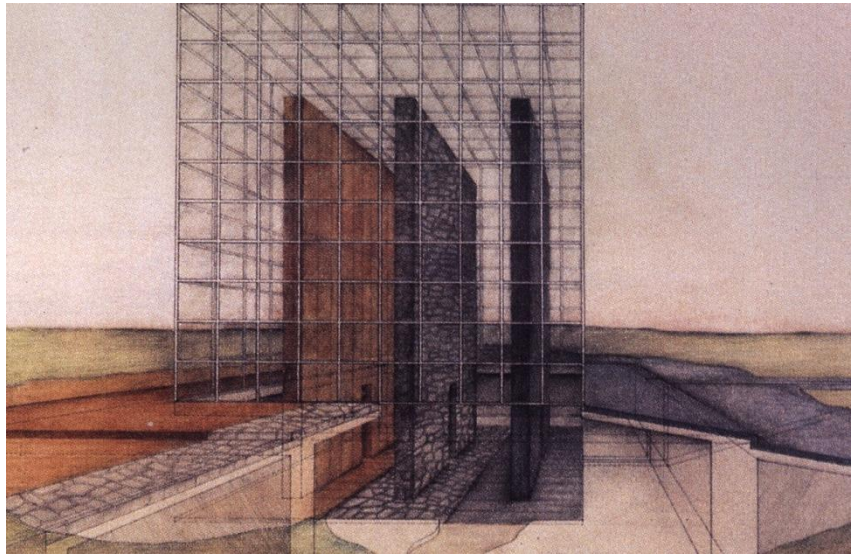
Argonne Training Program on Extreme-Scale Computing (ATPESC)

Presented to
ATPESC 2017 Participants



http://deathofdrawing.com/wp-content/gallery/raimund-abraham/RA-House-With-Three-Walls.jpg

Peter Kogge

Data Intensive Computing, the 3rd Wall, and the Need for Innovation in Architecture

Q Center, St. Charles, IL (USA)
Date 08/04/2017

EXASCALE COMPUTING PROJECT

When Do We Need New Architectures

- Long-lasting architectural advances occur when a “wall” must be overcome
- 1st Wall – Mid 90s: the **Memory Wall**
- 2nd Wall – 2004: the **Power Wall**
- 3rd Wall – Now: the **Locality Wall**

And this is largely due to emergence of apps with *Data Intensive* Characteristics

What Do I Mean by *Data Intensive*?

- Computation dominated by data access & movement – **not flops**
- Large sets of data often persistent
  - but little reuse during computation
- No predictable regularity
- Significantly different scaling
- Streaming becoming important

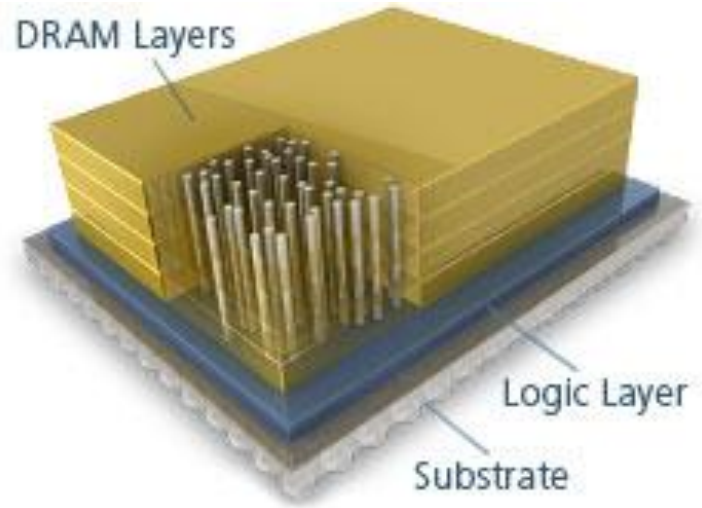
The "Locality" we have come to expect from our apps is disappearing

This Talk

- Moore's Law and the Prior Walls
- Today's Architectures
- Evidence of a New "Locality" Wall
  - Benchmarks
  - A Big Data Application
- Migrational Computing: a Possible Architectural Fix

Technology, Moore's Law, and Beyond

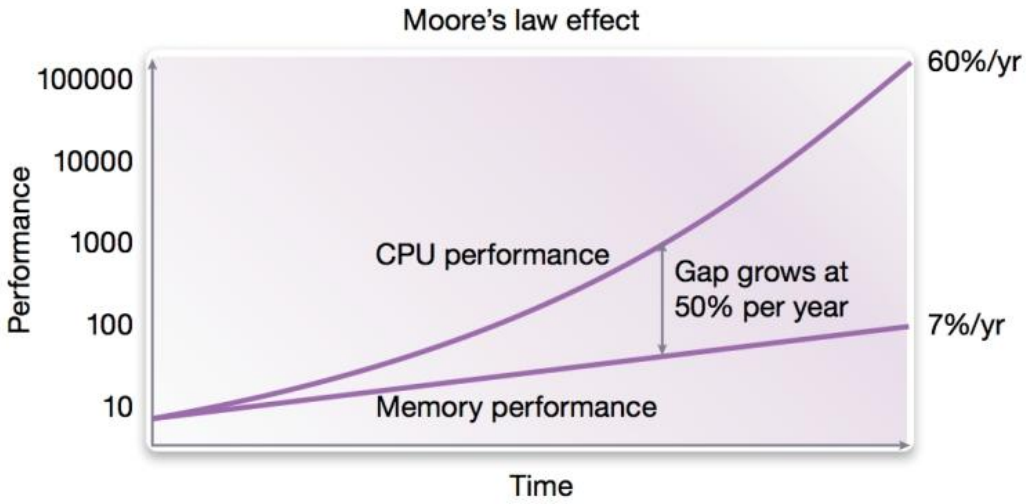
- Moore's Law: 2D transistors get smaller & faster
  - From 10um to 5nm feature size: **2,000X smaller & faster**
- Cores get smaller, faster, lower power
  - Power density approx. constant *as long as* V_{dd} declines
- Memory arrays get denser
  - To maximize density, access time *drops at best slowly*
  - Can increase bandwidth, but power skyrockets
- After Moore's Law: ***we're going 3D***!
  - With a mix of die types



http://www.micron.com/products/hybrid-memory-cube

The Memory Wall (mid 1990s)

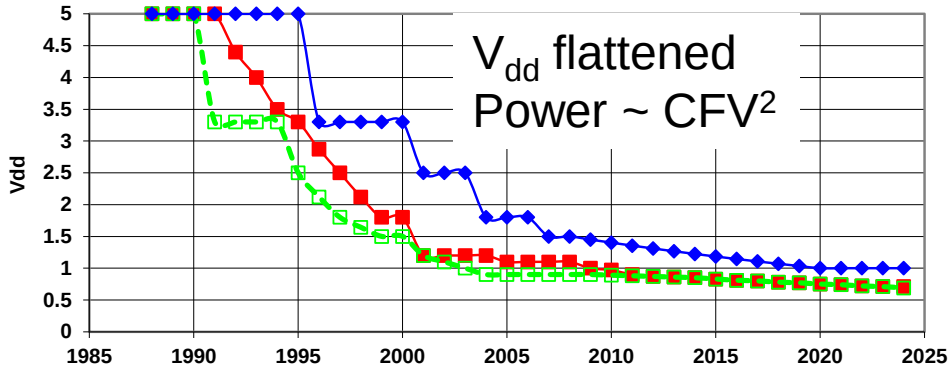
- Core clock speeds outran memory latency
- **Breaking the Wall**: Use extra transistors for
  - **Bigger on-chip SRAM caches**
  - More ILP to find more memory accesses
  - Add additional floating point capability
- Enablers: Applications had *plenty* of locality
- Example: Ax=b, A is large, dense, matrix
  - Tremendous temporal locality
  - Assume caches can save nxn patch of A
  - $O(n^2)$ to read nxn patch of A to cache
  - $O(n^3)$ operations on this patch
- With ***big enough cache***, don't care *how slow* memory is

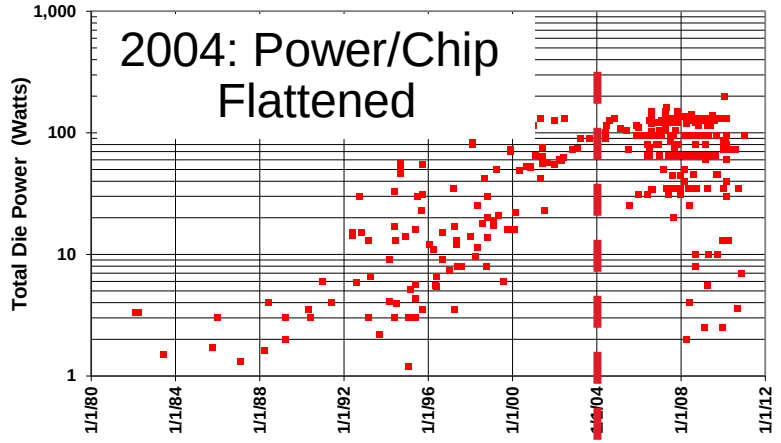


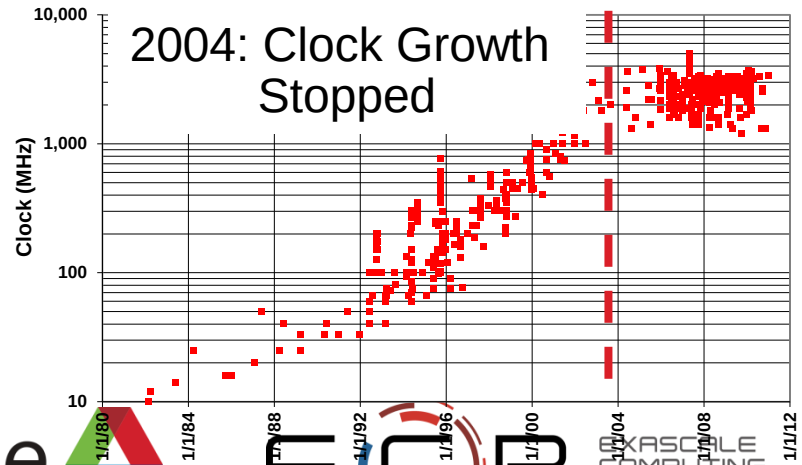
http://www.extremetech.com/wp-content/uploads/2014/07/140364245678419.jpg

The Power Wall (2004)

- Flattening V_{dd} increased power density
  - Bigger chips meant more logic to dissipate
- Result: at 120Watts, cooling uneconomical
- Breaking the wall:
  - ***Lower* the clock rate**
  - **Use *multiple* simpler cores**
  - **Increase *SIMD-style* parallelism**
- Side-effect: need more bandwidth
- Solution for dense apps: again **bigger caches**



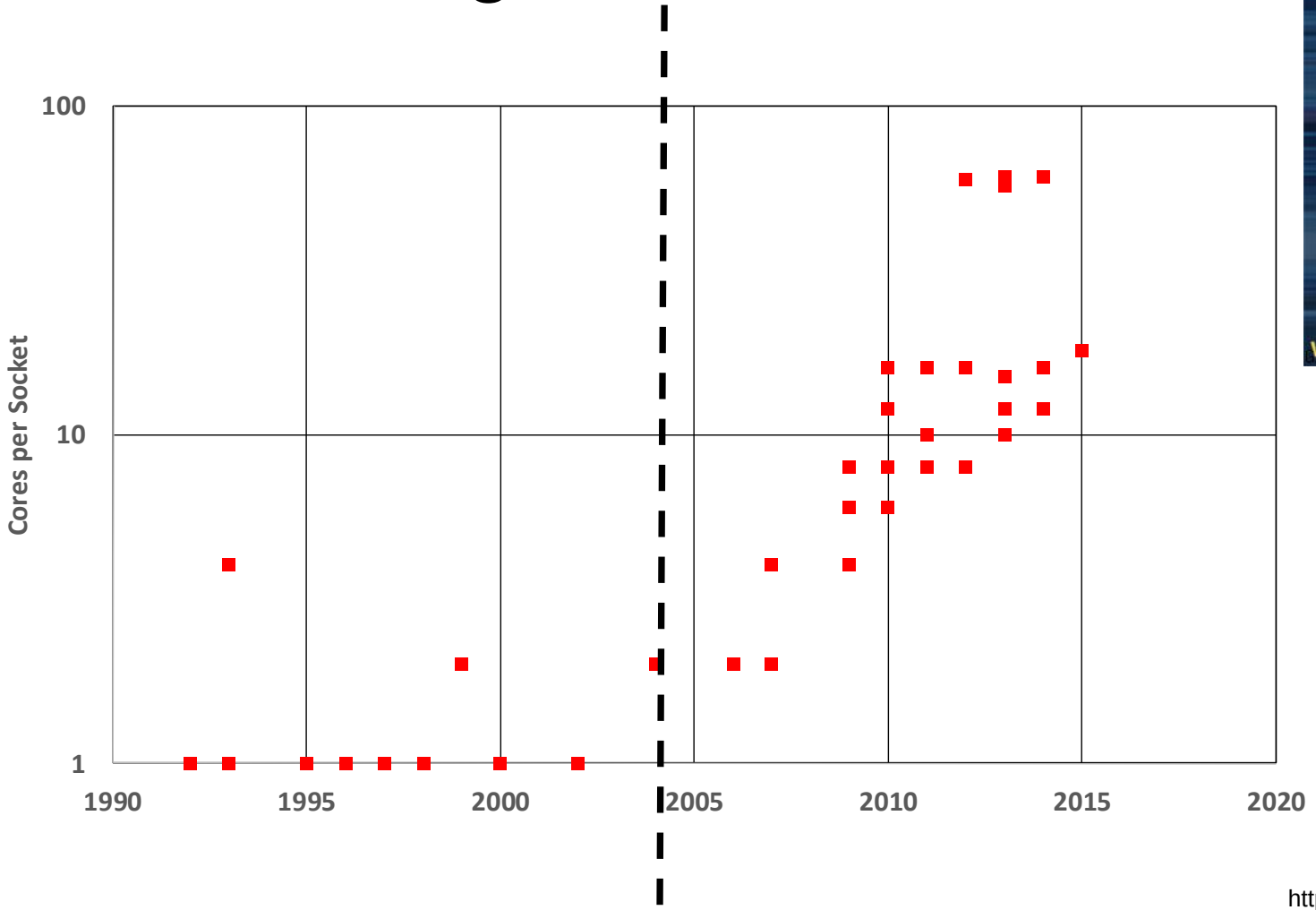


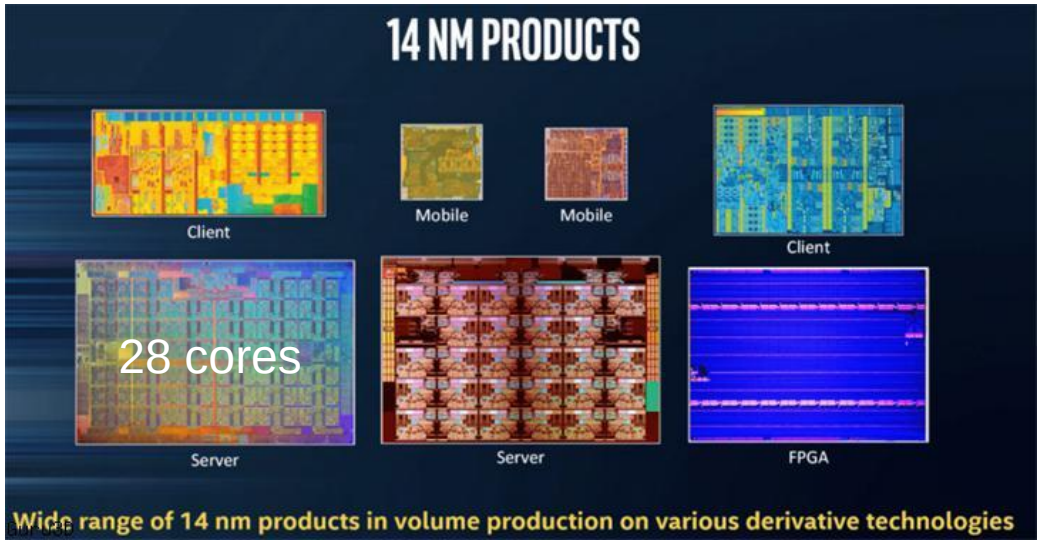


Argonne NATIONAL LABORATORY

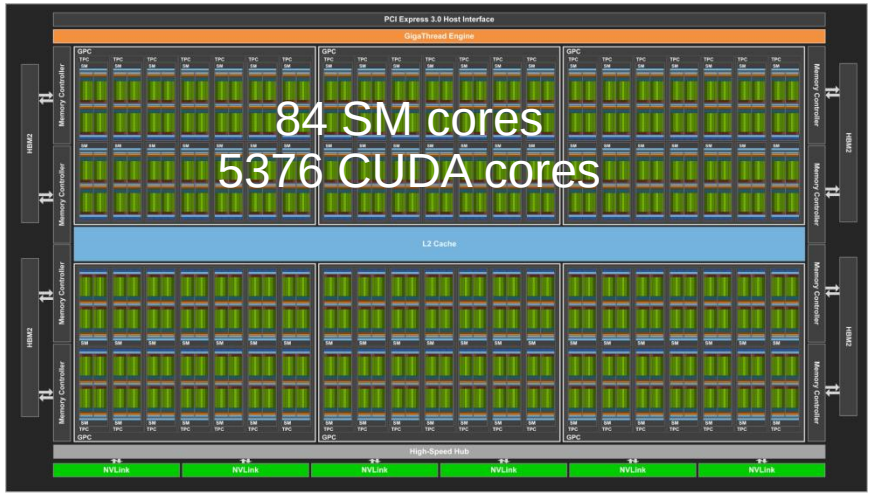
ECP EXASCALE COMPUTING PROJECT

2004: Emergence of Multi-core





http://www.guru3d.com/index.php?ct=news&action=file&id=19577



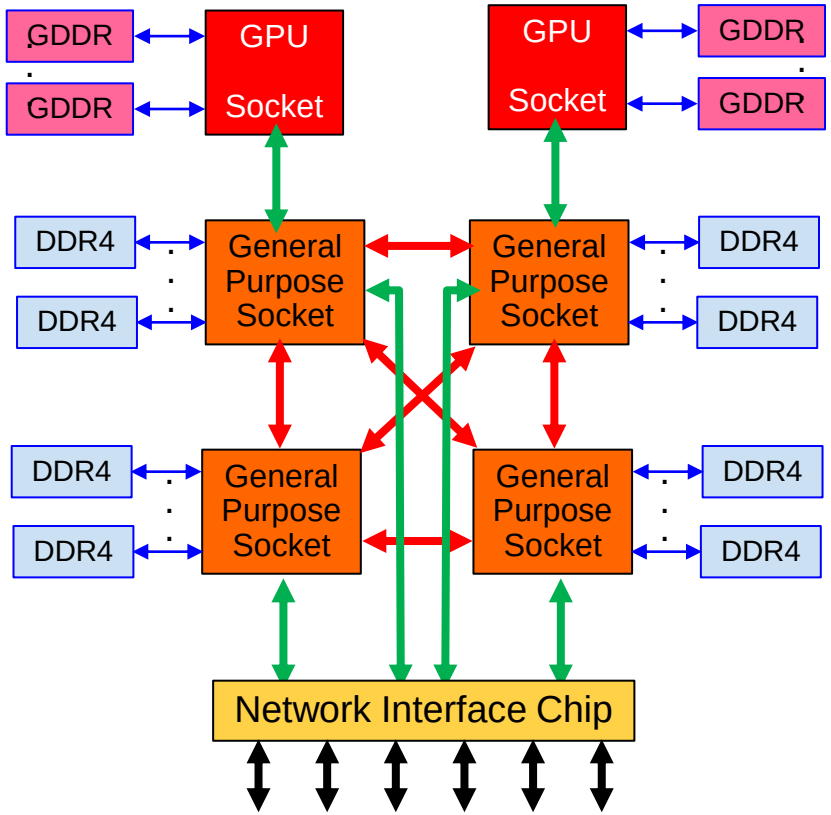
https://cdn.arstechnica.net/wp-content/uploads/sites/3/2017/05/voltablockdiagram.png

