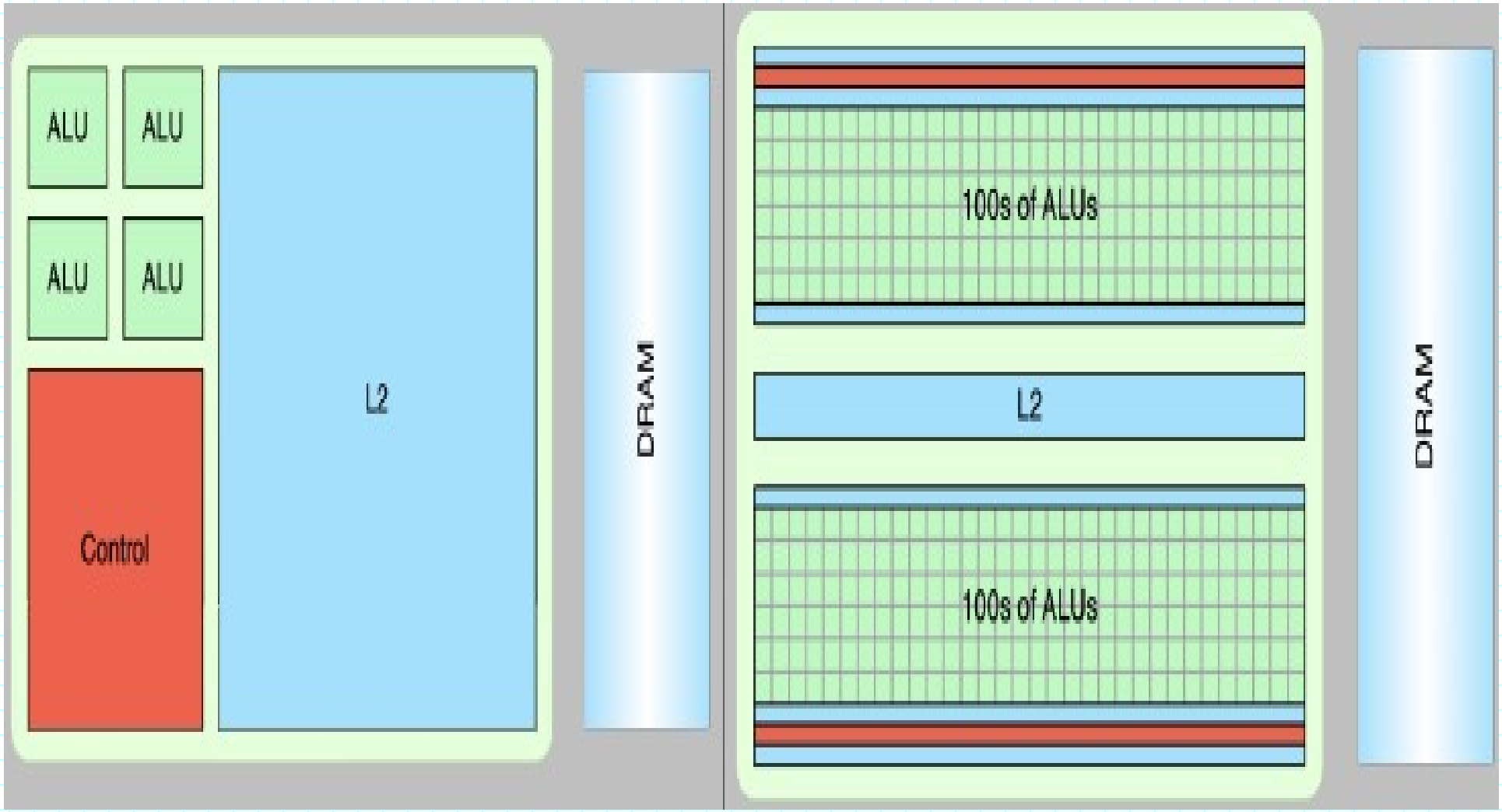
High Performance Computing Systems

- GPUs

Doug Shook

GPU Architecture



GPGPU vs. CPU

- Wasn't always viable due to missing functionality...
- Programming is significantly different
  - More restrictive
- Shared memory? Distributed memory?
- What application properties favor GPU over CPU?

