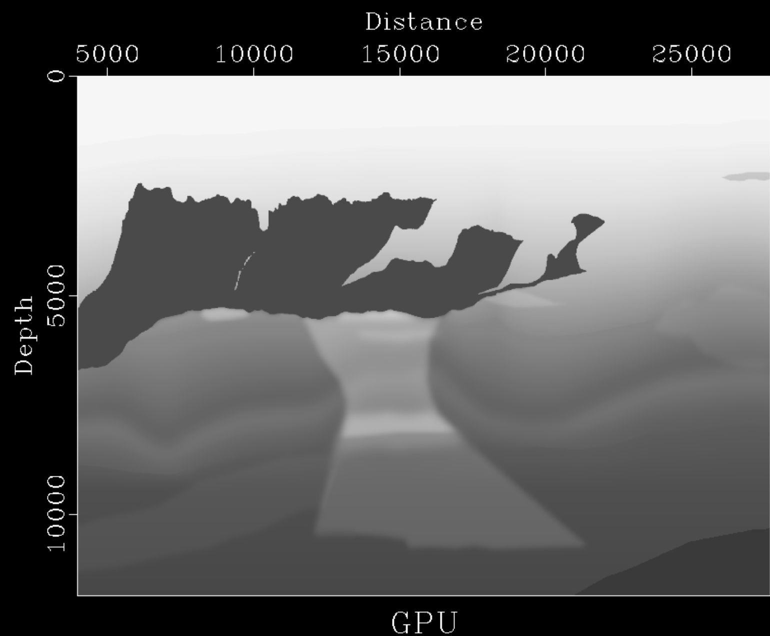
Micikevicius, 2009, *3D Finite Difference Computation on GPUs using CUDA, NVIDIA*

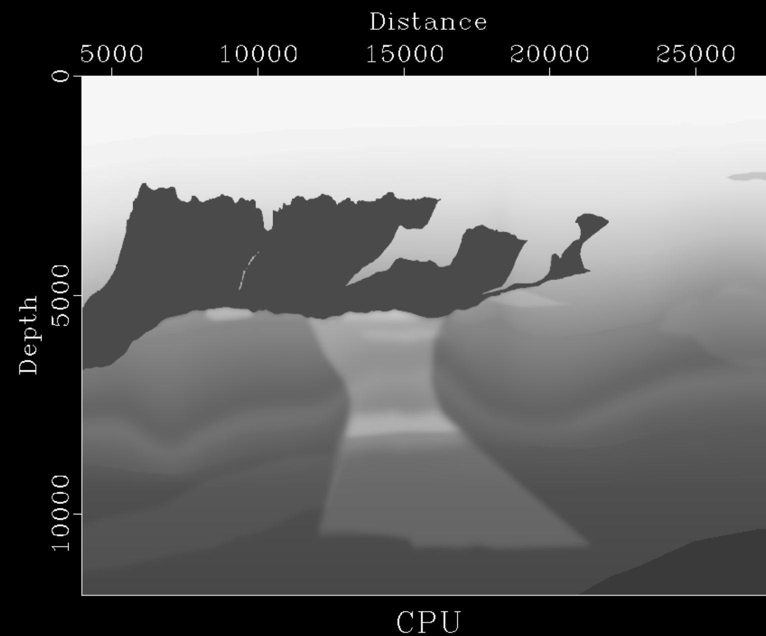

CPU
Start
Read Source Wavelet, Stiffness, and Density Fields
Initialize GPU data structures and copy model data to GPU
Write data to disk
Maximum # of Timesteps Completed?
No
Yes
Done
GPU
1
Displacement To Strain
2
Strain to Stress
Free Surface?
Yes
No
3
Near Surface BC
Stress Source?
Yes
No
4
Inject Stress Source
5
Stress To Acceleration
Accel Source?
6
Inject Accel Source
7
Advance Time
8
Boundary Conditions
9
Extract Receiver Data

Optimize: The usual suspects…

- Stash data in shared memory and/or registers
- Find good block dimensions
- Repeated expression replacement
- Kernel fusing
  - Keep an eye on register usage (`-Xptxas -v`)
- CUDA Visual Profiler is your friend

GPU-CPU Displacement Comparison



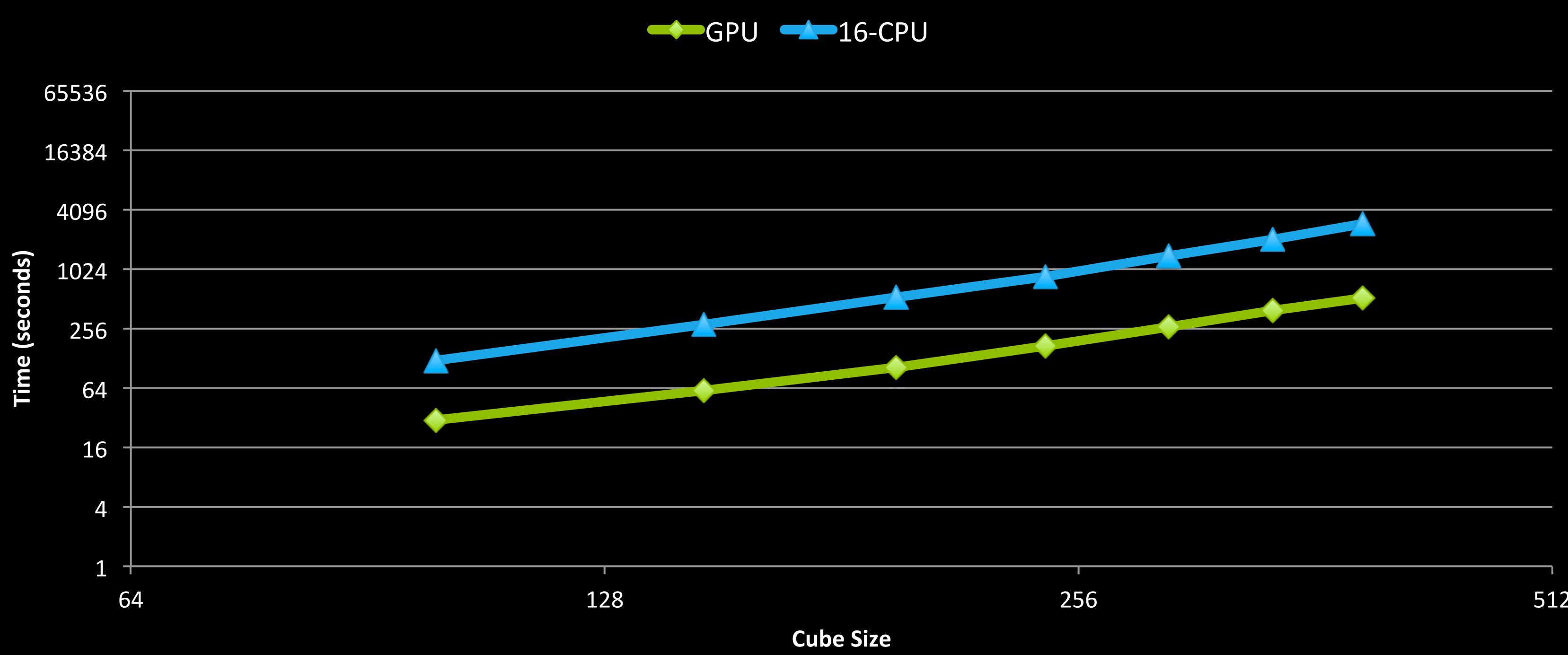


depth-component of displacement field overlaid on velocity model

Off to the races…

- All experiments run on N^3 data sets for 2000 timesteps
- CPU results computed with Intel Sandy Bridge E5-2670 2.60GHz
  - 2x 8-core, IBM iDataPlex dx360 M4
  - University of Chicago's Midway Compute Cluster
- GPU results computed with Nvidia K40
  - 4x GPUs per node
  - Nvidia PSG cluster – thanks, Nvidia!