Argonne NATIONAL LABORATORY

ECP EXASCALE COMPUTING PROJECT

Today's Hybrid Multi/Many Core/Socket Architecture

- ***Nothing*** is uniform about memory references
- Multiple memory domains
- Multiple memory ports & types
- Multiple different link protocols
- Higher bandwidth parts needed (at energy costs)
- Growing "width" of data returned from an access (spatial locality)



Argonne NATIONAL LABORATORY

ECP EXASCALE COMPUTING PROJECT

Energy Tightly Tied to Locality

Operation	Energy (pJ)
64-bit integer operation	1
64-bit floating-point operation	20
256 bit on-die SRAM access	50
256 bit bus transfer (short)	26
256 bit bus transfer (1/2 die)	256
Off-die link (efficient)	500
256 bit bus transfer(across die)	1,000
DRAM read/write (512 bits)	16,000
HDD read/write	$O(10^6)$

28nm CMOS, DDR3

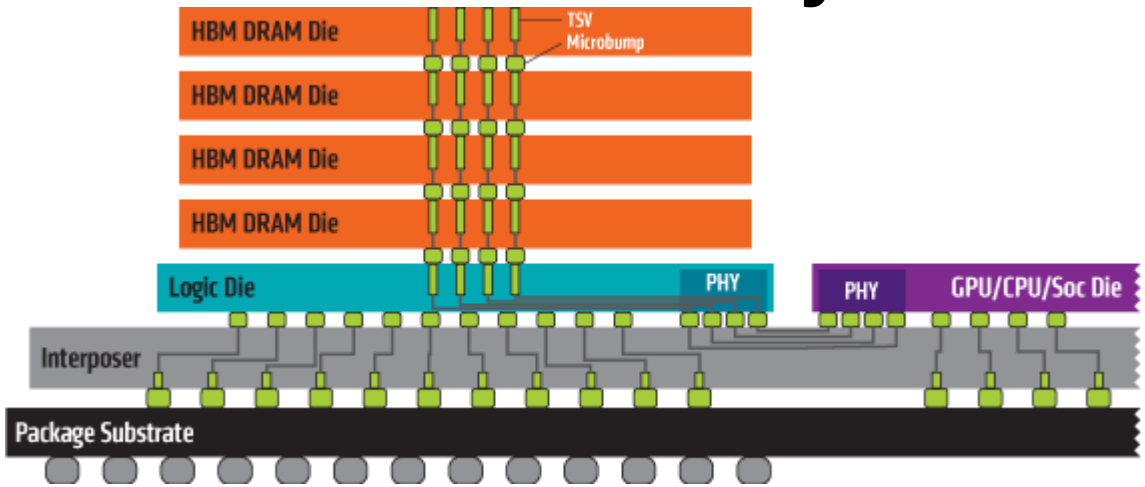
Greg Asfalk, HP

Perhaps 5 pJ in best of today

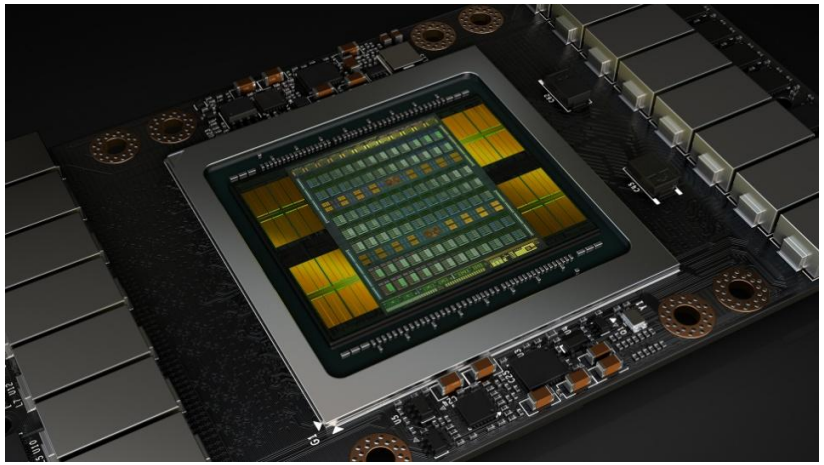
- Increasing with Non-Locality
- Largely unchanged by new technologies

Exascale goal of 20 pJ per flop unreachable if any memory references need to be made

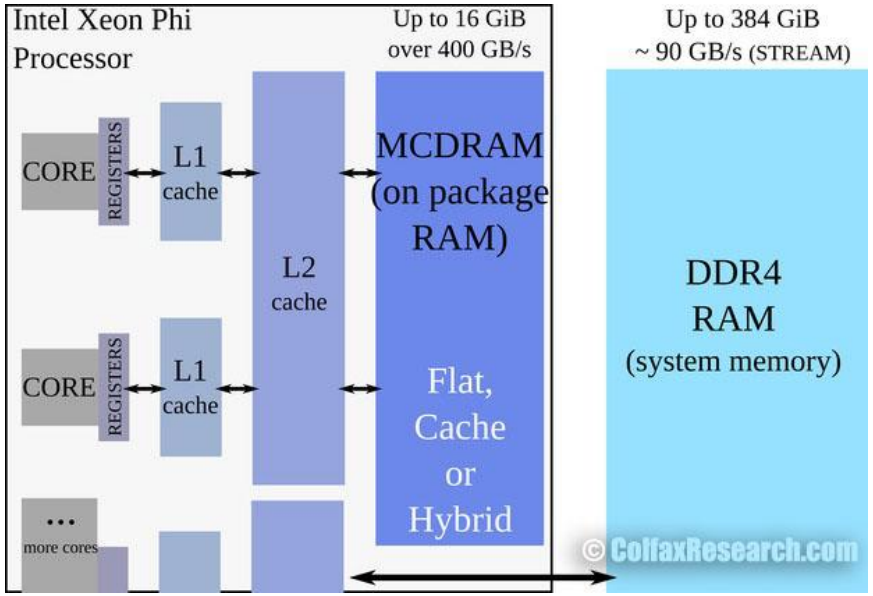
Need for More Memory Bandwidth – Multi-level Memories



http://www.amd.com/PublishingImages/graphics/illustrations/570px/6315-hbm-stacks-diagram.png



https://cdn.arstechnica.net/wp-content/uploads/sites/3/2017/05/NVIDIA-Telsa-V100.jpg



HBMs: 4-5X bandwidth, but wider transfer/access

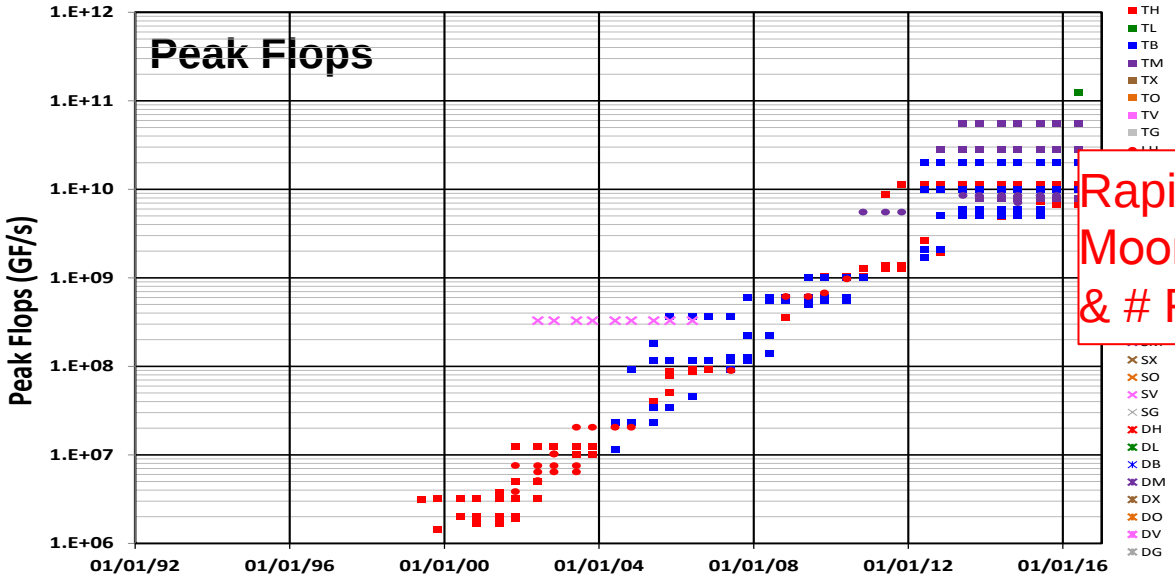
Argonne NATIONAL LABORATORY

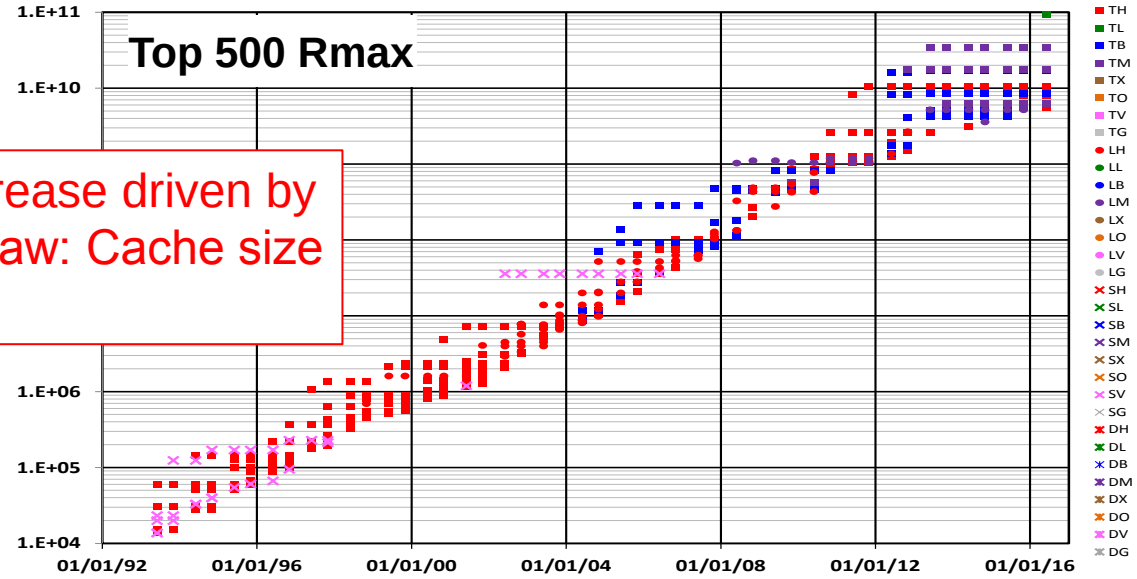
ECP EXASCALE COMPUTING PROJECT

And Apps Are Changing – Lets look at some Benchmarks

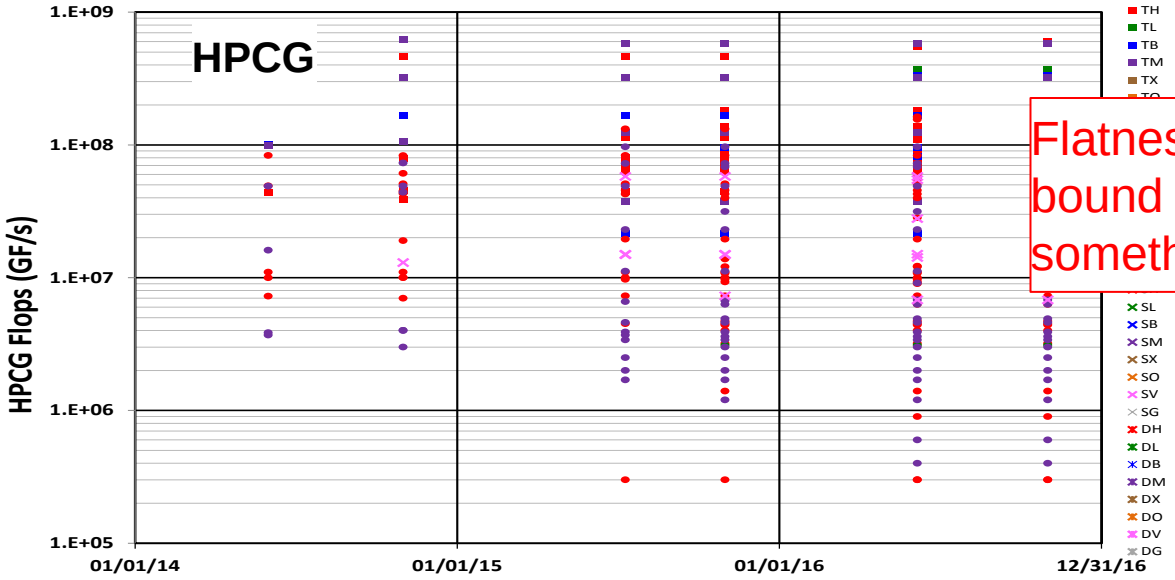
Benchmark Name	Function Performed
LINPACK	Solve Ax=b; A is dense
HPCG: Hi Perf Cong. Grad.	Ax=b; A sparse but regular
SpMV: Sparse Mat. Vec.	Ab; A sparse & irregular
BFS: Breadth First Search	Find all reachable vertices from root
FireHose	Find “events” in streams of data

