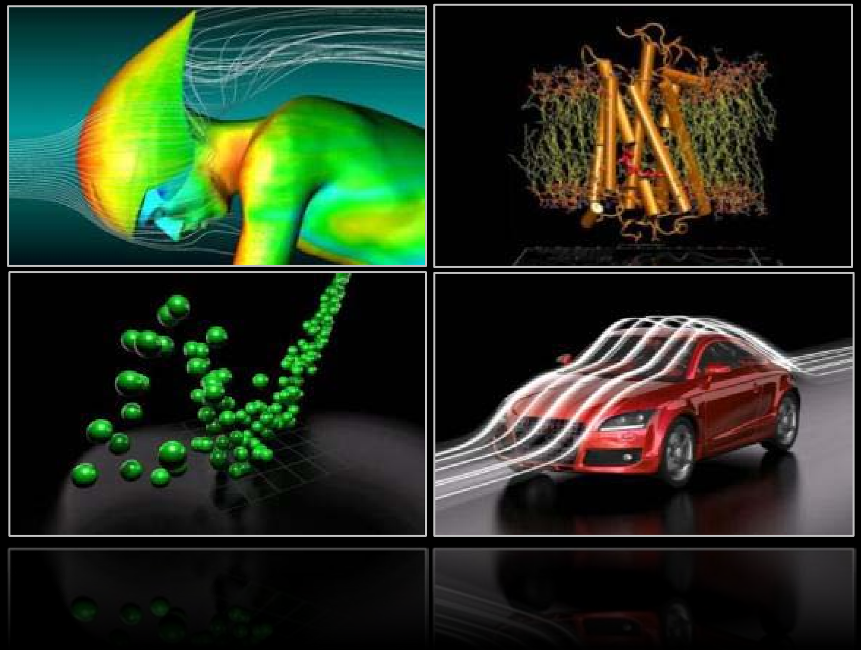
The Fermi GPU and HPC Application Breakthroughs



Peng Wang, PhD | HPC Developer Technology Group
Stan Posey | HPC Industry Development
NVIDIA, Santa Clara, CA, USA

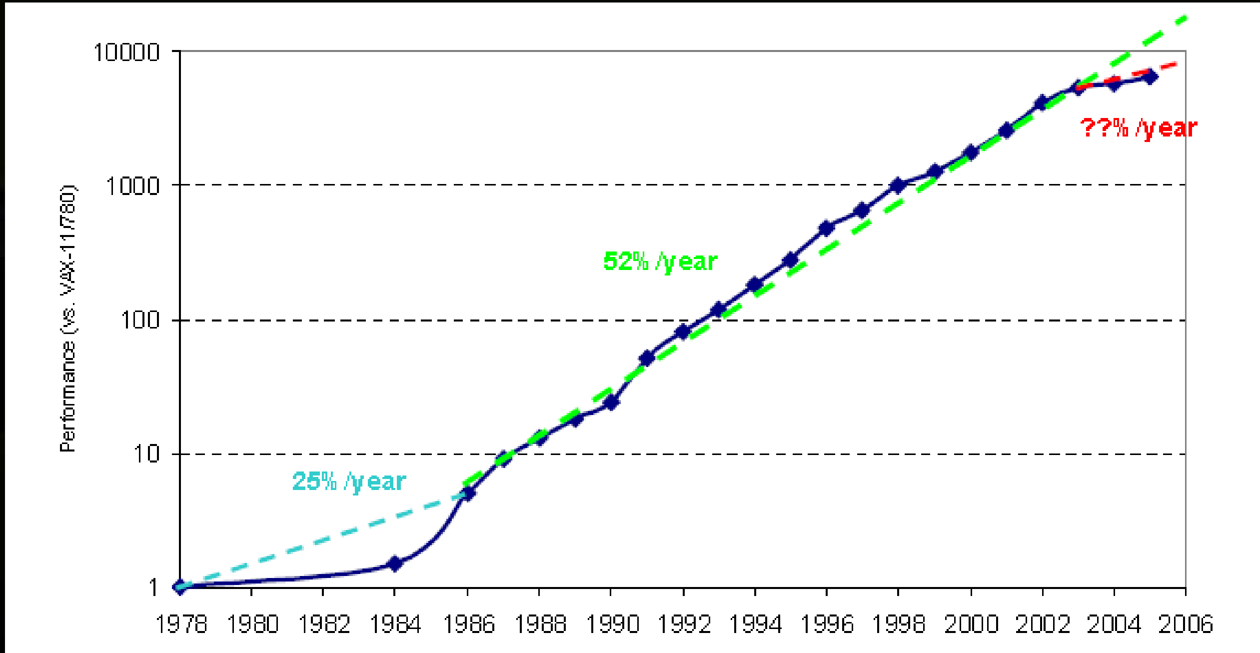
Overview

- GPU Computing: motivation
- Implicit computational structural mechanics (CSM) and direct solvers
- Explicit computational fluid dynamics (CFD) and flux computation
- Implicit CFD and iterative solver
- Molecular dynamics (MD) and particle method
- Summary