Off to the races…

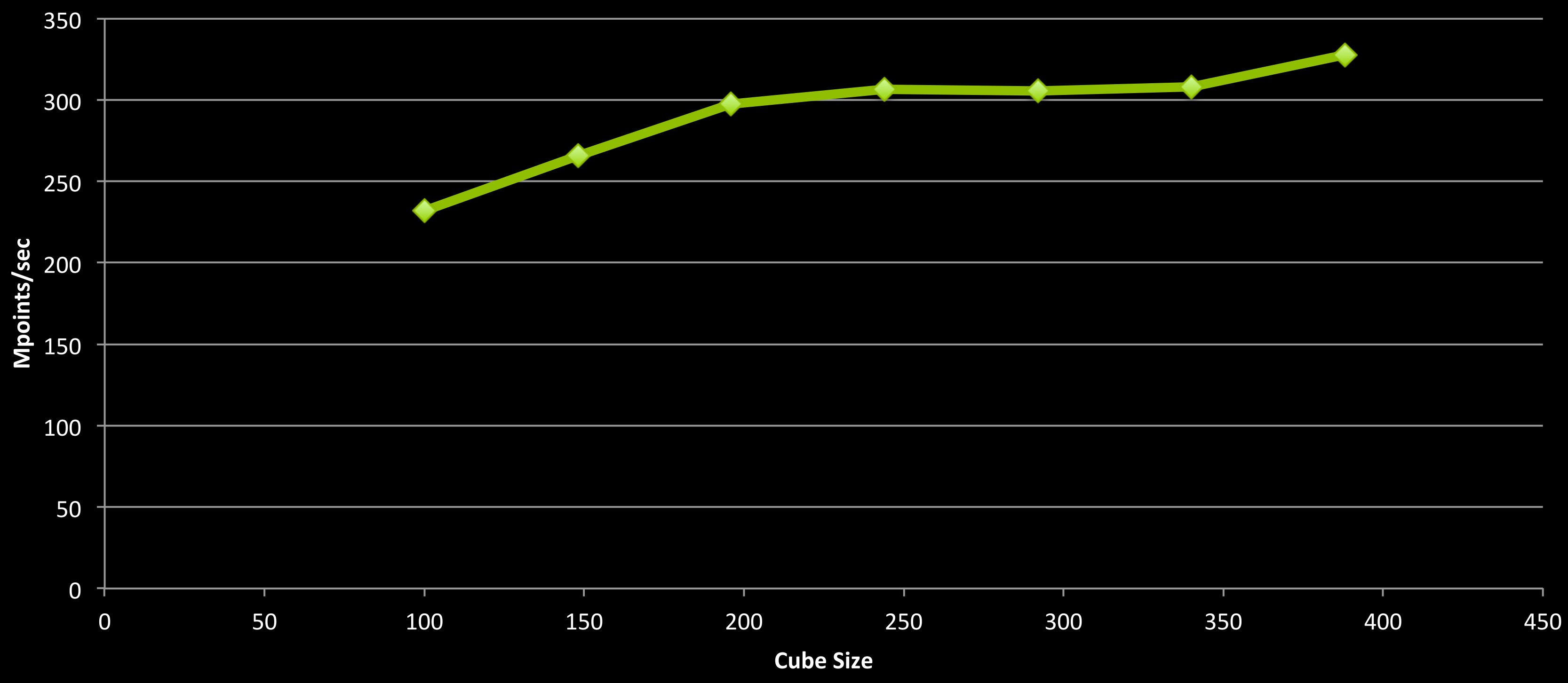


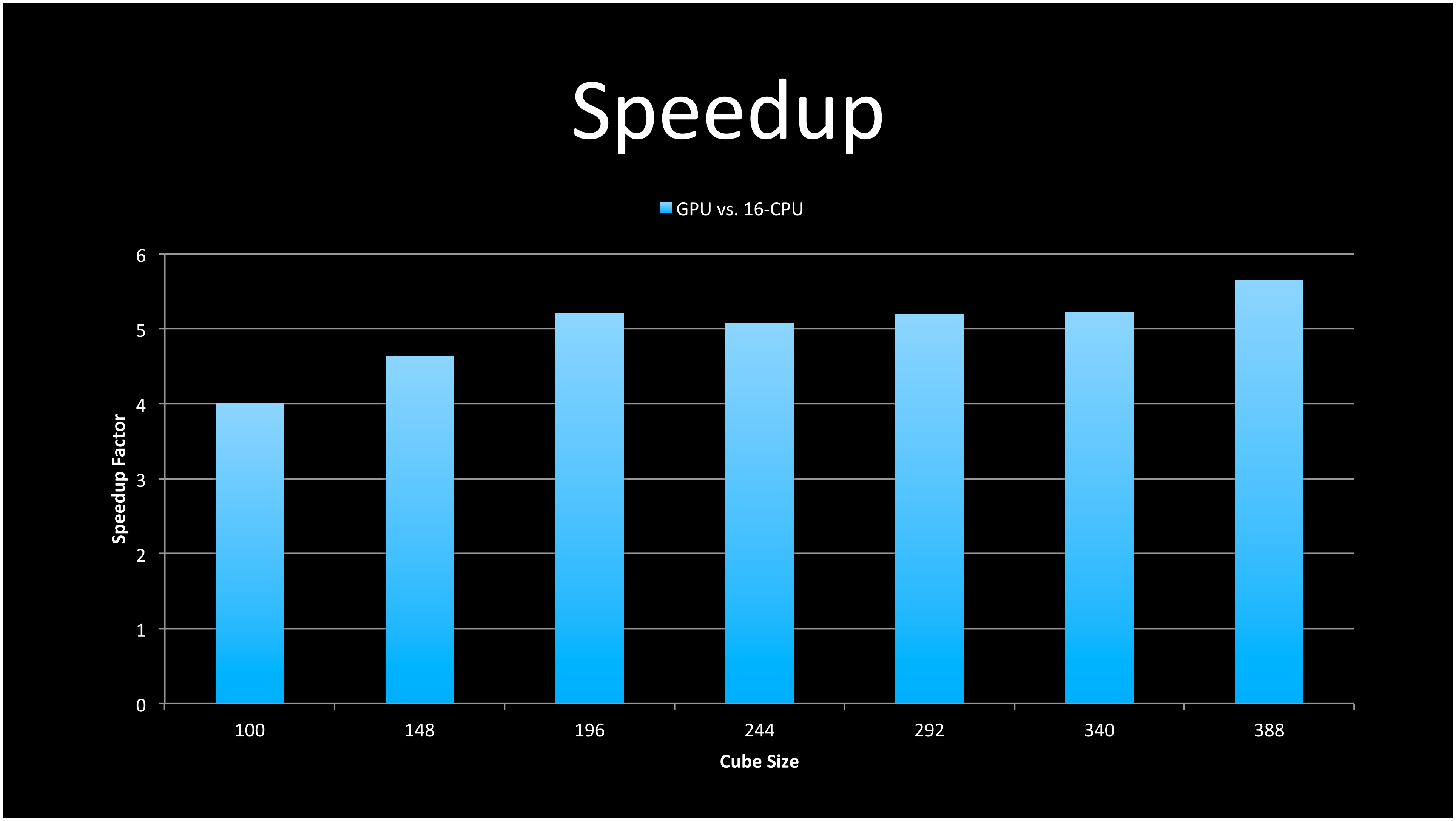
Off to the races…

Throughput (Mpoints/sec)

