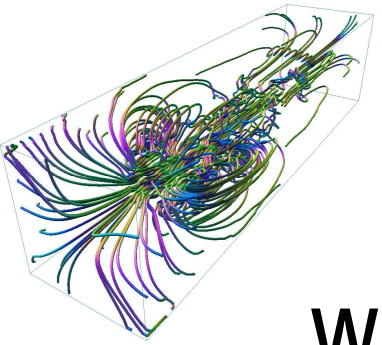
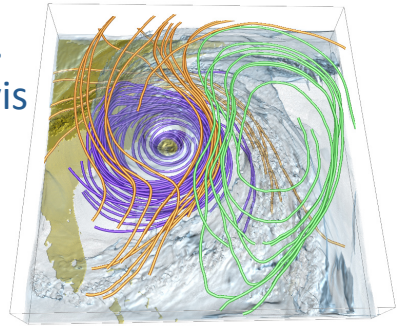
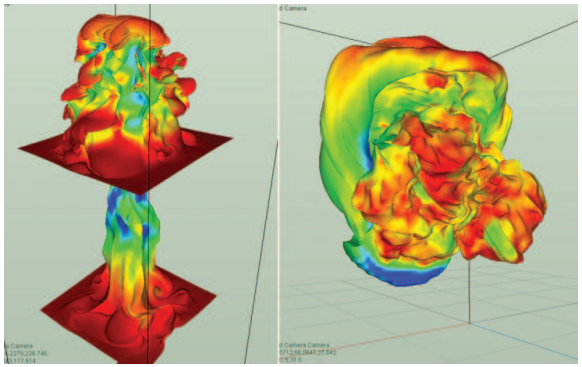
Visualization, C. Ho, J. Wei, K.-L. Ma, UC Davis
Simulations: NCAR



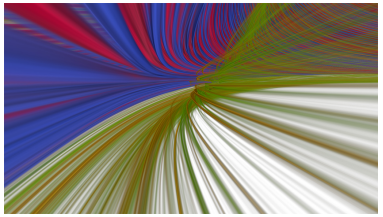


Visualization, S. Yan, N. Max, K.-L. Ma, UC Davis

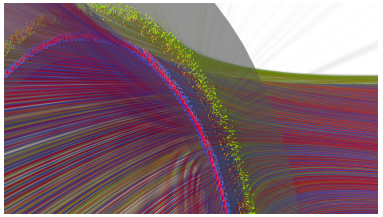


What's New with the GPU? GPUs for Scientific Computing and Visualization

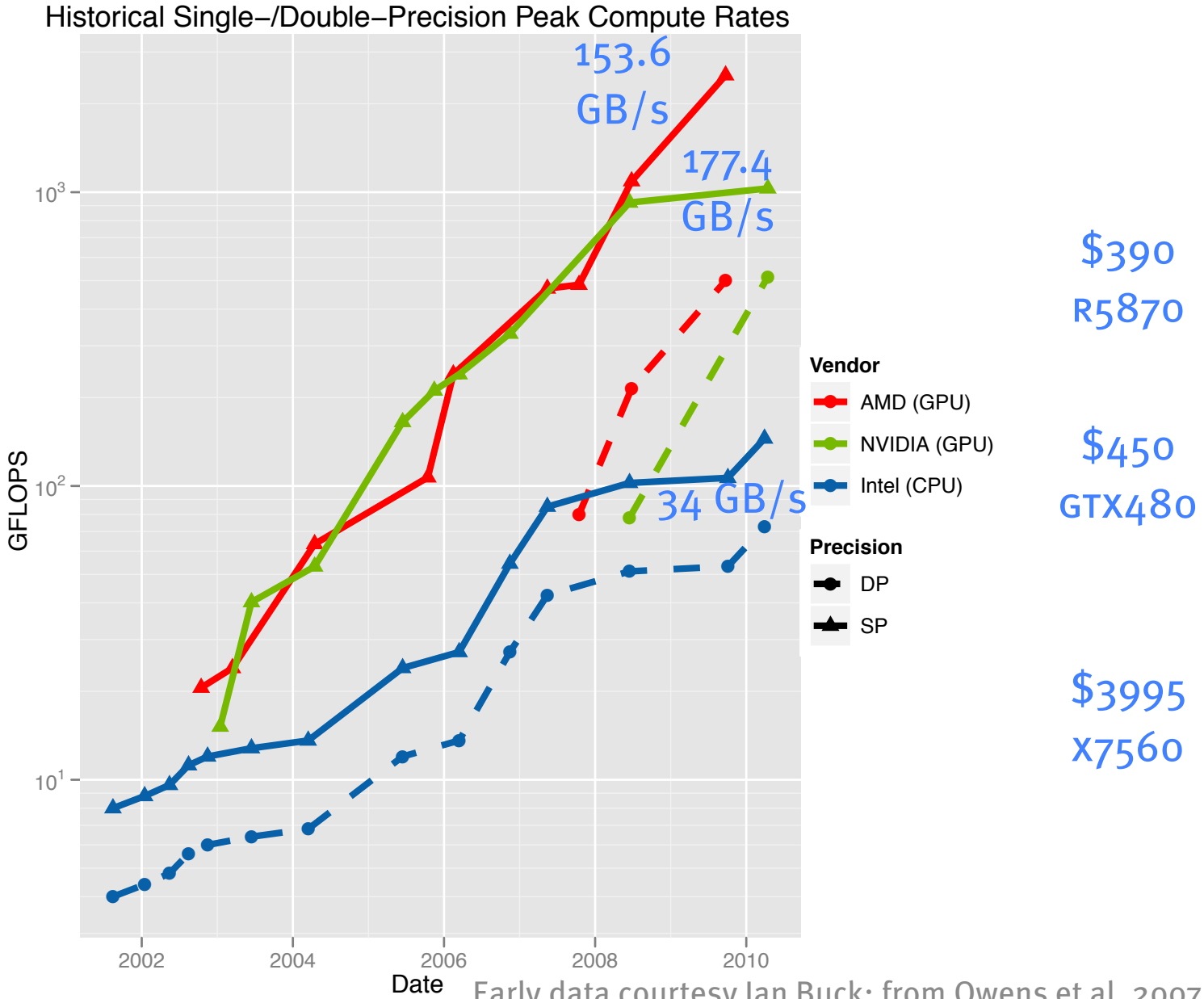
John Owens
Associate Professor, Electrical and Computer Engineering
SciDAC Institute for Ultrascale Visualization
University of California, Davis



Visualization, C. Ho, K.-L. Ma, UC Davis
Simulation, S. Ethier, PPPL



What's Good?

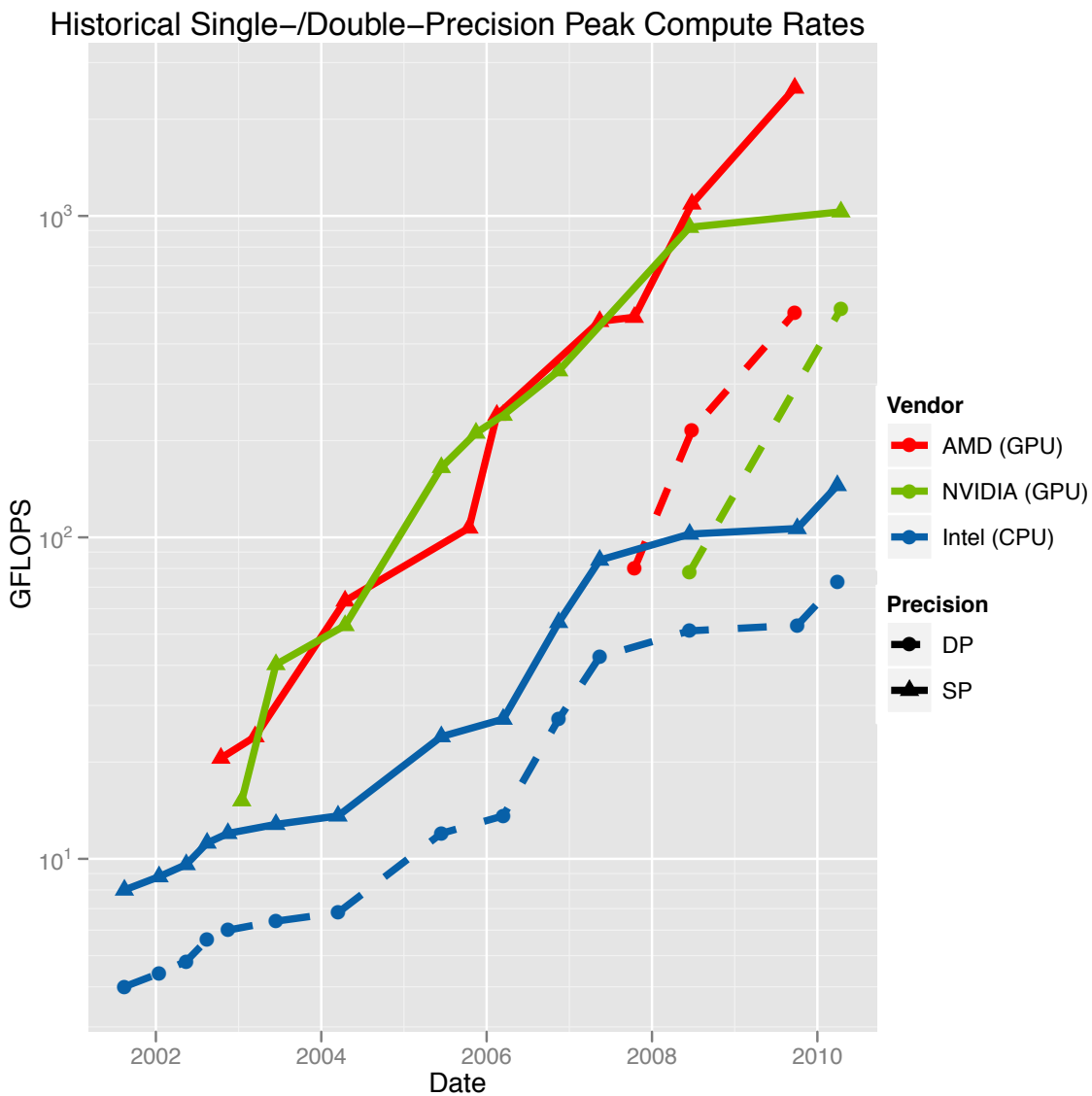


Early data courtesy Ian Buck; from Owens et al. 2007 [CGF]

What’s Bad?