Why Parallel Computing

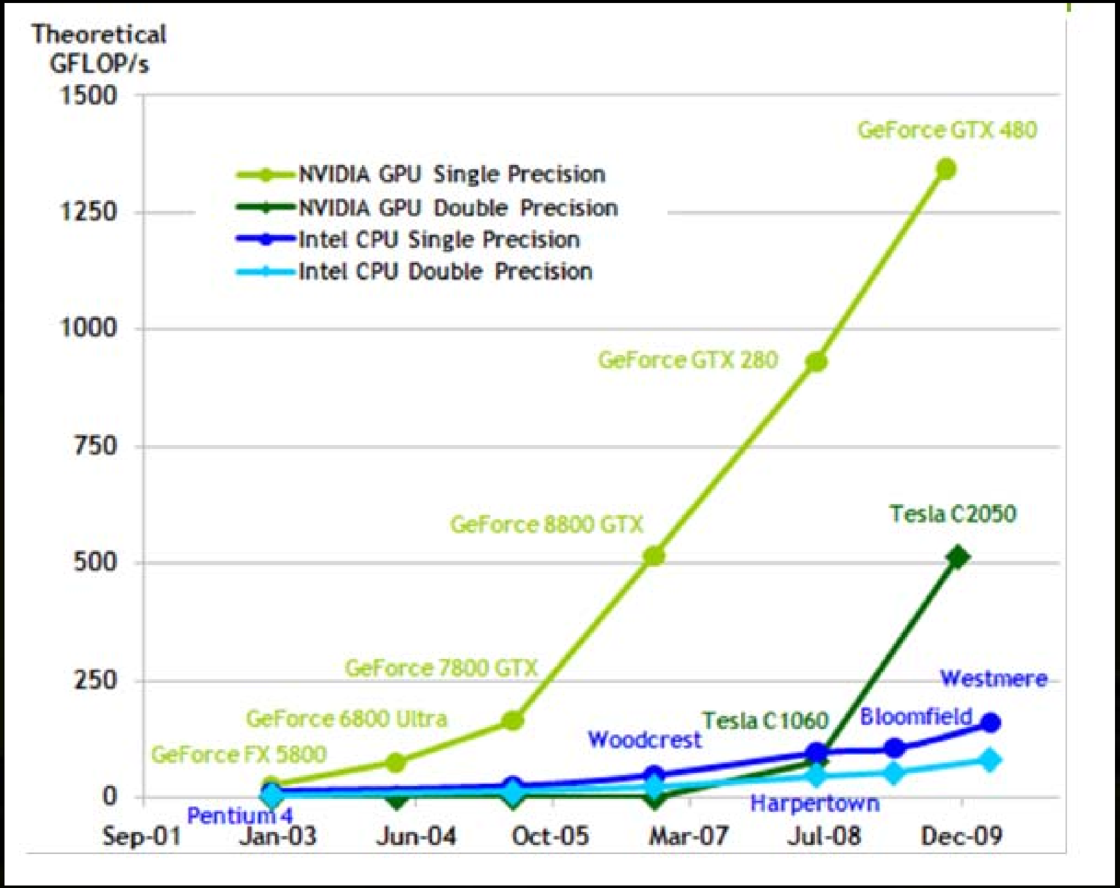
- **Parallel computing is mainstream: single core performance won't increase anymore**
  - **Single core frequency stopped increasing around 2004**
  - **Mainly because of the power wall**

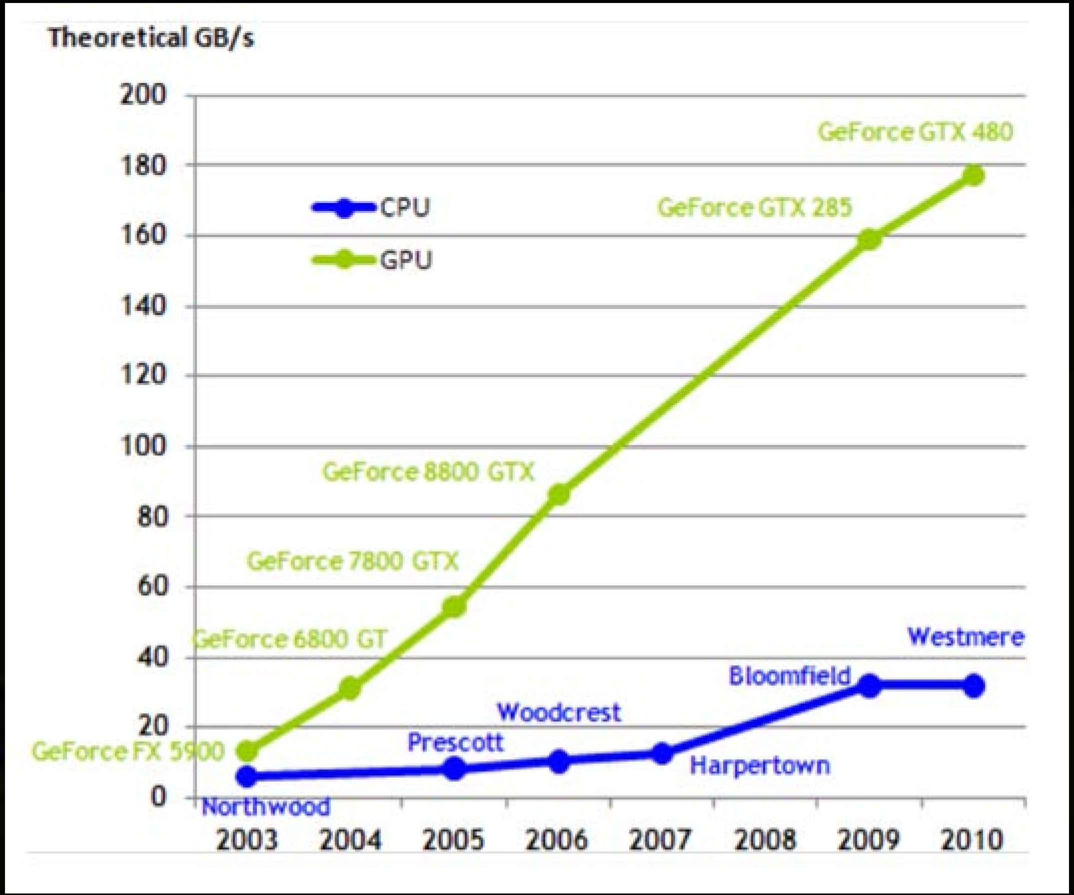
- **What's the best platform for parallel computing?**