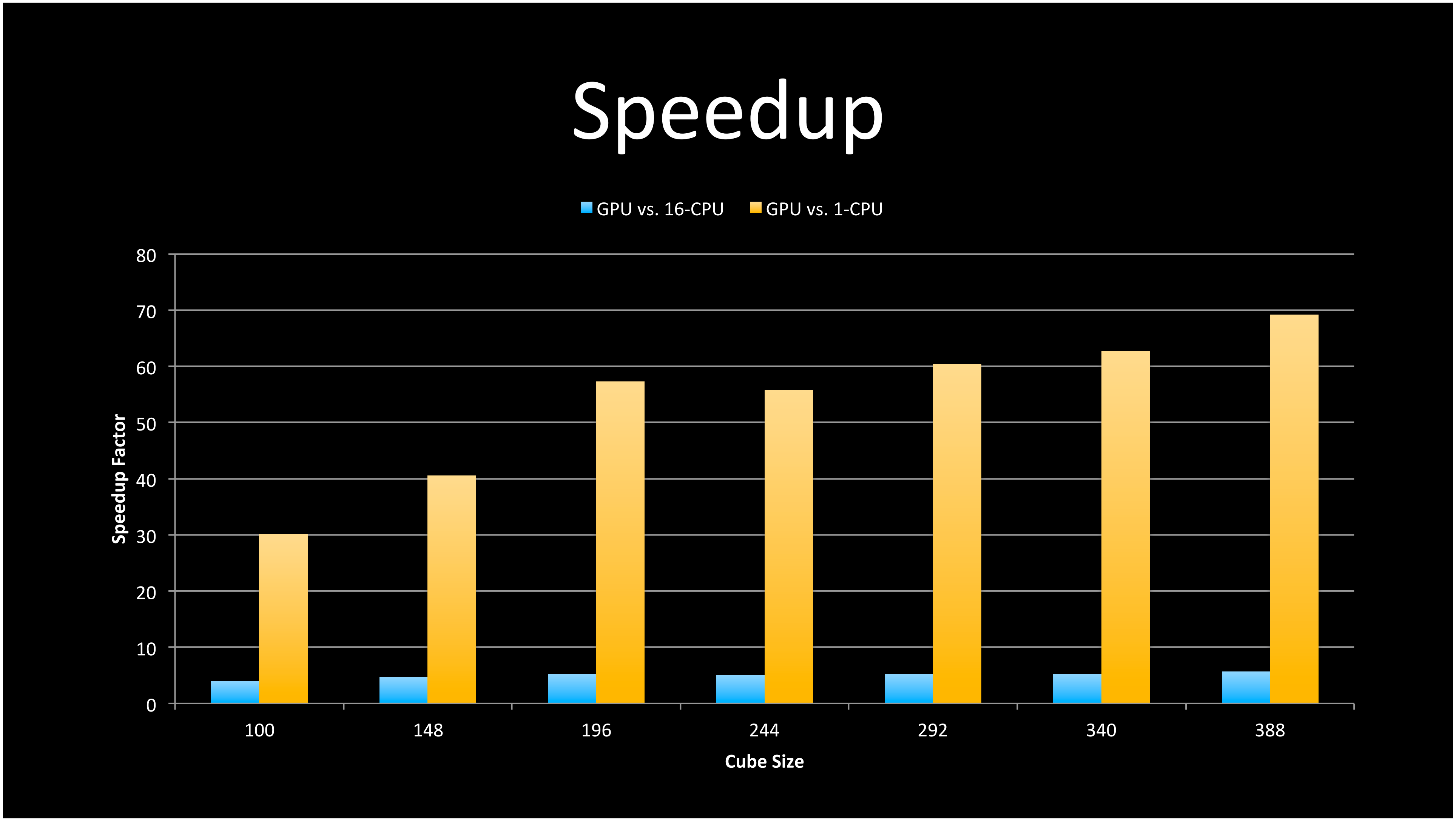
Cube Size	**Disp. To Strain**	Strain to Stress	**Stress to Accel.**	Advance Disp.
100	**1405**	1657	**1064**	2976
148	**1333**	1702	**1053**	3077
196	**1467**	1714	**1164**	3051
244	**1449**	1662	**1054**	3051
292	**1404**	1703	**993**	2950
340	**1282**	1721	**1021**	3170

Overall Throughput



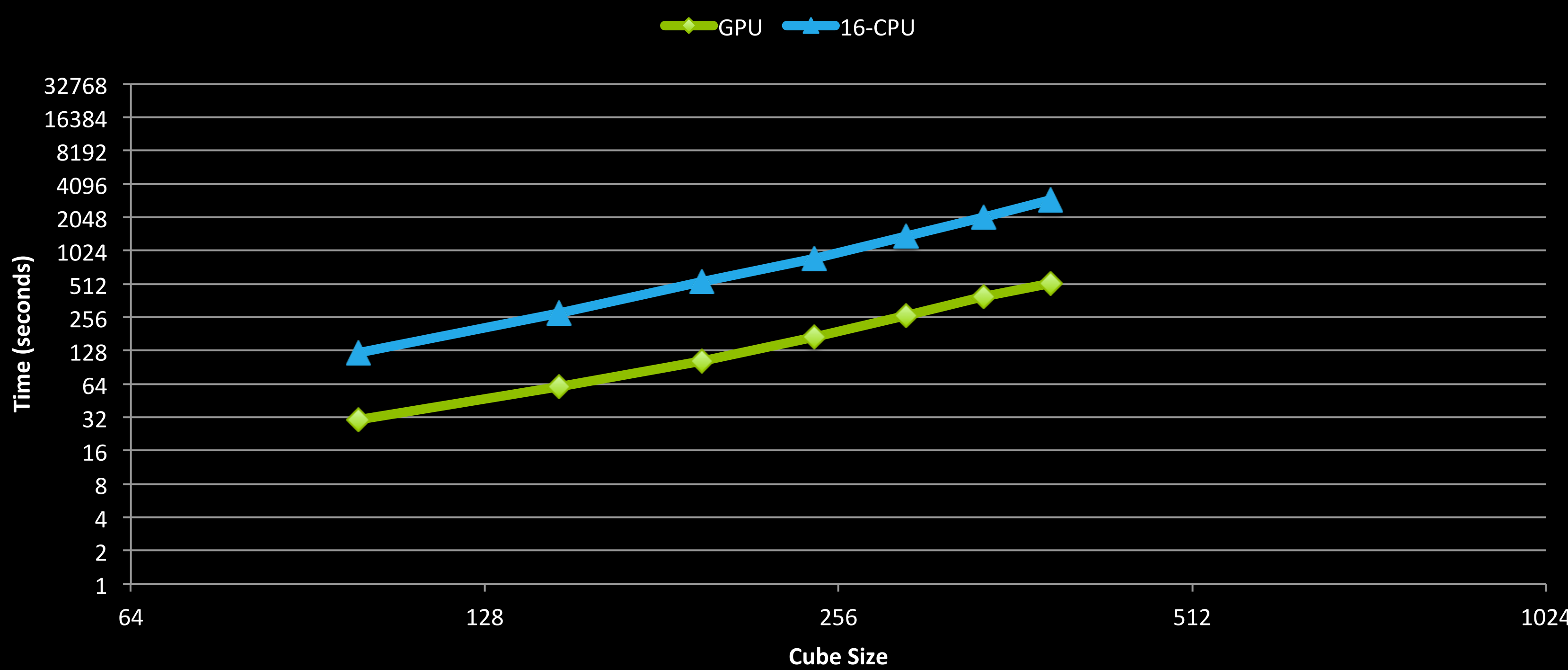


Speedup
GPU vs. 16-CPU
Speedup Factor
0
1
2
3
4
5
6
100
148
196
244
292
340
388
Cube Size