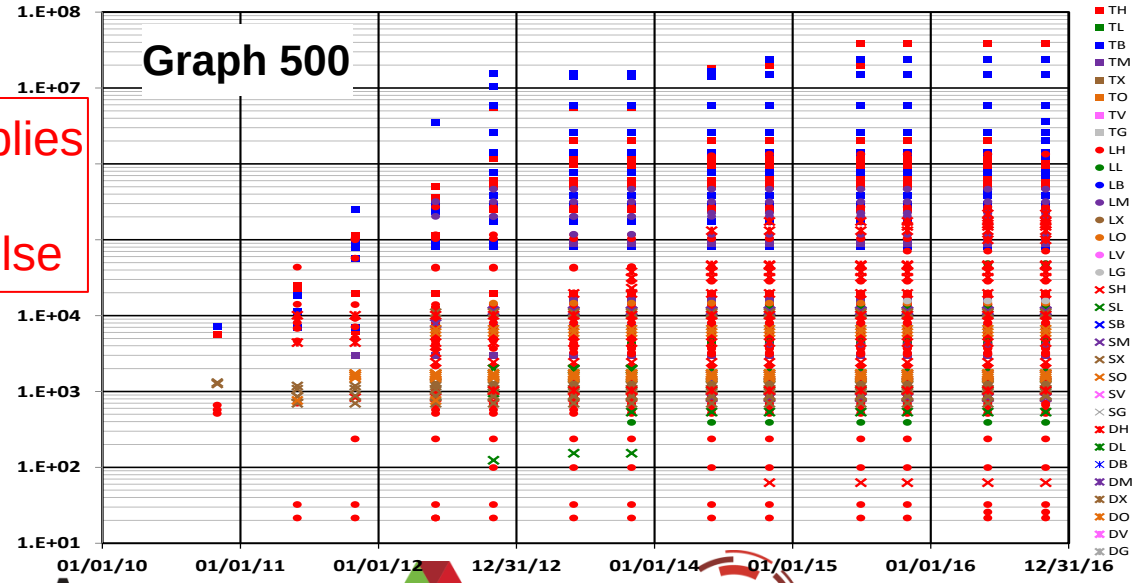
Performance vs Time





Rapid increase driven by Moore's Law: Cache size & # FPUs



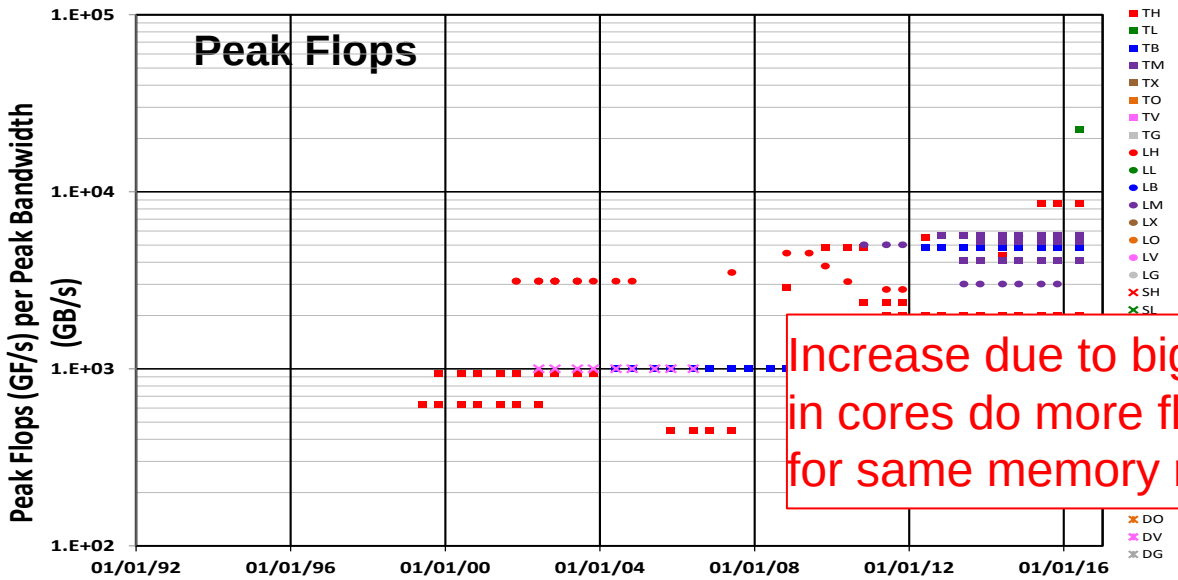


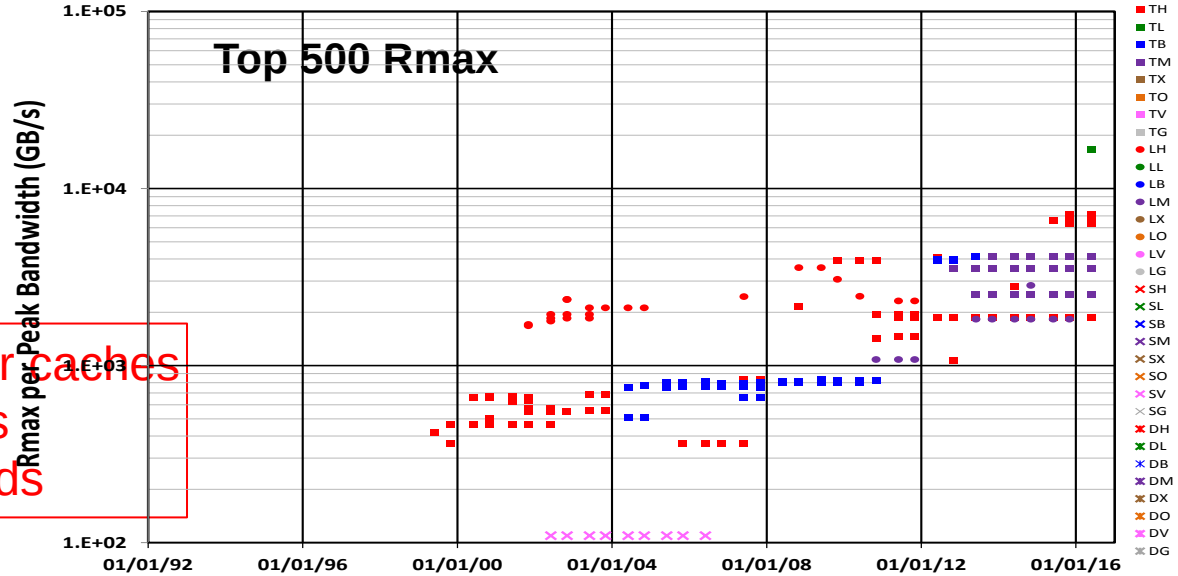
Flatness implies bound by something else

Argonne NATIONAL LABORATORY

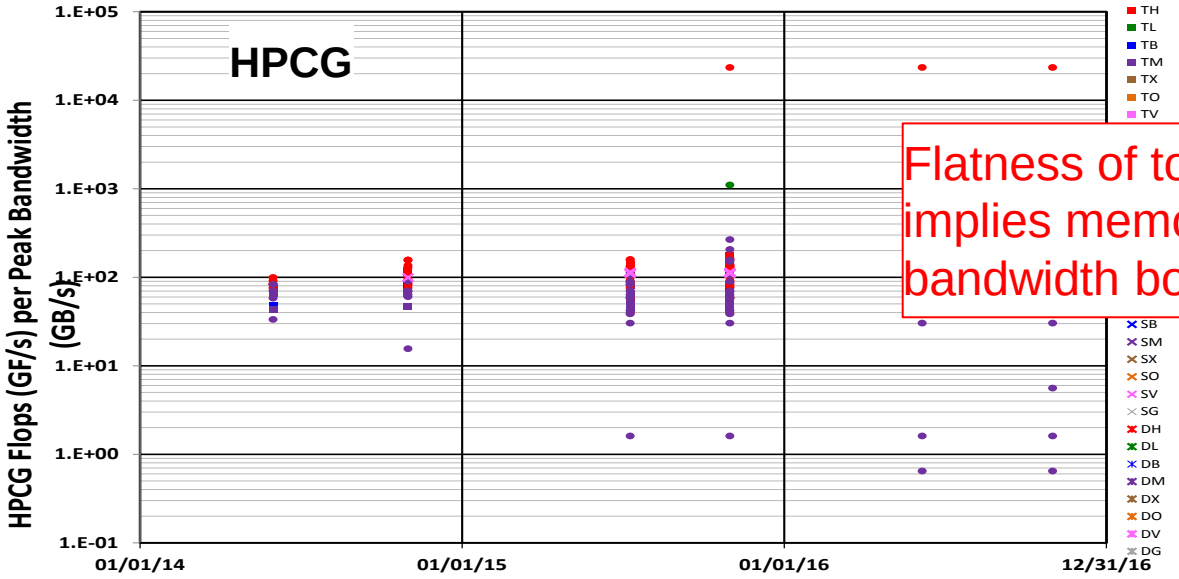
ECP EXASCALE COMPUTING PROJECT

Performance/Byte of B/W vs Time

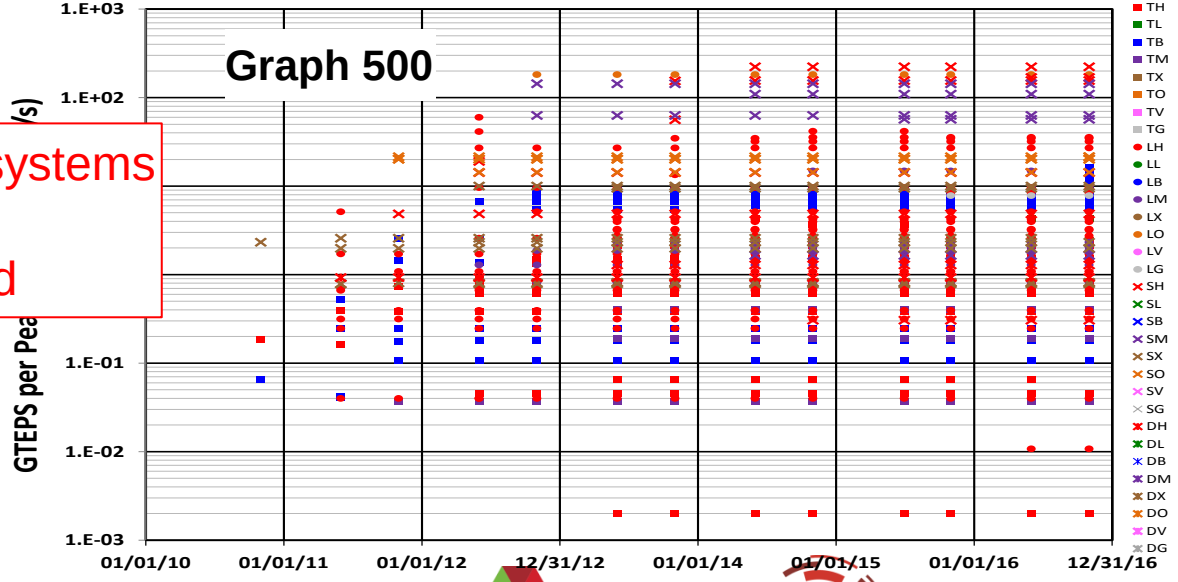




Increase due to bigger caches in cores do more flops for same memory reads



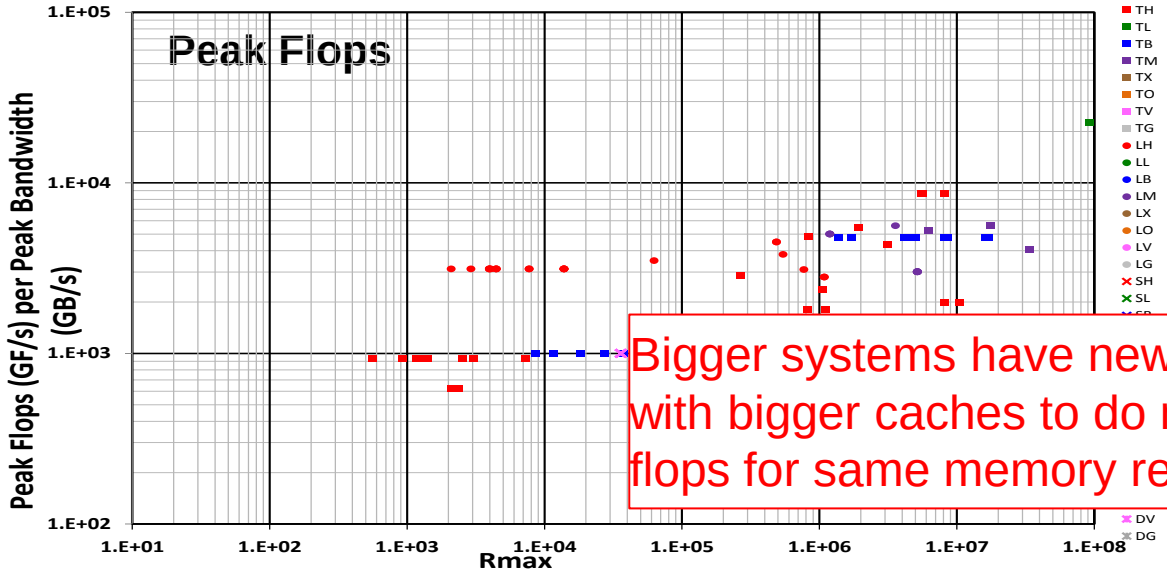
Flatness of top systems implies memory bandwidth bound

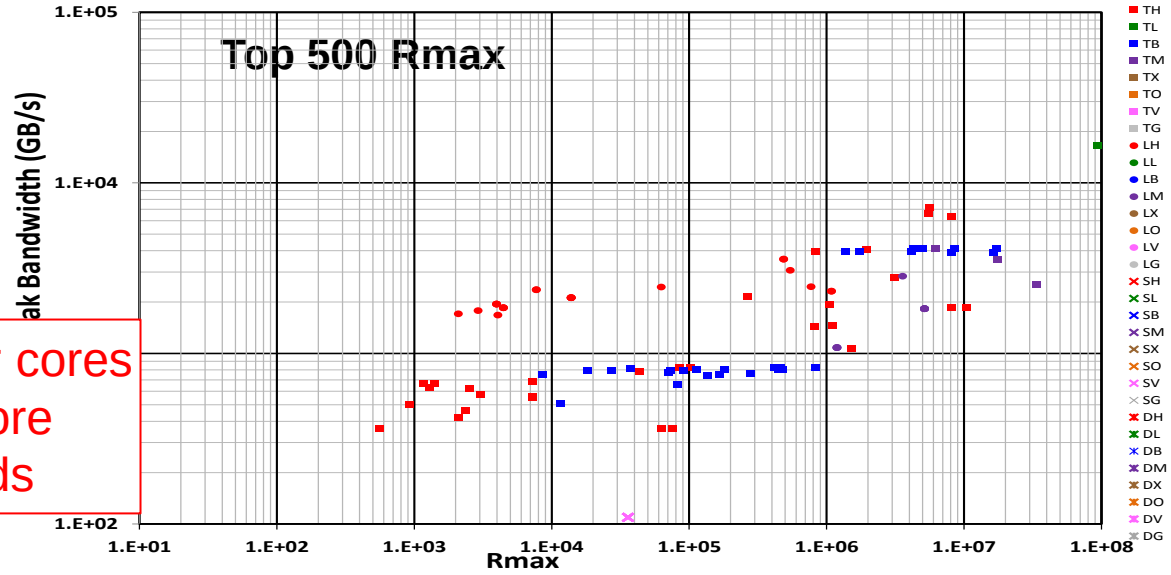


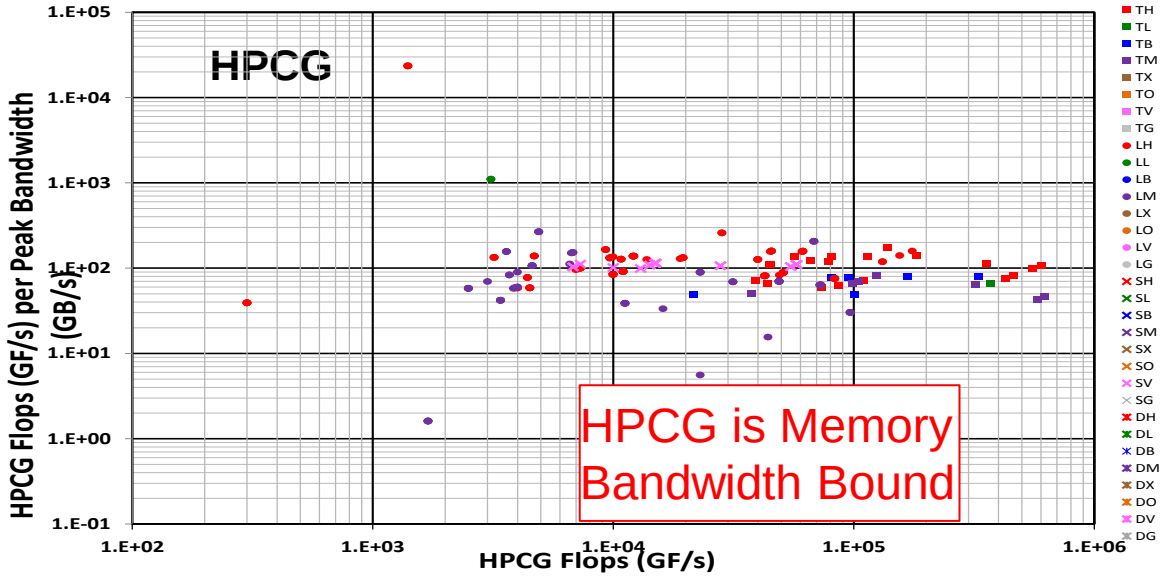
Argonne NATIONAL LABORATORY

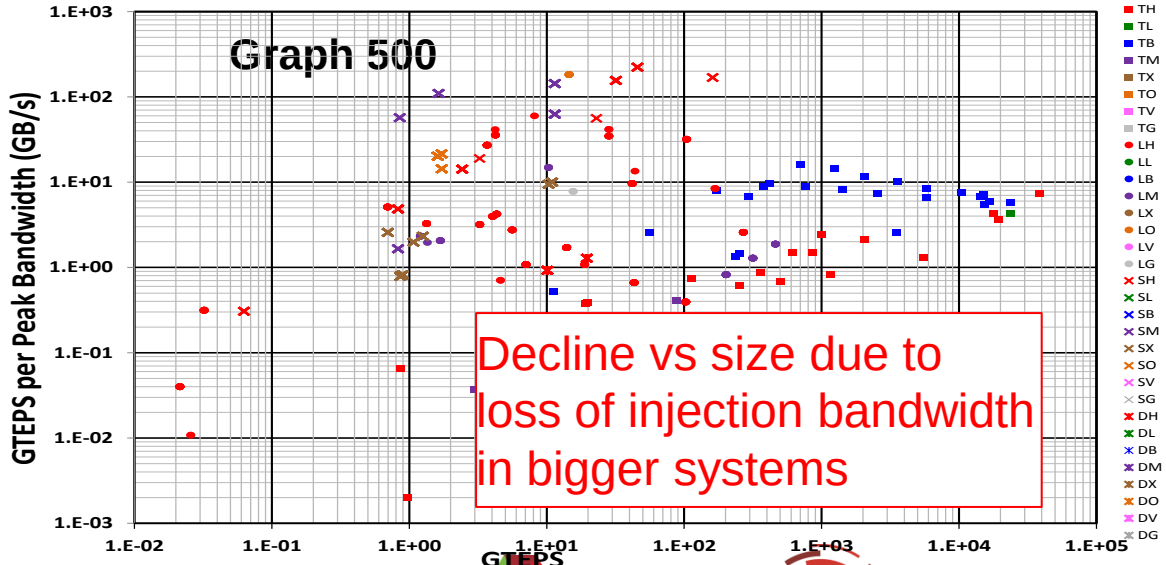
ECP EXASCALE COMPUTING PROJECT

Perf./Byte of B/W vs Perf.