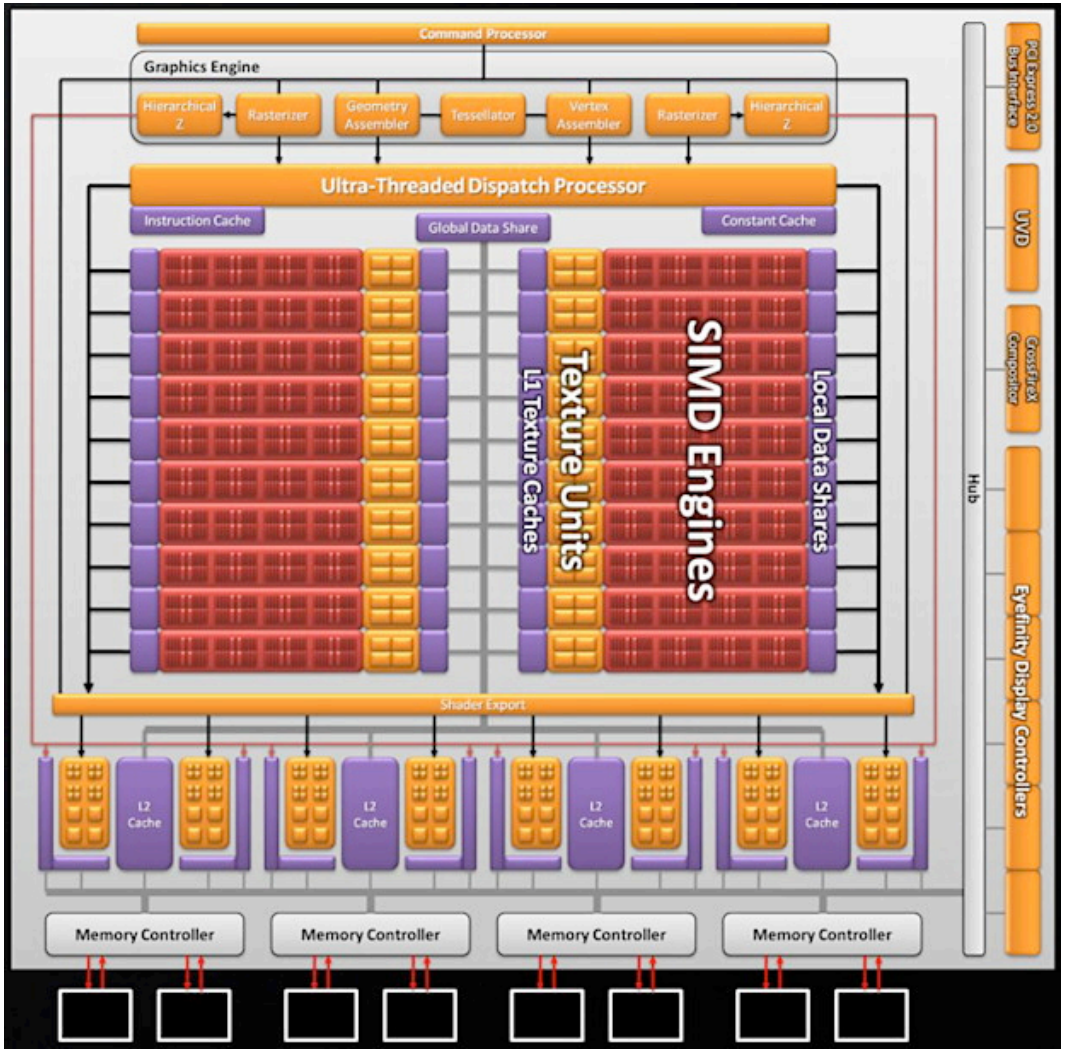
What's new?

- Double precision



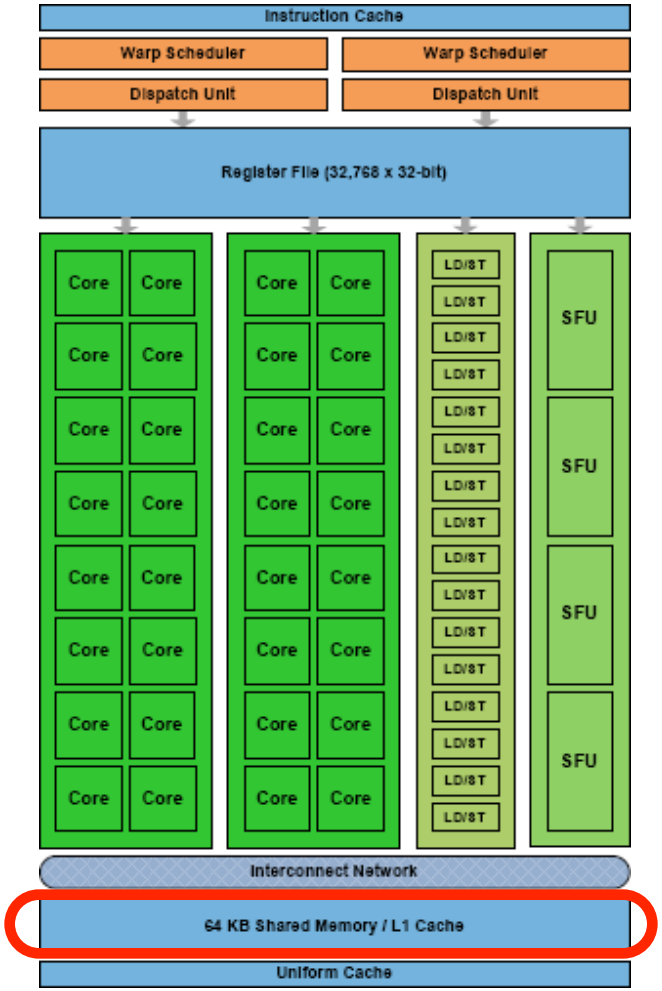
What's new?

- Double precision
- Fast atomics



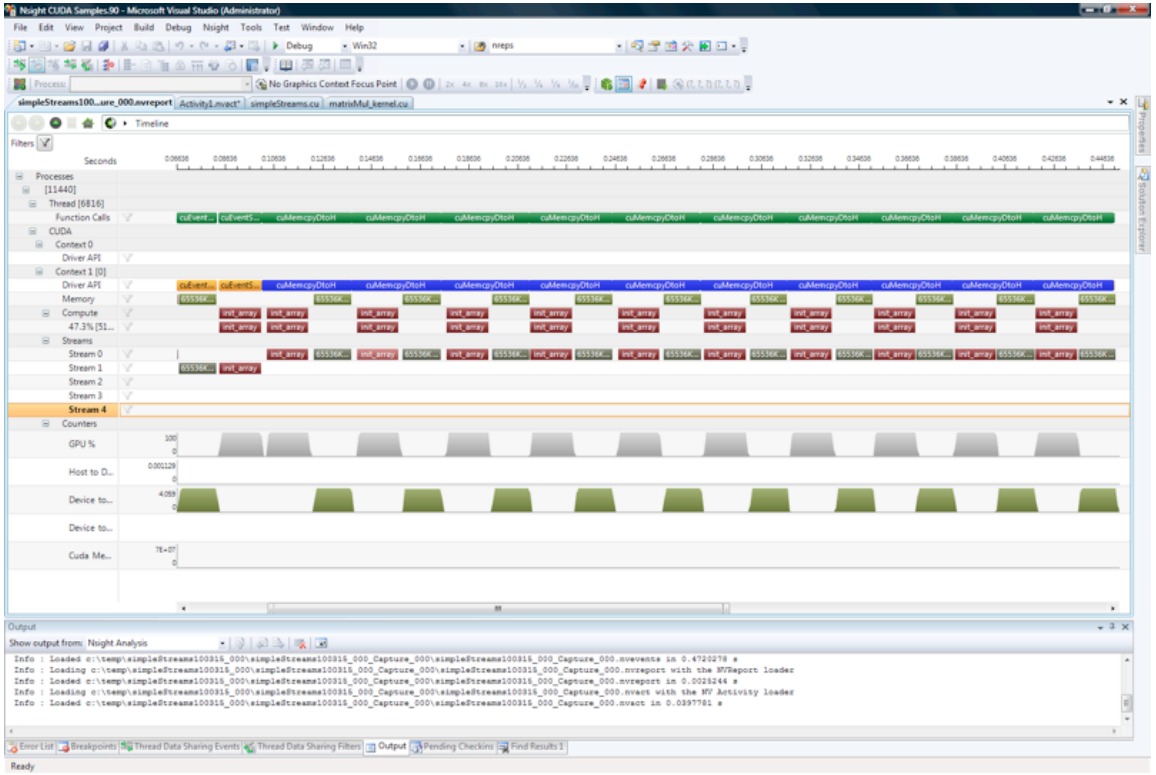
What's new?

- Double precision
- Fast atomics
- Hardware cache & ECC



What’s new?

- Double precision
- Fast atomics
- Hardware cache & ECC
- (CUDA) debuggers & profilers



Intel ISCA Paper (June 2010)

Debunking the 100X GPU vs. CPU Myth: An Evaluation of Throughput Computing on CPU and GPU

Victor W Lee[†], Changkyu Kim[†], Jatin Chhugani[†], Michael Deisher[†], Daehyun Kim[†], Anthony D. Nguyen[†], Nadathur Satish[†], Mikhail Smelyanskiy[†], Srinivas Chennupaty[⋆], Per Hammarlund[⋆], Ronak Singhal[⋆] and Pradeep Dubey[†]

victor.w.lee@intel.com

[†]Throughput Computing Lab, Intel Corporation

[⋆]Intel Architecture Group, Intel Corporation

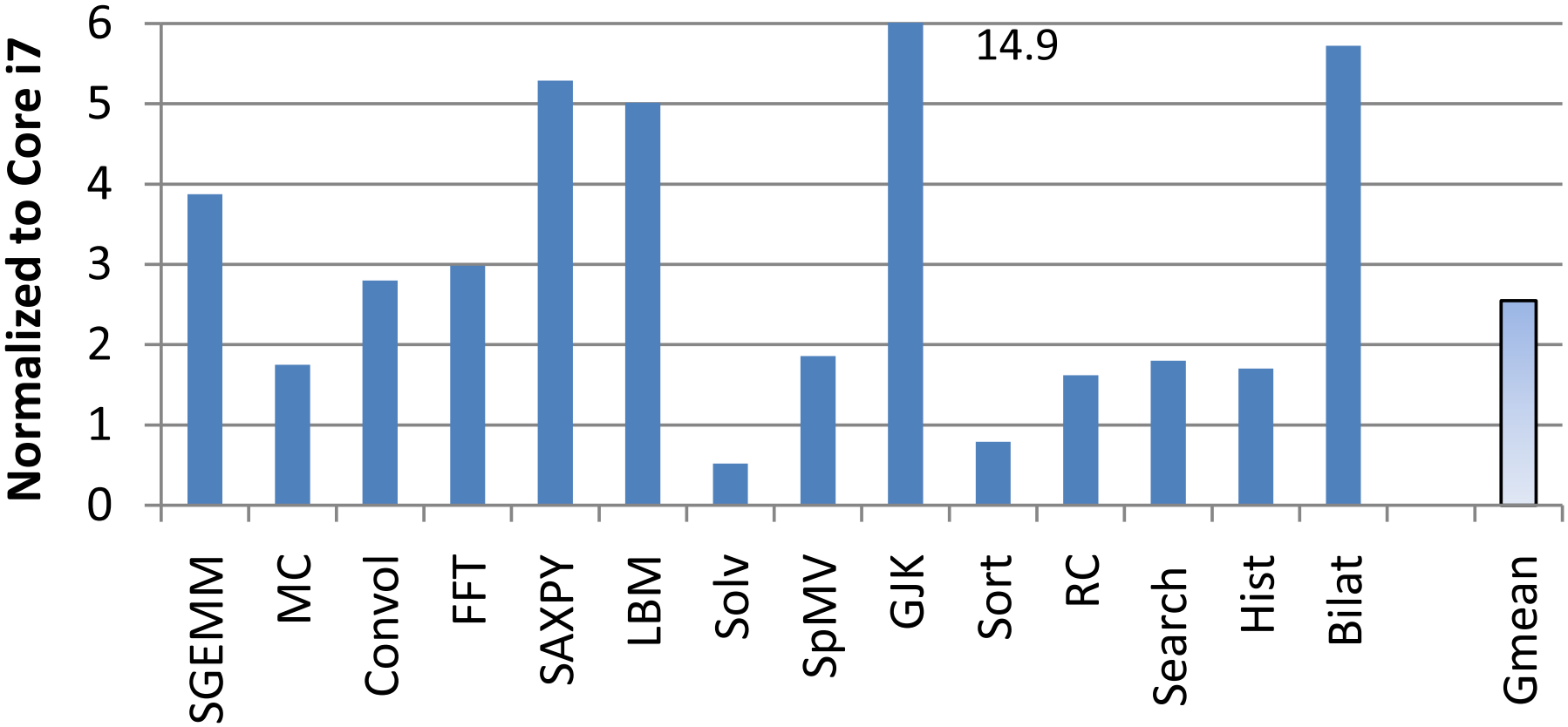
ABSTRACT

Recent advances in computing have led to an explosion in the amount of data being generated. Processing the ever-growing data in a timely manner has made throughput computing an important aspect for emerging applications. Our analysis of a set of important throughput computing kernels shows that there is an ample amount of parallelism in these kernels which makes them suitable for to-