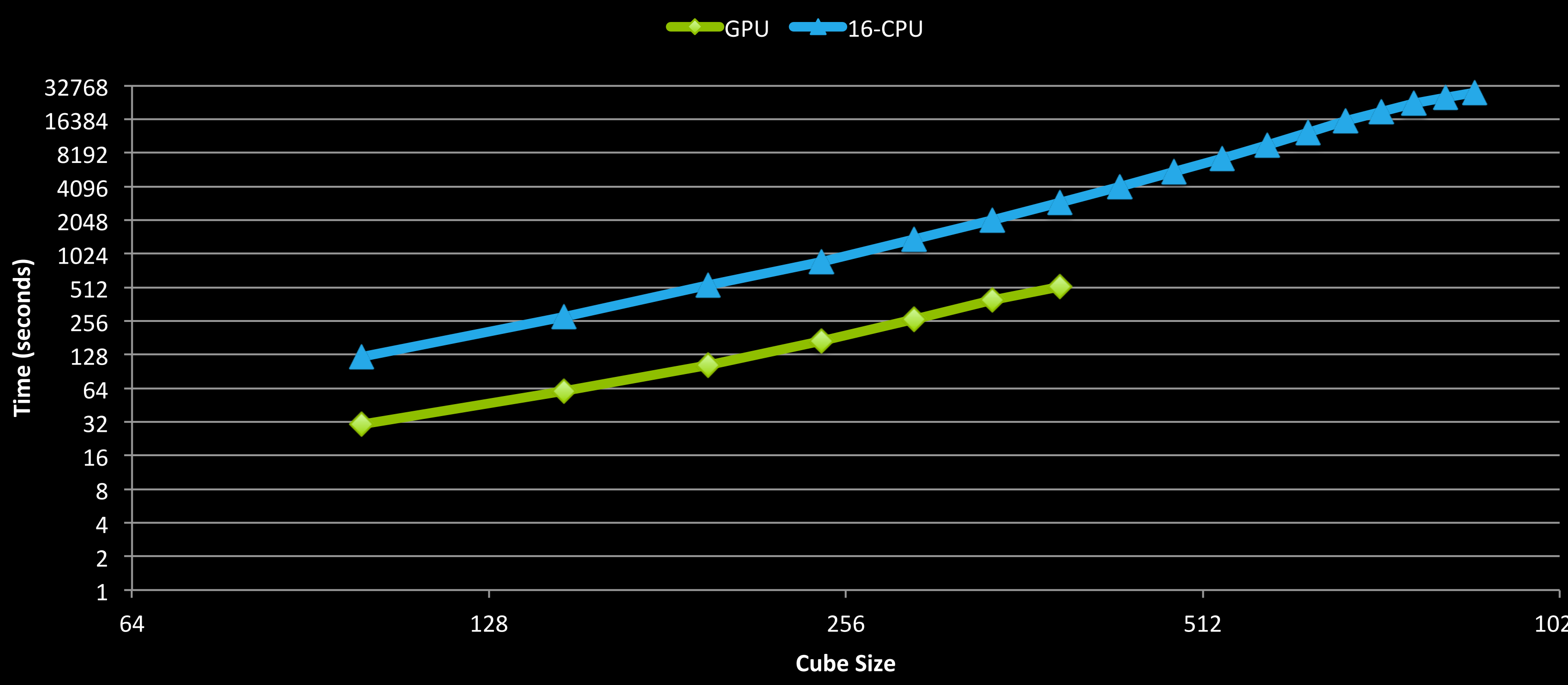


Speedup
GPU vs. 16-CPU
GPU vs. 1-CPU
80
70
60
50
40
30
20
10
0
Speedup Factor
100
148
196
244
292
340
388
Cube Size

While this is all well and good…



…CPU implementation has us beat

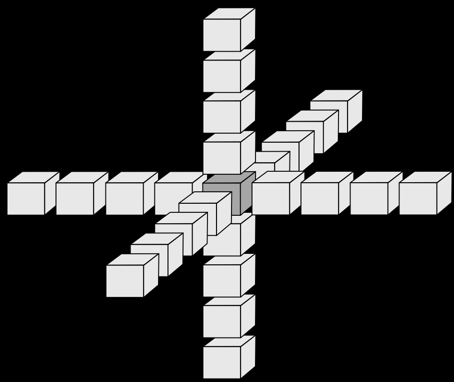


GPU Memory Wall

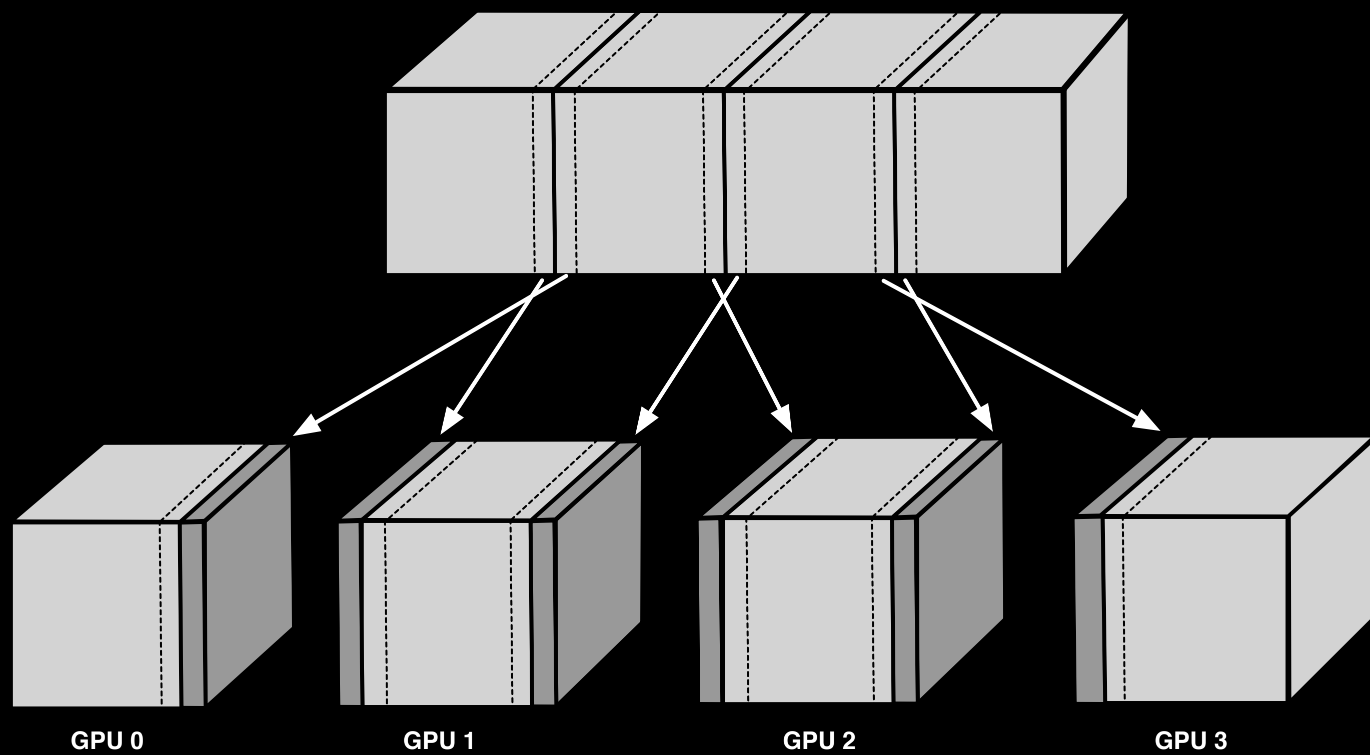
- Limited GPU memory becomes an issue
- In our 3D implementation, each grid point requires:
  - 1 density value
  - 9 stiffness coefficients
  - 3 displacement components (x3 time steps)
  - 6 strain/stress field components
  - 3 acceleration components
- GlobalMemory ≥ 28 * gridPoints * sizeof(float)

Multiple Devices

- Increase GPU memory (in aggregate) by distributing grid across multiple devices
- Not quite embarrassingly parallel
  - FD stencil needs halo cells from neighbor device(s)
  - Halo dance happens each iteration
  - Communication can become costly
- Slice on the slowest varying axis
  - Easy to compute stride

