Exascale Many-Task Computing with a Billion Processors

Ioan Raicu
Center for Ultra-scale Computing and Information Security
Department of Electrical Engineering & Computer Science
Northwestern University

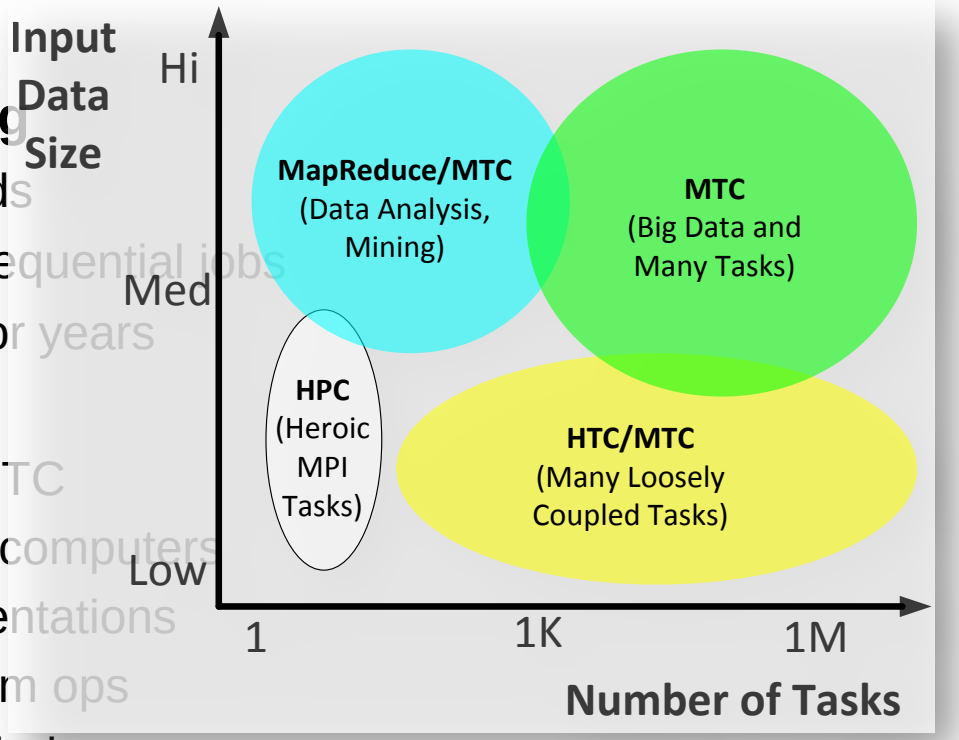
HPDC Program Committee Workshop
Northwestern University
March 22nd, 2010

Outline

- **Overview**
- Past Work
- Future Work
  - Motivation
  - Proposal
- Work-in-Progress

Many-Task Computing
HPC ← MTC → HTC

- **HPC: High-Performance Computing**
  - Synonymous with supercomputing
  - Tightly-coupled applications
  - Implemented using Message Passing Interface (MPI), needs low latency networks
  - Measured in FLOPS
- **HTC: High-Throughput Computing**
  - Typically applied in clusters and grids
  - Loosely-coupled applications with sequential jobs
  - Measured in operations per month or years
- **MTC: Many-Task Computing**
  - Bridge the gap between HPC and HTC
  - Applied in clusters, grids, and supercomputers
  - Loosely coupled apps with HPC orientations
  - Many activities coupled by file system ops
  - Many resources over short time periods