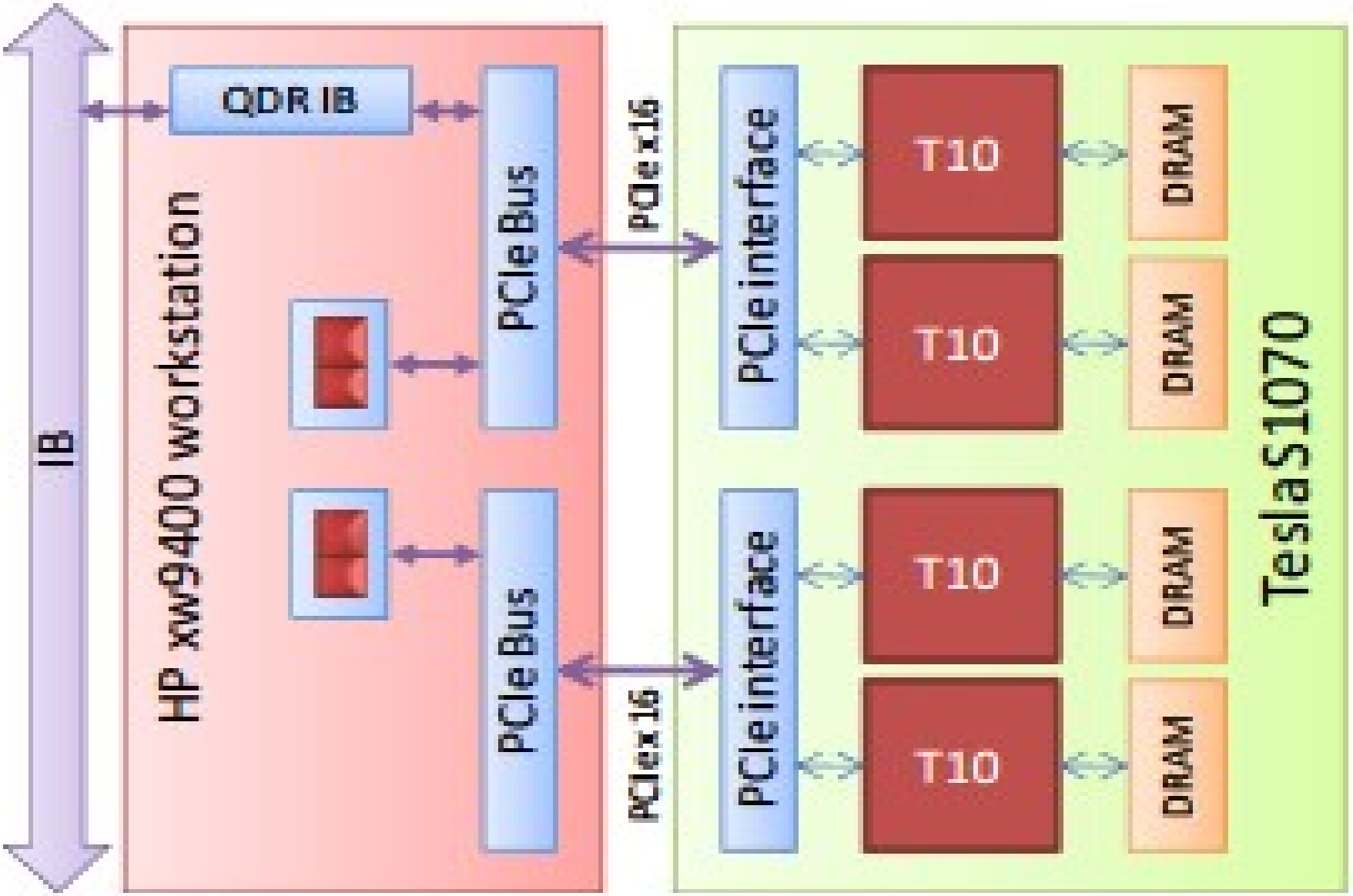
GPU Clusters

- Becoming more and more mainstream
  - Why?
- Not without problems
  - Development
  - Job scheduling
  - Resource Management
  - Security
- Kindratenko, et. al., "GPU Cluster for High-Performance Computing", 2010