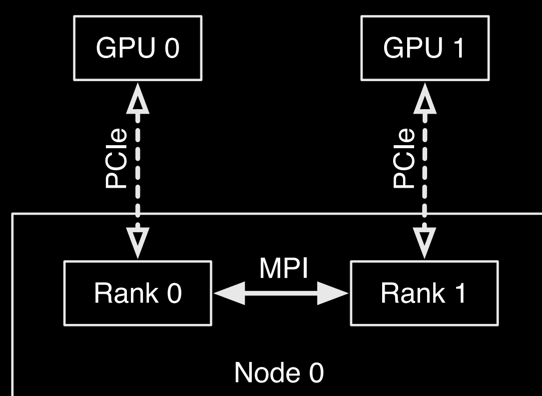


Multiple Devices



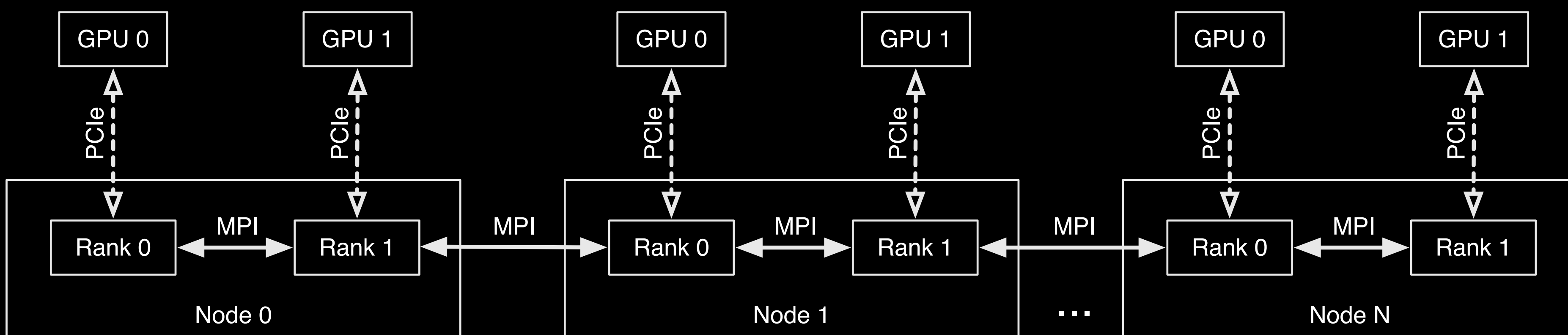
Multiple Devices: Take 1

- One host-process per device
- Communicate through MPI



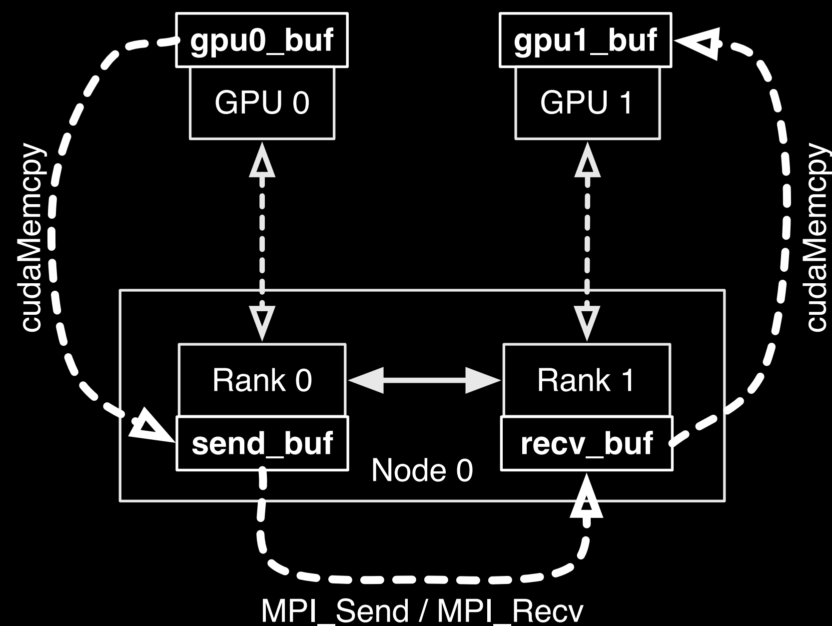
Multiple Devices: Take 1

- Generalizes nicely, scales to multiple nodes
- Handle arbitrary* number of devices & nodes

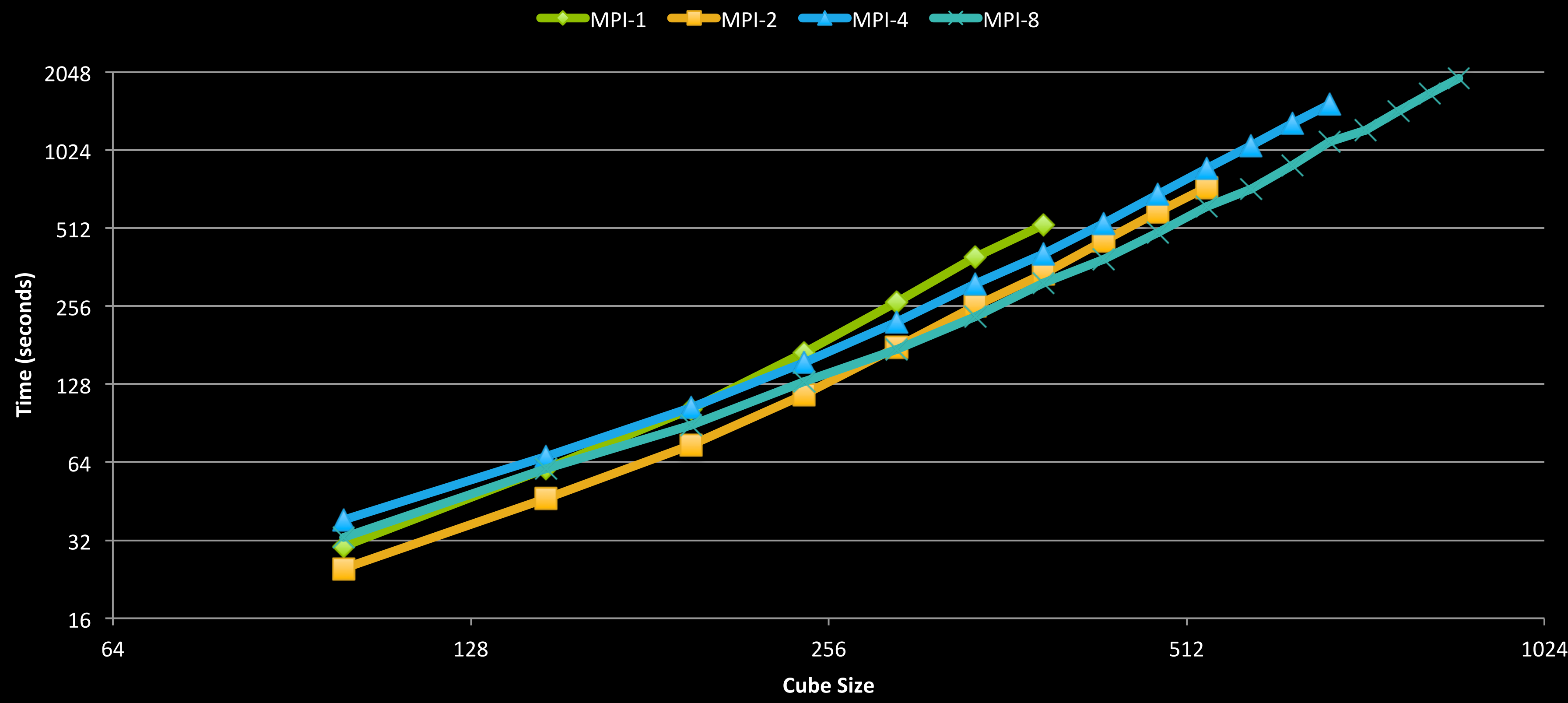


Multiple Devices: Take 1

- But: the data dance is awkward (read: **slow**)



