Distributed Storage Systems for Extreme-Scale Data-Intensive Computing

Ioan Raicu
Computer Science Department, Illinois Institute of Technology
Math and Computer Science Division, Argonne National Laboratory

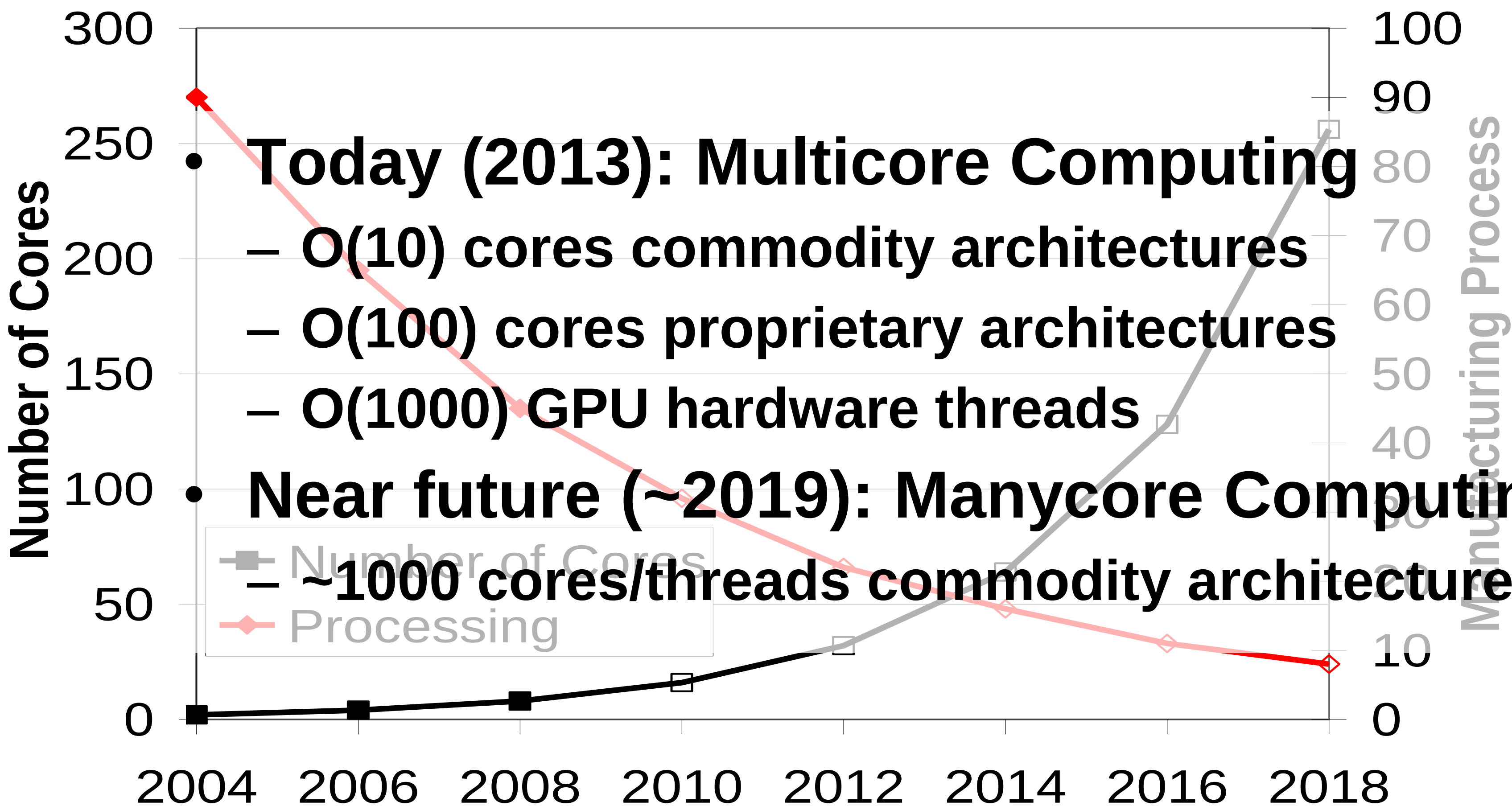
September 10th, 2013
2nd Annual CHANGES Workshop 2013

DataSys: Data-Intensive Distributed Systems Laboratory

- **Research Focus**
  - Emphasize designing, implementing, and evaluating systems, protocols, and middleware with the goal of supporting **data-intensive applications on extreme scale distributed systems**, from many-core systems, clusters, grids, clouds, and supercomputers
- **People**
  - Dr. Ioan Raicu (Director)
  - 6 PhD Students
  - 2 MS Students
  - 4 UG Students

- **Contact**
  - http://datasys.cs.iit.edu/
  - iraicu@cs.iit.edu

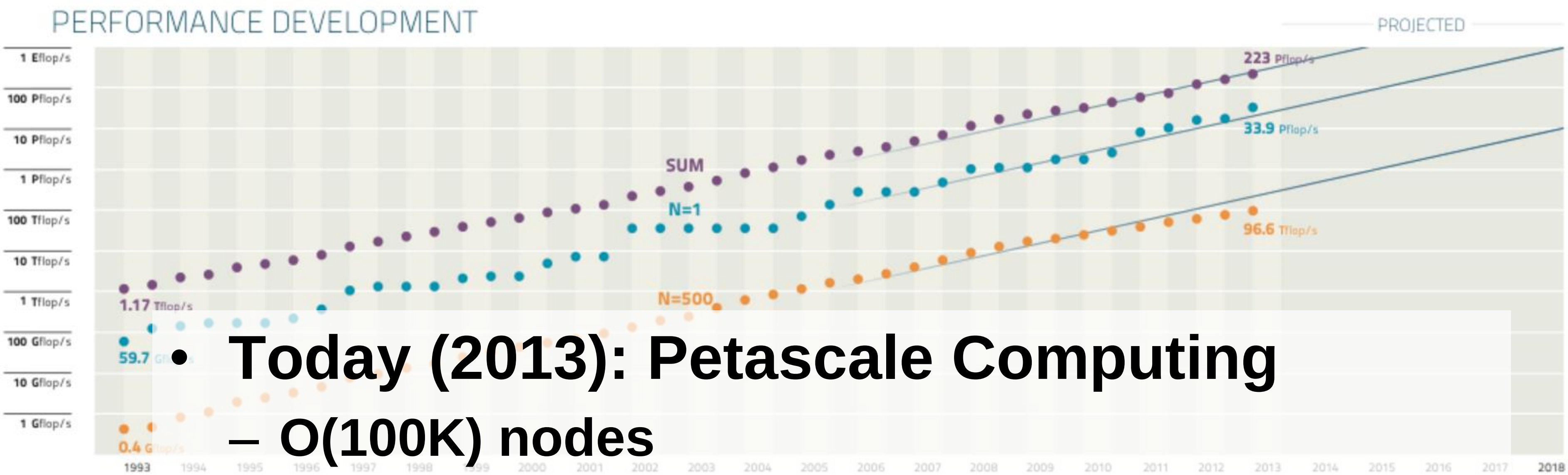
Manycore Computing



- **Today (2013): Multicore Computing**
  - **O(10) cores commodity architectures**
  - **O(100) cores proprietary architectures**
  - **O(1000) GPU hardware threads**
- **Near future (~2019): Manycore Computing**
  - **~1000 cores/threads commodity architectures**

Pat Helland, Microsoft, The Irresistible Forces Meet the Movable Objects, November 9th, 2007

Exascale Computing



- **Today (2013): Petascale Computing**
  - **O(100K) nodes**
  - **O(1M) cores**
- **Near future (~2018): Exascale Computing**
  - **~1M nodes (10X)**
  - **~1B processor-cores/threads (1000X)**

http://s.top500.org/static/lists/2013/06/TOP500_201306_Poster.png

Exascale Computing Architecture

- Compute
  - 1M nodes, with ~1K threads/cores per node
- Networking
  - N-dimensional torus
  - Meshes
  - Dragonfly
- Storage
  - SANs with spinning disks will replace today's tape
  - SANs with SSD acceleration/caching
  - SSDs likely to exist at every node (e.g. burst buffer moves storage closer to compute nodes)