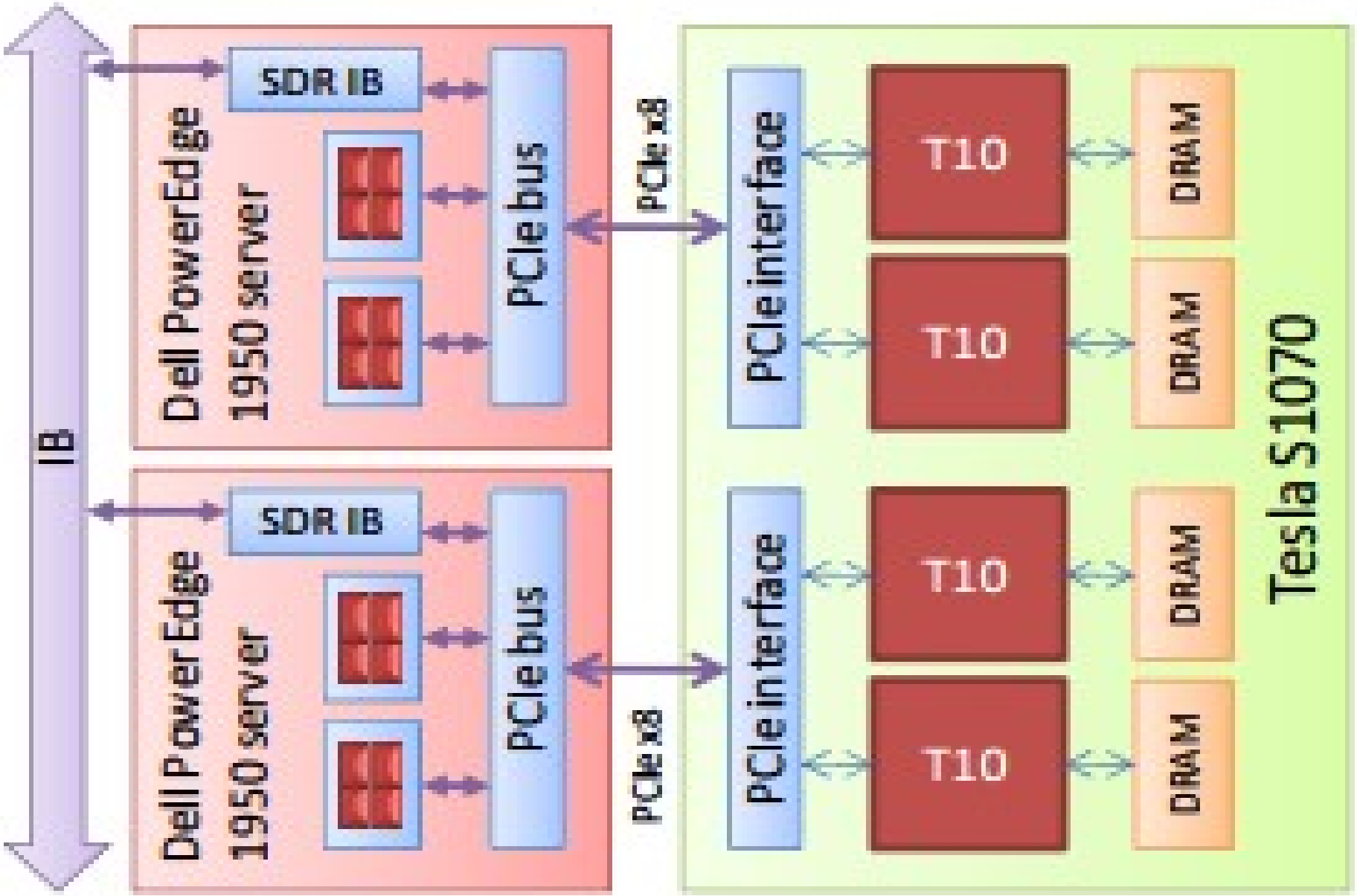
GPU Clusters

- Three primary components:
  - Hosts
  - GPUs
  - Interconnect
- Bandwidth considerations?