The past decade has seen a huge increase in digital content as more documents are being created in digital form than ever before. Moreover, the web has become the medium of choice for storing and delivering information such as stock market data, personal records, and news. Soon, the amount of digital data will exceed exabytes (10^{18}) [31]. The massive amount of data makes storing, cataloging, processing, and retrieving information challenging.

- Also see Vuduc et al., “On the Limits of GPU Acceleration”, HotPar 2010 (June 2010) [and yesterday afternoon]

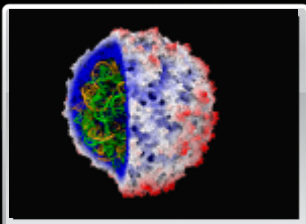
Top-Level Results



CUDA Successes

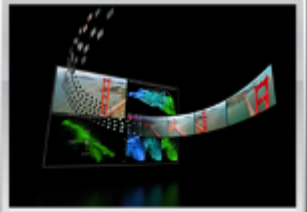
146X

Medical Imaging
U of Utah



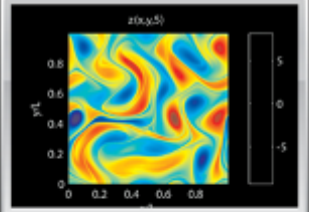
36X

Molecular Dynamics
U of Illinois



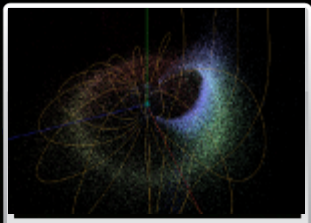
18X

Video Transcoding
Elemental Tech



50X

Matlab Computing
AccelerEyes

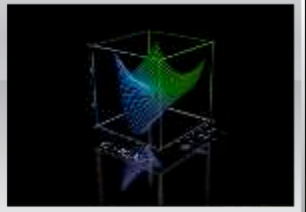


100X

Astrophysics
RIKEN

149X

Financial simulation
Oxford



47X

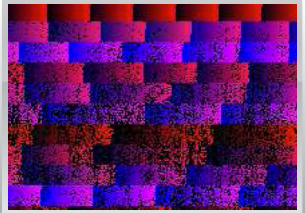
Linear Algebra
Universidad Jaime

20X

3D Ultrasound
Techniscan

130X

Quantum Chemistry
U of Illinois



30X

Gene Sequencing
U of Maryland

(c) 2010 NVIDIA Corporation

[courtesy David Luebke, NVIDIA]

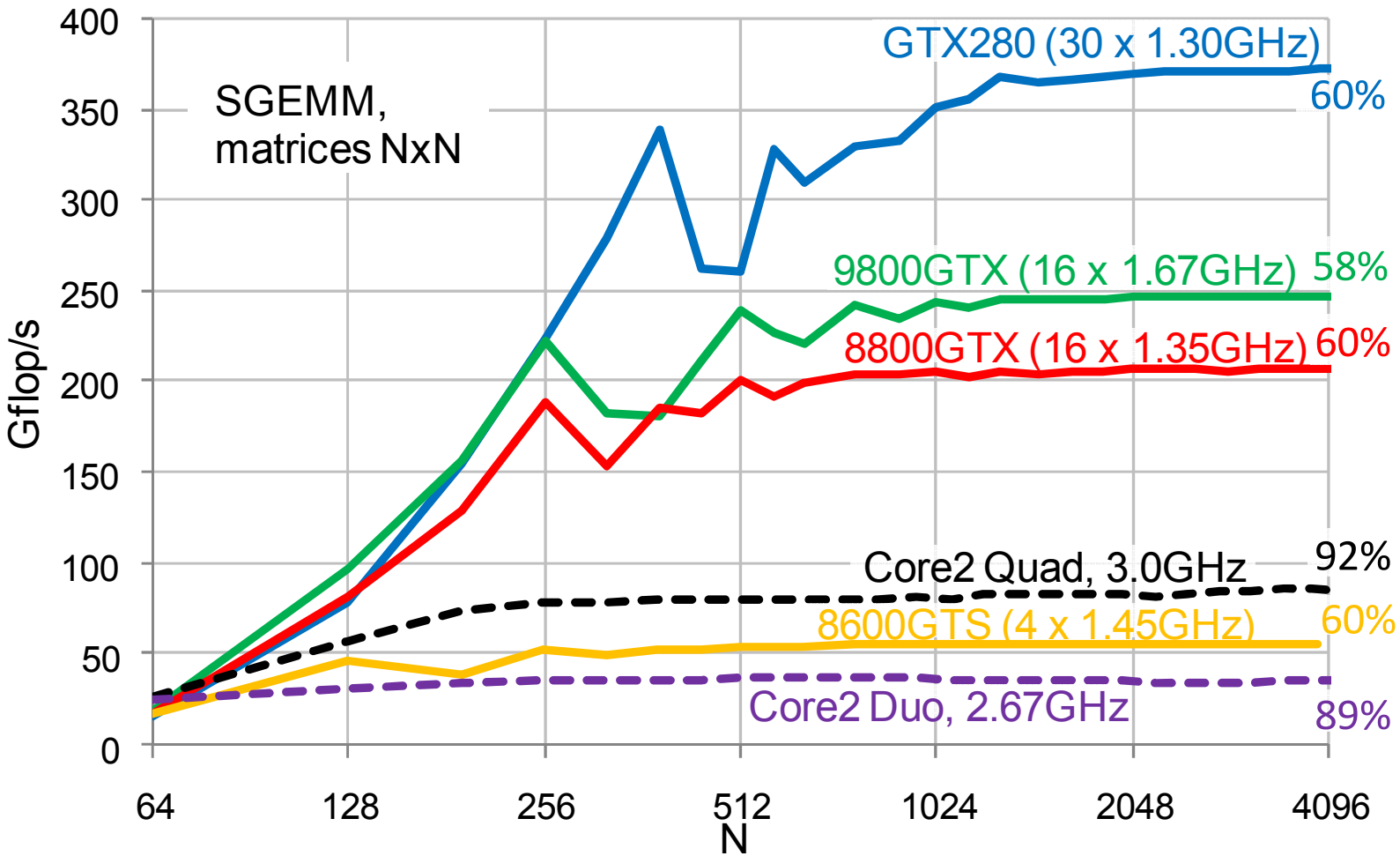
13 Dwarfs

- 1. Dense Linear Algebra
- 2. Sparse Linear Algebra
- 3. Spectral Methods
- 4. N-Body Methods
- 5. Structured Grids
- 6. Unstructured Grids
- 7. MapReduce
- 8. Combinational Logic
- 9. Graph Traversal
- 10. Dynamic Programming
- 11. Backtrack and Branch-and-Bound
- 12. Graphical Models
- 13. Finite State Machines

13 Dwarfs

- 1. Dense Linear Algebra
- 2. Sparse Linear Algebra
- 3. Spectral Methods
- 4. N-Body Methods
- 5. Structured Grids
- 6. Unstructured Grids
- 7. MapReduce
- 8. Combinational Logic
- 9. Graph Traversal
- 10. Dynamic Programming
- 11. Backtrack and Branch-and-Bound
- 12. Graphical Models
- 13. Finite State Machines

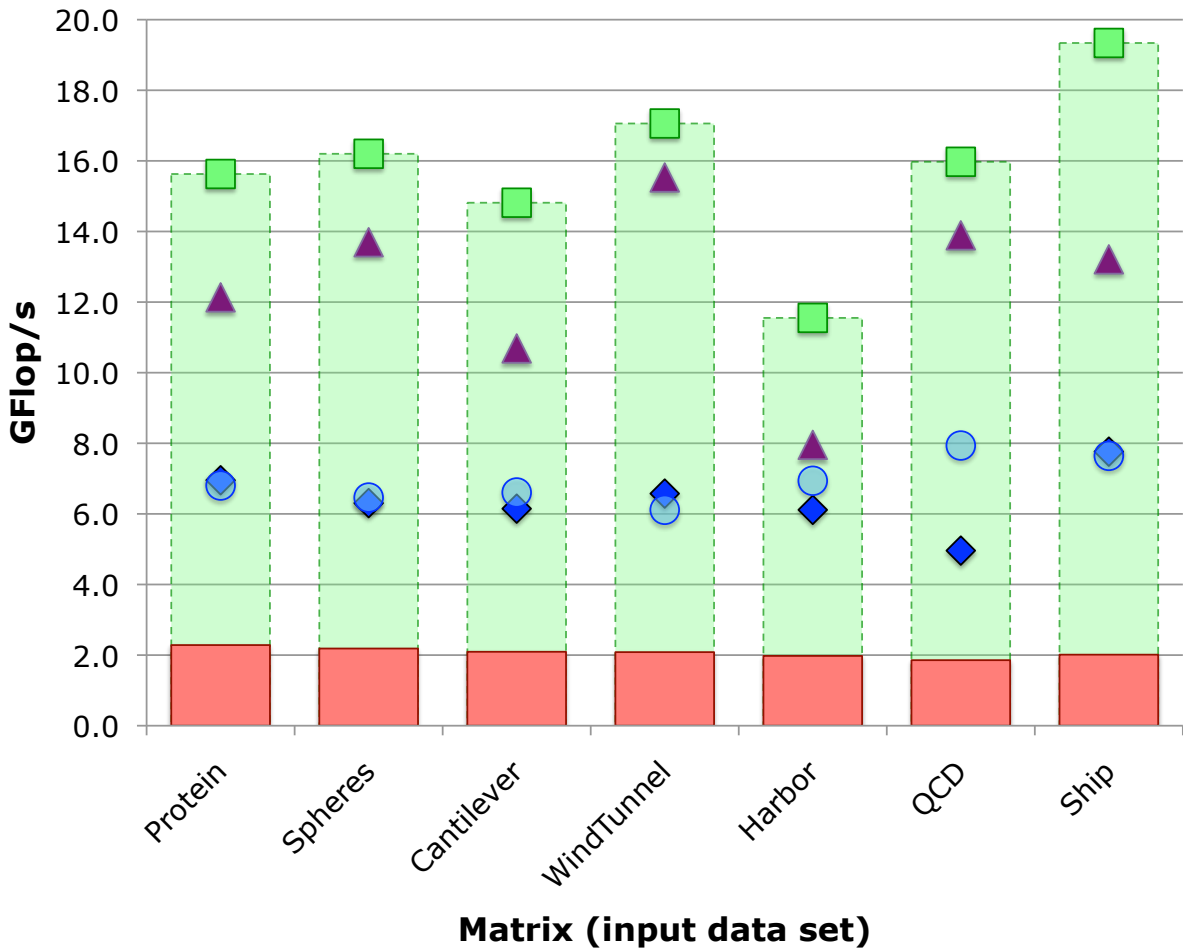
Dense Matrix Multiply



- *Benchmarking GPUs to Tune Dense Linear Algebra* by Vasily Volkov and James W. Demmel, Supercomputing 2008