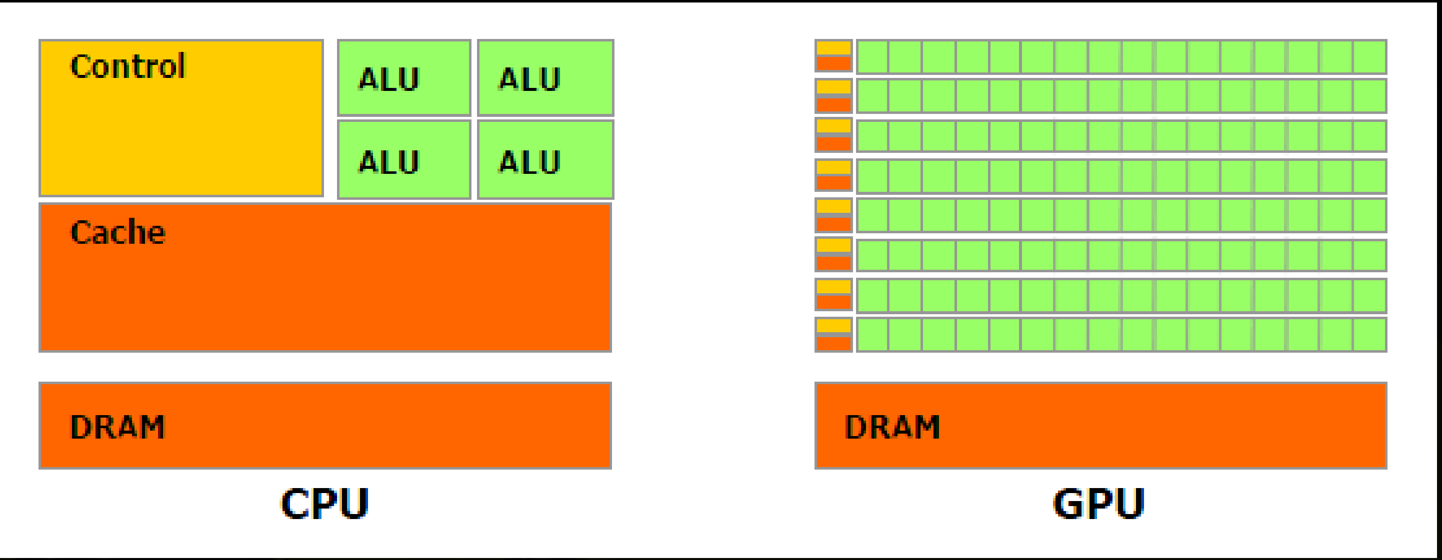
Hennessy & Patternson 2006

Why GPU Computing: Performance





Why GPU Has Much Higher Throughput



- CPU/multicore is optimized for latency, GPU/manycore is optimized for throughput.
- Heterogeneous computing: use latency-optimized cores for complicated logic/control part, use throughput-optimized cores for data-parallel, compute-intensive part.

Tesla C2050/C2070 GPU (Fermi)

C2050 Spec	
Processor clock	1.15 GHz
# SP core	448
# DP core	224
Memory clock	1.6 GHz
Memory bus width	384 bit
Memory size	3 GB (ECC-off), 2.6 (ECC-on)

C2070 has 6 GB memory

fp32 throughput = 448*2*1.15= 1030 GFlop/s

fp64 throughput = 224*2*1.15 = 515 GFlop/s

Memory throughput = 2*384*1.6/8 = 153.6 GB/s (decrease by ~20% with ECC on)

Patterns in Scientific Computing: Philip Colella's "Seven Dwarfs"

- Structured grids (including locally structured grids, e.g. AMR)
- Unstructured grids
- Fast Fourier Transform
- Dense Linear Algebra
- Sparse Linear Algebra
- Particles
- Monte Carlo

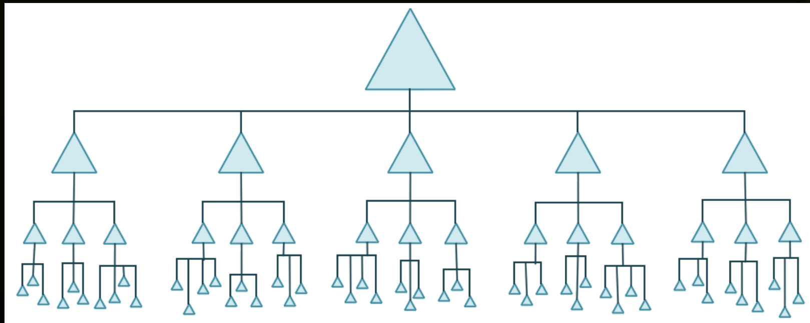
from "Defining Software Requirements for Scientific Computing", Phillip Colella, 2004

Colella's Dwarfs in Engineering Simulations

- CSM
  - E.g. Abaqus, ANSYS Mechanical, LS-DYNA, MSC.Nastran, Marc, NX Nastran, OptiStruct, RADIOSS, etc.
  - Direct sparse solver: dense linear algebra
- CFD
  - E.g. FLUENT, STAR-CCM+, OpenFOAM, AcuSolve, etc.
  - Explicit solver: flux computation on structured/unstructured grid.
  - Implicit solver/Iterative solver: sparse linear algebra & BLAS1
- CEM
  - E.g. HFSS, Xfdtd, etc.
  - Unstructured grid (FEM), structured grid (FDTD), dense linear algebra (direct solver)
- MD
  - E.g. Amber, NAMD, LAMMPS, Charmm, Gromacs, etc.
  - Particles, sparse linear algebra (neighbor/cell list), FFT/multilevel/tree