GPU Clusters



GPU Clusters



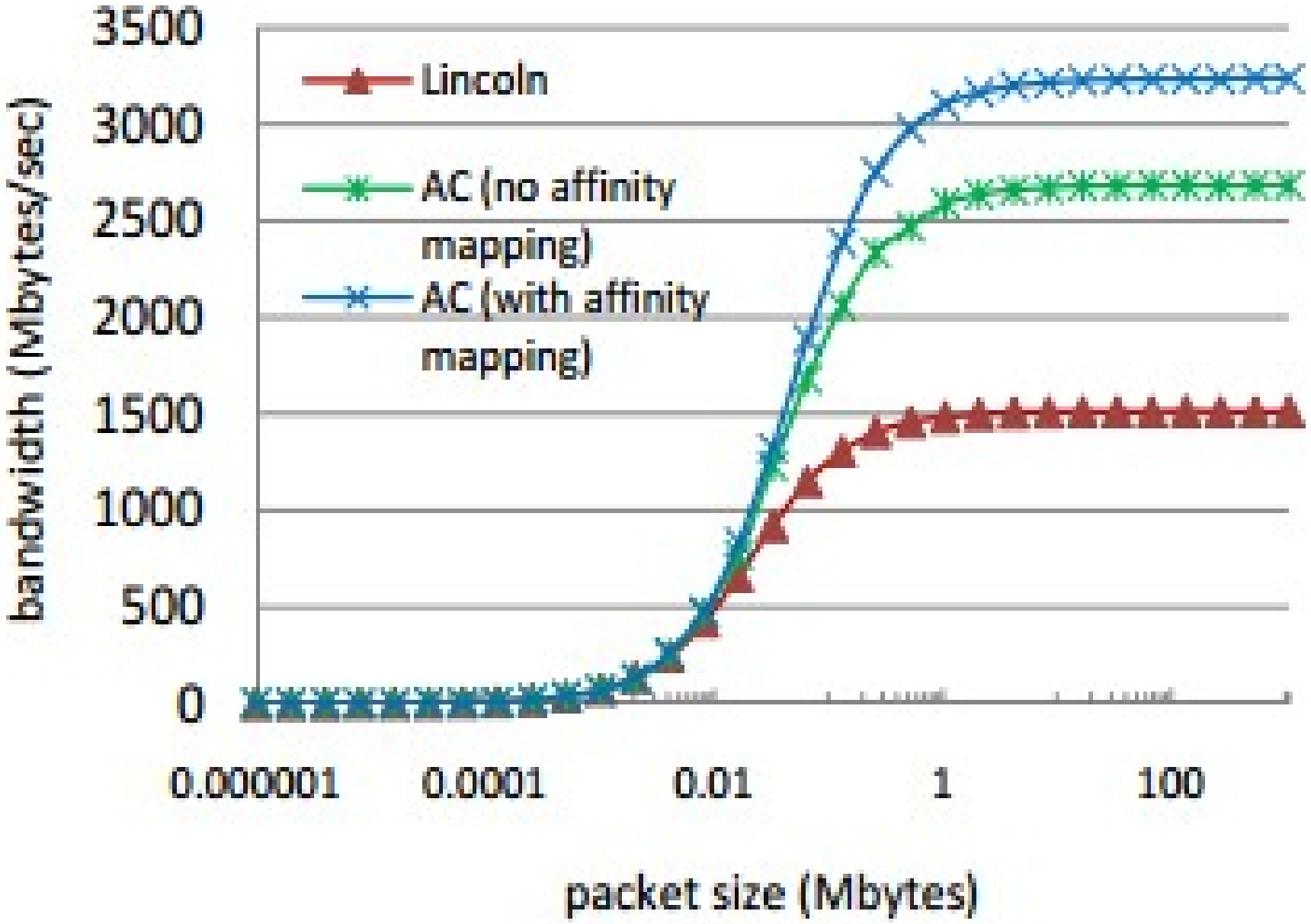
GPU Clusters

	AC	Lincoln
CPU Host	HP xw9400	Dell PowerEdge 1950 III
CPU	dual-core 2216 AMD Opteron	quad-core Intel 64 (Harpertown)
CPU frequency (GHz)	2.4	2.33
CPU cores per node	4	8
CPU memory host (GB)	8	16
GPU host (NVIDIA Tesla…	S1070-400 Turn-key	S1070 (a-la-carte)
GPU frequency (GHz)	1.30	1.44
GPU chips per node	4	2
GPU memory per host (GB)	16	8
CPU/GPU ratio	1	4
interconnect	IB QDR	IB SDR
# of cluster nodes	32	192
# of CPU cores	128	1536
# of GPU chips	128	384
# of racks	5	19
Total power rating (kW)	<45	<210