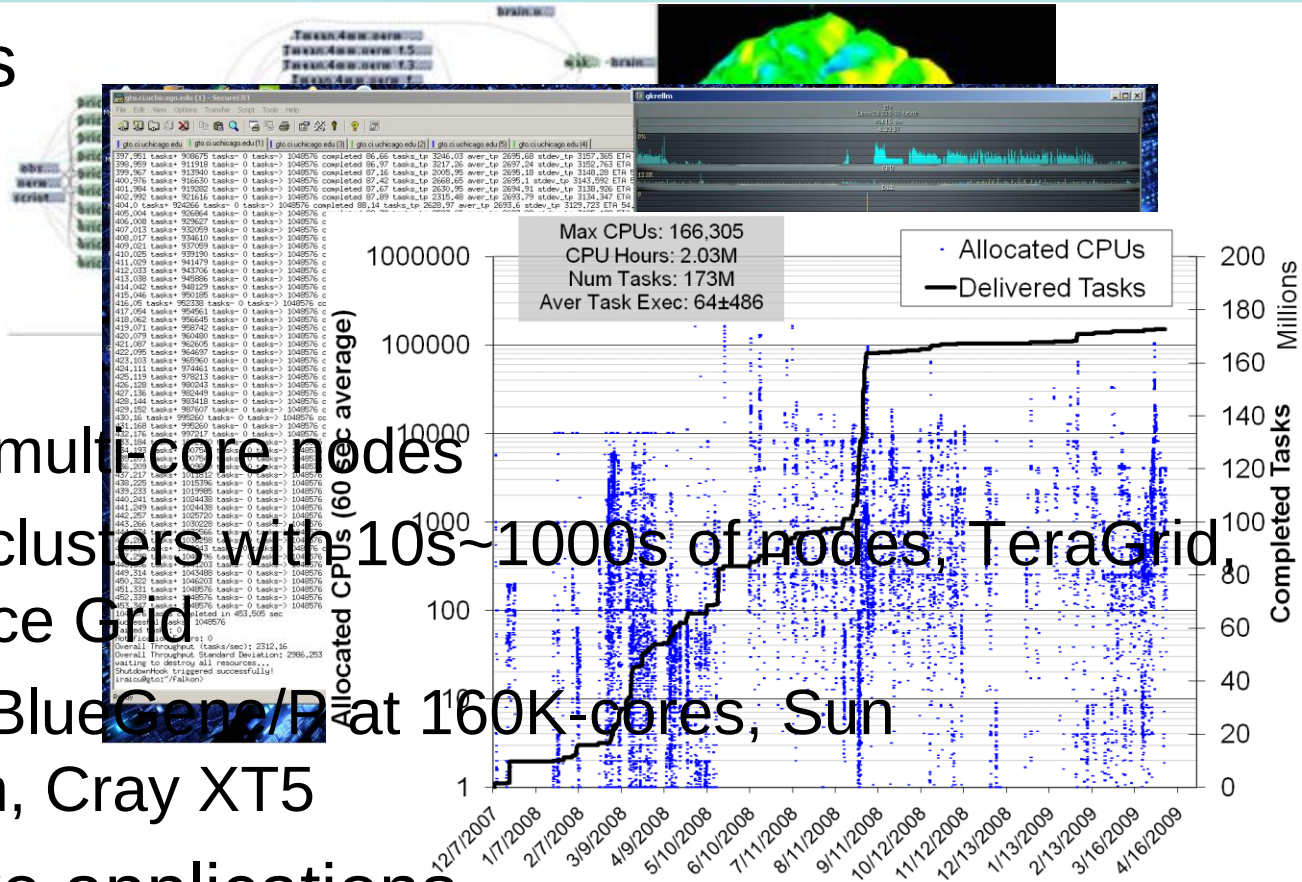


Outline

- Overview
- **Past Work**
- Future Work
  - Motivation
  - Proposal
- Work-in-Progress

Many-Task Computing Reality or Vaporware?

- Real systems
  - Swift
  - Falkon
- Scalability
  - **Gigascale**: multi-core nodes
  - **Terascale**: clusters with 10s~1000s of nodes, TeraGrid, Open Science Grid
  - **Petascale**: BlueGene/P at 160K-cores, Sun Constelation, Cray XT5
- Data intensive applications
  - Data diffusion
  - Collective data management



Many-Task Computing Applications

- Astronomy
- Astrophysics
- Economic Modeling
- Pharmaceutical Domain
- Chemistry
- Bioinformatics
- Neuroscience Domain
- Cognitive Neuroscience
- Data Analytics
- Data Mining
- Biometrics
- Molecular docking
- Uncertainty in economic models
- Structural equation modeling
- Posttranslational protein modification
- Climate modeling