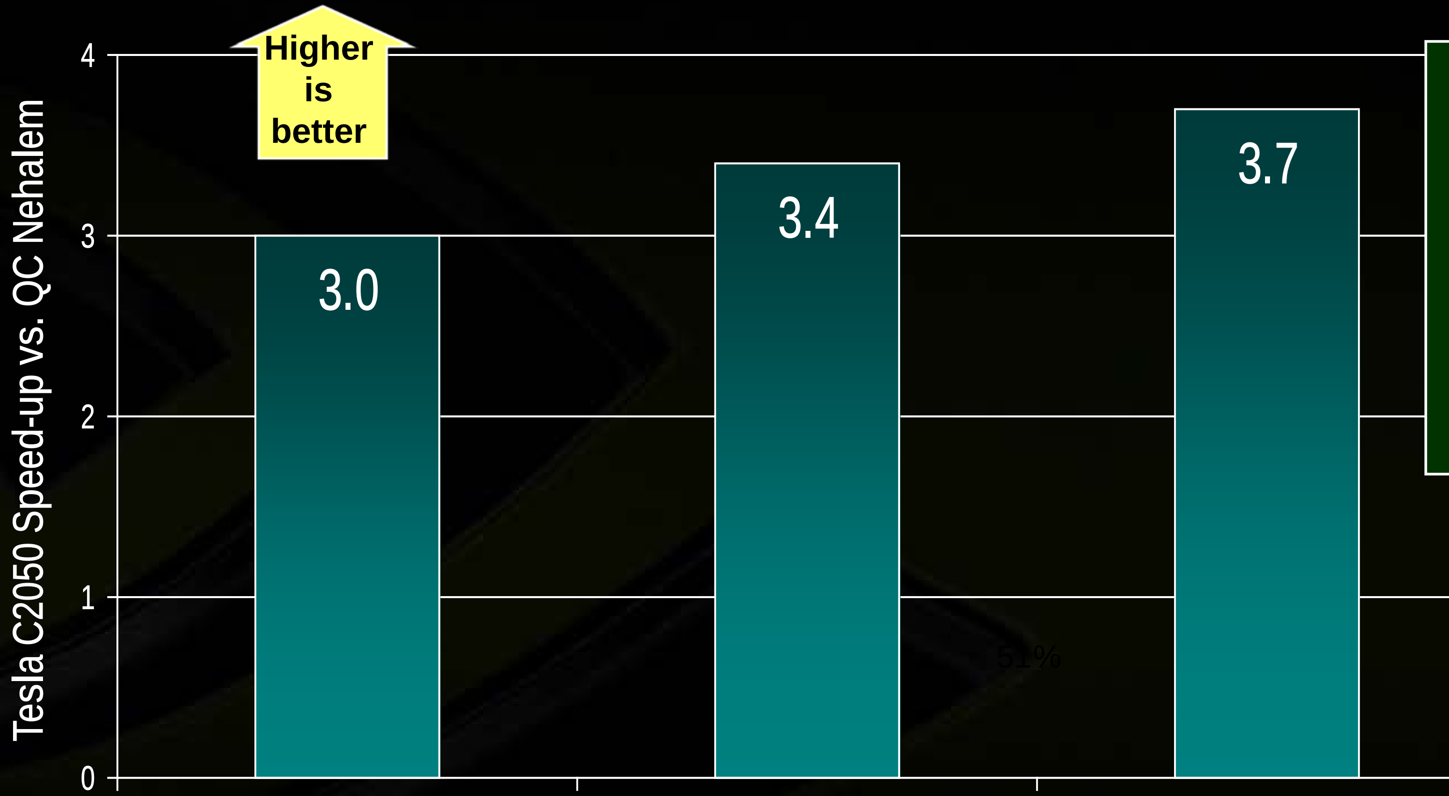
Direct Solvers

- The most time consuming part is dense matrix operations such as Cholesky factorization, Schur complement
- Put dense operations to GPU while keep the assembly tree trasversal on CPU
- GPU can achieve a good fraction of peak floating point performance on dense matrix algorithms.
- The data transfer time may be hide by concurrency depending on the solver implementation.



Preliminary Abaqus Results for Tesla C2050

Abaqus/Standard: Results of Multi-Frontal Direct Sparse Solver

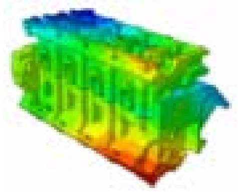


	S4b - 5 MM	Case 2 - 3.7 MM	Case 3 - 1.5 MM
DOFs	S4b - 5 MM	Case 2 - 3.7 MM	Case 3 - 1.5 MM
FP Ops	1.03E+13	1.68E+13	1.70E+13

Source: SIMULIA Customer Conference, 27 May 2010:

"Current and Future Trends of High Performance Computing with Abaqus"

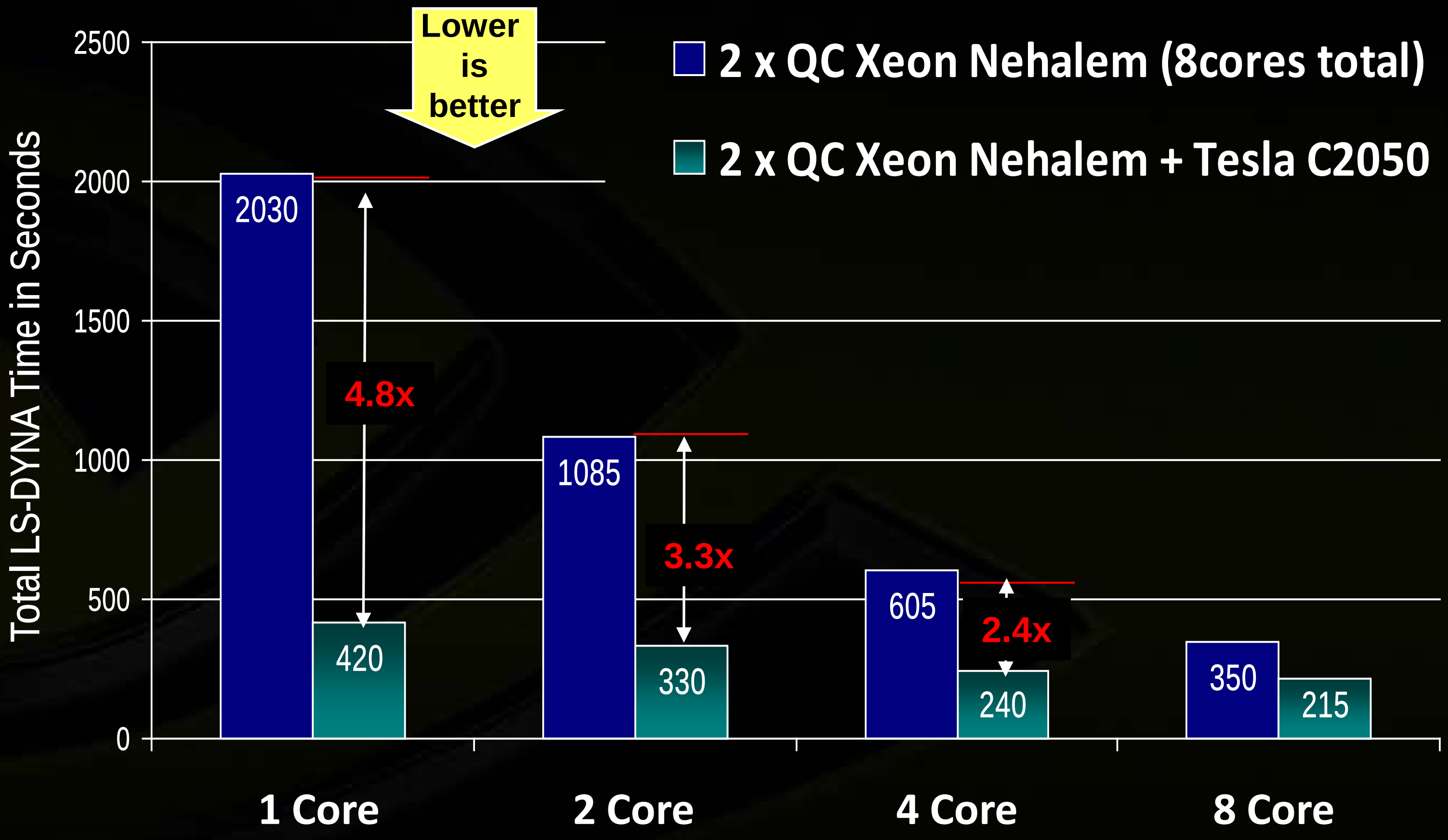
Presentation by Matt Dunbar



S4b: Engine Block Model of 5 MM DOF

LSTC and LS-DYNA 971 Results for Tesla C2050

NOTE: Results of End-to-End LS-DYNA Wallclock Time for 300K DOF



OUTER3 Model

~300K DOF, 1 RHS

NOTE: LSTC results in just two weeks on C2050 (Fermi)