Tuned SpMV

Best Single-node Double-precision SpMV



Our code, PPoPP'10
1.1 to **1.5x**

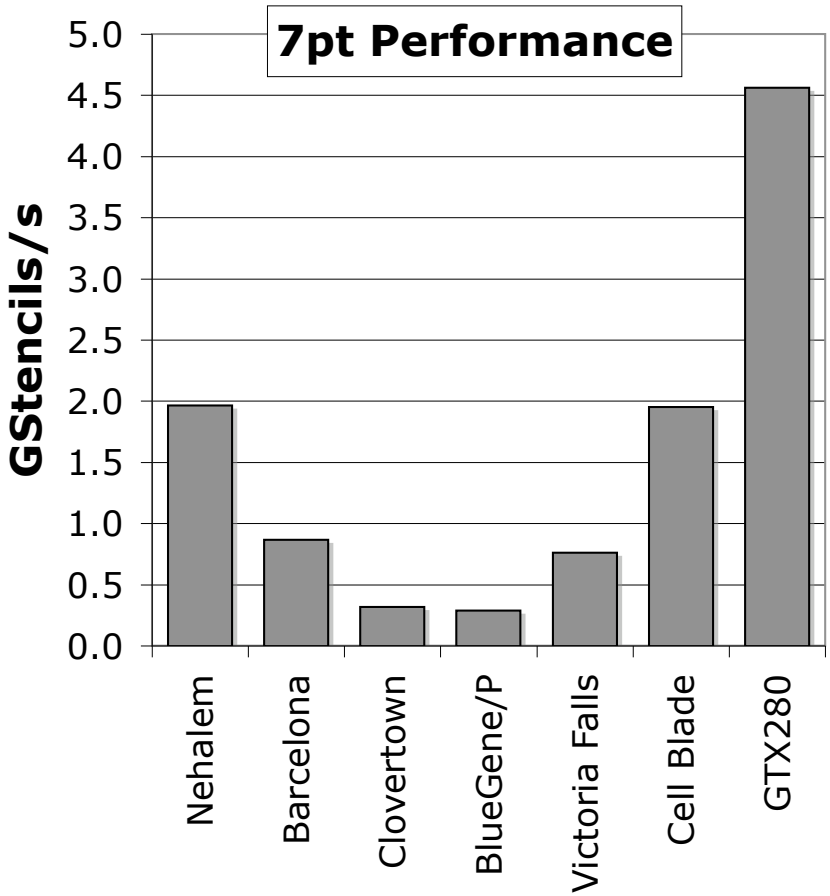
NVIDIA, SC'09
GTX285

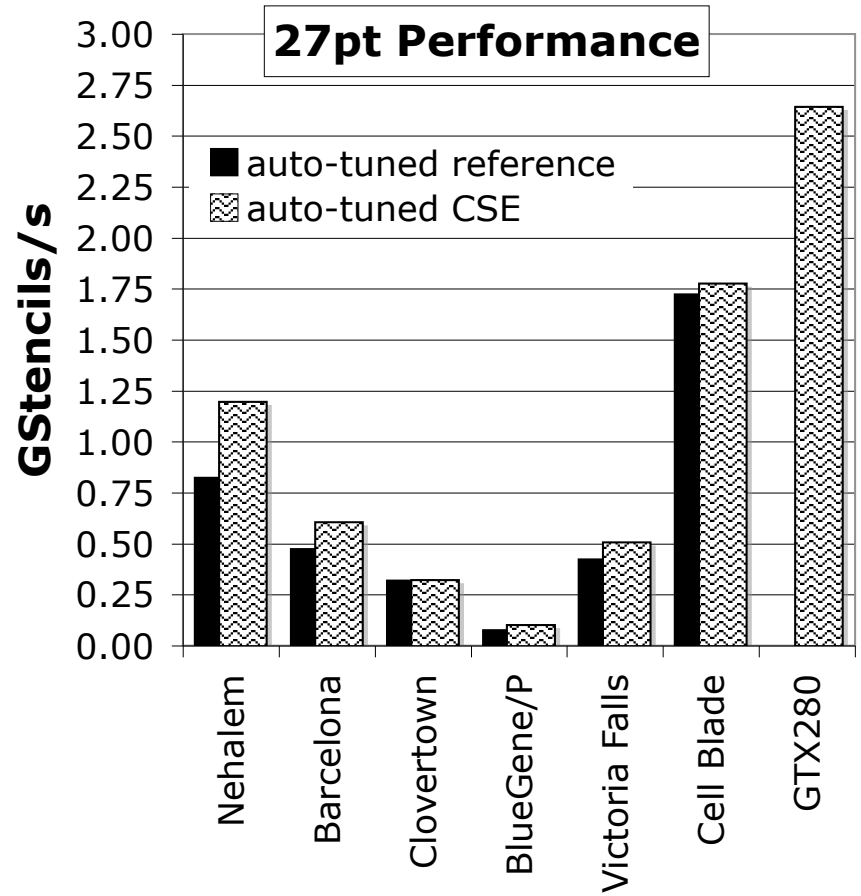
ParCo'09
Circles:
Tuned Nehalem
Diamonds:
STI PowerXCell 8i

Baseline:
Intel Nehalem,
Multithreaded
But Untuned
(8 cores)

- Richard Vuduc, Aparna Chandramowlishwaran, Jee Choi, Murat (Efe) Guney, and Aashay Shringarpure. *On the Limits of GPU Acceleration*, HotPar '10 (June 2010).

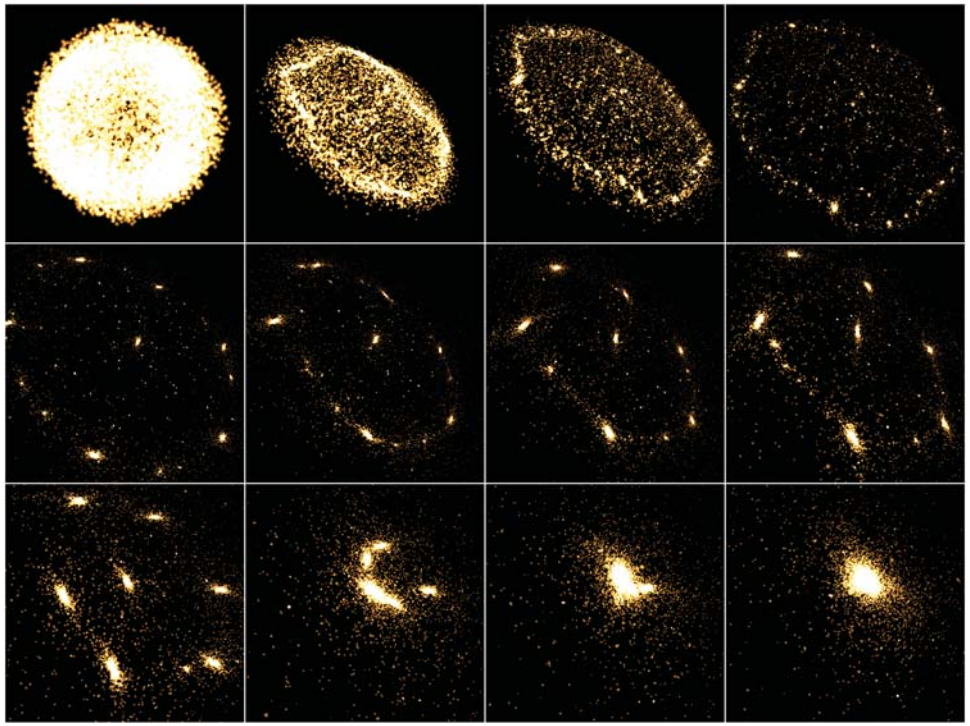
Stencil Computations





- "Auto-tuning Stencil Computations on Multicore and Accelerators" by Kaushik Datta, Samuel Williams, Vasily Volkov, Jonathan Carter, Leonid Oliker, John Shalf, and Katherine Yelick. To appear in *Scientific Computing with Multicore and Accelerators*, January 2011.

N-Body



GPU: Approaches theoretical compute peak, uses fast square-root and reciprocal hw. >10B gravitational forces/s, 200+ GFLOPS. (Nyland et al., 2007)

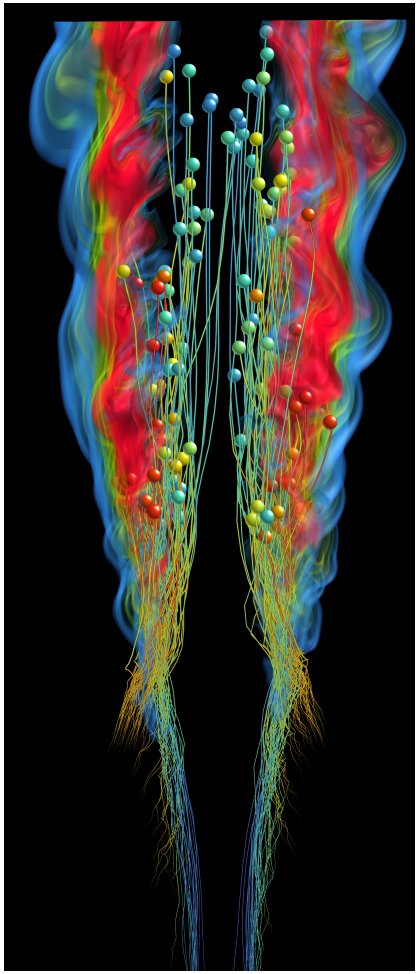
CPU: SSE implementation achieves 3.8 GFLOPS (Elsen et al., 2006)

- F*ast N-Body Simulation with CUDA* by Lars Nyland, Mark Harris, and Jan Prins, GPU Gems 3, 2007.

UltraVis
SciDAC Institute for Ultrascale Visualization

In-situ Analysis and Visualization

- Feature extraction & data reduction
- Analysis for uncertainty quantification
- Rendering particles, fields, & extracts
- Visual monitoring and steering
- Optimal parallel image composition algorithms (UCD and Argonne National Lab)
- GPGPU library (CUDPP, Owens et al.)