Bigger systems have newer cores with bigger caches to do more flops for same memory reads

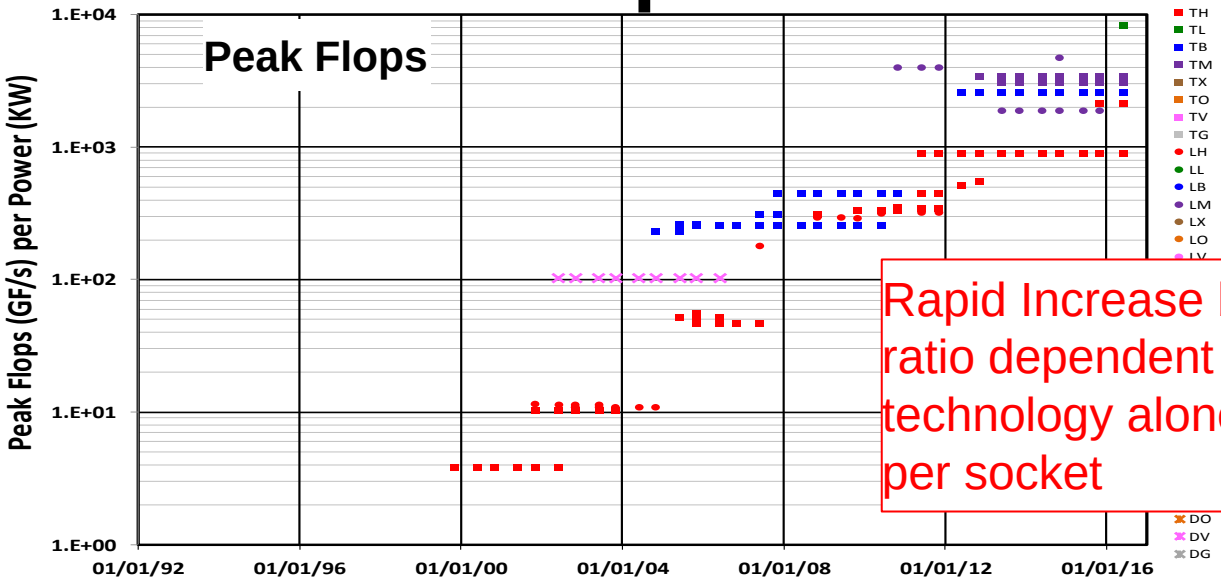
HPCG is Memory Bandwidth Bound

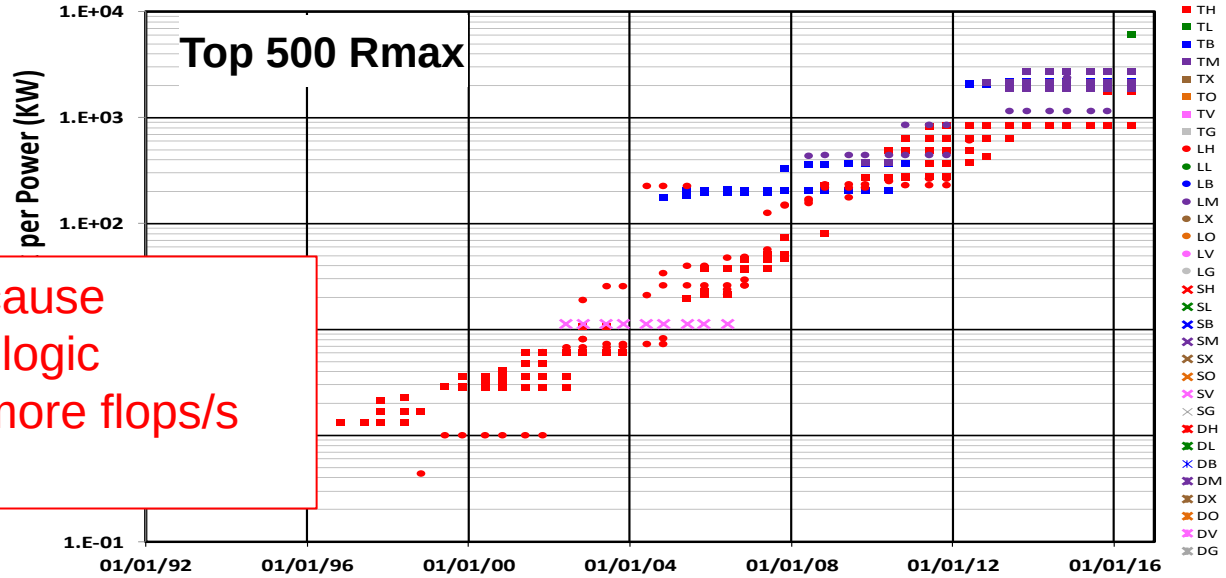
Decline vs size due to loss of injection bandwidth in bigger systems

Argonne NATIONAL LABORATORY

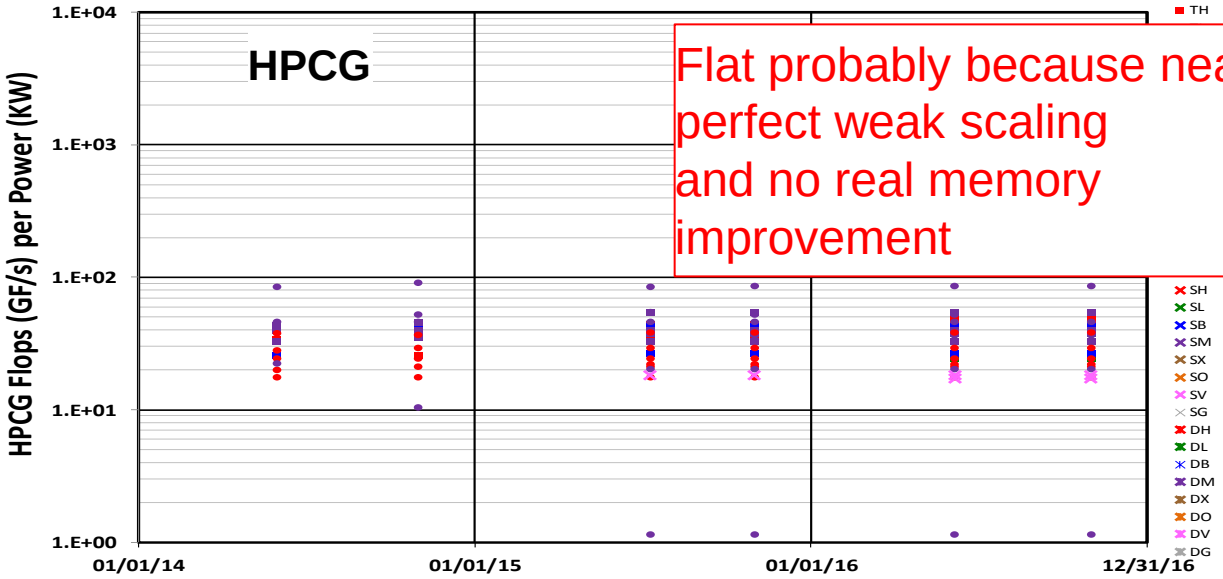
ECP EXASCALE COMPUTING PROJECT

Performance per Watt vs Time

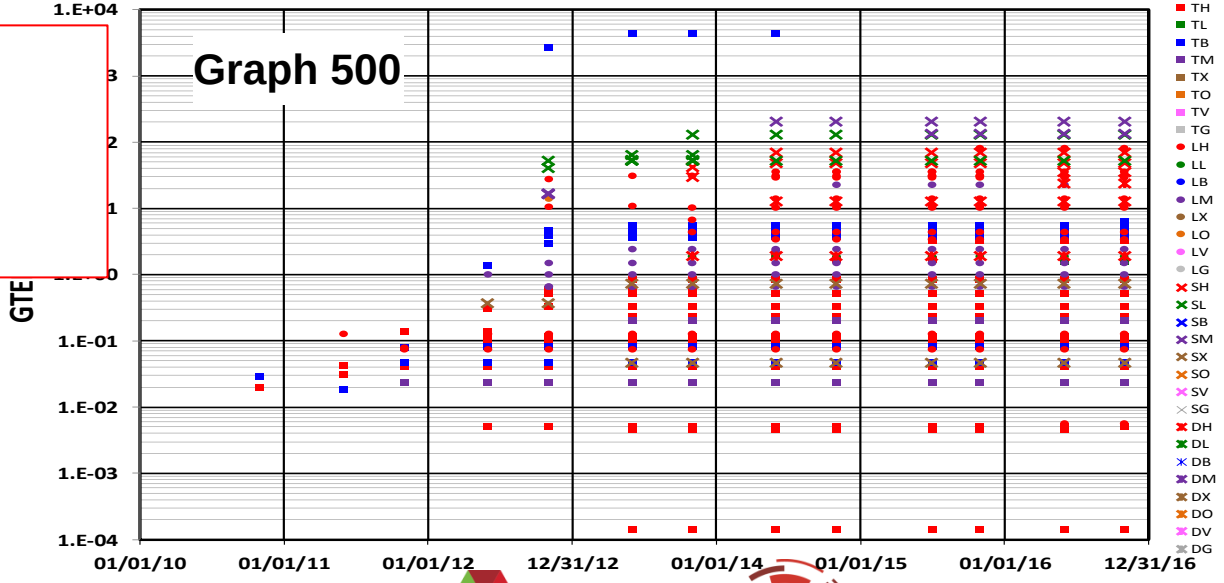




Rapid Increase because ratio dependent on logic technology alone; more flops/s per socket



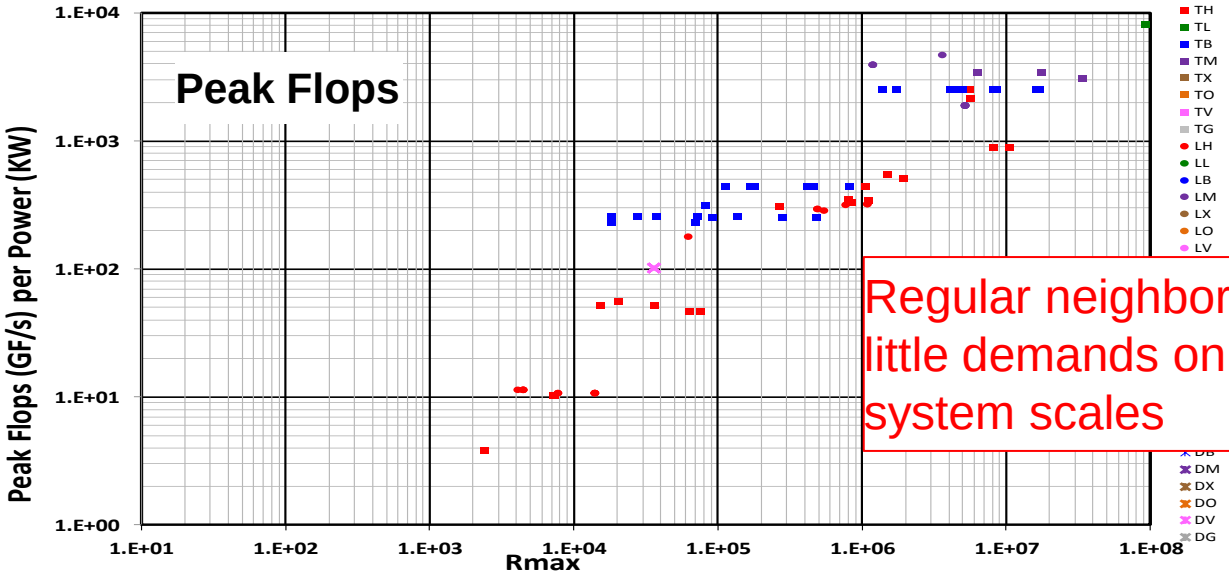
Flat probably because near perfect weak scaling and no real memory improvement



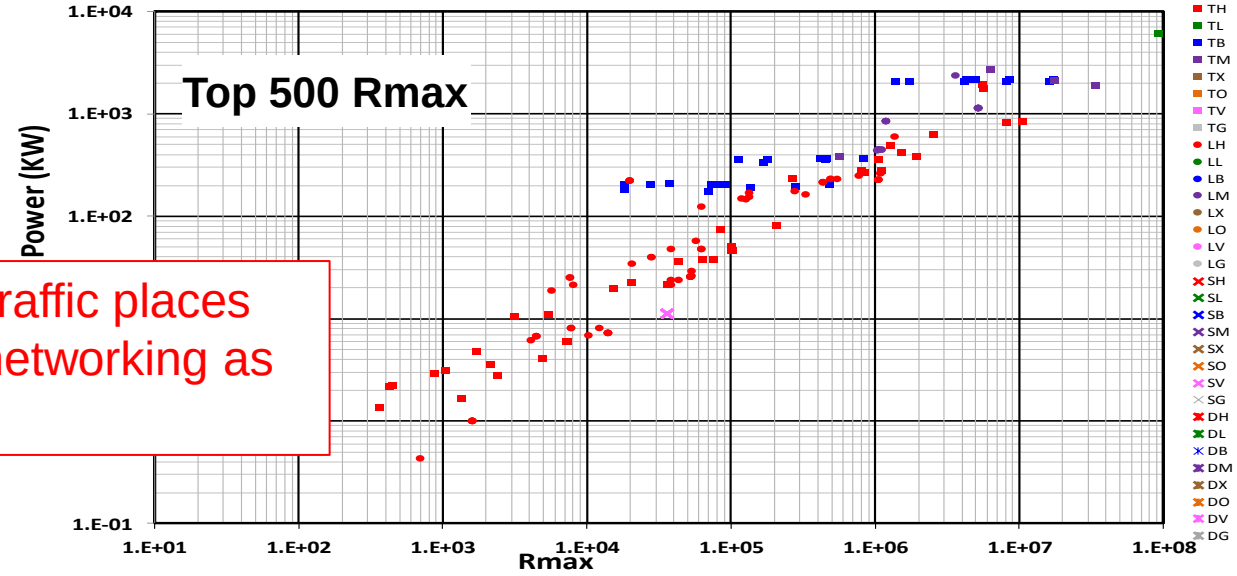
Argonne NATIONAL LABORATORY

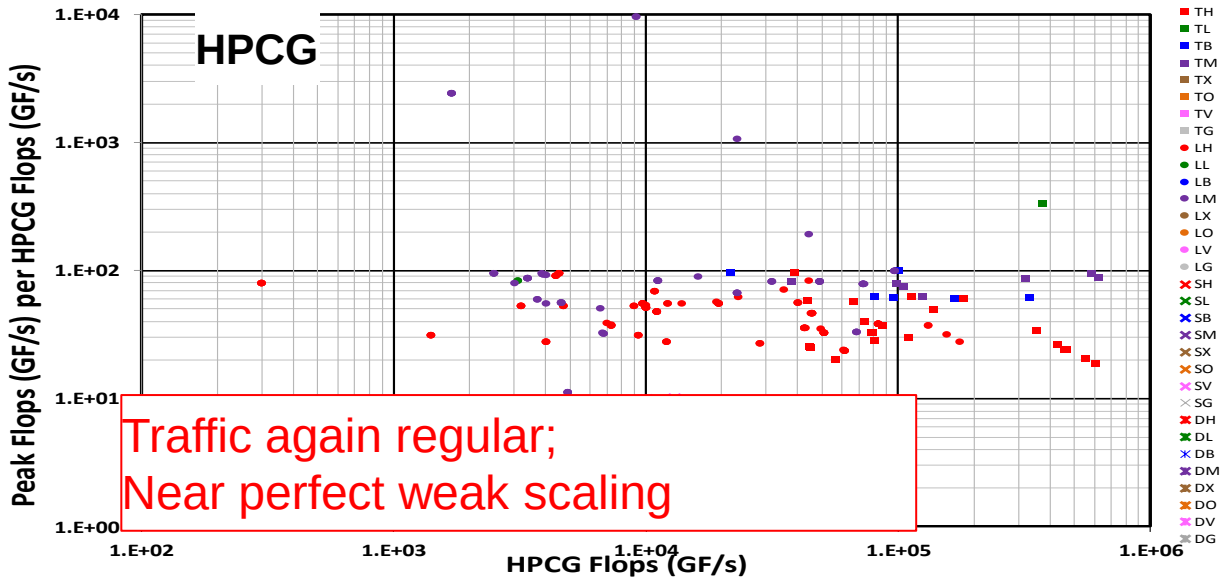
ECP EXASCALE COMPUTING PROJECT

Perf. per Watt vs Perf.