Multiple Devices: Take 1

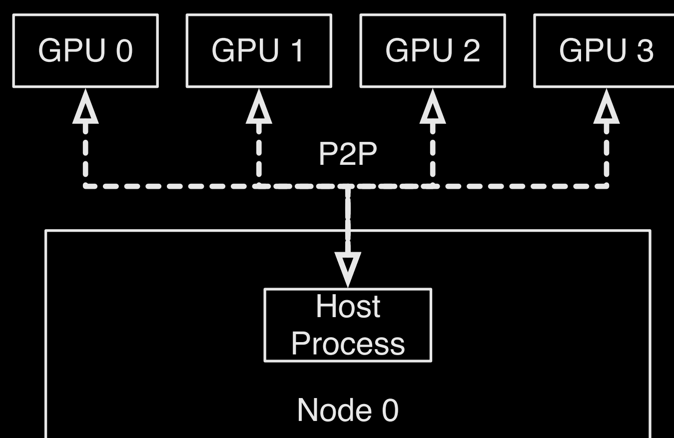


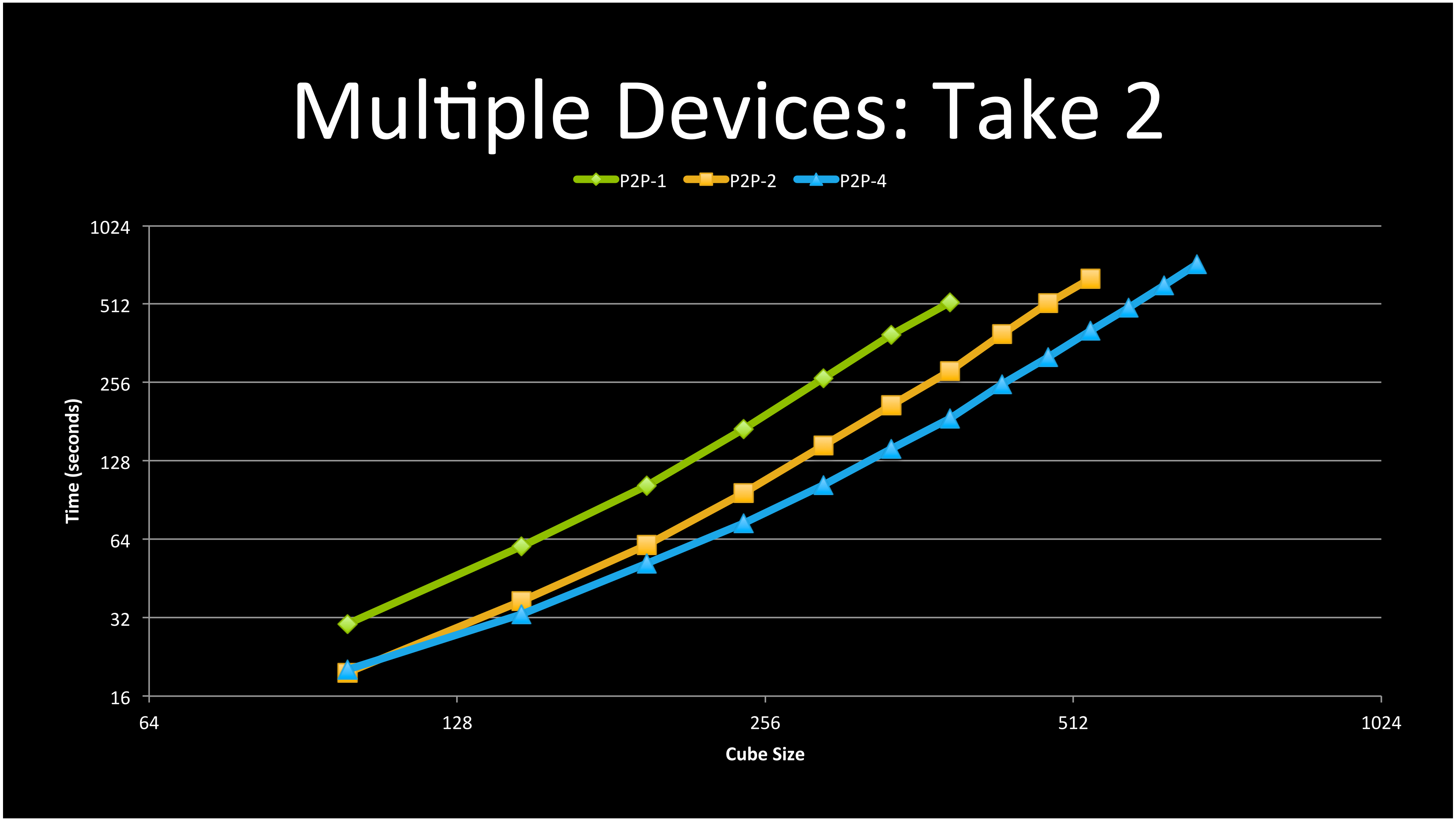
Multiple Devices: Take 1

- Can we fix this?
  - CUDA-aware MPI…?
  - GPU Direct…?
  - RDMA…?
  - Black magic…?
- Open-source research code considerations
  - Keep it simple!

Multiple Devices: Take 2

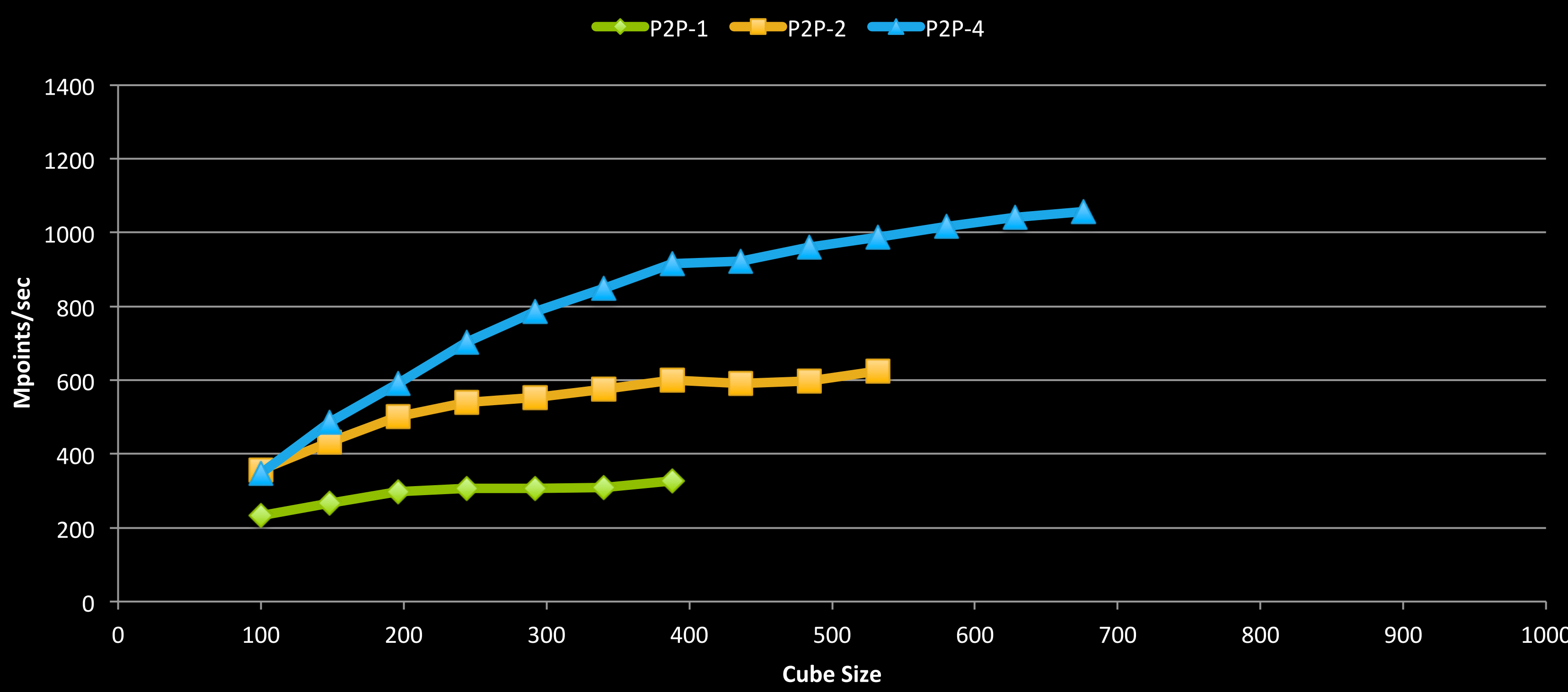
- One host-process *per node*
- Communicate through peer-to-peer memcpy





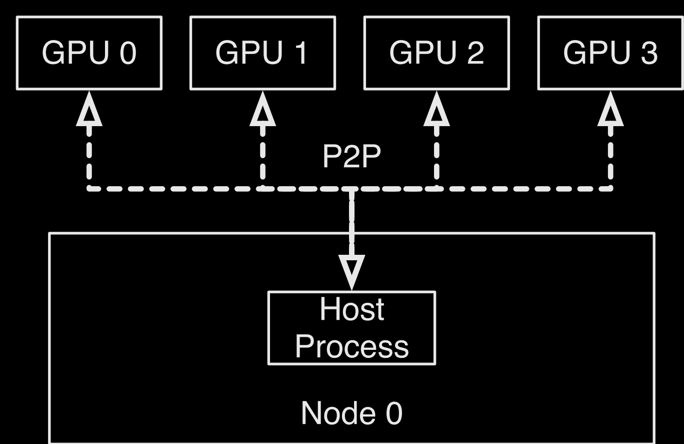
Multiple Devices: Take 2
P2P-1
P2P-2
P2P-4
1024
512
256
128
64
32
16
Time (seconds)
64
128
256
512
1024
Cube Size