Visualization: H. Yu, Sandia; J. Wei, K.-L. Ma, UC Davis
Simulation: J. Chen, Sandia

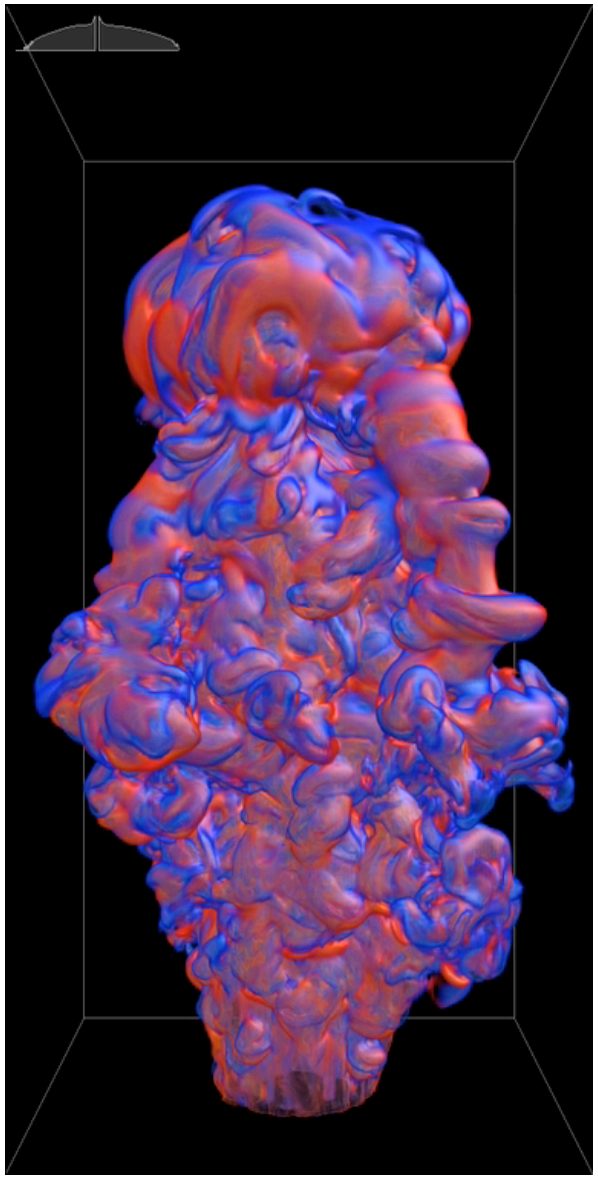
Lagrangian Visualization & Analysis of Fluid Flow

- Example:
  Finite-Time Lyapunov Exponents
  - Measures exponential separation rate between neighboring particles
  - Visualizes forward time and backward time exponents with Volume Rendering
  - Formation of turbulent structures can be observed directly
  - Jet flow:
    256×1282, 735 timesteps, ~3.1B particles
  - Time to movie:
    CPU ~60min (compute bound),
    GPU ~4min (I/O bound)

[courtesy Christoph Garth, UC Davis]

Lagrangian Visualization & Analysis of Fluid Flow

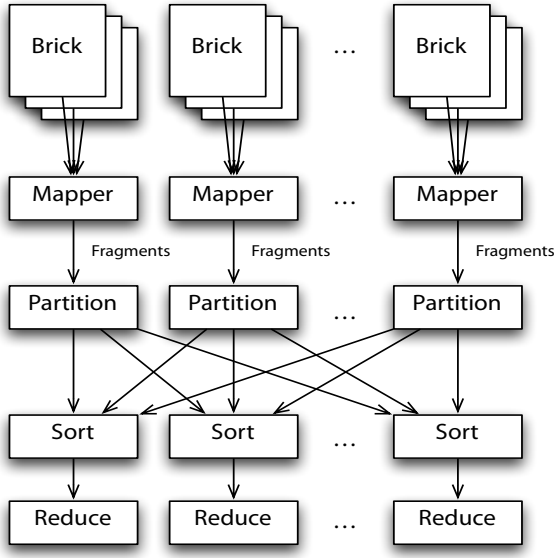
- Example:
  Finite-Time Lyapunov Exponents
  - Measures exponential separation rate between neighboring particles
  - Visualizes forward time and backward time exponents with Volume Rendering
  - Formation of turbulent structures can be observed directly
  - Jet flow:
    256×1282, 735 timesteps, ~3.1B particles
  - Time to movie:
    CPU ~60min (compute bound),
    GPU ~4min (I/O bound)

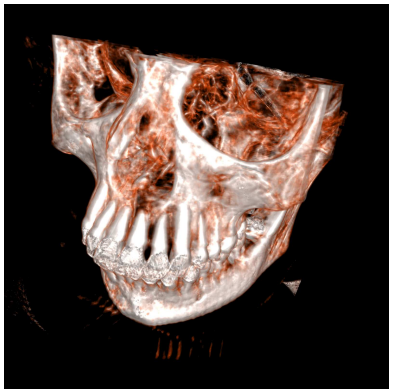


[courtesy Christoph Garth, UC Davis]

GPU Volume Rendering

- Use MapReduce parallel programming model
- Lessons:
  - Too few nodes: too much work per node
    - Sweet spot for us: 8 nodes, 1024^3
  - Too many nodes: too much communication
  - Can (easily) afford more computation per ray
  - Disk I/O would dominate runtime

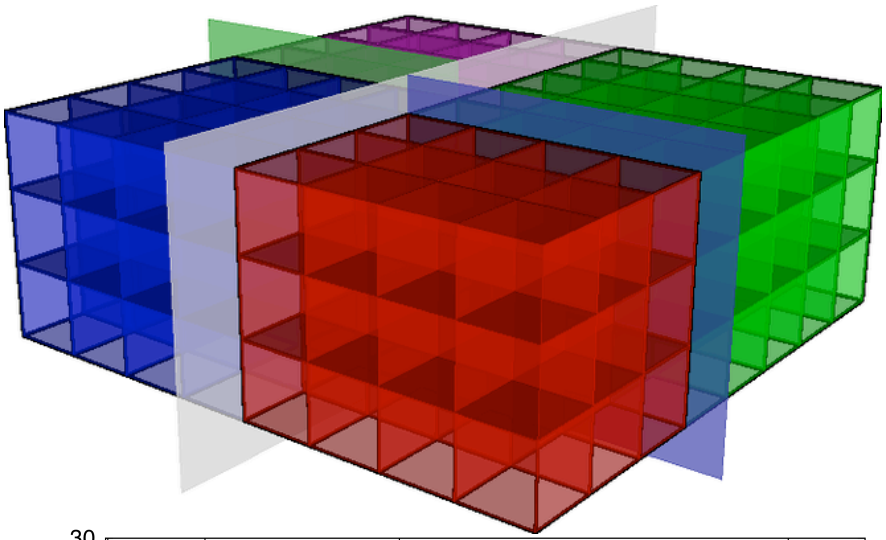


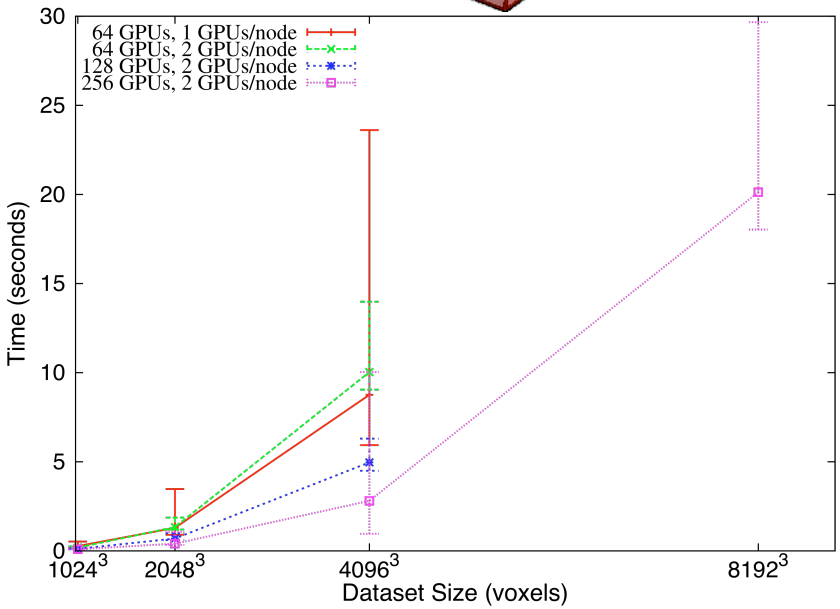


Multi-GPU Volume Rendering using MapReduce by Jeff A. Stuart, Cheng-Kai Chen, Kwan-Liu Ma, John D. Owens. MapReduce '10, June 2010.

GPU Volume Rendering

- Highest scalability of GPU-based volume renderers
- *k*-d trees for dynamic load-balancing
- ~2 frames/second for 2048^3 volumes
- Integrated into VisIt





Large Data Visualization on Distributed Memory Multi-GPU Clusters by Thomas Fogal, Hank Childs, Siddharth Shankar, Jens Krüger, R. Daniel Bergeron and Philip Hatcher. High Performance Graphics '10, June 2010.

Goal: Sparse Volume Rendering

- Challenge: Real-time rendering of data that is sparse in the analysis space
- Production vis packages don't offer scalable multi-GPU sparse volume renderers
  - Integrate with query-driven visualization, image compositing, parallel I/O
- Ultravis lead: Jian Huang, Tennessee

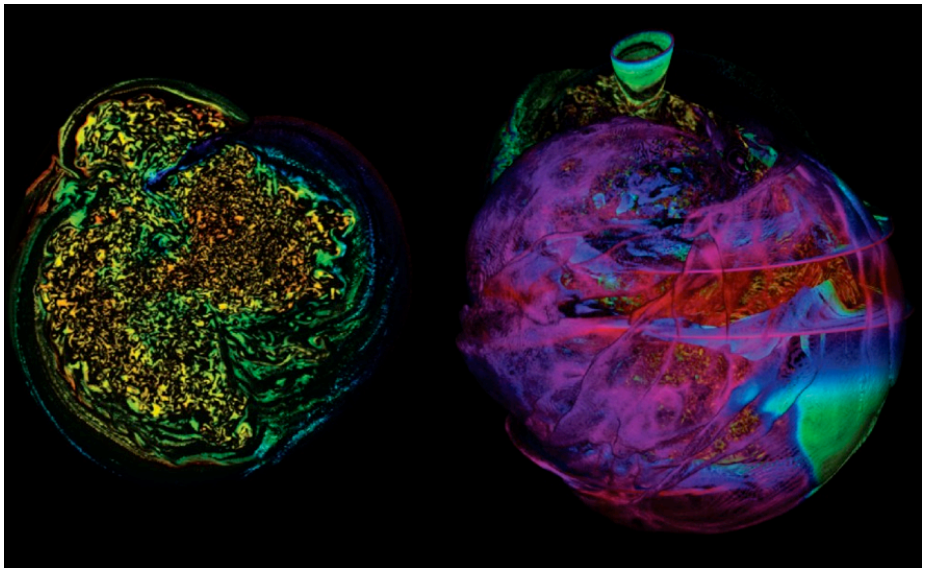


Figure from Terascale Supernova Initiative, courtesy Jian Huang

Goal: Adaptive Volume Rendering

- Goal: Visualize volumes with varying resolution
- Challenges: Sampling at boundaries, load distribution