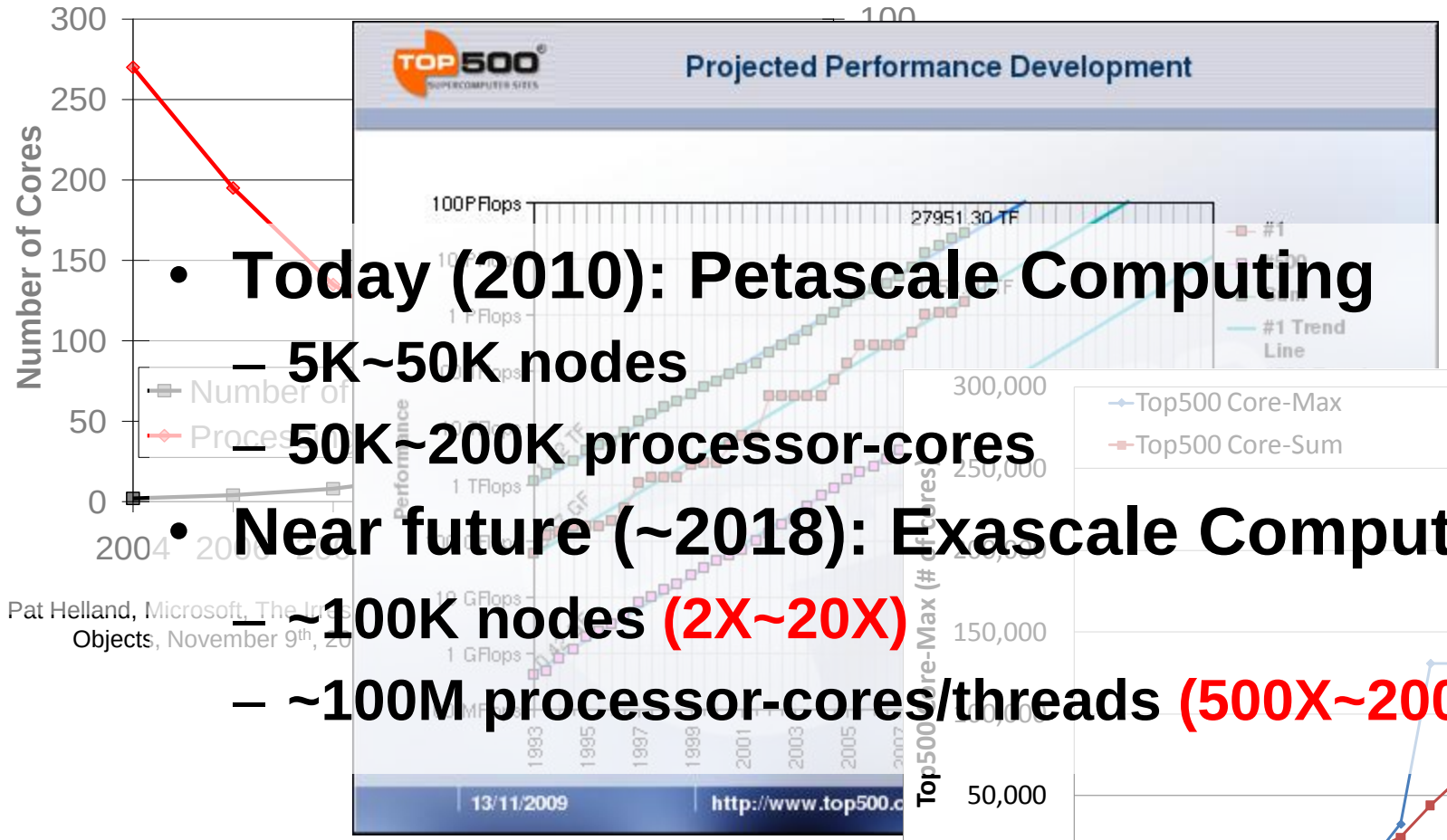
Many-Task Computing Building a community

- Publications
  - 30+ articles over 5 years, 800+ citations, 30+ collaborators
  - 2 dissertations (Zhao 2007, Raicu 2009)
- Building a community
  - 2008-2009: 102 abstracts, 78 papers, 3 venues
  - ACM MTAGS 2008, at SC08 (*6 papers*)
  - IEEE MTAGS 2009, at SC09 (*14 papers*)
  - IEEE TPDS, SI on MTC, 2010 (*~13 papers*)
- Other related activities
  - ACM ScienceCloud2010 at HPDC2010
  - Courses: Big Data, Data-Intensive Distributed Computing

Outline

- Overview
- Past Work
- **Future Work**
  - **Motivation**
  - Proposal
- Work-in-Progress

Projected Growth Trends



Pat Helland, Microsoft, The Irre
Objects, November 9th, 20

Top500 Projected Development,
http://www.top500.org/lists/2009

- **Today (2010): Petascale Computing**
  - **5K~50K nodes**
  - **50K~200K processor-cores**
- **Near future (~2018): Exascale Computing**
  - **~100K nodes (2X~20X)**
  - **~100M processor-cores/threads (500X~2000X)**