Some Challenges to Overcome at Exascale Computing

- Programming paradigms
  - HPC is dominated by MPI today
  - Will MPI scale another 3 orders of magnitude?
  - Other paradigms (including loosely coupled ones) might emerge to be more flexible, resilient, and scalable
- Storage systems will need to become more distributed to scale ➔ Critical for resilience of HPC
- Network topology must be used in job management, data management, compilers, etc
- Power efficient compilers and run-time systems

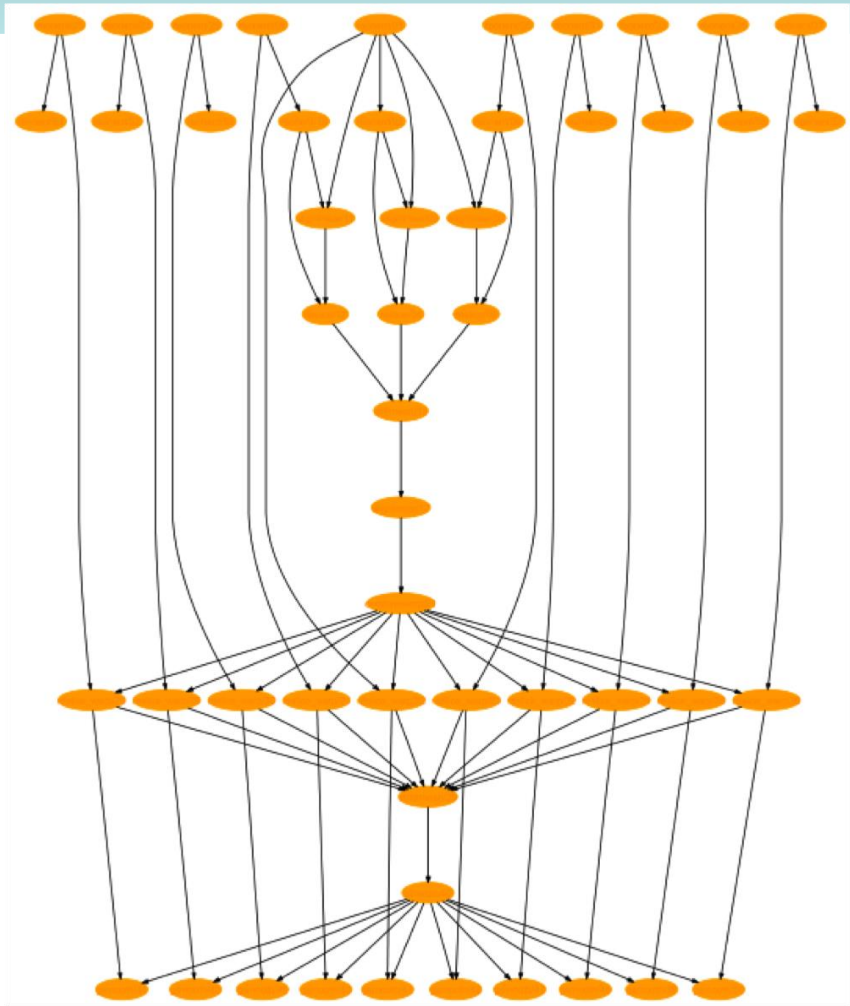
Critical Technologies Needed to achieve Extreme Scales

- Fundamental Building Blocks (with a variety of resilience and consistency models)
  - **Distributed hash tables (DHT)**
    - Also known as NoSQL data stores or key/value stores
    - Examples: Chord, Tapestry, memcached, Dynamo, MangoDB, Kademlia, CAN, Tapestry, Memcached, Cycloid, Ketama, RIAK, Maidsafe-dht, Cassandra and C-MPI, BigTable, HBase
  - **Distributed message queues (DMQ)**
    - Example: SQS, RabitMQ, Couch RQS, ActiveMQ, KAFKA, Hedwig

DHT and DMQ → Future generation distributed systems

- Global File Systems, Metadata, and Storage
  - Data-diffusion, FusionFS, ZHT
- Job Management Systems
  - Falkon, MATRIX, CloudKon, GeMTC
- Workflow Systems
  - Swift
- Monitoring Systems
  - Built on top of ZHT
- Provenance Systems
  - Built on top of ZHT and FusionFS
- Data Indexing
  - Future work on top of ZHT/FusionFS
- Relational Databases
  - Future work on top of ZHT

Many-Task Computing (MTC)



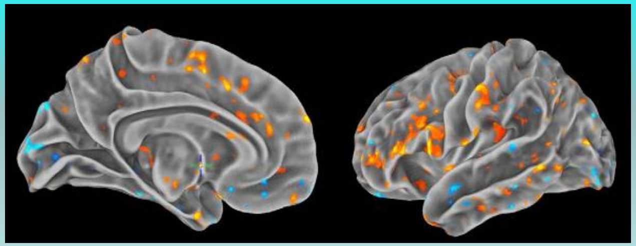
MTC emphasizes:

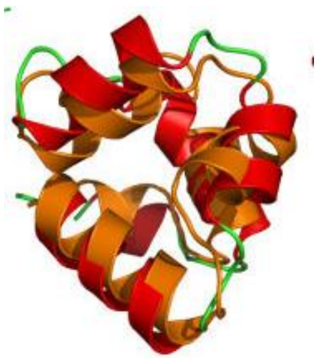
- bridging HPC/HTC
- many resources
  - short period of time
- many computational tasks
- dependent/independent tasks
- tasks organized as DAGs
- primary metrics are seconds

Advantages:

- Improve fault tolerant
- Maintain efficiency
- Programmability & Portability
- support embarrassingly parallel and parallel applications

Swift/T and Applications





- Swift/T
  - Active research project (CI UChicago & ANL)
  - Parallel Programming Framework
  - Throughput ~25k tasks/sec per process
  - Shown to scale to 128k cores
- Application Domains Supported
  - Astronomy, Biochemistry, Bioinformatics, Economics, Climate

Swift lets you write parallel scripts that run many copies of ordinary programs concurrently, using statements like this:

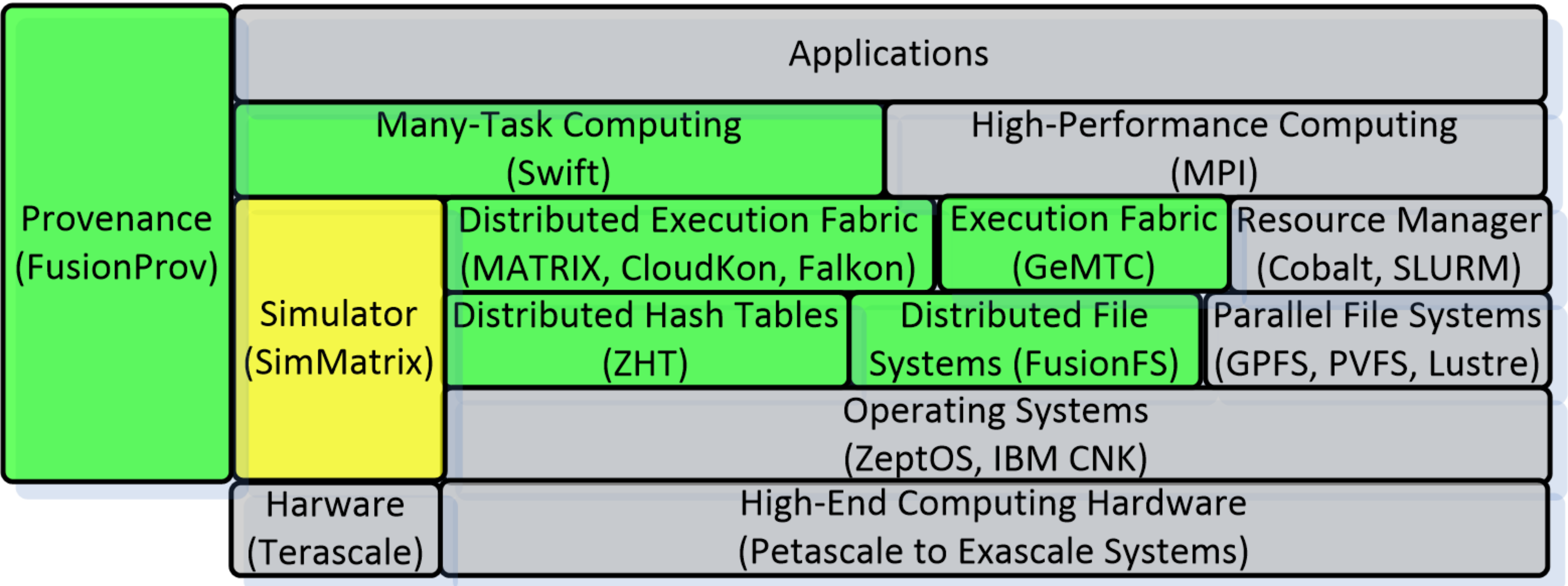
```
foreach protein in proteinList {
  runBLAST(protein);
}
```