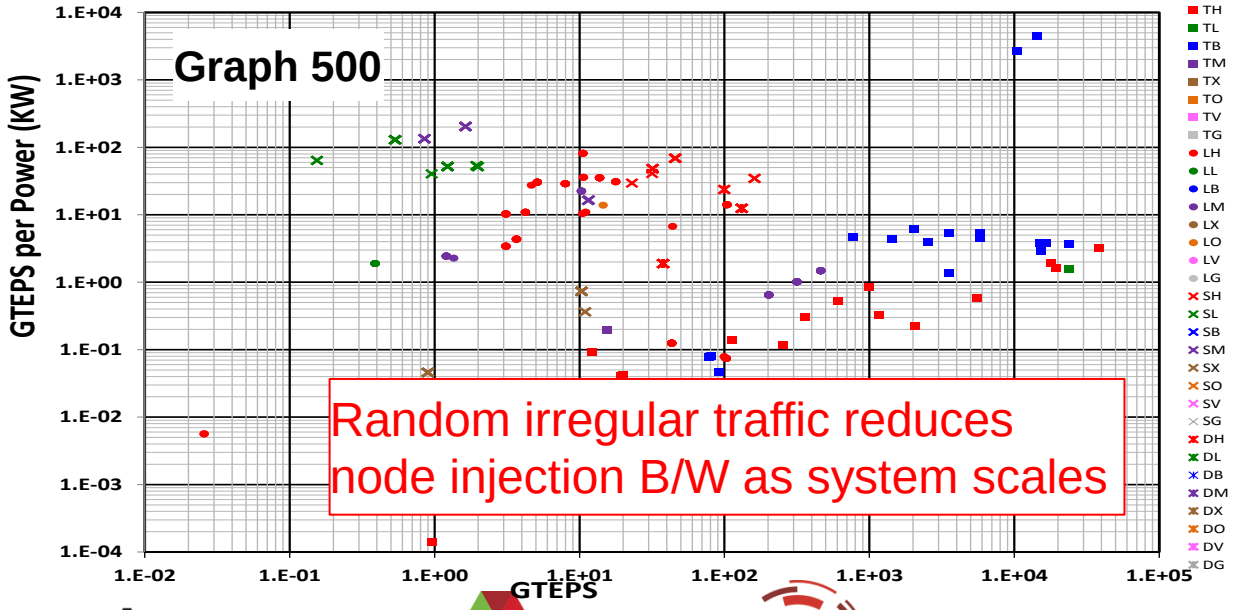


Regular neighbor traffic places little demands on networking as system scales





Traffic again regular;
Near perfect weak scaling

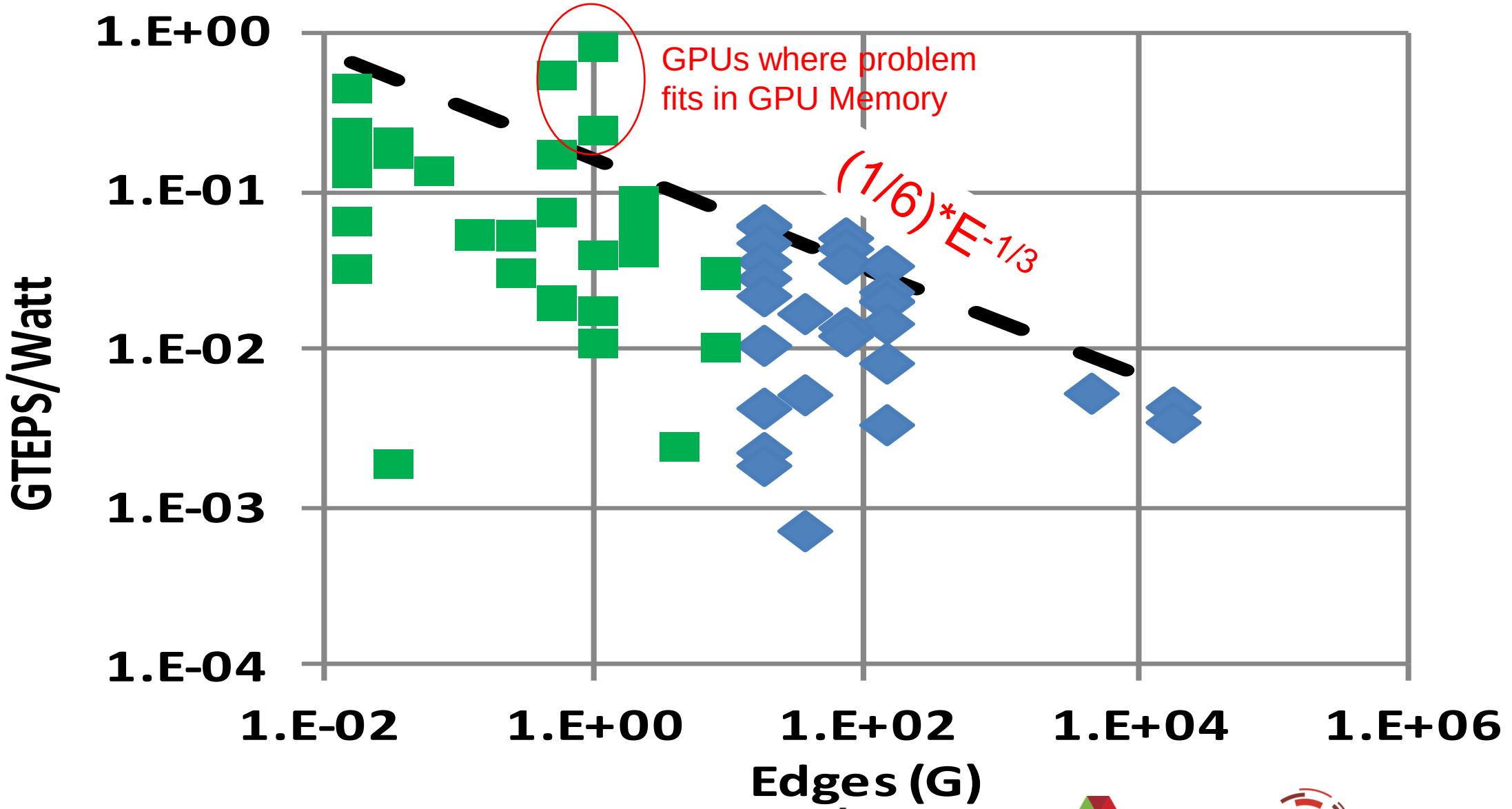


Random irregular traffic reduces node injection B/W as system scales

Argonne NATIONAL LABORATORY

ECP EXASCALE COMPUTING PROJECT

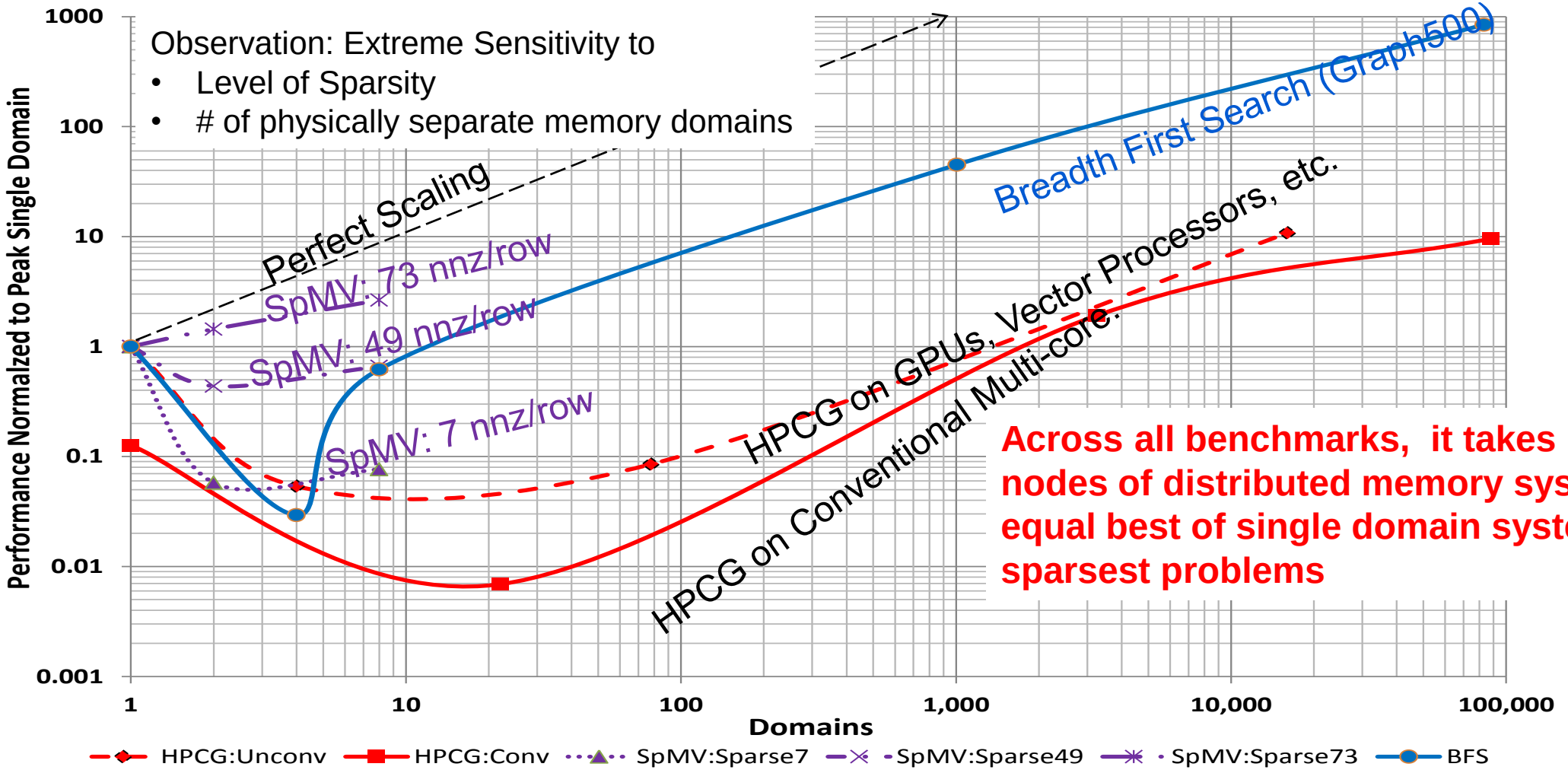
Green-GRAPH500



Argonne NATIONAL LABORATORY

ECP EXASCALE COMPUTING PROJECT

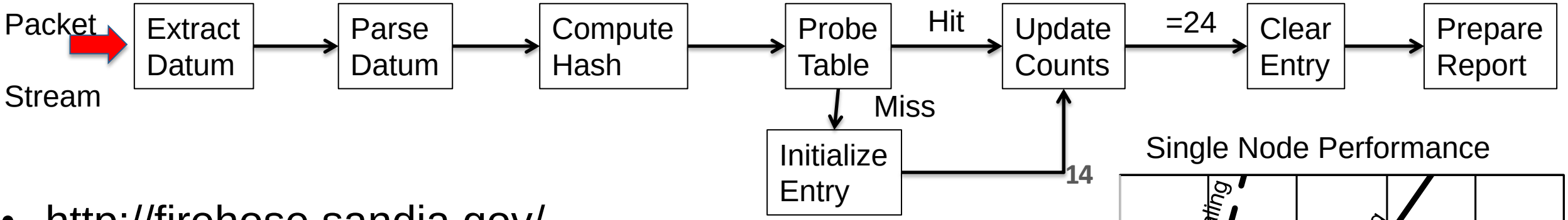
Sparsity & Conventional Scalability



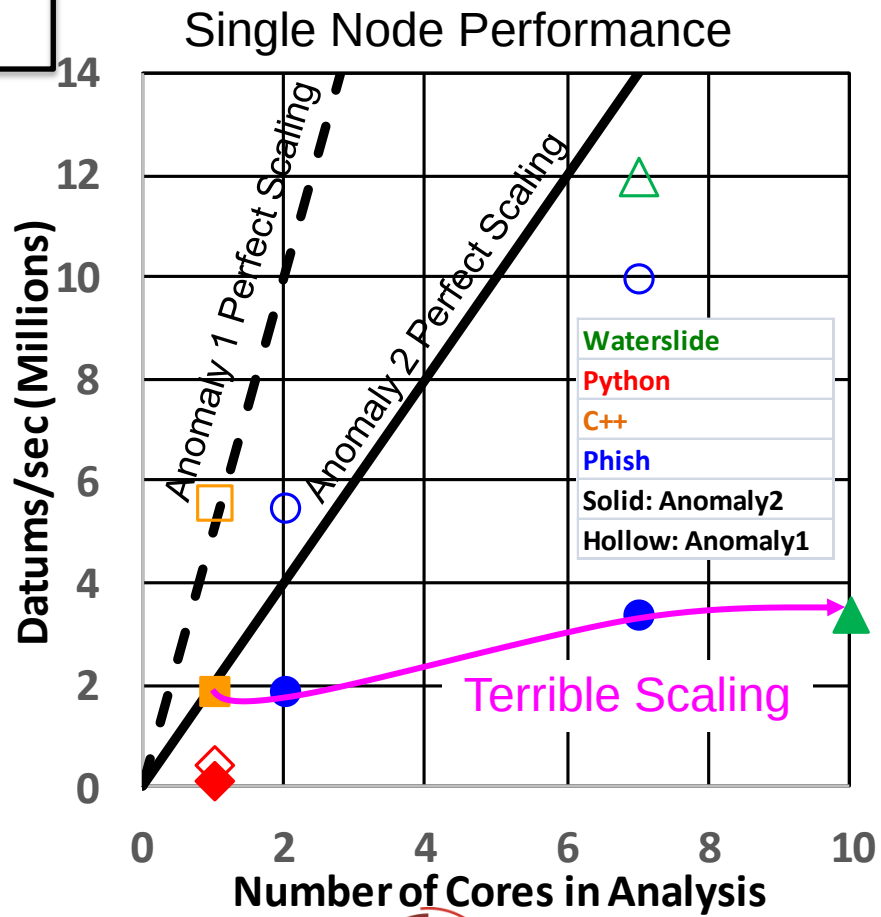
Across all benchmarks, it takes 10-1000 nodes of distributed memory systems to equal best of single domain systems for the sparsest problems

Bylina et al., "Performance Analysis of Multicore and Multinodal Implementation of SpMV Operation", 2014. www.graph500.org. http://www.hpcg-benchmark.org/

Firehose Streaming Benchmark



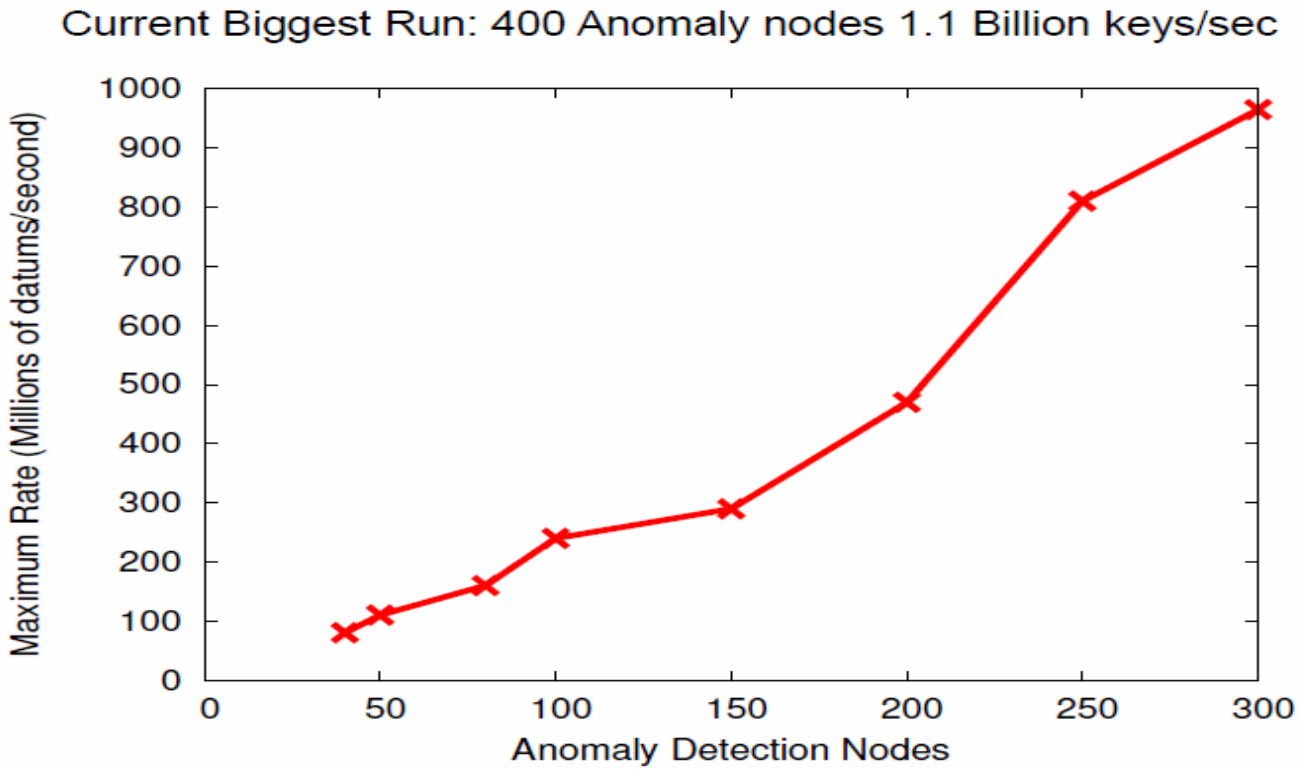
- http://firehose.sandia.gov/
- **Datum**: Comma separated ASCII string
  - **Key**: ASCII string representing 64b uint (IP adr)
  - **Value**: depends on benchmark variant
  - **Truth flag**: was the stream from this key biased
- **Event**: detection of 24 datums with same "key"
- **Anomaly**: value distribution biased towards 0s
  - 3 variants defined
- **Performance metric**: **Datums/sec**



Argonne NATIONAL LABORATORY

ECP EXASCALE COMPUTING PROJECT

Large Scale Anomaly 1 Processing



- MPI with PHISH runtime library
- Approx 2.75 M datums/s per node
- Or about 220 M/s per rack

- SNL SkyBridge, Cray-CS300 1848 2-socket nodes at 16 cores/node
- From “Stateful Streaming in Dist. Memory Supercomputers,” Berry & Porter, CLSAC 2016

Scaling line is fairly linear
BUT at 2.75M datums/s per *32 core node*,
0.09M datums/s per core is ***1/60*** that of a single core

Summary: Basic Benchmarks – Non-traditional Have Locality Issues

Benchmark Name	Function Performed	Performance Limiters
LINPACK	Solve Ax=b; A is dense	Cache size & # FPUs
HPCG: Hi Perf Cong. Grad.	Ax=b; A sparse but regular	Memory B/W
SpMV: Sparse Mat. Vec.	Ab; A sparse & irregular	Memory B/W; some Network
BFS: Breadth First Search	Find all reachable vertices from root	Network B/W; Remote atomics
FireHose	Find “events” in streams of data	Managing the streaming