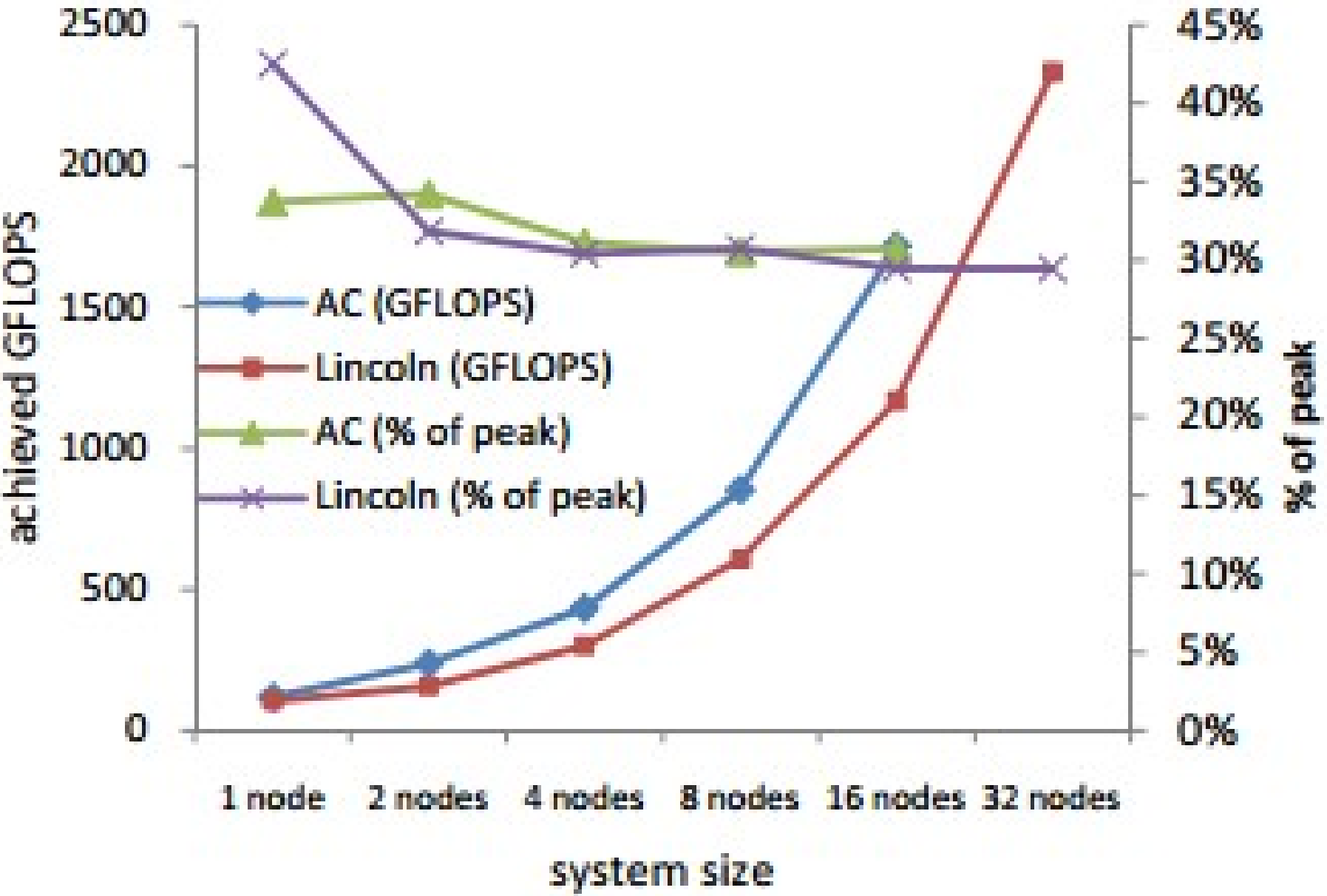
Power Consumption

State	Host Peak (Watt)	Tesla Peak (Watt)	Host power factor (pf)	Tesla power factor (pf)
power off	4	10	.19	.31
start-up	310	182		
pre-GPU use idle	173	178	.98	.96
after NVIDIA driver module unload/reload[1]	173	178	.98	.96
after deviceQuery[2]	173	365	.99	.99
memtest # 10 [9]	269	745	.99	.99
VMD Madd	268	598	.99	.99
after memtest kill (GPU left in bad state)	172	367	.99	.99
after NVIDIA module unload/reload[3]	172	367	.99	.99
NAMD GPU ApoA1 [10]	315	458	.99	.95
NAMD GPU STMV [10]	321	521	.97-1.0	.85-1.0[4]
NAMD CPU only ApoA1	322	365	.99	.99
NAMD CPU only STMV	324	365	.99	.99

Bandwidth


3500
3000
2500
2000
1500
1000
500
0
bandwidth (Mbytes/sec)
Lincoln
AC (no affinity mapping)
AC (with affinity mapping)
0.000001
0.0001
0.01
1
100
packet size (Mbytes)

HPLinpack


2500
2000
1500
1000
500
0
achieved GFLOPS
45%
40%
35%
30%
25%
20%
15%
10%
5%
0%
% of peak
AC (GFLOPS)
Lincoln (GFLOPS)
AC (% of peak)
Lincoln (% of peak)
1 node
2 nodes
4 nodes
8 nodes
16 nodes
32 nodes
system size

Cluster Management

- Resource allocation is usually intended for CPUs...

- The systems are also NUMA-like

Development Tools

- A few options available:
  - Abstract libraries
  - Lightweight toolkits
  - Autogenerated kernels

- Which should be used and when?