Images from Swift Case Studies - http://www.ci.uchicago.edu/swift/case_studies/

Swift Applications

Field	Description	Characteristics	Status
Astronomy	Creation of montages from many digital images	Many 1-core tasks, much communication, complex dependencies	E
Astronomy	Stacking of cutouts from digital sky surveys	Many 1-core tasks, much communication	E (Falkon)
Biochemistry	Analysis of mass-spec data for post-translational protein modifications	10,000 – 100,000 K jobs for proteomic searches using custom serial codes	D
Biochemistry	Protein folding using iterative fixing algorithm, also exploring other biomolecule interactions	100s to 1000s of 1-1000 core simulations & data analysis	O
Biochemistry	Identification of drug targets via computational screening	Up to 1M x 1 core	O (Falkon)
Bioinformatics	Metagenome modeling	1000's of 1-core integer programming problems	D
Business economics	Mining of large text corpora to study media bias	Analysis and comparison of 70M+ text files of news articles	D
Climate	Ensemble climate model runs and analysis of output data	10s to 100s of 100-1000 core simulations	E
Economics	Generation of response surfaces for various economic models	1K to 1M 1-core runs (10K typical), then data analysis	O
Neuroscience	Analysis of functional MRI datasets	Comparison of images; connectivity analysis with SEM, many tasks (100K+)	O
Radiology	Training of computer aided diagnosis algorithms	Comparison of images; many tasks, much communication	D
Radiology	Image processing and brain mapping for neurosurgical planning research	1000's of MPI application executions	O

Proposed Software Stack in Large-Scale Distributed Systems



ZHT: Zero-Hope Distributed Hash Table

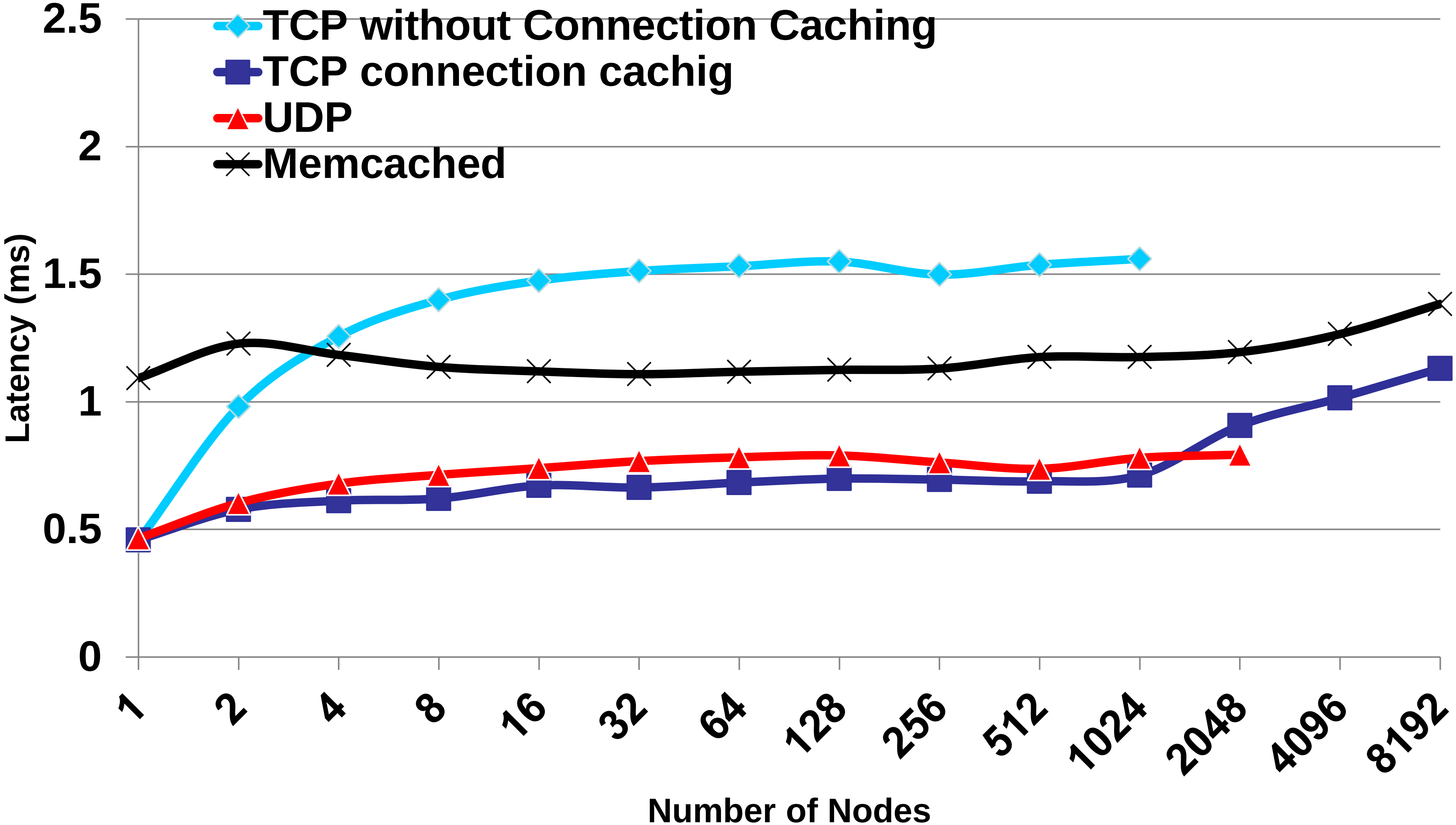
- ZHT: A distributed Key-Value store
  - Light-weighted
  - High performance
  - Scalable
  - Dynamic
  - Fault tolerant
  - Strong Consistency
  - Persistent
  - Versatile: works from clusters, to clouds, to supercomputers