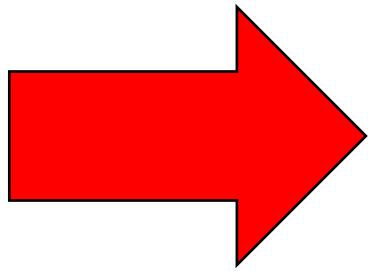
Real World Challenge Data Intensive Problem (From Lexis Nexis)

Auto Insurance Co: “Tell me about giving auto policy to Jane Doe” in < 0.1sec

- **2012: 40+ TB of Raw Data**
- **Periodically clean up & combine to 4-7 TB**
- **Weekly “Boil the Ocean” to precompute answers to all standard queries**
  - **Does X have financial difficulties?**
  - **Does X have legal problems?**
  - **Has X had significant driving problems?**

Relationships

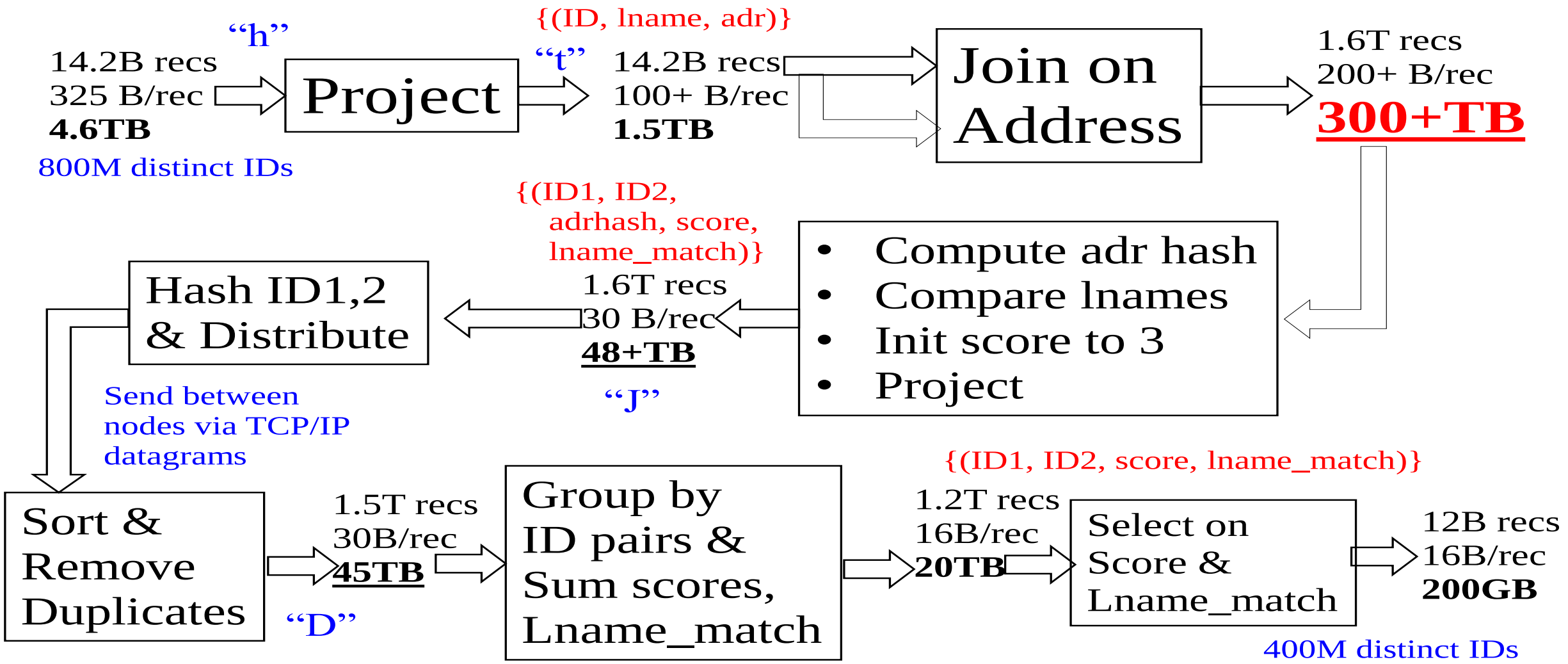
  - **Who has shared addresses with X?**
  - **Who has shared property ownership with X?**



Look up answers to precomputed queries for “Jane Doe”, and combine

“Jane Doe has no indicators
But
she has shared multiple addresses with Joe Scofflaw
Who has the following negative indicators ….”

Traditional Approach: Runaway Intermediate Data



Argonne NATIONAL LABORATORY

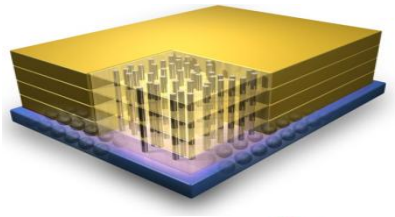
ECP EXASCALE COMPUTING PROJECT

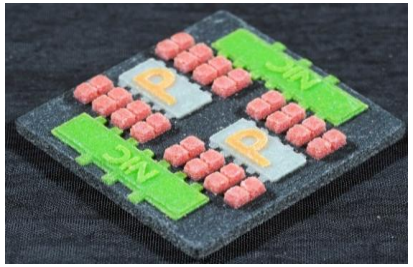
Projecting Performance for LexisNexis' Implementation

2012: 400 2-socket nodes (10 racks)

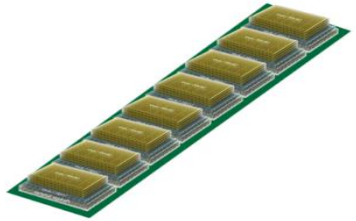
2013 study looked at "future" alternatives:

- Upgrades to conventional
- "Lightweight" systems
  - Lower power, lower performance cores
  - Study assumed Calxeda 4-core ARMs
  - but systems like HP Moonshot similar
- Sandia's X-Caliber project
  - Heavyweight with HMC-like memories
  - Resembles Intel's Knights Landing
- All processing on bottom of 3D stack
  - System = "sea" of stacks

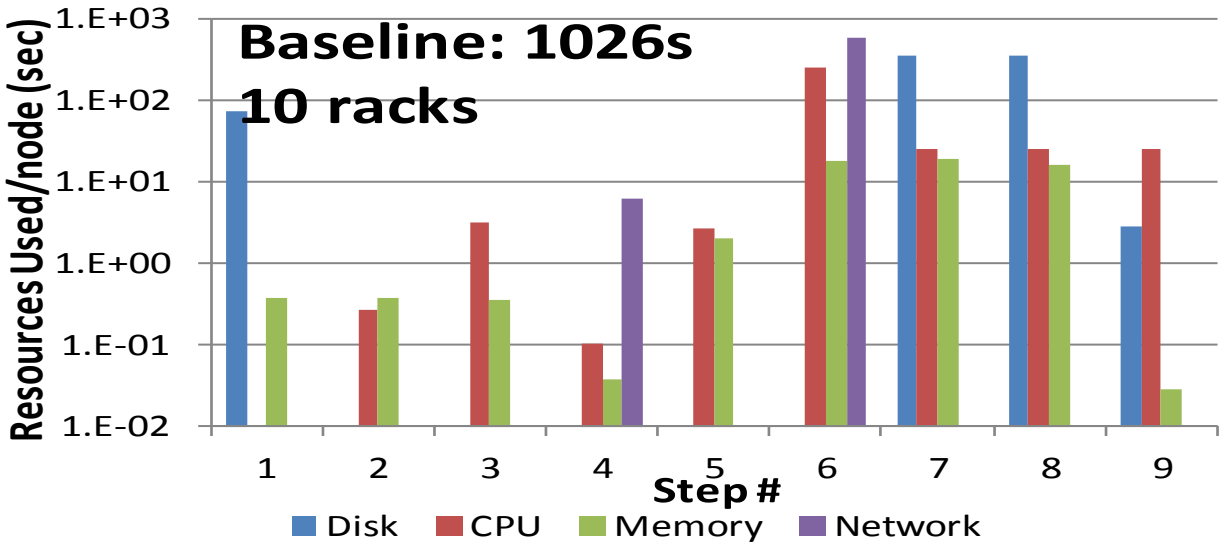




(b) X-caliber Node Mockup



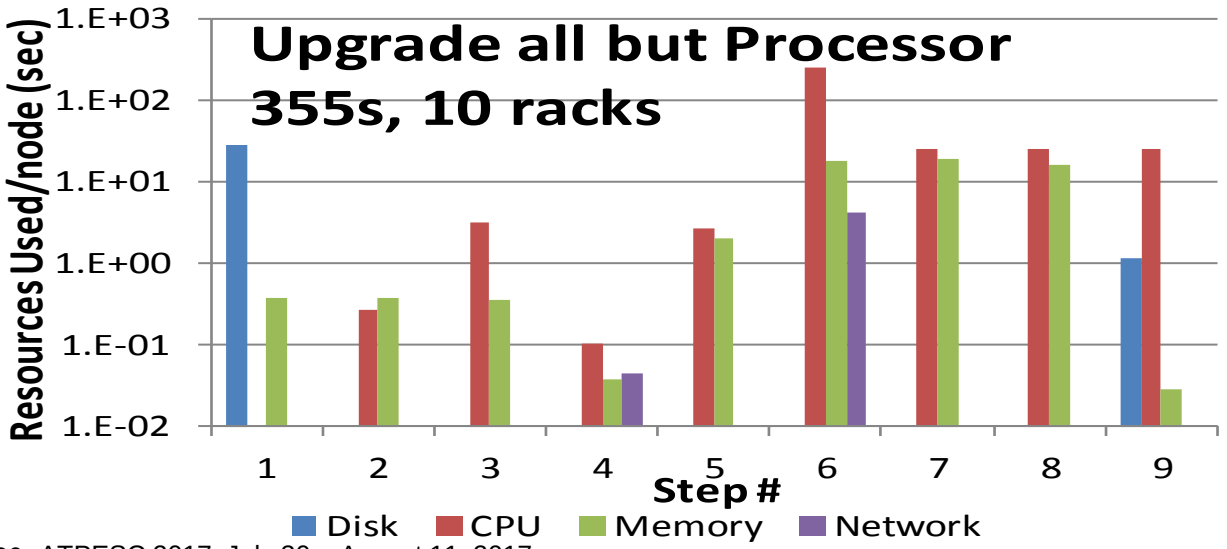
Heavyweight Alternatives Using LN's App Flow

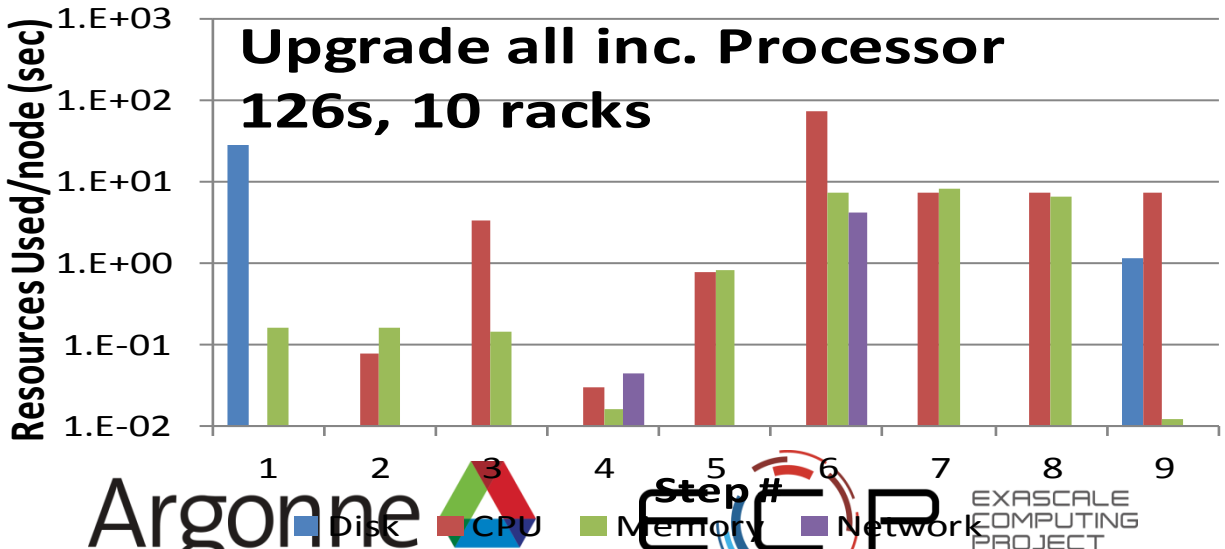


Performance Options:

- Socket: 6C to 24C
- Memory B/W by 3X
- Disk to SSD or RAMDisk
- Network to Infiniband