Target performance is ~2x faster on GPU and multi-GPU for Q4 LS-DYNA release

Explicit Finite Difference/Volume Codes

- Highly compressible flows, Maxwell equation, wave equations
- Flux computation can be memory or compute bound depending on the flux formula, physics model.
- GPU's shared memory is well-suited to reduce redundant memory access in FD/FV
- Unstructured grid: reorder grid data structure to enhance memory coalecsing (e.g. FEFLO, Corrigan, et al. 2009)

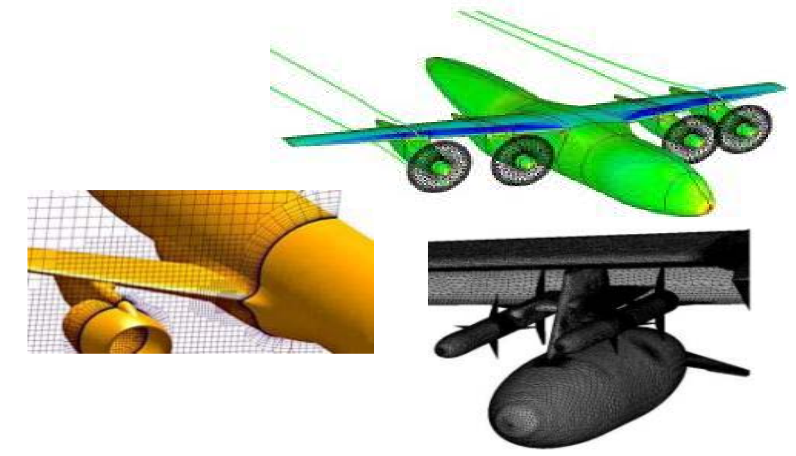
Aerodynamical CFD

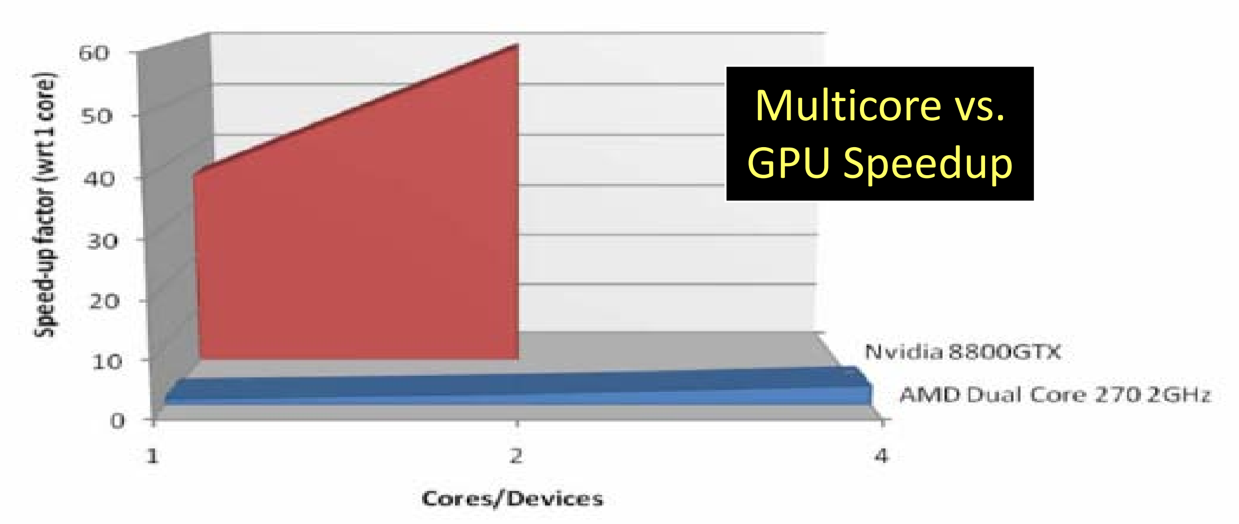
BAE Systems

BAE SYSTEMS

Technology and Engineering Services

- BAE-developed unstructured URANS/DES/LES/DG code Veloxi
- Fast turnaround time crucial for exploring the aerodynamical design space
- Arbitrary polyhedra
- 3D explicit finite volumn
- 2nd order time and space
- Two equation turbulence model
- 60x over single core Intel i7 CPU with use of 2 x GeForce 8800GTX SP