ZHT: Zero-Hope Distributed Hash Table

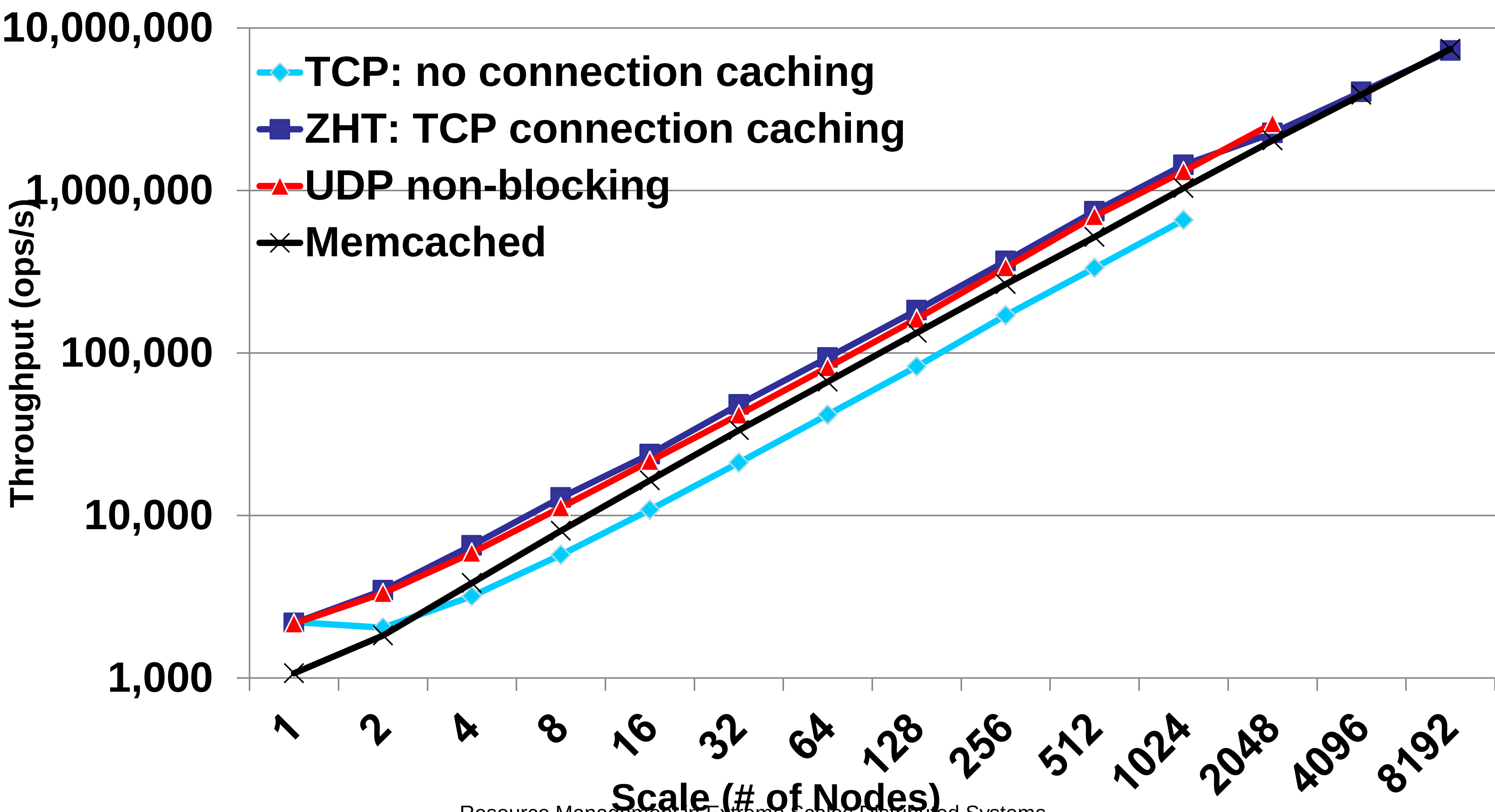
- Many DHTs: Chord, Kademlia, Pastry, Cassandra, C-MPI, Memcached, Dynamo
- Why another?

Name	Impl.	Routing Time	Persistence	Dynamic membership	Append Operation
Cassandra	Java	Log(N)	Yes	Yes	No
C-MPI	C	Log(N)	No	No	No
Dynamo	Java	0 to Log(N)	Yes	Yes	No
Memcached	C	0	No	No	No
ZHT	C++	0 to 2	Yes	Yes	Yes