No one option grows performance more than 45%

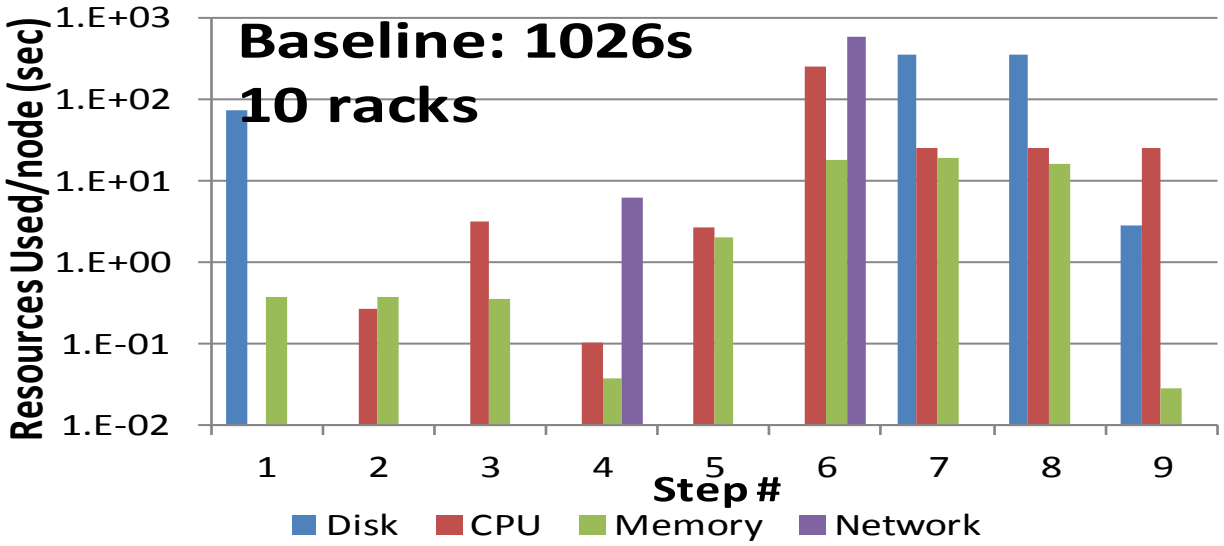


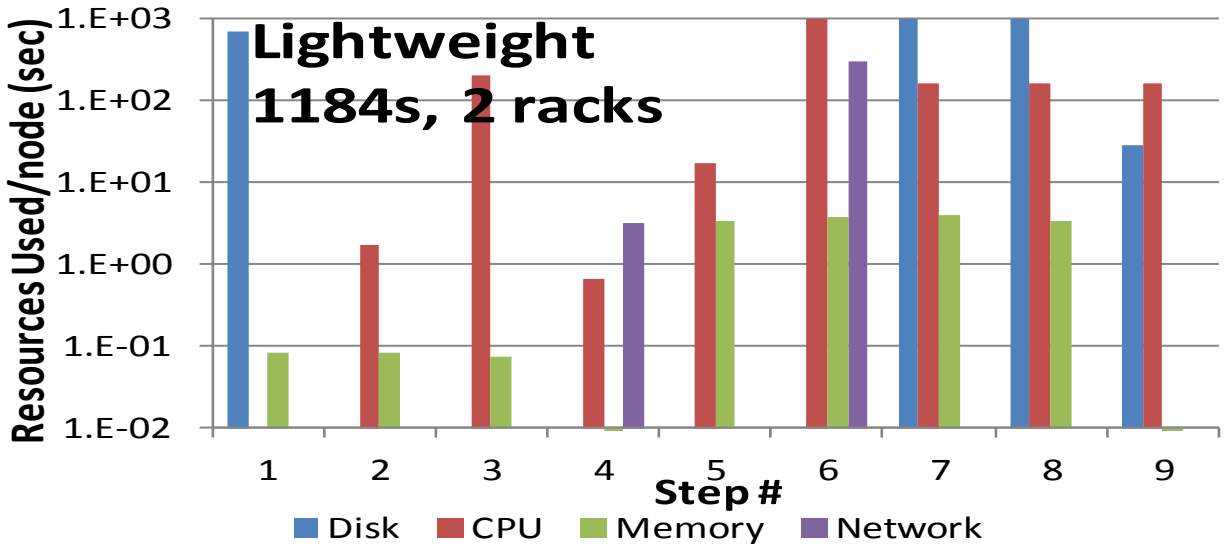


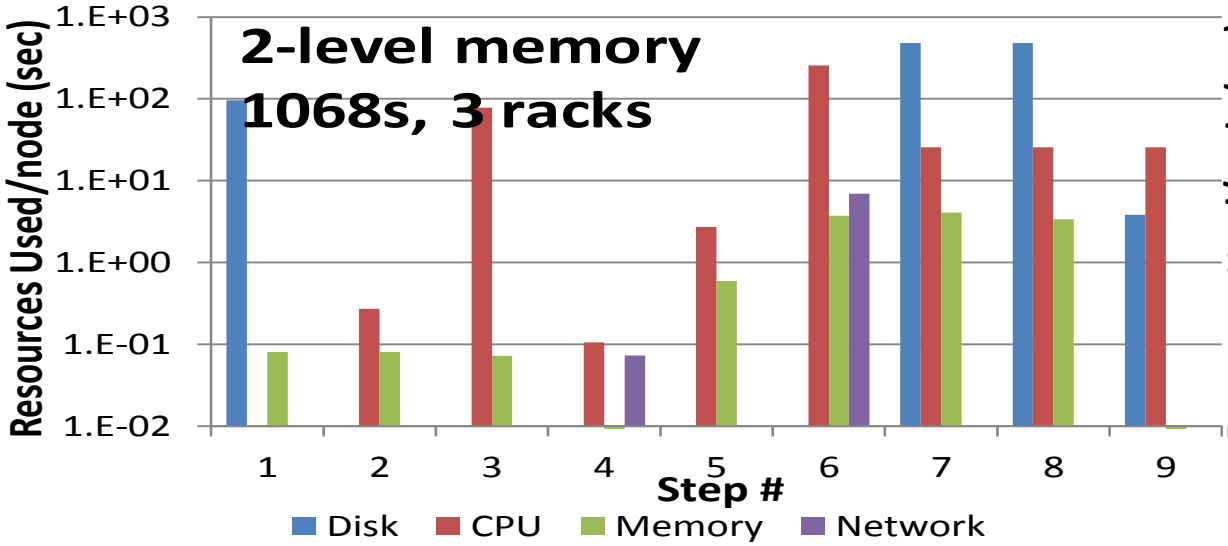
Argonne NATIONAL LABORATORY

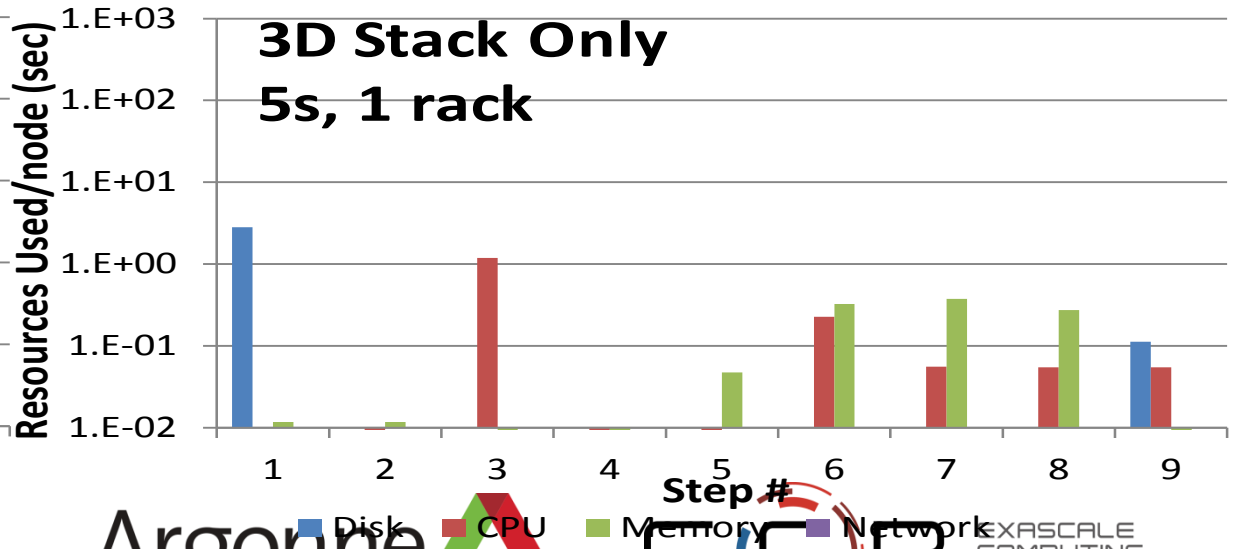
ECP EXASCALE COMPUTING PROJECT

Unconventional Alternatives









Argonne NATIONAL LABORATORY

ECP EXASCALE COMPUTING PROJECT

Migrational Computing: An Alternative Architecture

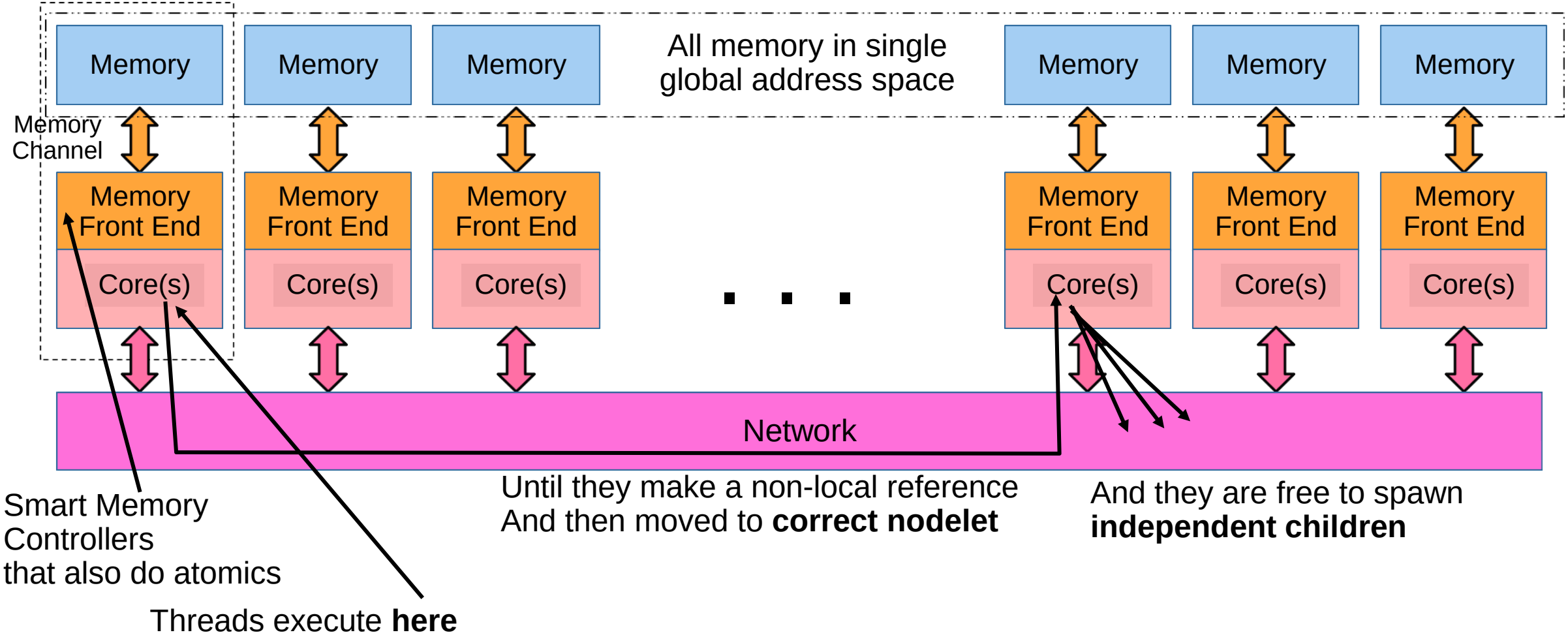
- **Thread Migration**: *move* site of a thread's execution
- Rationale: make memory reference ***LOCAL***!
- Today: either invisible (e.g. during I/O call) or explicit (as in Chapel)
- New idea: make migration automatic on remote memory access

Argonne NATIONAL LABORATORY

ECP EXASCALE COMPUTING PROJECT

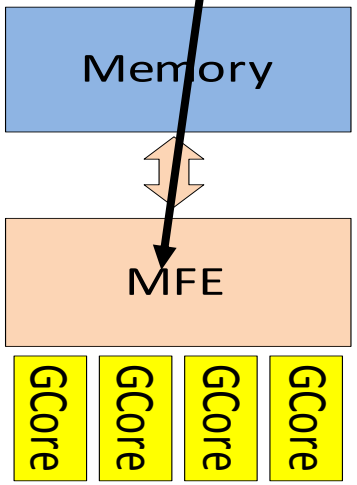
A Migrational Architecture

Nodelet: New unit of parallelism



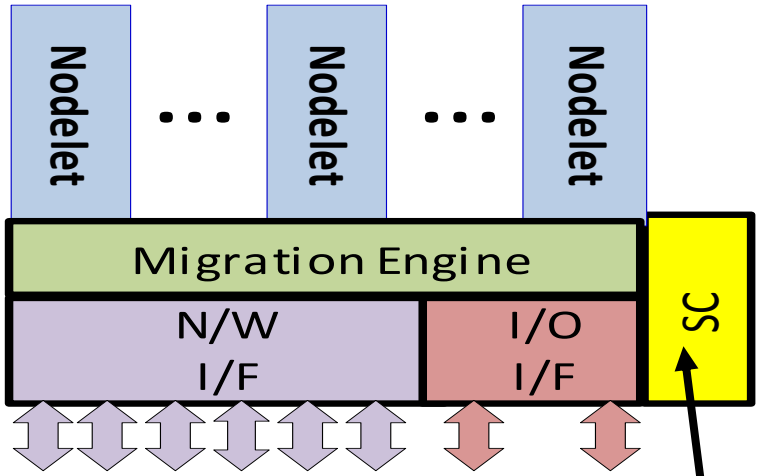
A Real Migrational System

Atomics run in Memory Front End (MFE)



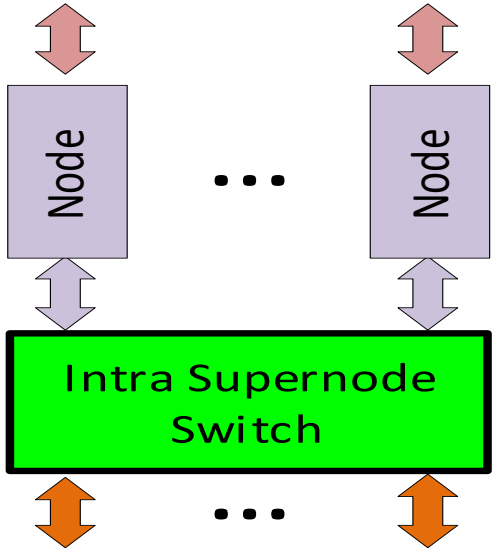
(a) Nodelet

Multi-Threaded Cores



(b) Node

Migrating Threads are major traffic on Network

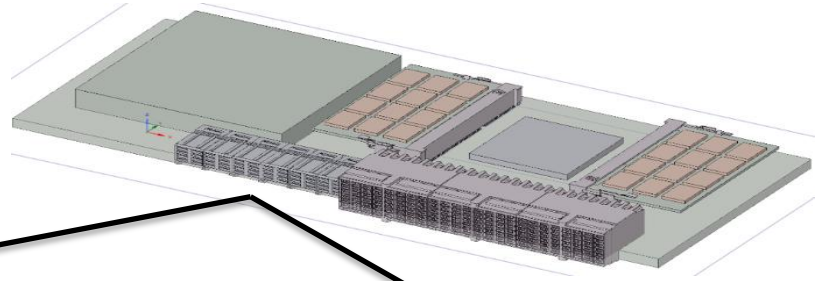


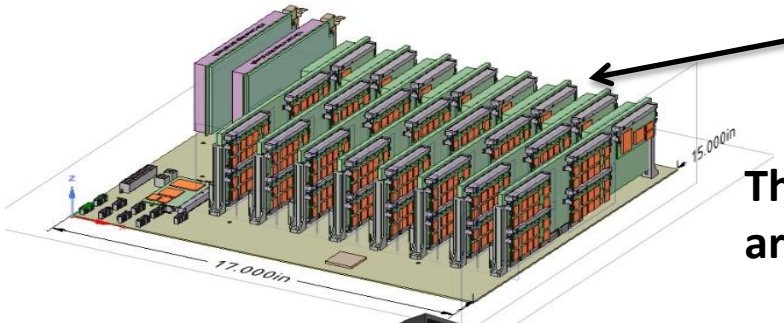
Up Links

(c) SuperNode

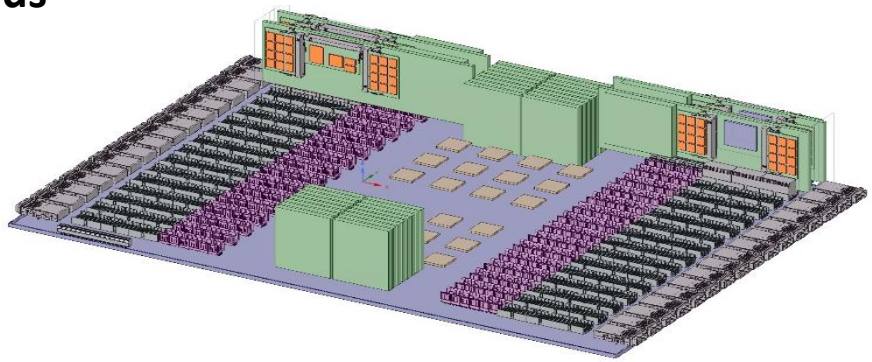
Stationary Core Runs OS, Launches Jobs

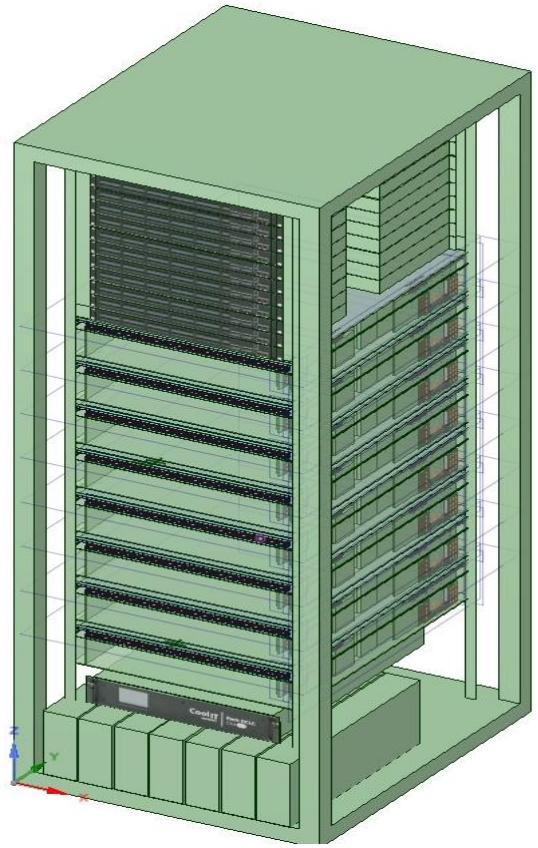
Near Term Scaling





The node boards are the same!