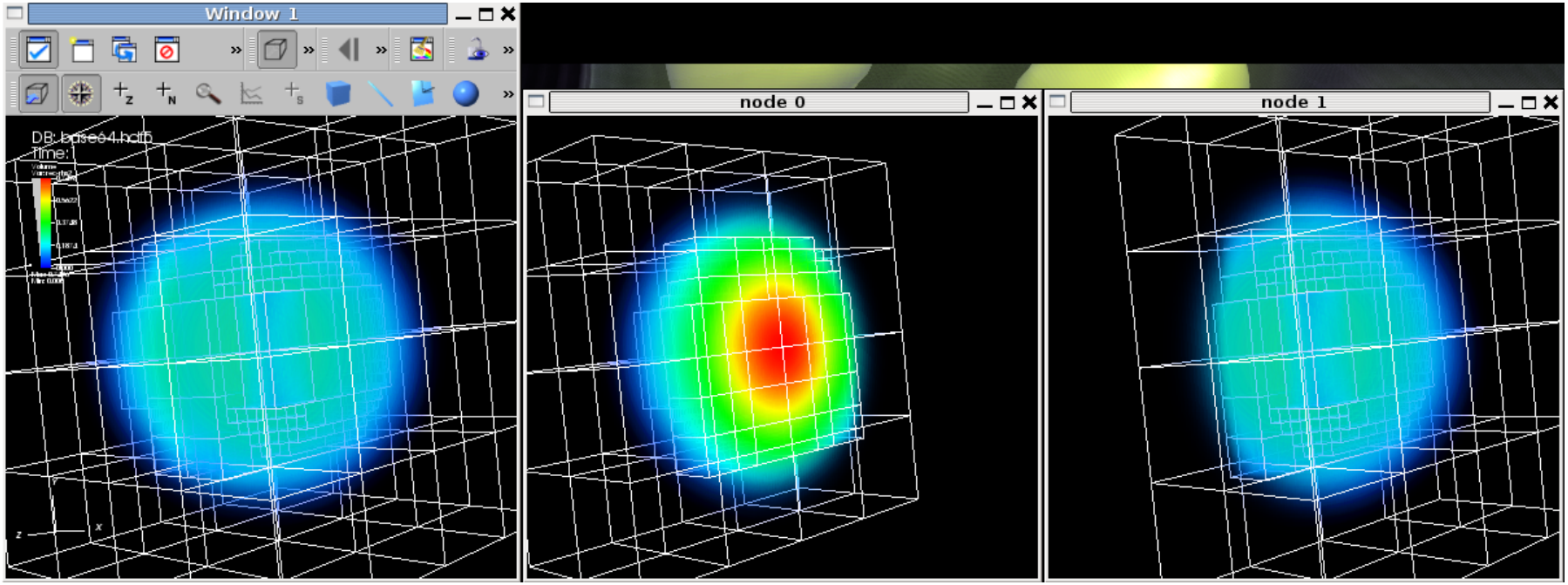
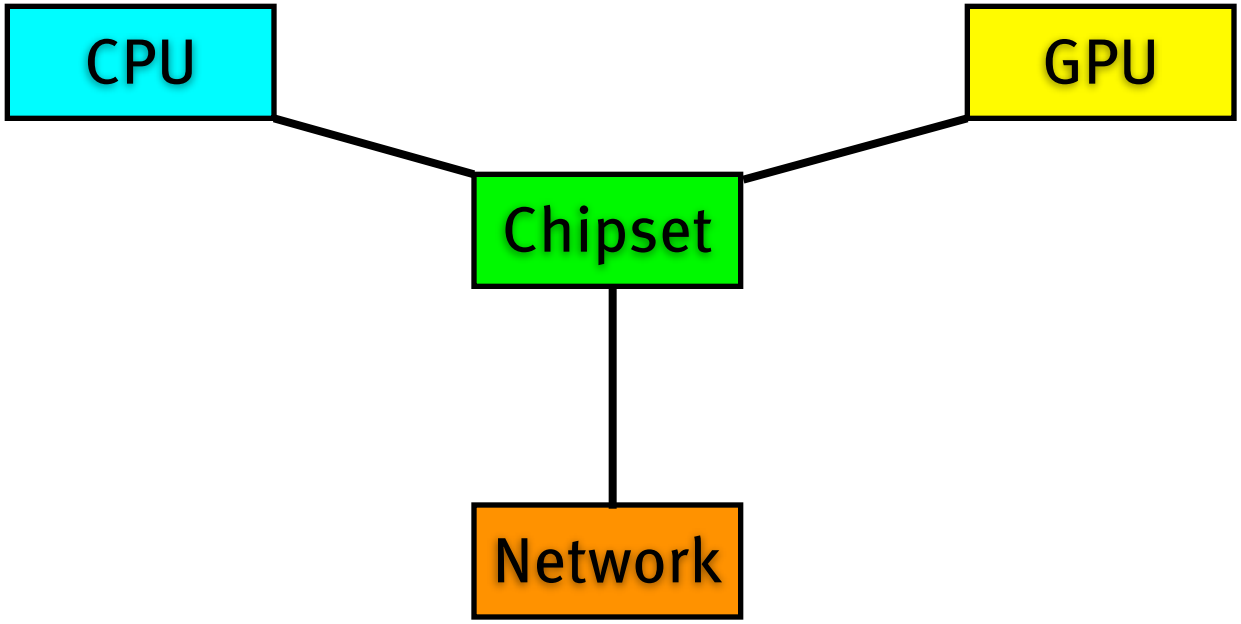
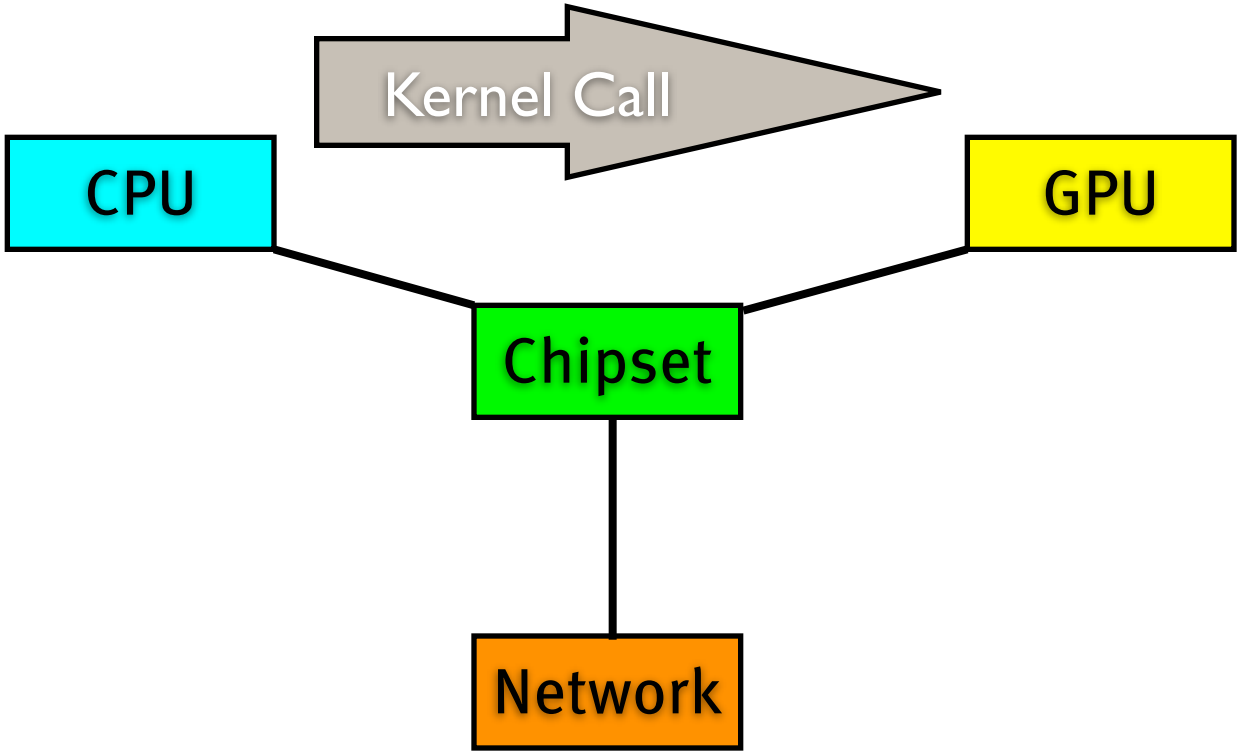