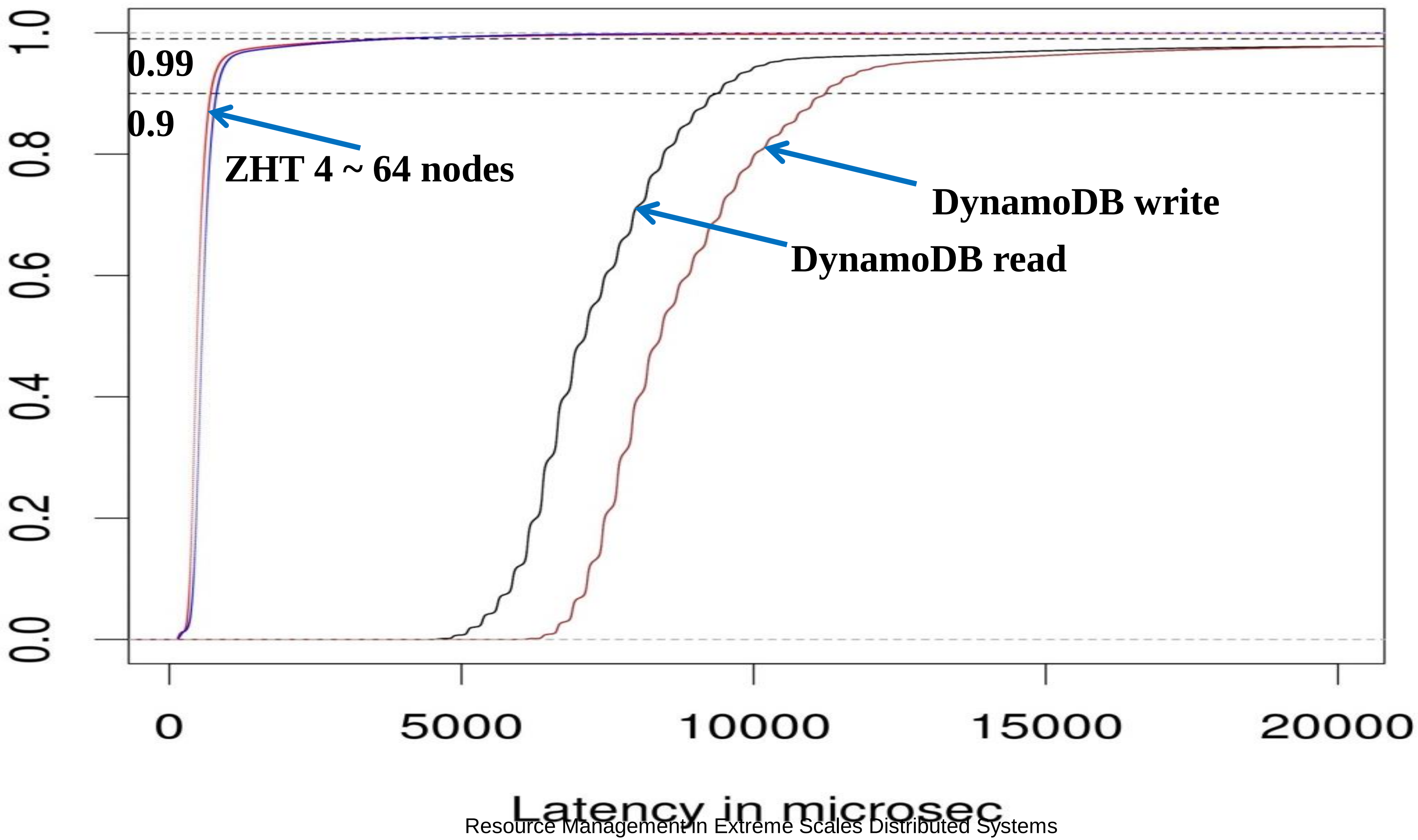
ZHT: Zero-Hope Distributed Hash Table



ZHT: Zero-Hope Distributed Hash Table

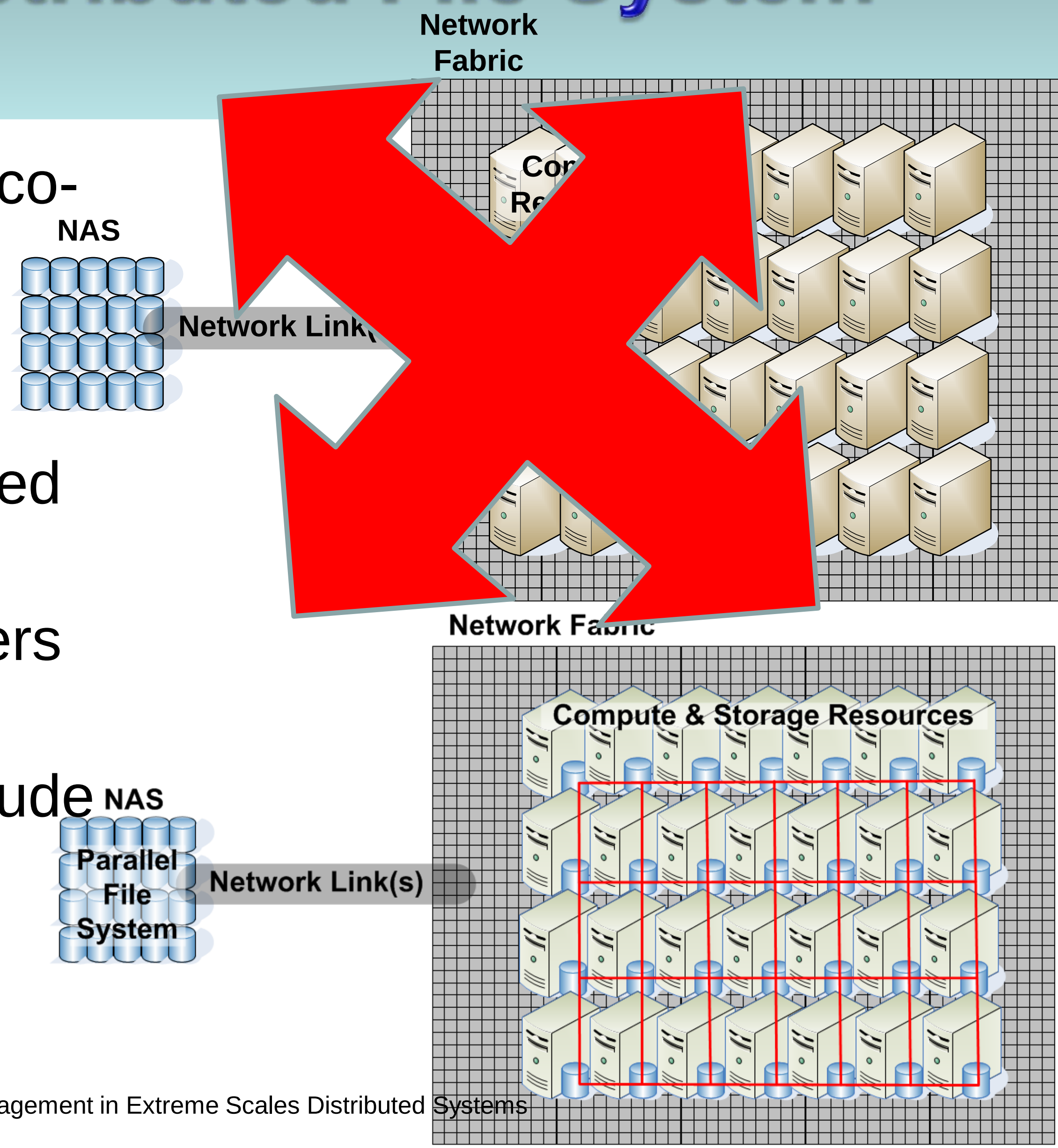


ZHT: Zero-Hope Distributed Hash Table

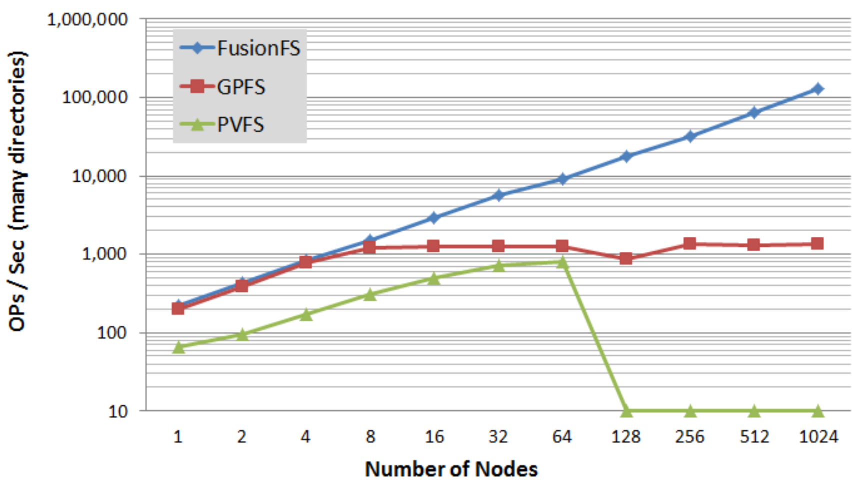


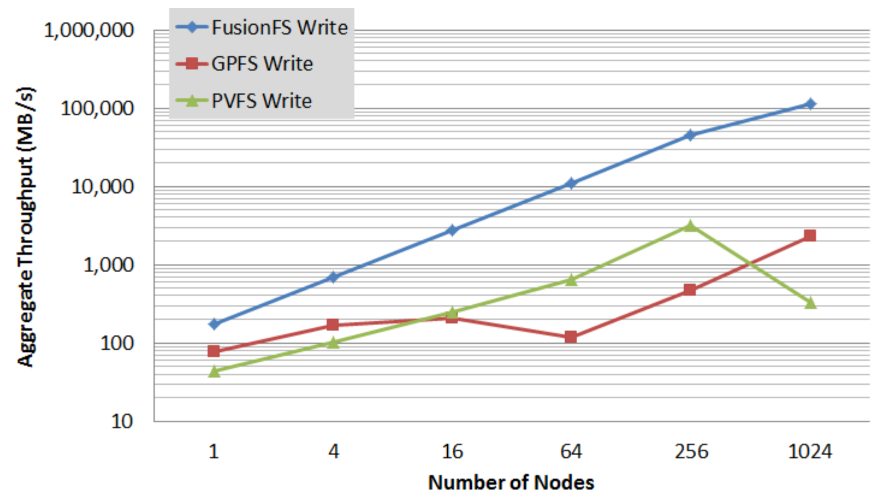
FusionFS Distributed File System

- A distributed file system co-locating storage and computations, while supporting POSIX
- Everything is decentralized and distributed
- Aims for millions of servers and clients scales
- Aims at orders of magnitude higher performance than current state of the art parallel file systems