Overall Throughput



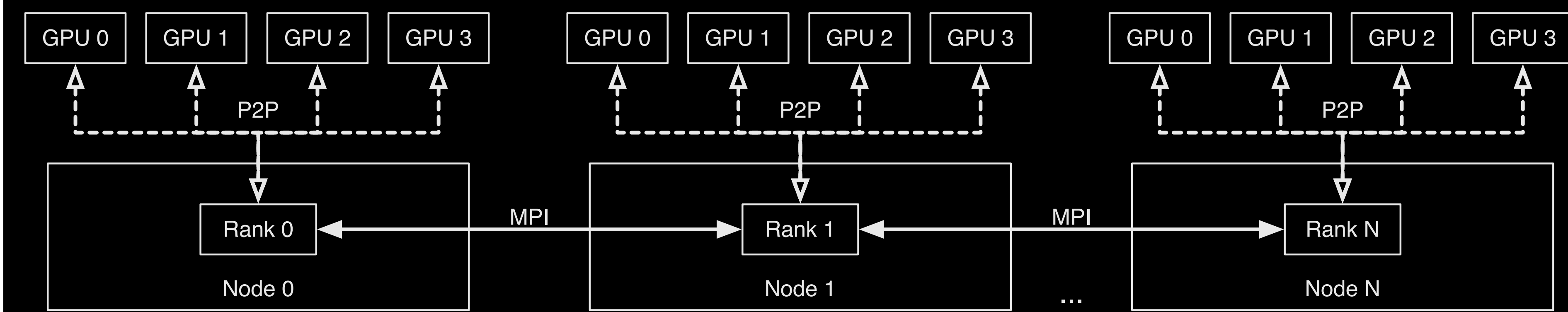
Multiple Devices: Take 2

- But we're still limited to one node…

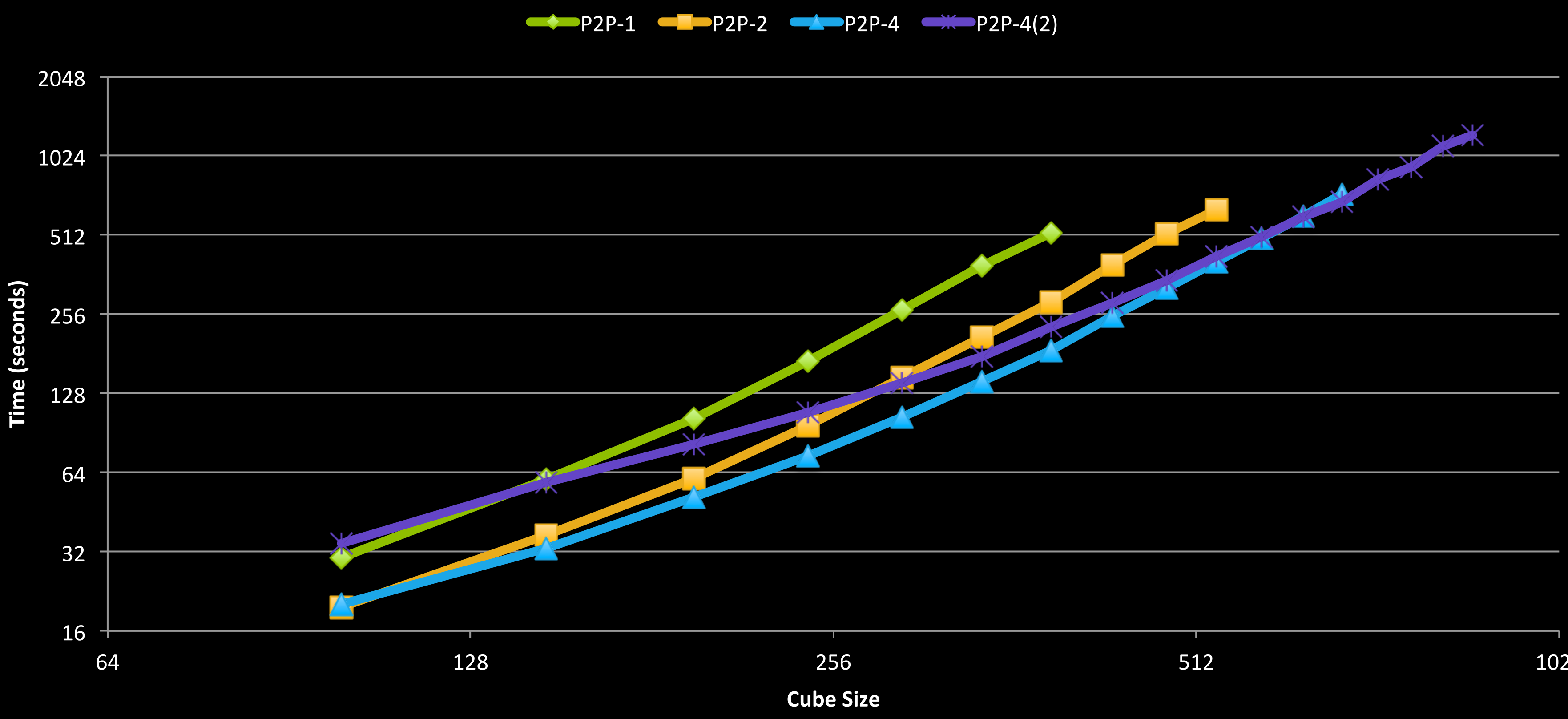


Multiple Devices: Take 2.1

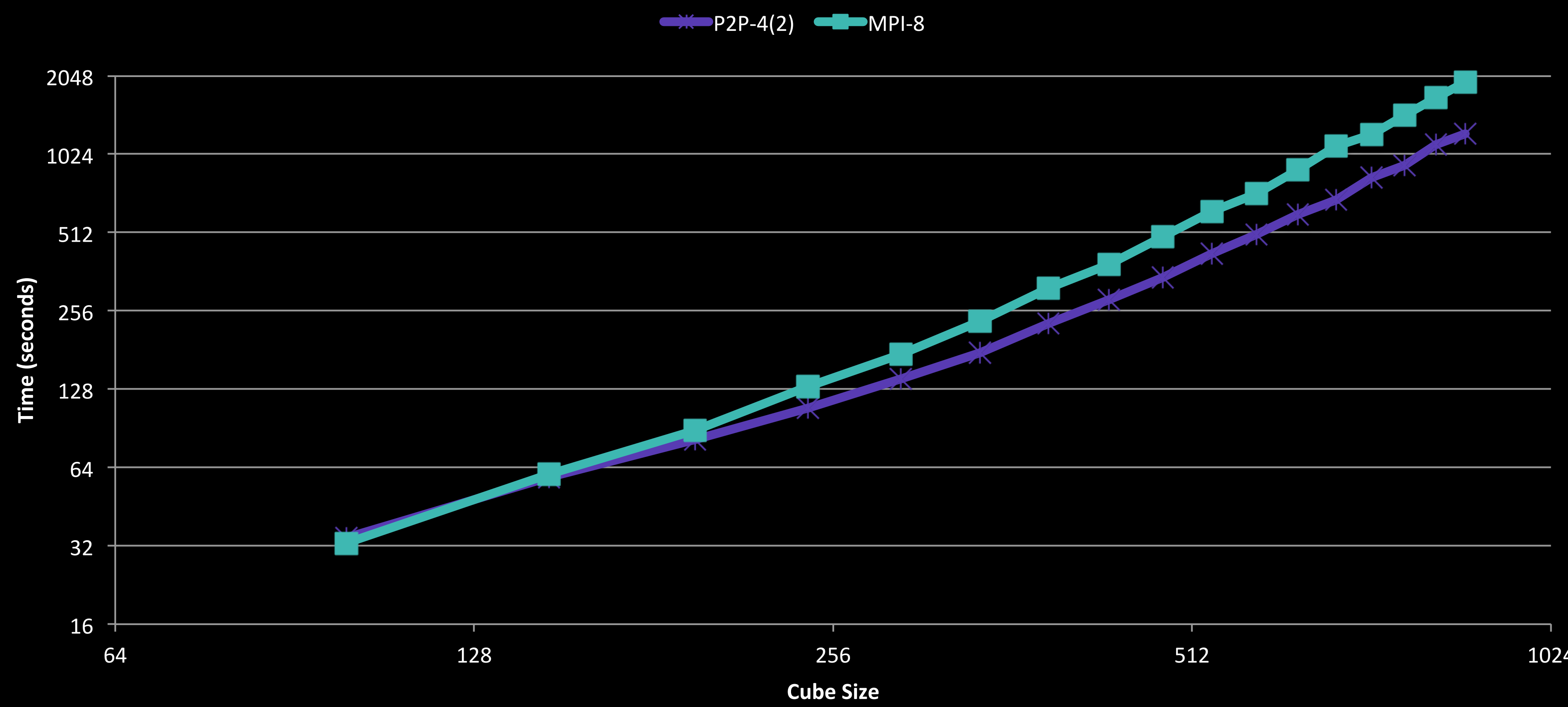
- Scale to multiple nodes with MPI
- Handle arbitrary* number of devices & nodes