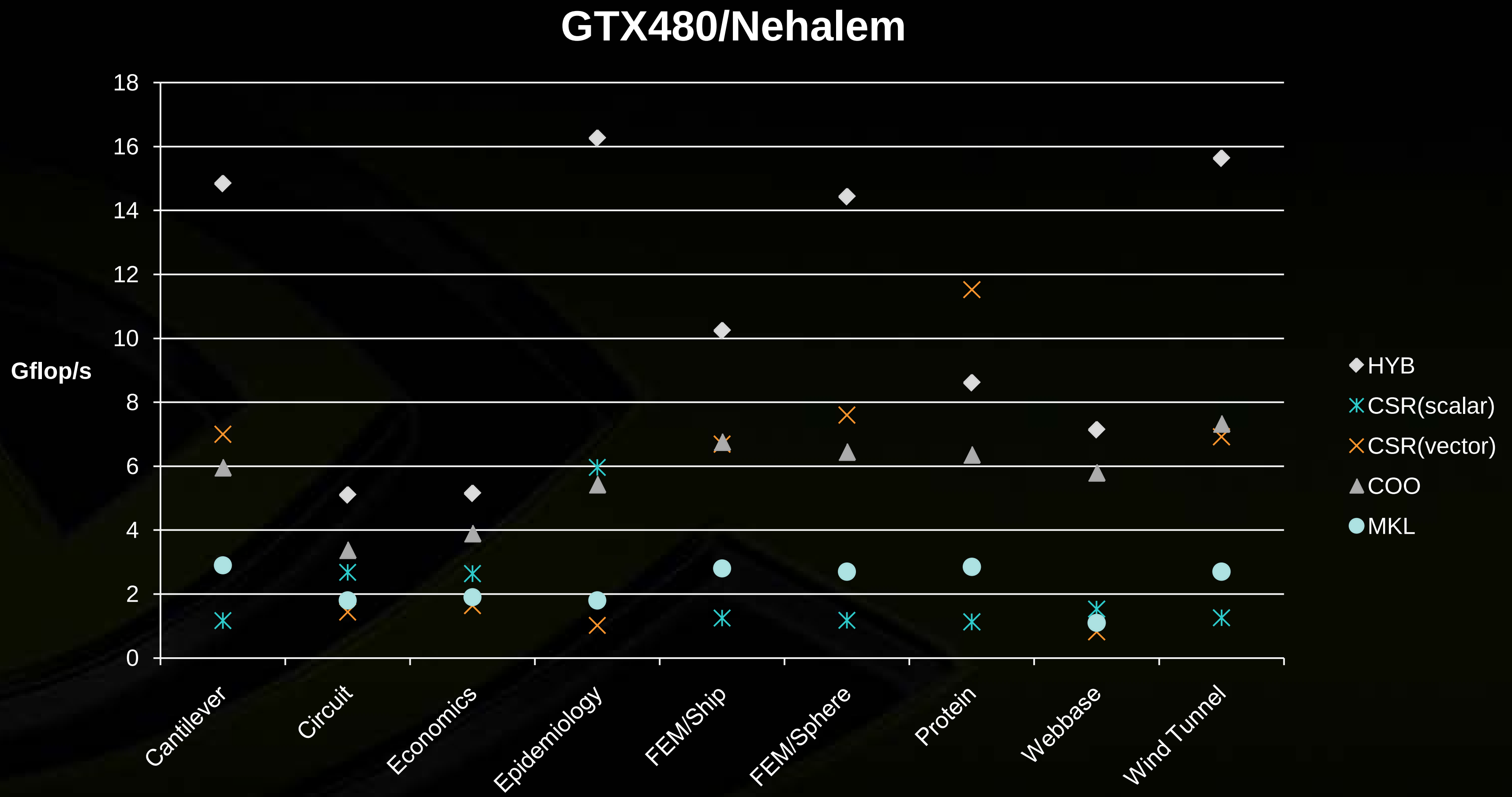


Jamil Appa, GTC, 2009

Iterative Solvers

- Sparse-matrix vector multiply & BLAS1
  - Memory-bound
- GPU can deliver good SpMV performance
  - ~10-20 Gflops for unstructured matrices in double precision
- The best sparse matrix data structure on GPU can be different from CPU
  - Hybrid data structure: unpacked regular storage plus packed storage
  - Best balance between performance and storage space
- A massively parallel preconditioner is key

Unstructured Matrices



GTX480
BW=160 GB/s

Flops=2*nz/t, BW = (2*sizeof(double)+size(int))/t
=> Flops~BW/10~16 Gflop/s

ACUSIM and Release of AcuSolve 1.8 for Tesla

NVIDIA

ACUSIM SOFTWARE

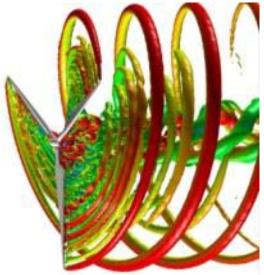
NVIDIA Tesla GPUs Enable Shorter Design Cycles, Improved Product Quality Using AcuSolve

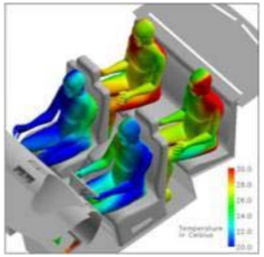
Latest Release of the AcuSolve CFD Software Doubles Performance of Flow Simulations using NVIDIA Tesla C2050 GPUs

SANTA CLARA, Calif. —May 27, 2010—NVIDIA announced today that ACUSIM Software, a leading provider of computational fluid dynamics (CFD) solutions widely used by engineers and scientists involved in product design, has integrated support for NVIDIA® Tesla™ 20-series GPUs into the company's latest AcuSolve 1.8 release.

Performance tests of the general-purpose finite-element-based CFD flow solver have demonstrated up to a 2x boost in performance with the Tesla C2050 GPU processor, compared with the latest quad-core CPU running the same simulation.

AcuSolve is used in a broad range of mechanical design applications and deployed by research organizations and Fortune 500 companies including Bechtel, Chevron, John Deere, Procter & Gamble, Sanyo, Visteon and Whirlpool. They use CFD simulations to replace costly physical tests during product development, which leads to shorter design times and improved product quality.