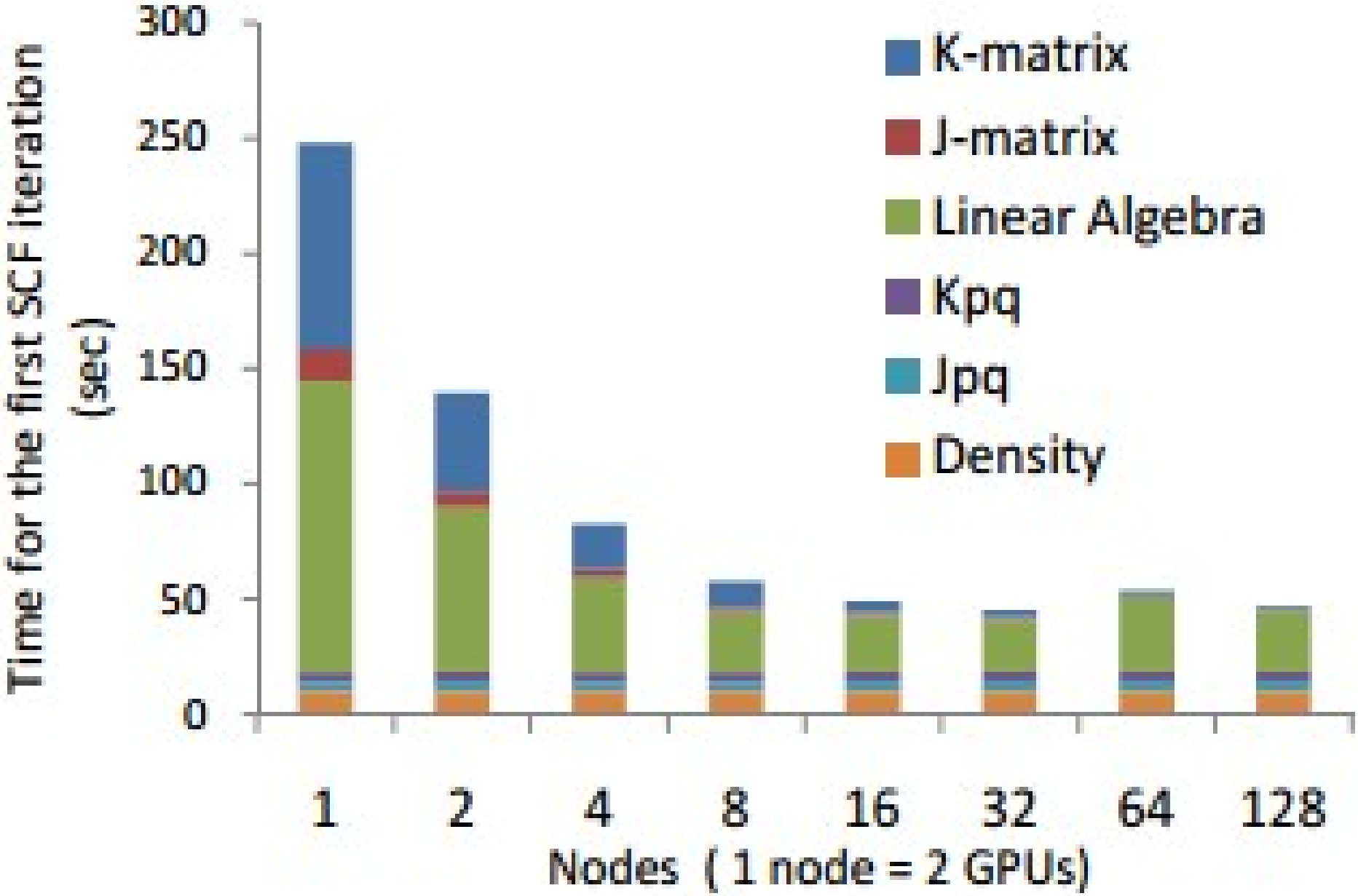
Development Tools

- Tool comparison:
  - CUDA
  - OpenCL
  - PGI x86+GPU Compiler
  - CUDA + MPI

Example Applications

- TPACF
- NAMD
- DSCF

DSCF


Time for the first SCF iteration (sec)
300
250
200
150
100
50
0
K-matrix
J-matrix
Linear Algebra
Kpq
Jpq
Density
1
2
4
8
16
32
64
128
Nodes (1 node = 2 GPUs)

Conclusions

- CUDA still needs some work
- Need an efficient way to reuse legacy MPI code
  - Scheduling

GPU Speedup

- If GPUs are so great, why not use them for everything?
  - 10X – 1000X speedup
- Are they really that great?
- Lee, et. al., "Debunking the 100X GPU vs. CPU Myth", 2010

Goals

- Examining GPU performance claims
- Optimization techniques for CPU and GPU
- How does architecture affect performance?

Kernels

Kernel	Application	SIMD	TLP	Characteristics
SGEMM (**SGEMM**) [48]	Linear algebra	Regular	Across 2D Tiles	Compute bound after tiling
Monte Carlo (**MC**) [34, 9]	Computational Finance	Regular	Across paths	Compute bound
Convolution (**Conv**) [16, 19]	Image Analysis	Regular	Across pixels	Compute bound; BW bound for small filters
FFT (**FFT**) [17, 21]	Signal Processing	Regular	Across smaller FFTs	Compute/BW bound depending on size