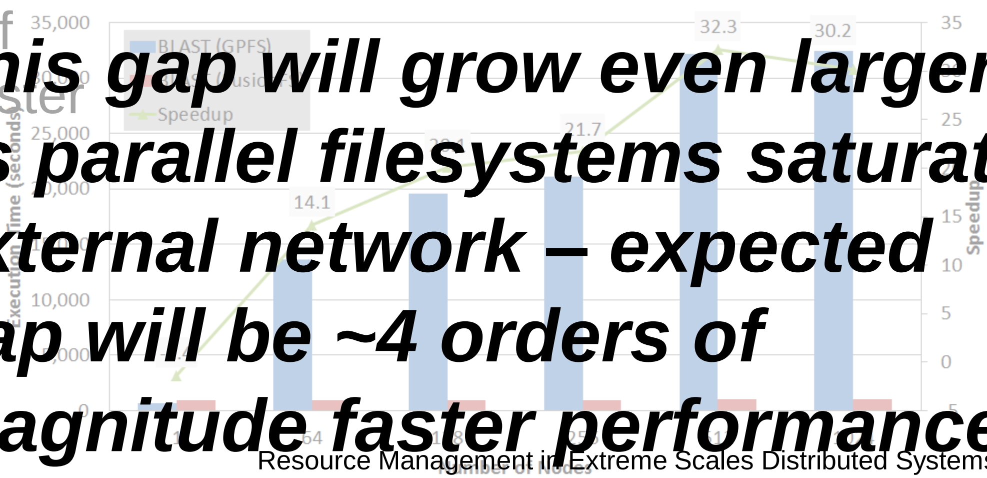


FusionFS Distributed File System





^ ~2 orders of magnitude faster metadata



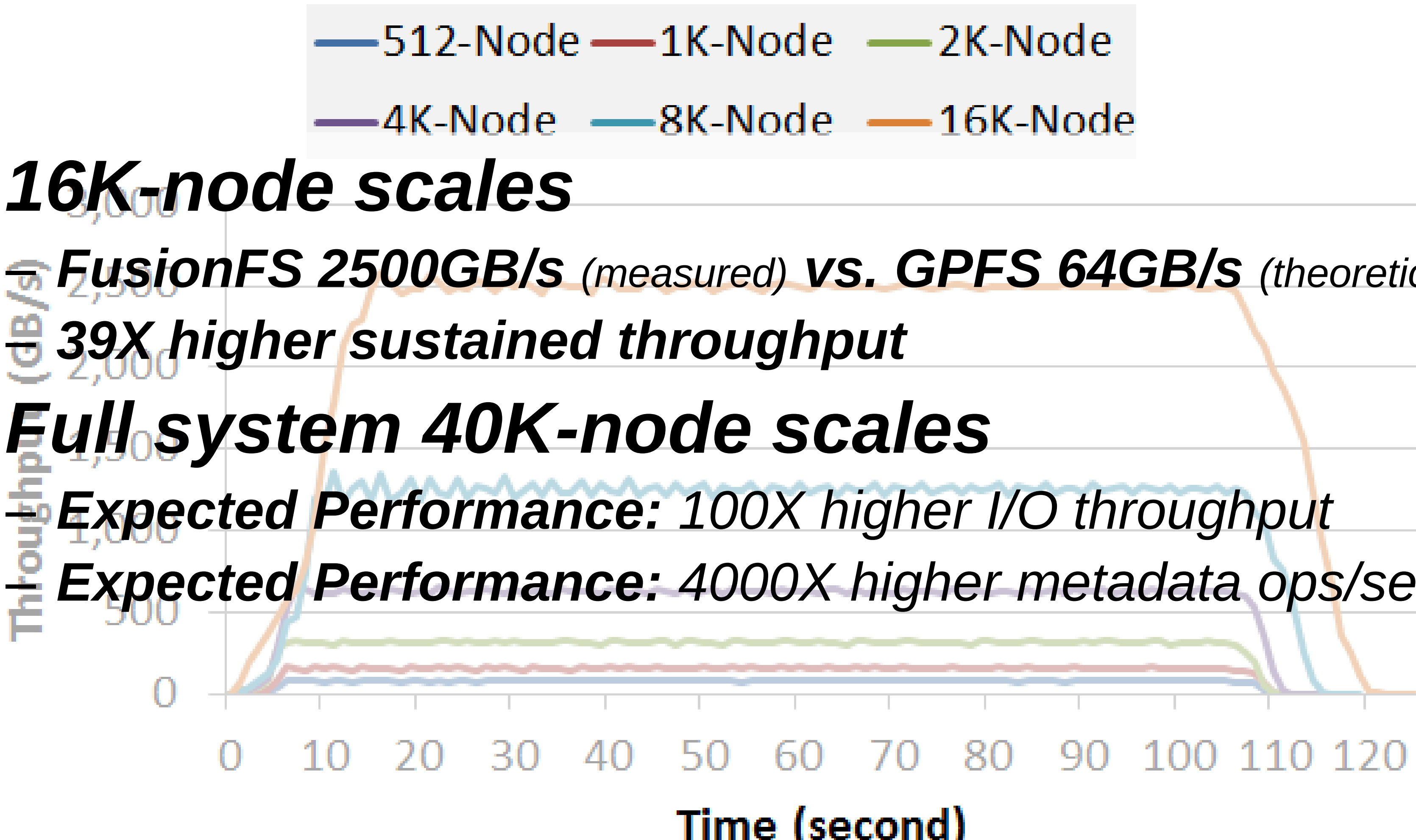
This gap will grow even larger as parallel filesystems saturate external network – expected gap will be ~4 orders of magnitude faster performance

^ ~1.5 order of magnitude faster I/O

< ~1.5 order of magnitude faster runtime for real application

FusionFS Distributed File System

- ***16K-node scales***
  - ***FusionFS 2500GB/s** (measured) **vs. GPFS 64GB/s** (theoretical)*
  - ***39X higher sustained throughput***
- ***Full system 40K-node scales***
  - ***Expected Performance:** 100X higher I/O throughput*
  - ***Expected Performance:** 4000X higher metadata ops/sec*



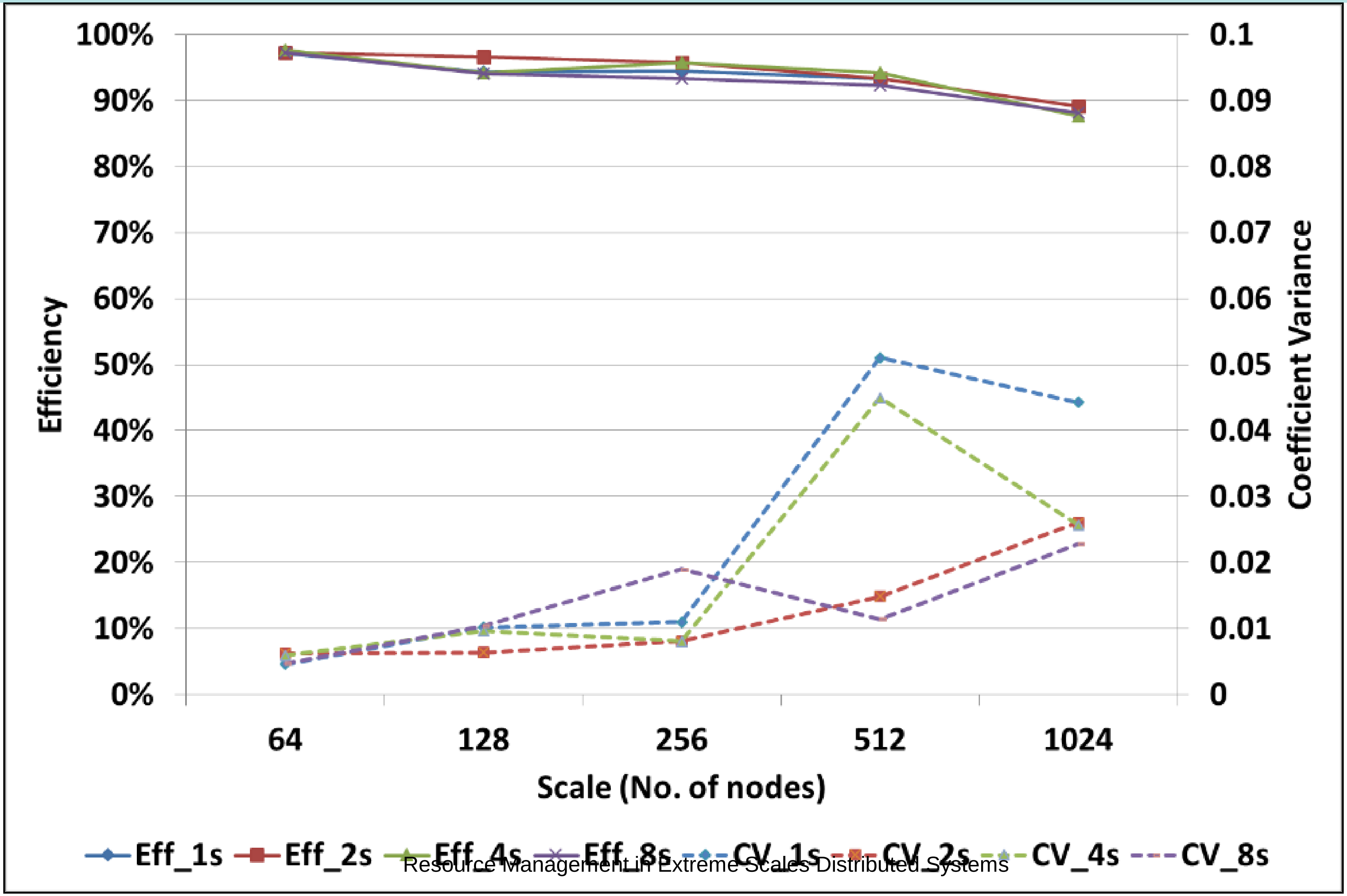
FusionFS Distributed File System

- Many hot topics related to distributed storage
  - Provenance (FusionProv) – uses ZHT
  - Information Dispersal Algorithms (IStore) – uses GPUs
  - SSD+HHD hybrid caching (HyCache)
  - Data Compression
- Improvements on the horizon
  - Non-POSIX interfaces (e.g. Amazon S3)
  - Explore viability of supporting HPC checkpointing
  - Deep indexing and search