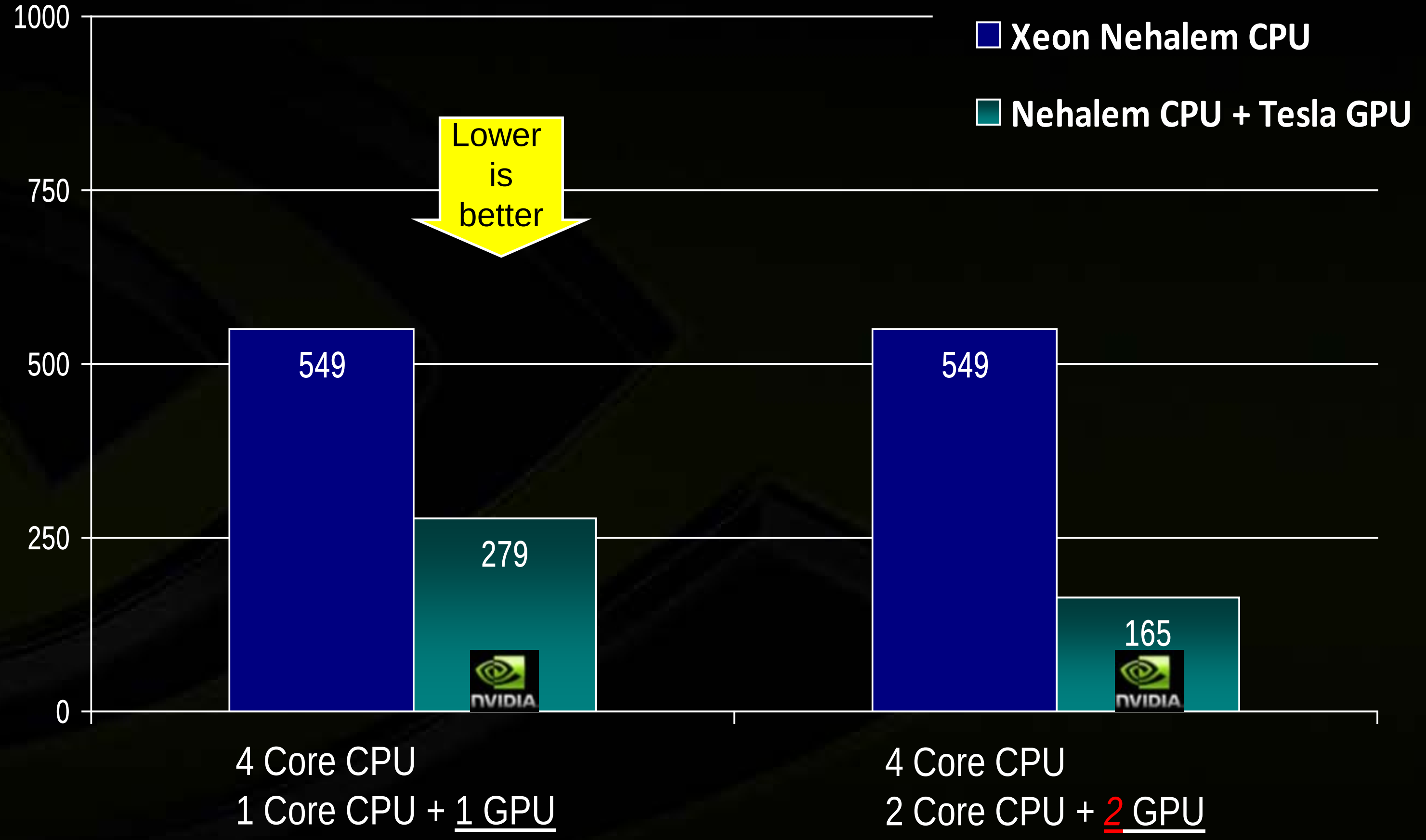


P&G

"It's always about computing speed," said Tom Lange, Director of Modeling and Simulation at Procter & Gamble. "GPU-accelerated CFD allows for more realism, helping us replace slow and expensive physical learning cycles with virtual ones. This transforms engineering analysis from the study of failure to true virtual trial and error, and design optimization."

Performance of AcuSolve 1.8 on Tesla C2050

AcuSolve: Comparison of Multi-Core Xeon CPU vs. Xeon CPU + Tesla GPU



S-duct with 80K DOF

Hybrid MPI/Open MP for Multi-GPU test

Select Success in Full Range of Continuum CFD

NVIDIA

Range of Time (Flow Regimes) and Spatial Discretisation Schemes

	Structured Grid	Unstructured
Explicit [usually compressible]	UNIVERSITY OF CAMBRIDGE **TurboStream (MBS, FV)** Sandia National Laboratories / Oak Ridge National Laboratory **S3D (FV)** **~15x**	BAE SYSTEMS **SOLAR (Poly, FV)** George Mason University **FEFLO (FE)** **~10x**
Implicit [usually incompressible]	**~5x** Pratt & Whitney A United Technologies Company **Enigma (MBS, FV)** NCSA **DNS (FV)**	**~2x** ACUSIM SOFTWARE **AcuSolve (FE)** Autodesk **Moldflow (FE)**

~x Factors Based on Comparisons with Xeon QC Nehalem CPUs

MBS = multiblock structured
Poly = polyhedral
FV = finite volume
FE = finite element

Particle Methods

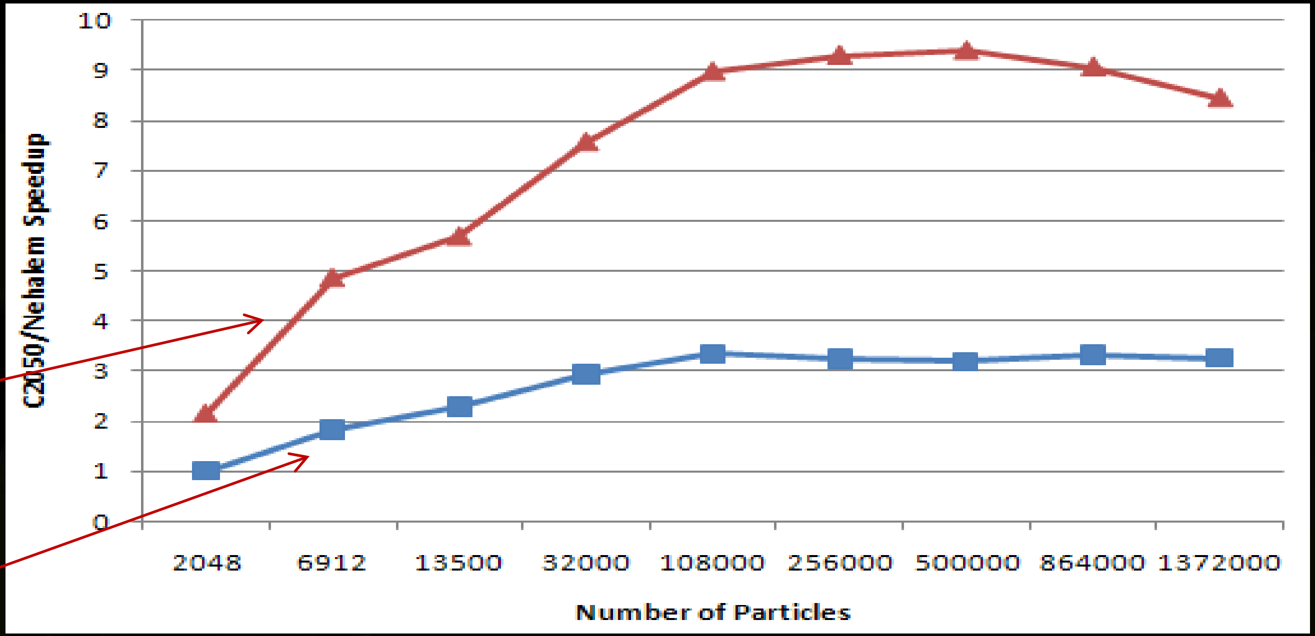
- N-Body like force kernel
- Force calculation in MD/SPH is essentially a SpMV: the neighbor-list is the sparse matrix, position the input vector and force the output vector.
  - Use similar hybrid data structure as in the SpMV case
- Typically memory-bound: design data structure that are optimized for good memory access pattern
- Building neighbor-list is essentially sparse matrix assembly
  - Can be done entirely on GPU