SAXPY (**SAXPY**) [46]	Dot Product	Regular	Across vector	BW bound for large vectors
LBM (**LBM**) [32, 45]	Time Migration	Regular	Across cells	BW bound
Constraint Solver (**Solv**) [14]	Rigid body physics	Gather/Scatter	Across constraints	Synchronization bound
SpMV (**SpMV**) [50, 8, 47]	Sparse Solver	Gather	Across non-zero	BW bound for typical large matrices
GJK (**GJK**) [38]	Collision Detection	Gather/Scatter	Across objects	Compute Bound
Sort (**Sort**) [15, 39, 40]	Database	Gather/Scatter	Across elements	Compute bound
Ray Casting (**RC**) [43]	Volume Rendering	Gather	Across rays	4-8MB first level working set, over 500MB last level working set
Search (**Search**) [27]	Database	Gather/Scatter	Across queries	Compute bound for small tree, BW bound at bottom of tree for large tree
Histogram (**Hist**) [53]	Image Analysis	Requires conflict detection	Across pixels	Reduction/synchronization bound
Bilateral (**Bilat**) [52]	Image Analysis	Regular	Across pixels	Compute Bound

Compute Platforms

- CPU: Intel Core i7
  - Quad core
  - 3.2GHz
  - Hyperthreading
  - SIMD Units
  - 32kb/256kb/8MB caches
- Nvidia GTX280 GPU
  - 1.3GHz
  - Hundreds of cores
  - Small caches/memory