Figure courtesy Thomas Fogel

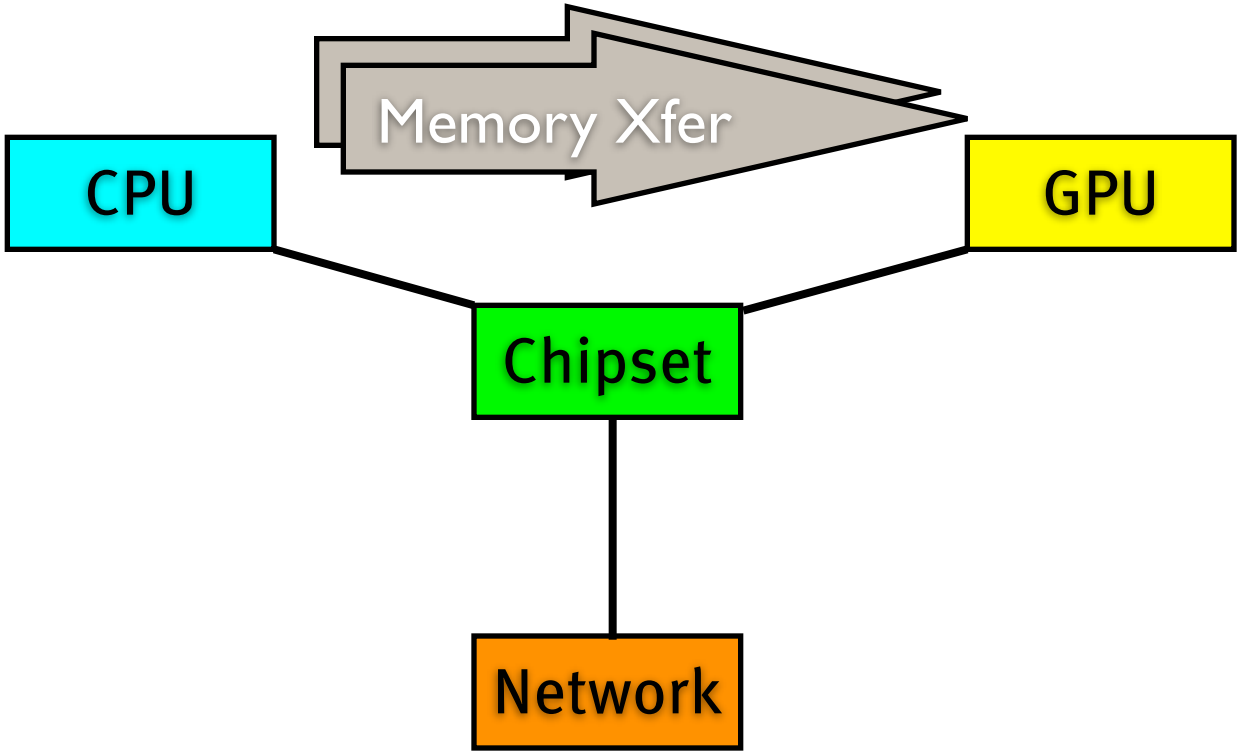
A Modern Computer



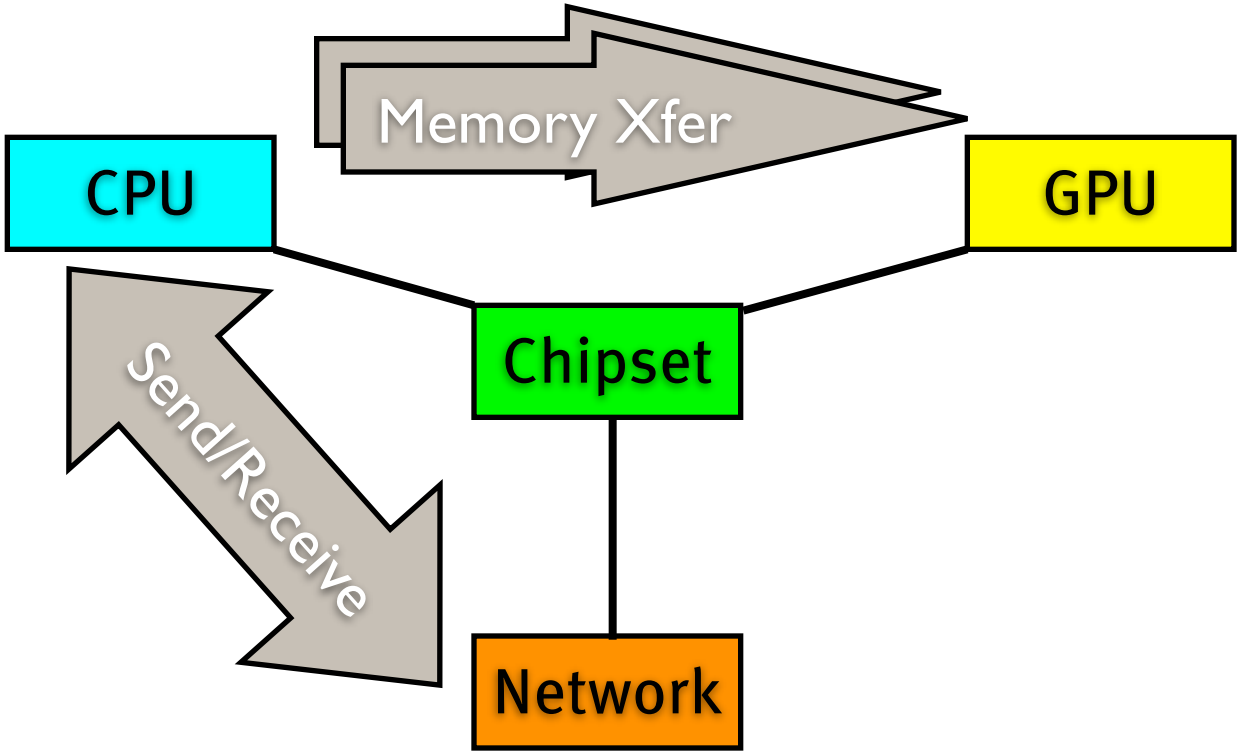
A Modern Computer



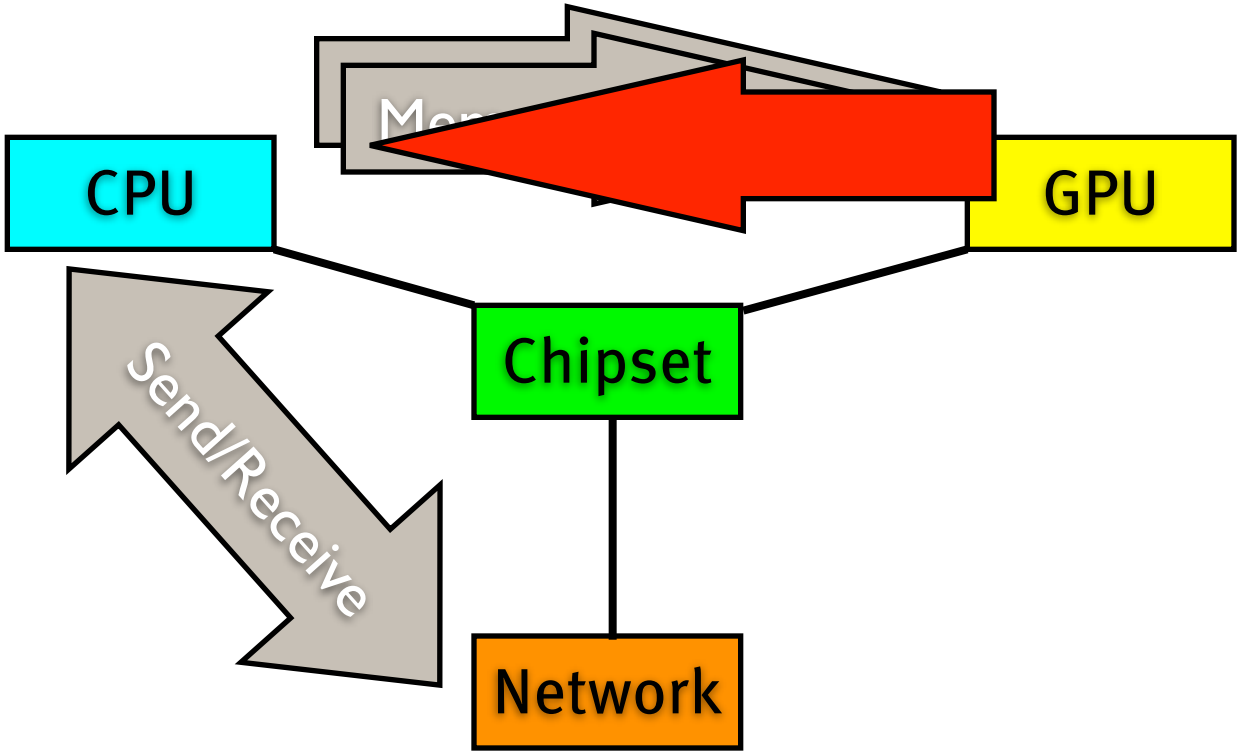
A Modern Computer



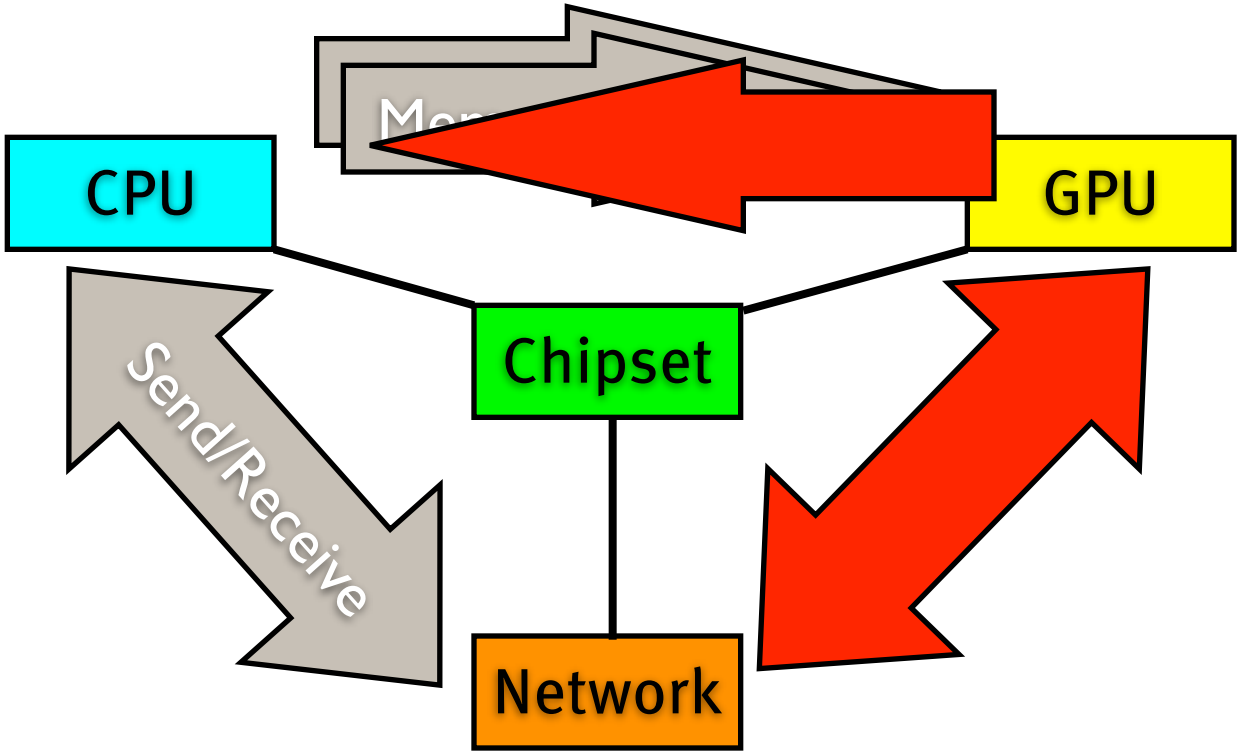
A Modern Computer



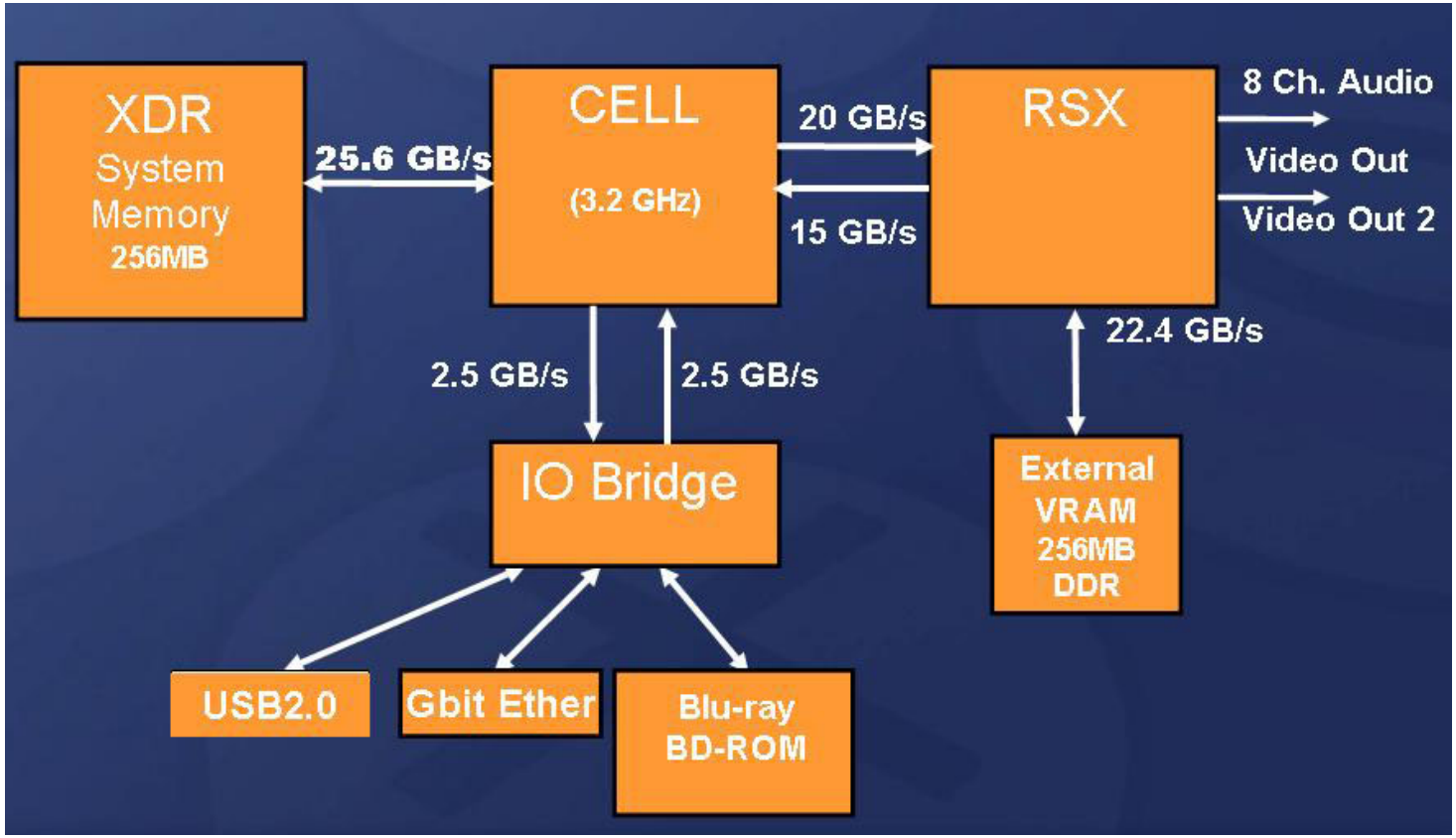
A Modern Computer



A Modern Computer



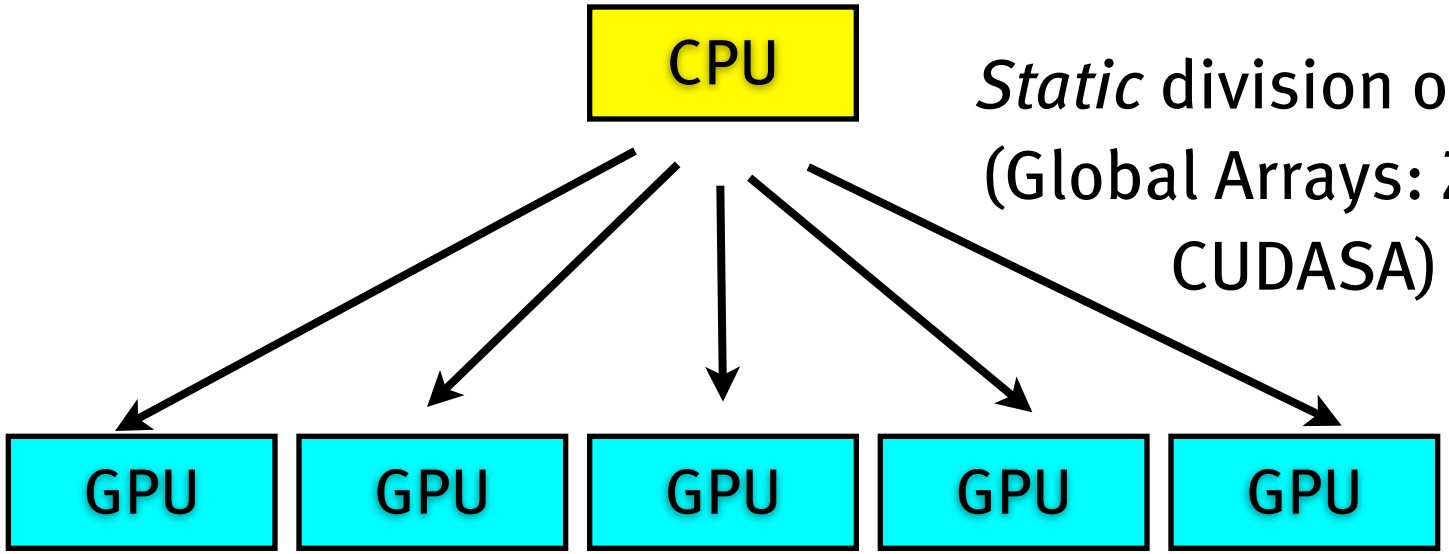
Fast & Flexible Communication



- CPUs are good at creating & manipulating data structures?
- GPUs are good at accessing & updating data structures?

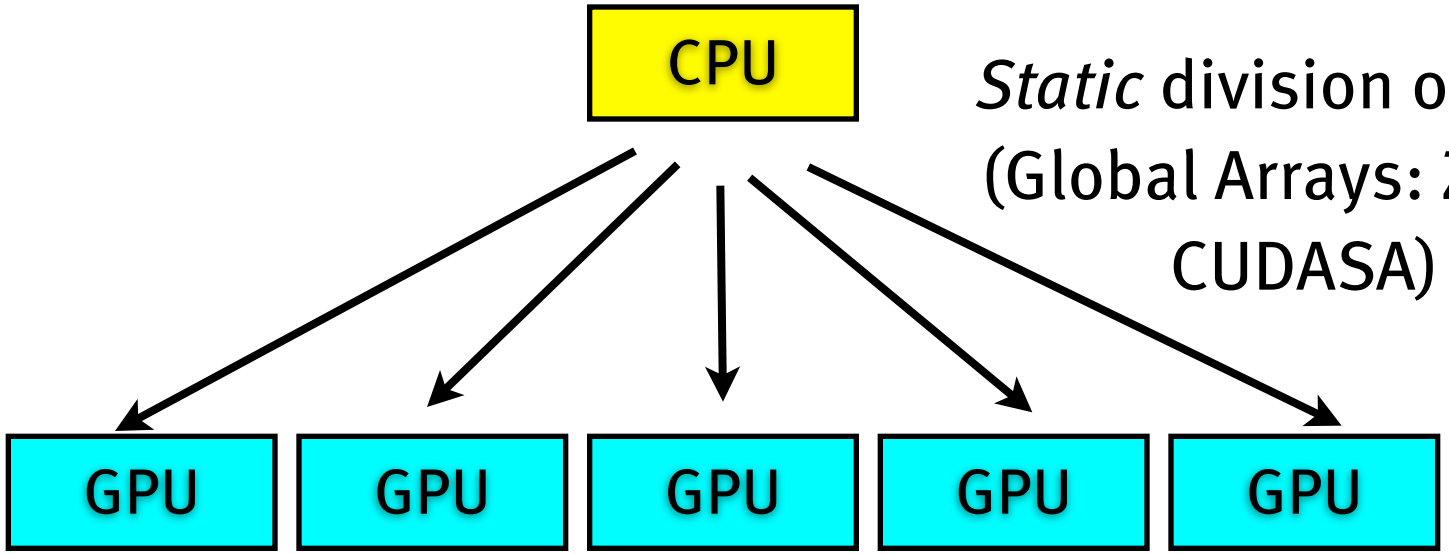
http://www.watch.impress.co.jp/game/docs/20060329/3dps303.jpg

Structuring Multi-GPU Programs



Static division of work (Global Arrays: Zippy, CUDASA)

Structuring Multi-GPU Programs

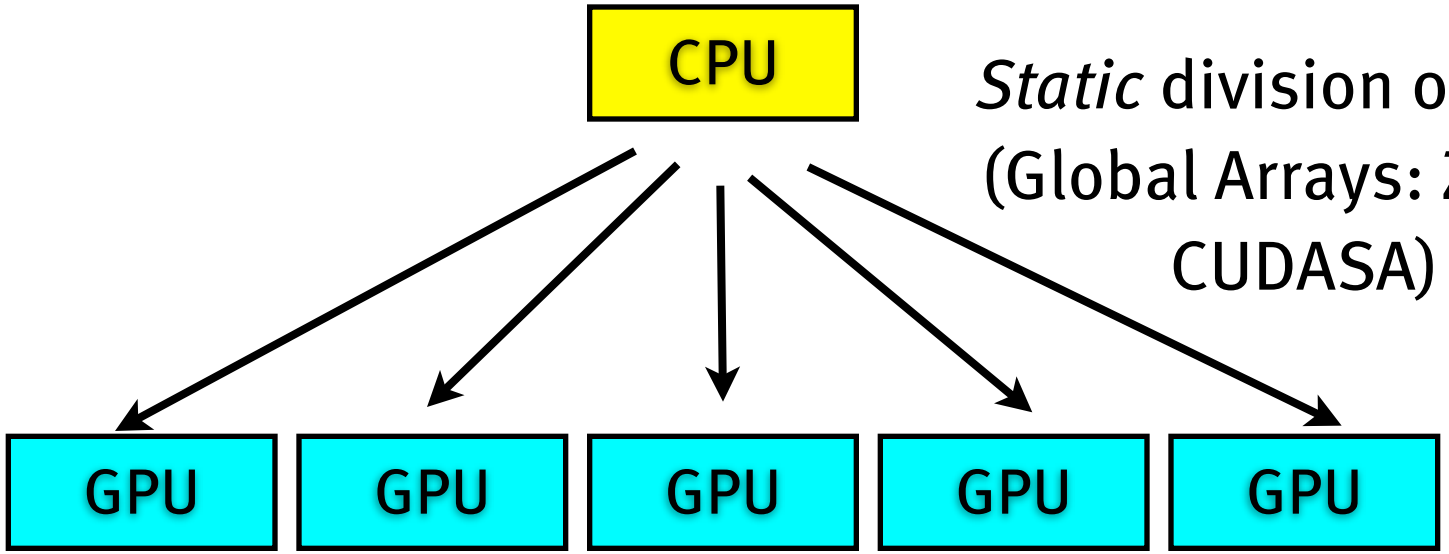


Static division of work (Global Arrays: Zippy, CUDASA)

Want to run on GPU:

```
if (foo == true) {
  GPU[x][bar] = baz;
} else {
  bar = GPU[y][baz];
}
```

Structuring Multi-GPU Programs



Static division of work (Global Arrays: Zippy, CUDASA)

Want to run on GPU:

```
if (foo == true) {
  GPU[x][bar] = baz;
} else {
  bar = GPU[y][baz];
}
```

Instead, *GPU as slave*.

Goal: GPU as first-class citizen.

Making Development Easier

- Better tools
- Higher-level / domain-specific languages
- Robust self-tuning

- Benchmarks

Porting Existing Codes

- Static analysis
  - Hotspot identification and transformation
  - Profiling
  - Code transformations
  - Portable self-tuning
- Dynamic techniques
  - Scheduling
  - Load-balancing
  - Memory management

GPU Computing Challenges

- Addressing other dwarfs
- Sparseness & adaptivity
- Scalability: Multi-GPU algorithms and data structures
- Irregularity
- Incremental data structures
- Abstract models of GPU computation

Thanks to …

- Thomas Fogel, Christoph Garth, Jian Huang, Kwan-Liu Ma, Rich Vuduc, Sam Williams
- NVIDIA for hardware donations; University of Illinois / NCSA and Argonne for cluster access
- Funding agencies: Department of Energy (SciDAC Institute for Ultrascale Visualization, Early Career Principal Investigator Award), NSF, LANL, BMW, NVIDIA, HP, Intel, UC MICRO, Microsoft, ChevronTexaco, Rambus