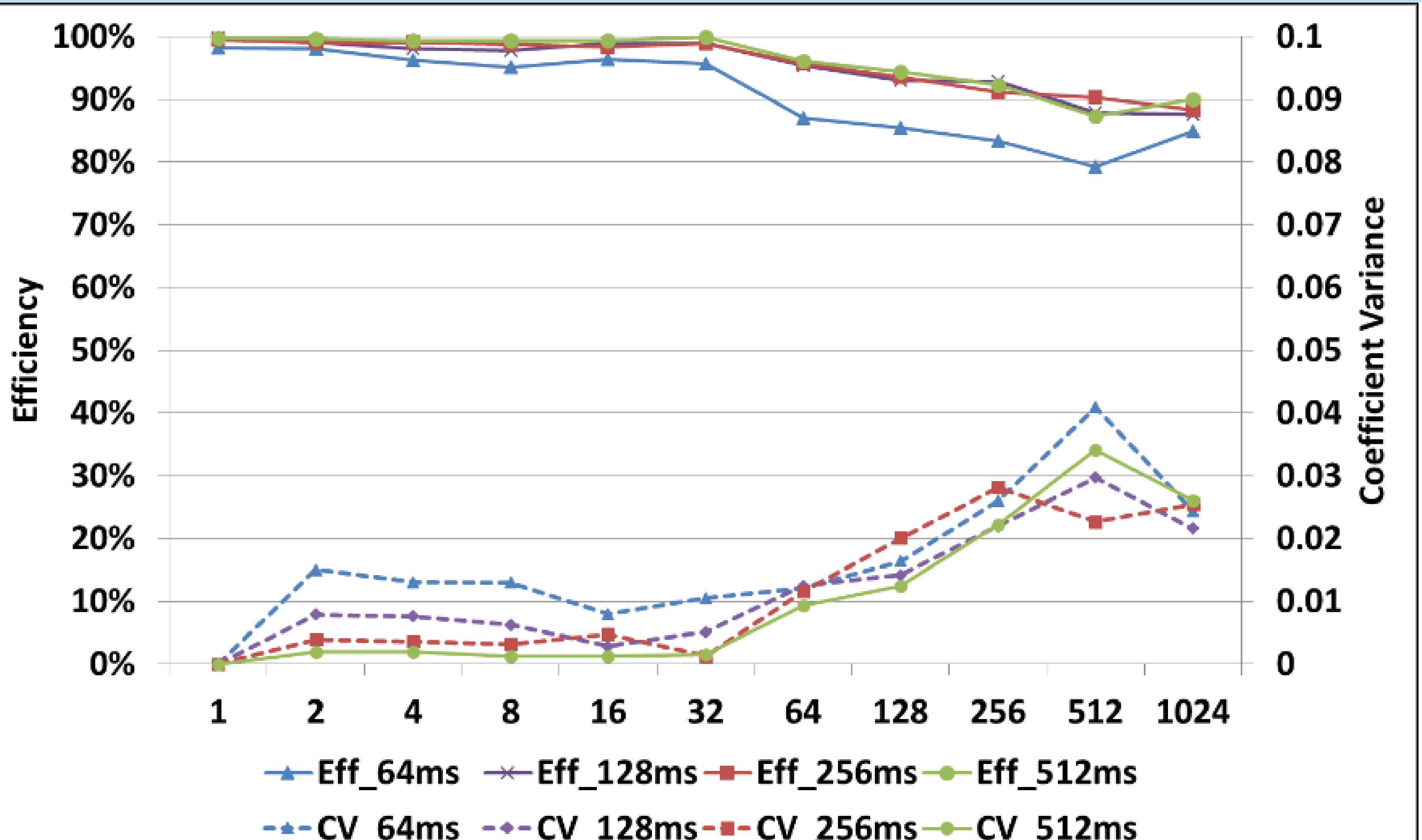
MATRIX MTC execution Framework at Extreme Scales

- MATRIX - distributed MTC execution framework for distributed load balancing using Work Stealing algorithm
  - Distributed scheduling is an efficient way to achieve load balancing, leading to high job throughput and system utilization
  - Dynamic job scheduling system at the granularity of node/core levels for extreme scale applications

MATRIX MTC execution Framework at Extreme Scales



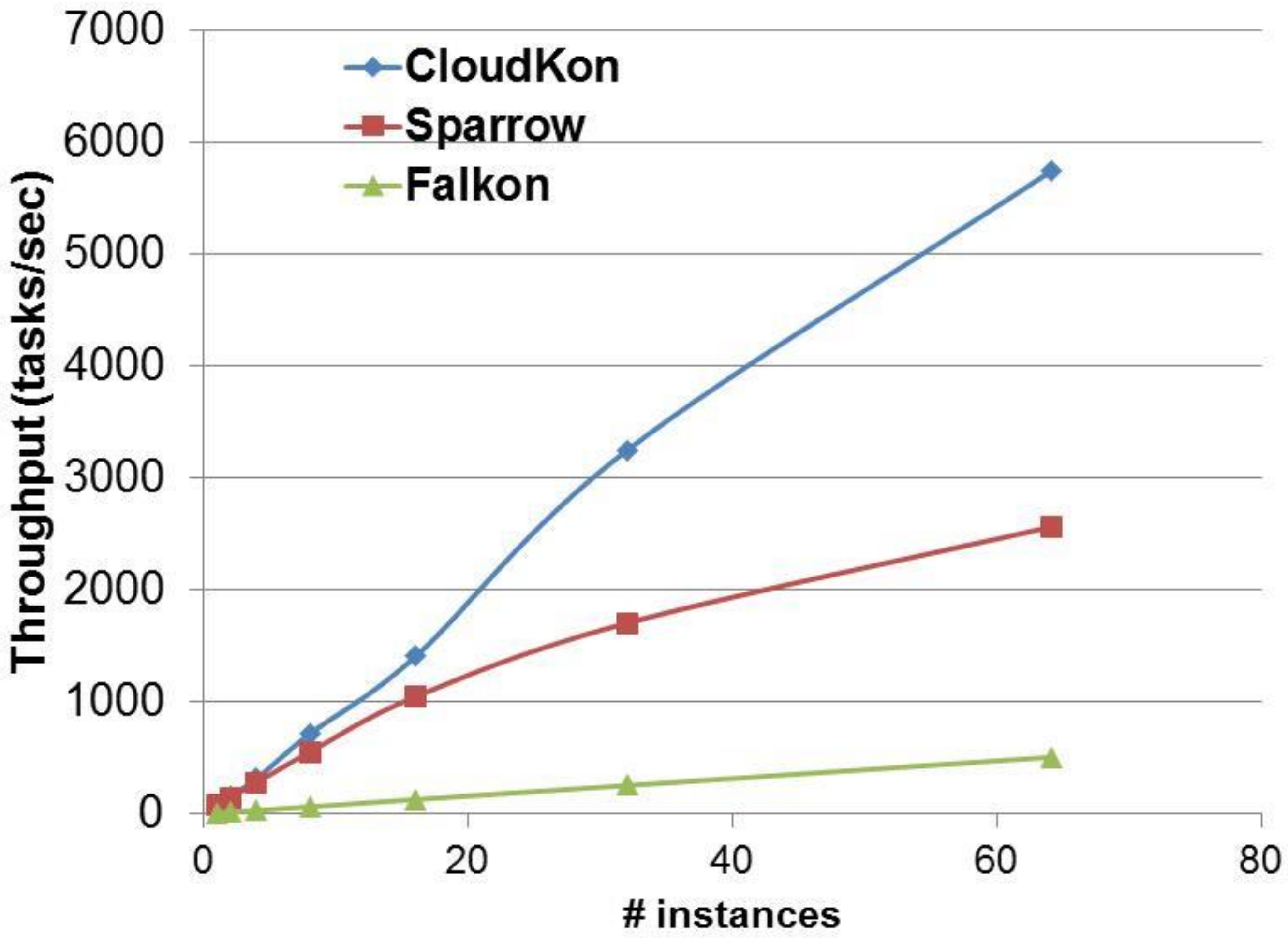
MATRIX MTC execution Framework at Extreme Scales



CloudKon: Cloud-Enbled Distributed Task Execution Framework

- Use Amazon services as building blocks
  – SQS, DynamoDB, and EC2
- Distributed load balancing
- Dynamic and Elastic
- Light-weight and Fast (2X+)
- Small codebase (1K-LOC, 5%)