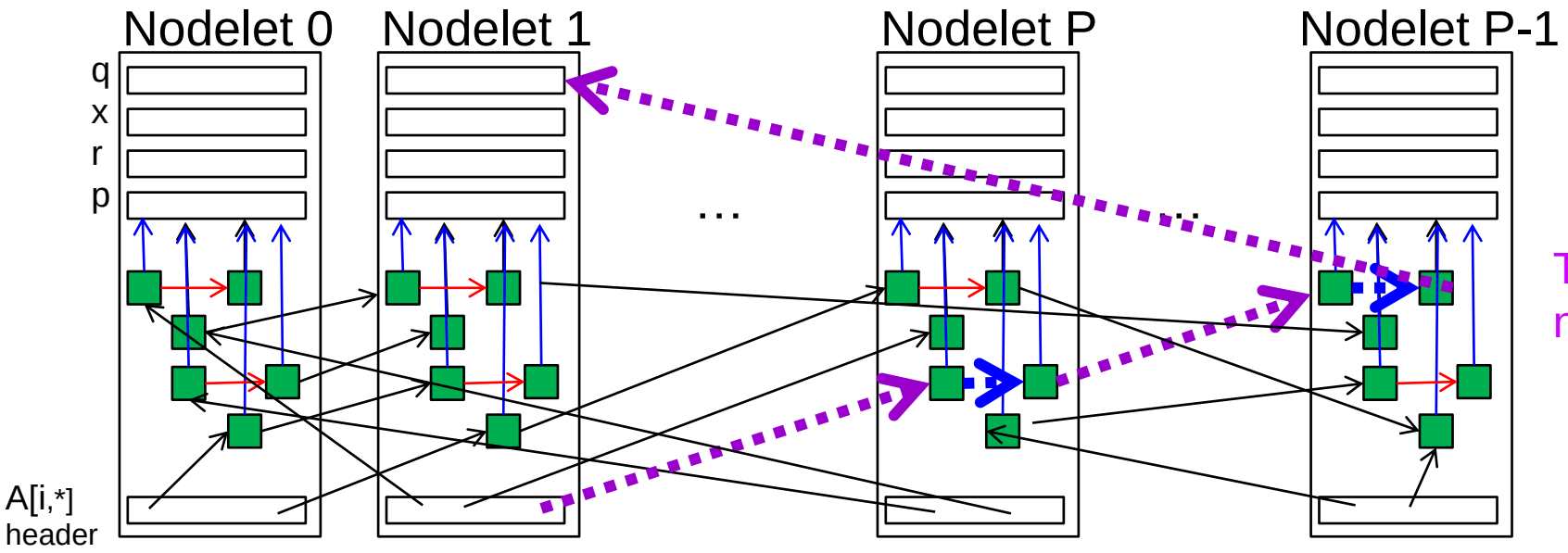
Emu Chick

- 8 Nodes, 64 Memory Channels
- Copy room environment

Emu1 Memory Server

- 256 nodes, 2048 Memory Channels
- Server room environment

Sparse Matrix-Dense Vector with Migrating Threads



Thread carries i, sum, nextj with it

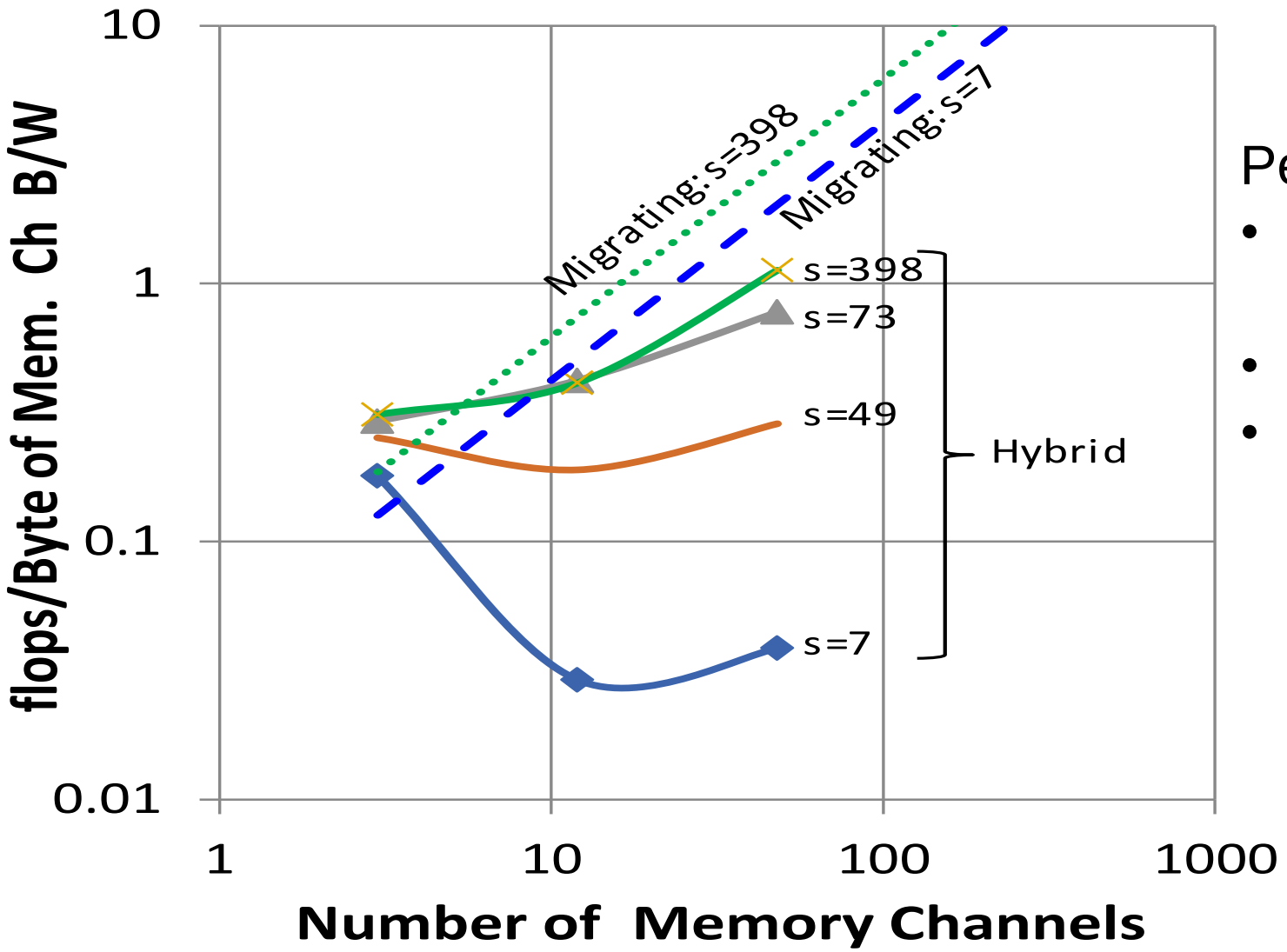
```
struct Aelt {
   int col;
   Aelt *next_rowelt};
nextj = Ahdr[i];
sum=0;
while (nextj != 0) {
       sum += x[(*nextj).col];
       nextj = (*nextj).next_rowelt;
};
q[i] = sum;
```

- One Aelt for each non-zero in some row of A
  - Non-zero value
  - Column index
  - Pointer to next non-zero

Migration before access

No migration before access

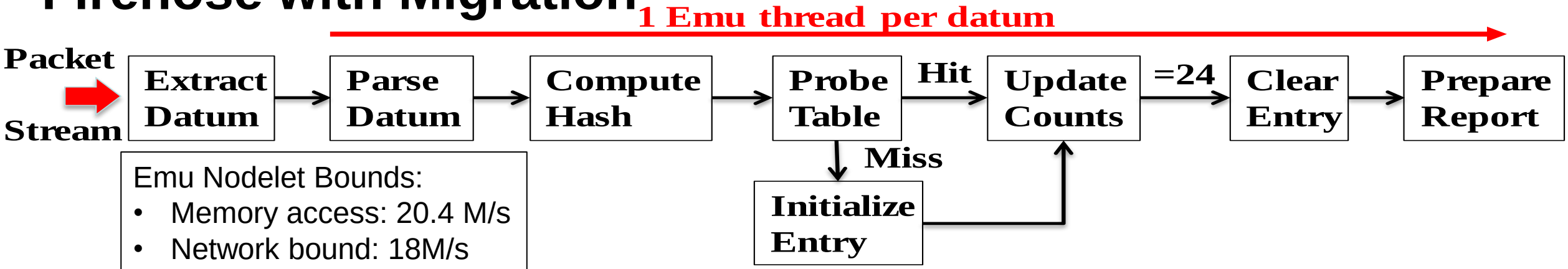
SpMV with Migrating Threads



Per row for migrating threads:

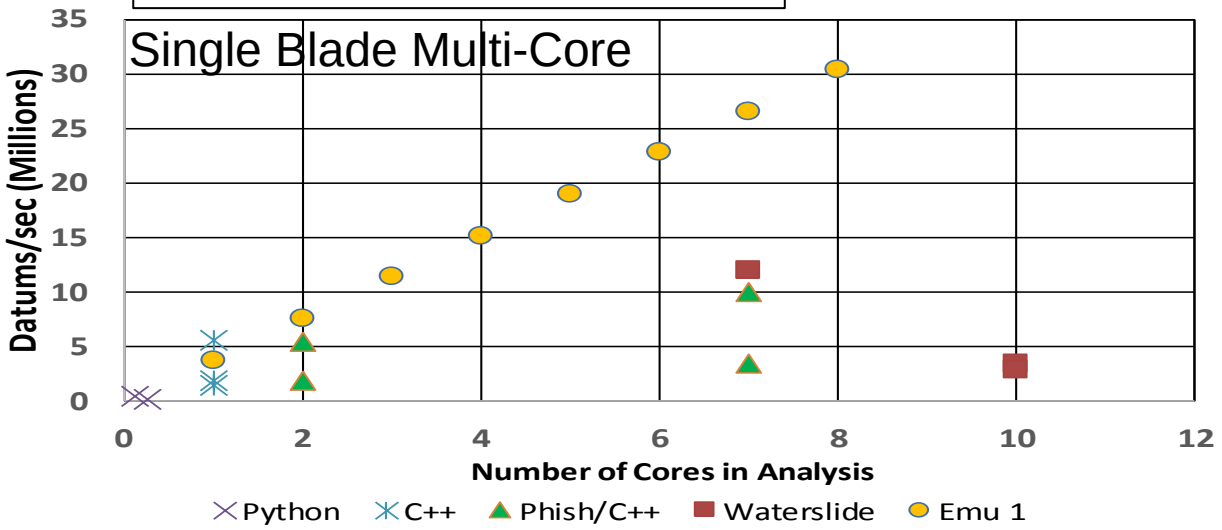
- Stinger-like multiple CSR blocks
- 32s+108 bytes
- At most s+1 migrations

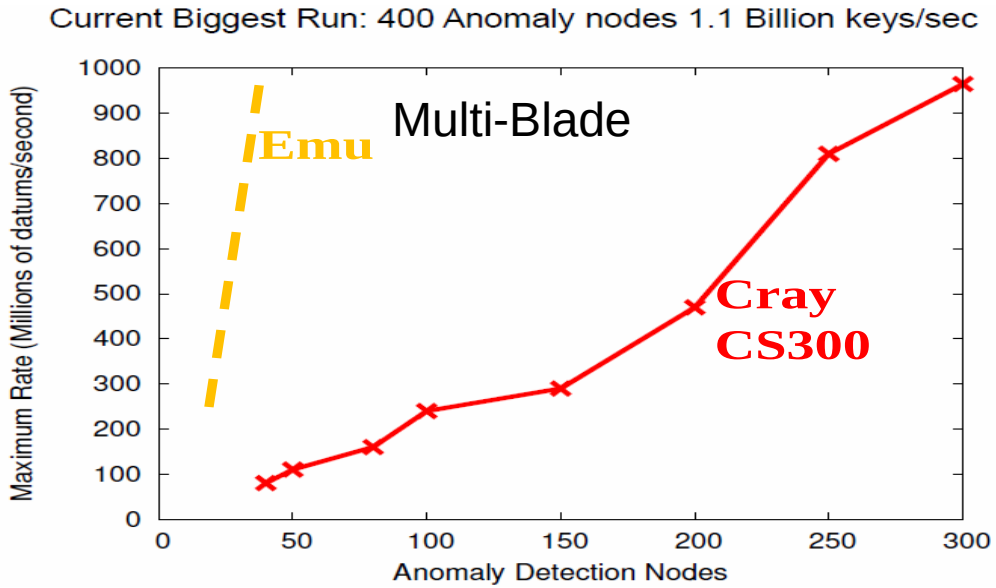
Firehose with Migration



Emu Nodelet Bounds:

- Memory access: 20.4 M/s
- Network bound: 18M/s
- Instr. Rate bound: 3.8M/s





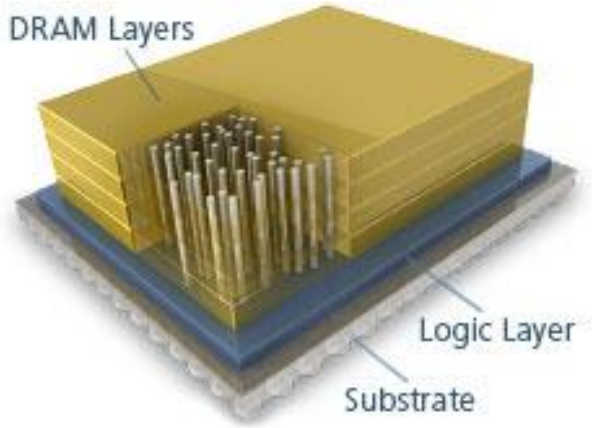
- 1 Emu Chick (64 nodelets) = 88X a CS300 node
- 1 Emu Rack (2048 nodelets) = 35X a CS300 rack

Argonne NATIONAL LABORATORY

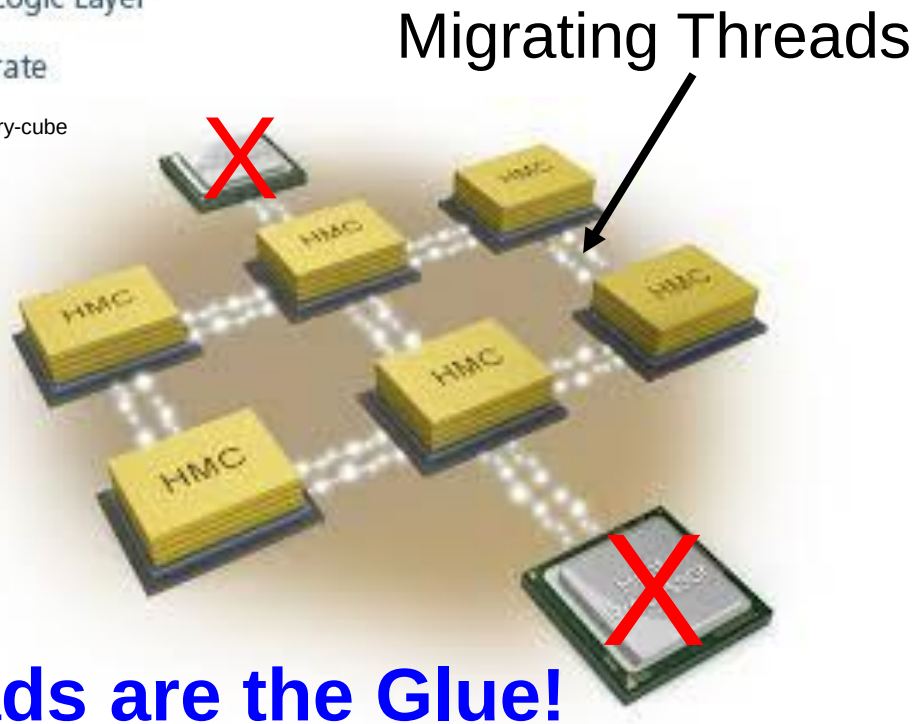
ECP EXASCALE COMPUTING PROJECT

Ultimate Scaling: Sea of Memory Stacks

- Add Cores below each vault
- Upgrade off-stack interfaces to full peer-peer protocol
- Add in second stack of non-volatile
- Result: standalone stack with 32 independent nodelets



http://www.micron.com/products/hybrid-memory-cube

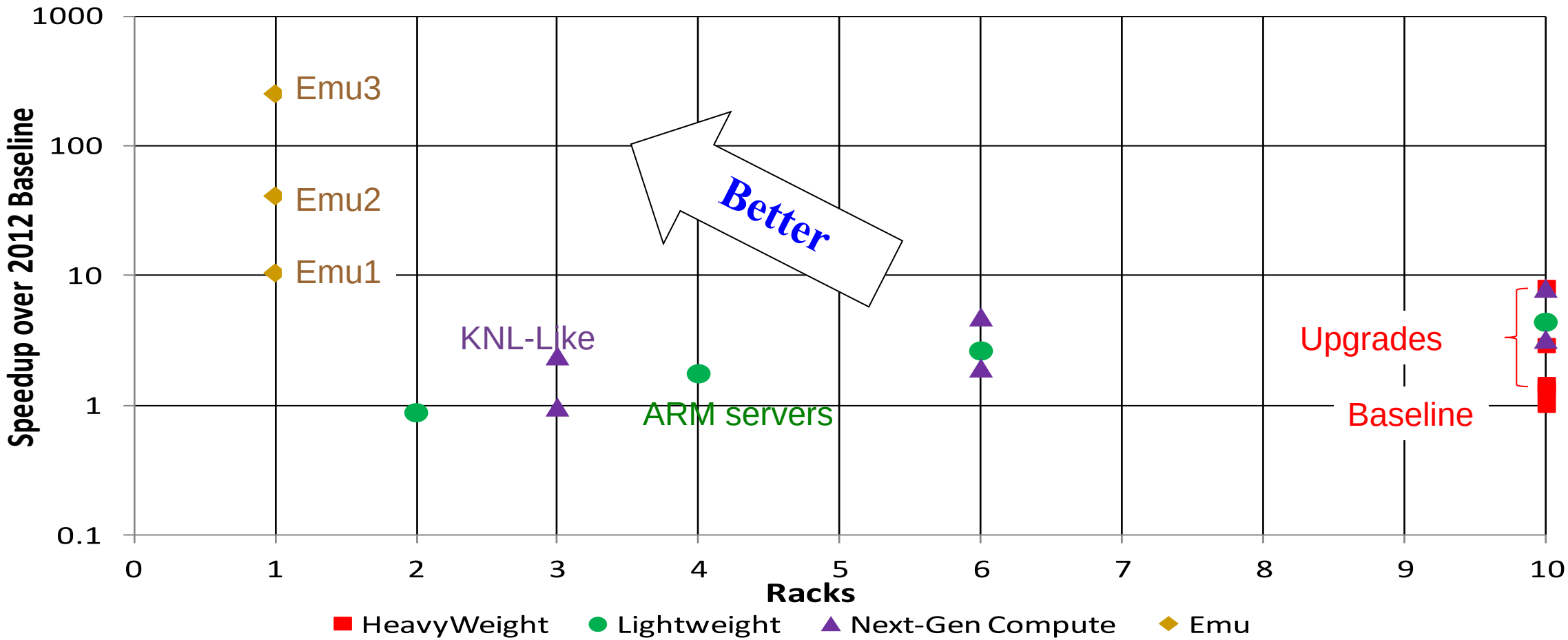


Migrating Threads are the Glue!

Argonne NATIONAL LABORATORY

ECP EXASCALE COMPUTING PROJECT

Projection for Massive "Batch-Mode"LexisNexis Problem



Emu1 assumes 400MHz GCs
2400 MT/s DRAM Channels

Real-Time Streaming Version Even Better

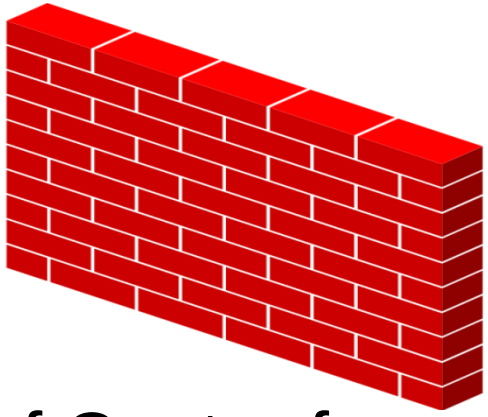
Argonne NATIONAL LABORATORY

ECP EXASCALE COMPUTING PROJECT

Conclusions

- Non-locality increasing rapidly in real apps
- Current architectures becoming badly inefficient
- The problem is in the memory & scaling
- Growing need for “remote functions”
- Migrating threads greatly simplify all
- Natural projection to 3D systems

Acknowledgements



Benchmark Analysis in part by:

- NSF grant CCF-1642280
- US Dept. of Energy, Award# DE-NA0002377, as part of Center for Shock-Wave Processing of Advanced Reactive Materials (C-SWARM) at the Univ. of Notre Dame, Notre Dame, IN

Emu hardware design by Emu Solutions, Inc