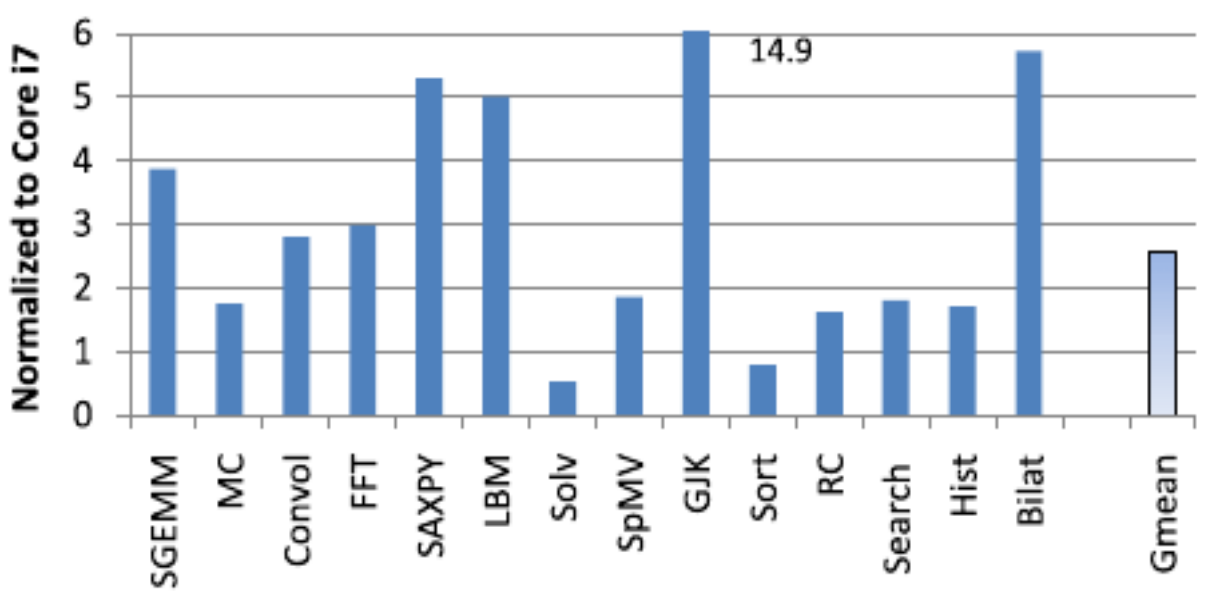
Compute Platforms

	Num. PE	Frequency (GHz)	Num. Transistors	BW (GB/sec)	SP SIMD width	DP SIMD width	Peak SP Scalar FLOPS (GFLOPS)	Peak SP SIMD Flops (GFLOPS)	Peak DP SIMD Flops (GFLOPS)
Core i7-960	4	3.2	0.7B	32	4	2	25.6	102.4	51.2
GTX280	30	1.3	1.4B	141	8	1	116.6	311.1/933.1	77.8

Compute Platforms - Features

- Processing Element
- Cache Size / Multi-threading
- Bandwidth
- Other differences

Performance



Apps.	SGEMM	MC	Conv	FFT	SAXPY	LBM	Solv	SpMV	GJK	Sort	RC	Search	Hist	Bilat
Core i7-960	94	0.8	1250	71.4	16.8	85	103	4.9	67	250	5	50	1517	83
GTX280	364	1.4	3500	213	88.8	426	52	9.1	1020	198	8.1	90	2583	475

Table 3: Raw performance measured on the two platforms. From the left, metrics are Gflops/s, billion paths/s, million pixels/s, Gflops/s, GB/s, million lookups/s, FPS, Gflops/s, FPS, million elements/s, FPS, million queries/s, million pixels/s, million pixels/s.

Performance Analysis

- Bandwidth
- Flops
- Cache
- Gather / Scatter
- Reduction / Synchronization
- Fixed Function

Hardware Optimizations

- CPU
- GPU

Hardware Recommendations

- Bandwidth
- Flops
- Cache
- Gather / Scatter
- Reduction / Synchronization
- Fixed Function

Conclusions