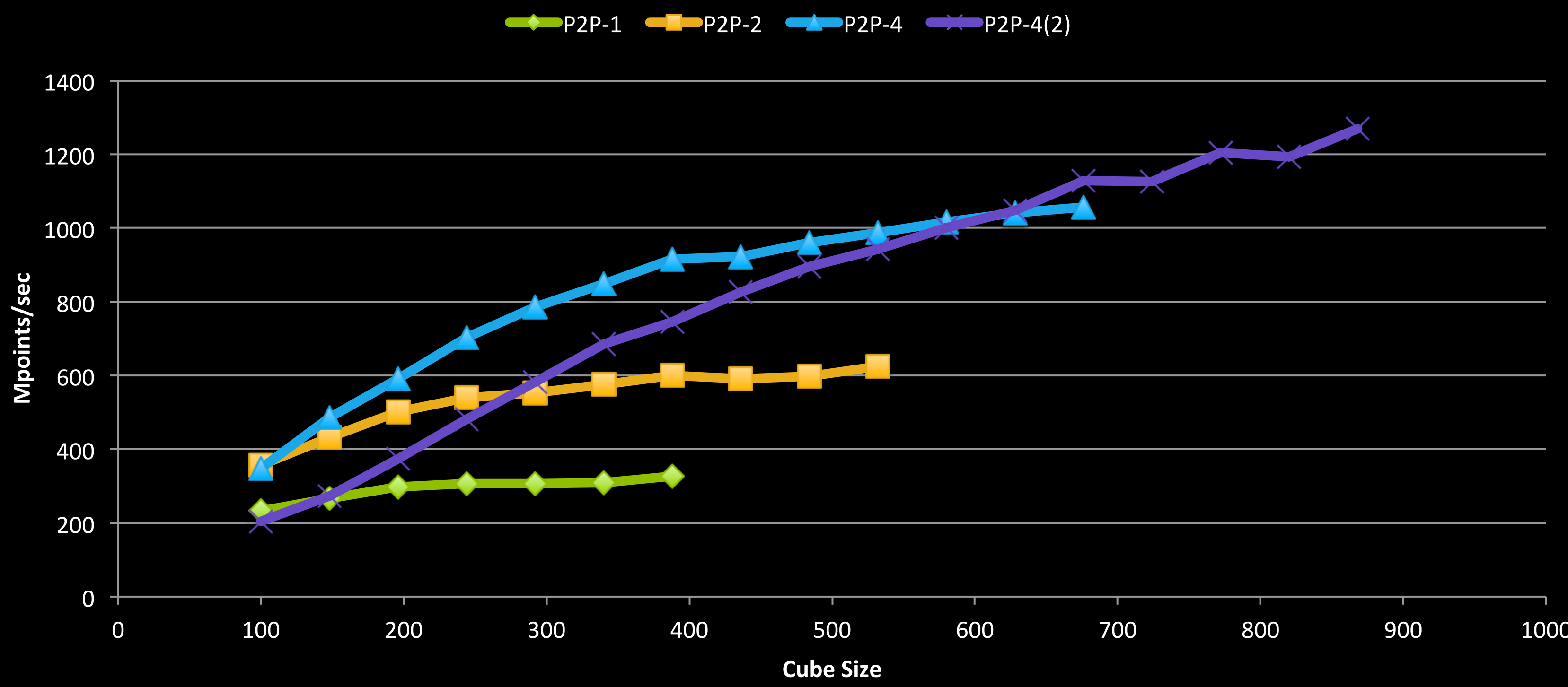
Multiple Devices: Take 2.1

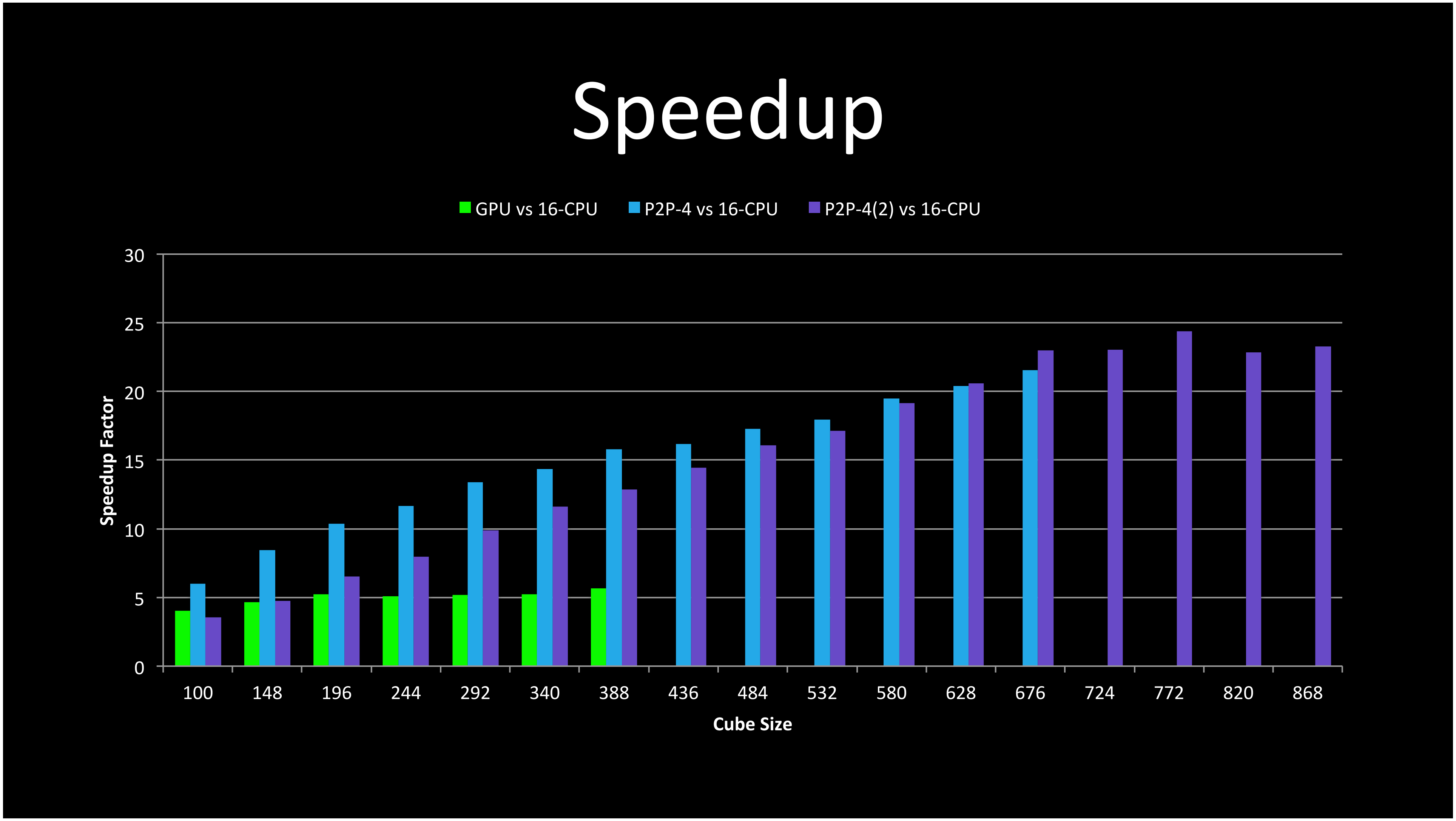


Multiple Devices: Take 2.1



Overall Throughput





Speedup
GPU vs 16-CPU
P2P-4 vs 16-CPU
P2P-4(2) vs 16-CPU
Speedup Factor
0
5
10
15
20
25
30
Cube Size
100
148
196
244
292
340
388
436
484
532
580
628
676
724
772
820
868

Wrapping up

- Elastic wave modeling represents a complex computation problem
  - But, GPU is a promising route for acceleration
  - Flexible multi-GPU implementation offers considerable speedup on large grids
- Represents a seed for further research, modifications, and extensions

- Check out our code: www.reproducibility.org
  - Exec: `sfewefd2d_gpu`, `sfewefd3d_gpu_p2p`, `sfewefd3d_multiNode`
  - Source: user/rweiss
  - Examples: book/uwa/geo2013ElasticModellingGPU/
  - Improvements and extensions welcomed...

...bug reports welcome as well

S4599: An Adventure in Porting:

Adding GPU-Acceleration to Open-Source 3D Elastic Wave Modeling

Robin M. Weiss and Jeffery Shragge

March 26, 2014
GPU Technology Conference

THE UNIVERSITY OF CHICAGO | Research Computing Center

CPGCO2
Centre for Petroleum Geoscience & CO2 Sequestration
The University of Western Australia