LAMMPS: Leonard-Jones Single GPU Performance

Speedup of a C2050 over 4 core Nehalem

Force+Neighor list

Overall speedup

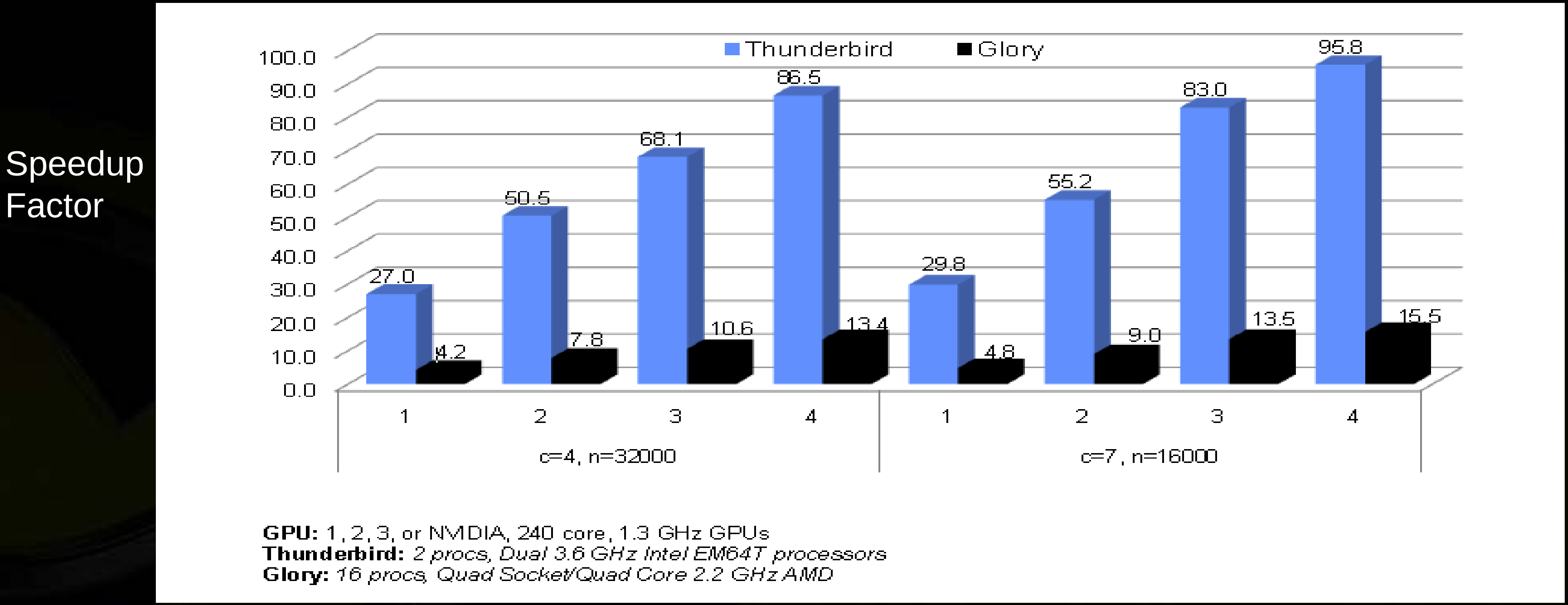


Number of Particles

P. Wang, M. Brown, P. Crozier, 2010

- LJ speedup will be the least significant.
- Overall speedup limited by Amdahl's law.

LAMMPS Gay-Berne Force Performance



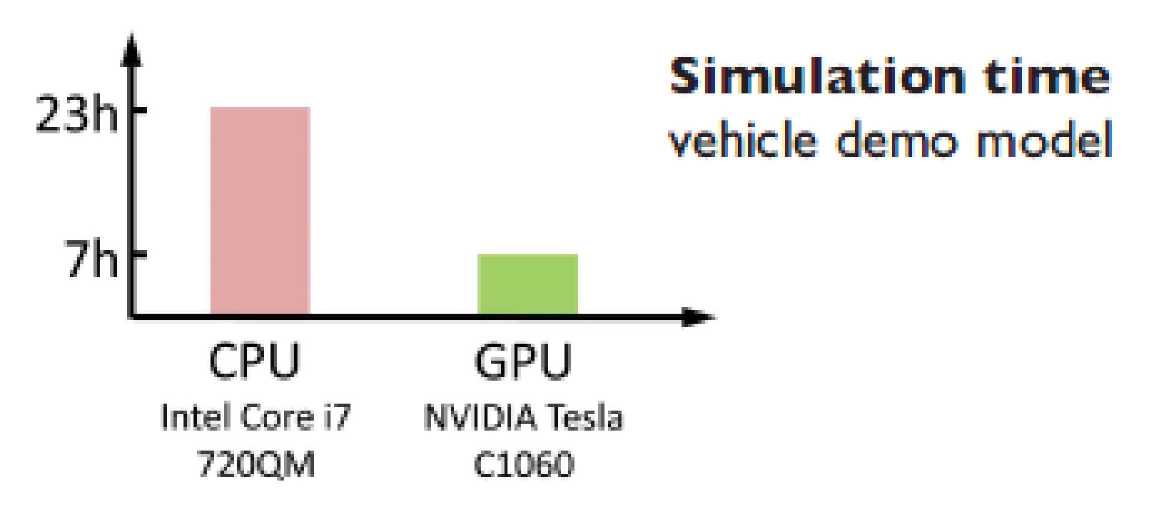
GPU: C1060

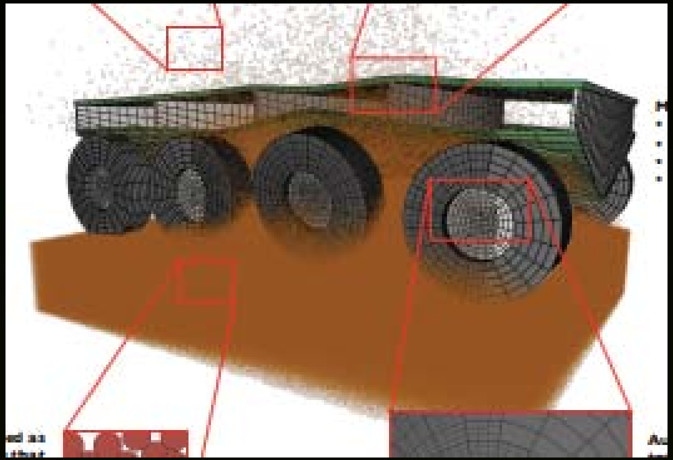
Blue: speedup compared to Dual-core 3.6 GHz Intel EM64T

Black: speedup compared to Quad Socket/Quad Core 2.2 GHz AMD (use 16 cores)

IMPETUS AFEA

- An explicit finite element code for full scale blast simulations
- SPH modeling of liquids





175K finite elements
3M particles

Summary

- For problems that fit, GPU can deliver excellent performance
- All seven scientific computing patterns map well to GPU
- Key for success: expose massively parallelism & may need to redesign data structure

Thank you!

Questions?