CloudKon: Cloud-Enbled Distributed Task Execution Framework



GeMTC:
GPU-Enabled Many-Task Computing

GPU

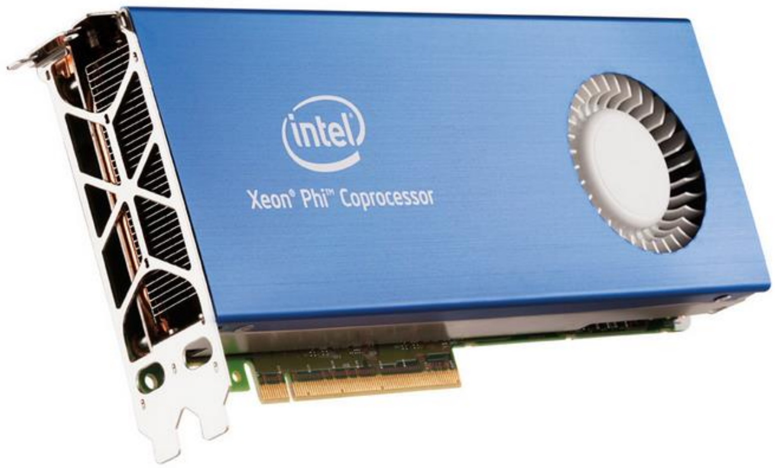
- Streaming Multiprocessors (15 SMXs on Kepler K20)
- 192 warps * 32 threads

Coprocessors

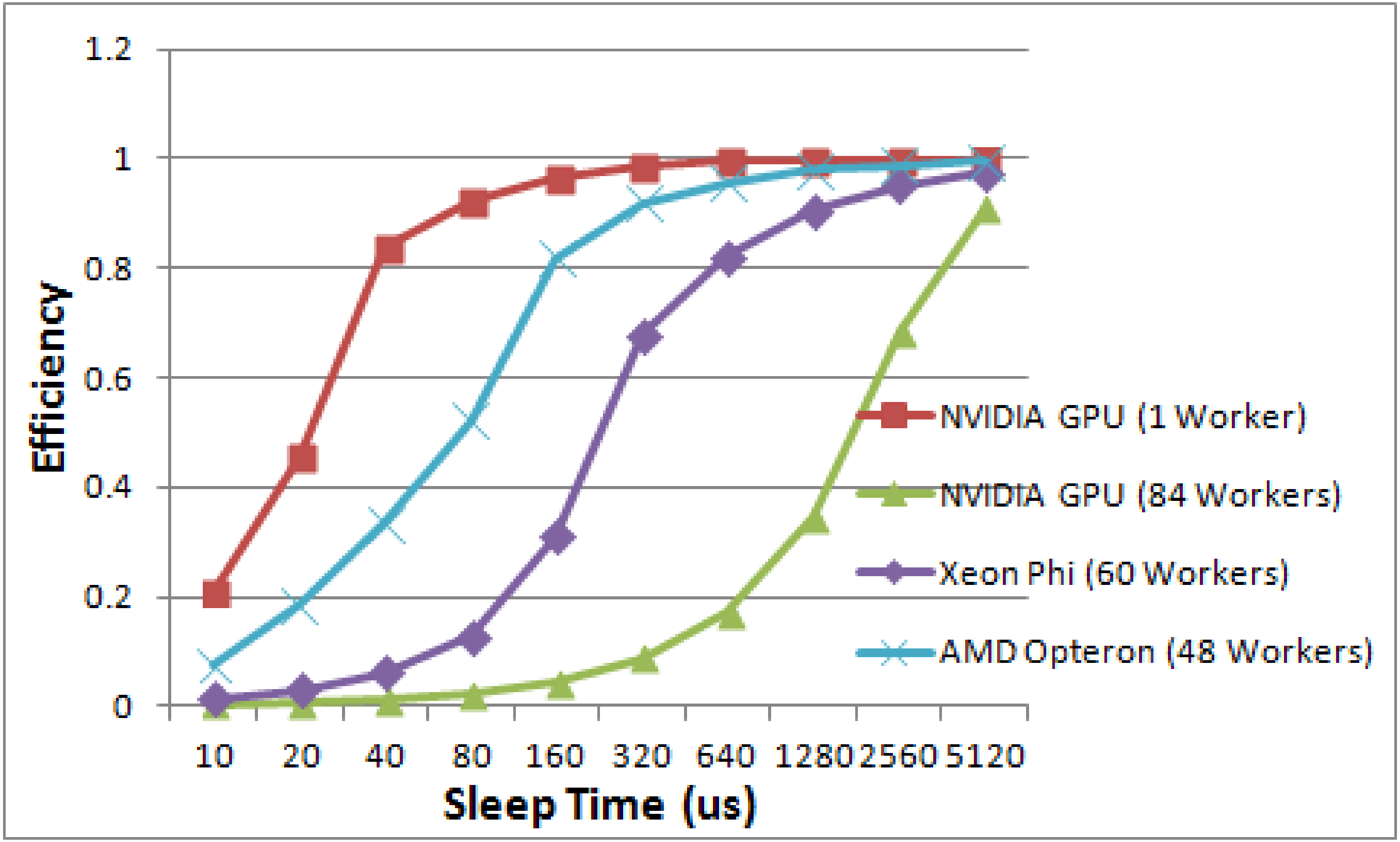
- Intel Xeon Phi
- 60 cores * 4 threads per core = 240 hardware threads

GeMTC

- Efficient support for MTC on accelrators



GeMTC: GPU-Enabled Many-Task Computing



Main Message

- ***Decentralization is critical***
  – Computational resource management
  – Storage systems
- ***Preserving locality is critical!***
  – POSIX I/O on shared/parallel file systems ignore locality
  – Data-aware scheduling coupled with distributed file systems that expose locality is the key to scalability over the next decade
- *Co-locating storage and compute is* ***GOOD***
  – Leverage the abundance of processing power, bisection bandwidth, and local I/O

Active Funding ($)

- **NSF CAREER 2011 – 2015: $486K**
  - "*Avoiding Achilles' Heel in Exascale Computing with Distributed File Systems*", NSF CAREER
- **DOE Fermi 2011 – 2013: $84K**
  - "Networking and Distributed Systems in High-Energy Physics", DOE FNAL
- **DOE LANL 2013: $75K**
  - "Investigation of Distributed Systems for HPC System Services", DOE LANL
- **IIT STARR 2013: $15K**
  - **"***Towards the Support for Many-Task Computing on Many-Core Computing Platforms***",** IIT STARR Fellowship
- **Amazon 2011 - 2013: $18K**
  - **"***Distributed Systems Research on the Amazon Cloud Infrastructure***",** Amazon
- **NVIDIA 2013 – 2014: $12K**
  - "CUDA Teaching Center", NVIDIA

Funding (Time)

- **DOE 2011 – 2013: 450K hours**
  - “*FusionFS: Distributed File Systems for Exascale Computing*”, DOE ANL ALCF; 450,000 hours on the IBM BlueGene/P
- **XSEDE 2013: 200K hours**
  - “*Many-Task Computing with Many-Core Accelerators on XSEDE*”, NSF XSEDE; 200K hours on XSEDE
- **GLCPC 2013: 6M hours**
  - “*Implicitly-parallel functional dataflow for productive hybrid programming on Blue Waters*”, Great Lakes Consortium for Petascale Computation (GLCPC); 6M hours on the Blue Waters Supercomputer
- **NICS 2013: 320K hours**
  - “Many-Task Computing with Many-Core Accelerators on Beacon”, National Institute for Computational Sciences (NICS); 320K hours on the Beacon system

Service Activities

- IEEE Transactions on Cloud Computing
  - Special Issue on Scientific Cloud Computing
- Springer's Journal of Cloud Computing: Advances, Systems and Applications
- IEEE/ACM MTAGS 2013 @ SC13
- IEEE/ACM DataCloud 2013 @ SC13
- ACM ScienceCloud 2014 @ HPDC14
- IEEE CCGrid 2014 in Chicago
- GCASR 2014 in Chicago
- Others:
  - IEEE/ACM SC 2013, ACM HPDC 2014, IEEE IPDPS 2014, IEEE ICDCS 2014, IEEE eScience 2014

More Information

- More information:
  - http://www.cs.iit.edu/~iraicu/
  - http://datasys.cs.iit.edu/
- Contact:
  - iraicu@cs.iit.edu
- Questions?