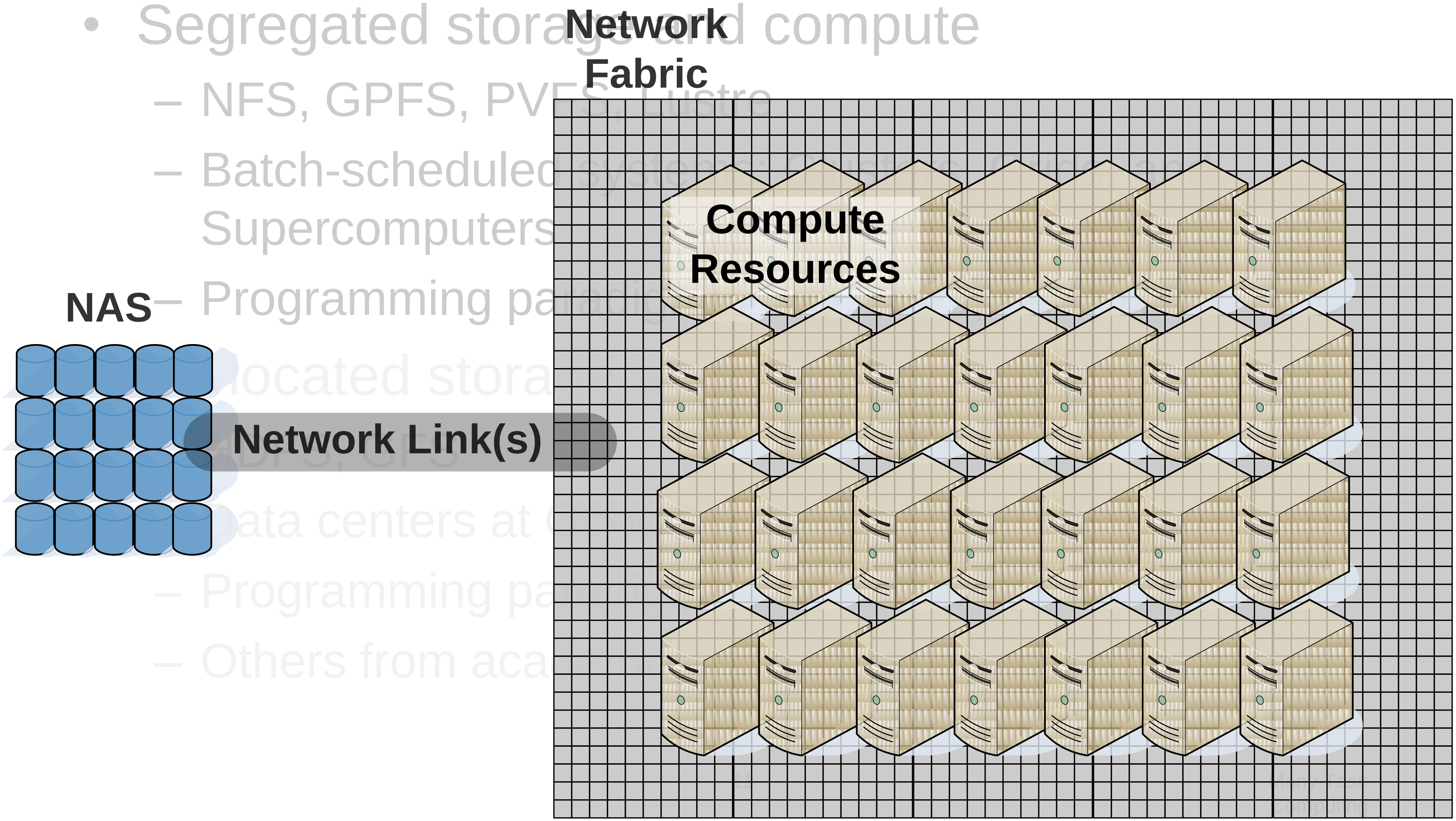
http://www.top500.org/

Storage Performance Trends

- Single Node Disk Performance
  - 2002 (2-cores): 70GB SCSI, 100MB/s (~50MB/core)
  - 2010 (8-cores)
    - 2TB SATA, 140MB/s (~18MB/core)
    - 256GB SSD, 260MB/s (~33MB/core)
    - 1TB SSD (RAID), 870MB/s (~109MB/core)
    - 10TB SATA (RAID), 1424MB/s (~178MB/core)
- Network Attached Storage
  - 2002 (2K-cores): BG/L, GPFS, 1GB/s (~0.5MB/core, 100X reduction)
  - 2010 (160K-cores): BG/P, GPFS, 65GB/s (~0.4MB/core, 438X reduction)
  - 2011 (1.2M-threads): Bluewaters needs ~480GB/s to sustain ~0.4MB/thread
  - 2018 (100M-threads): Exascale needs ~40TB/s to sustain ~0.4MB/thread

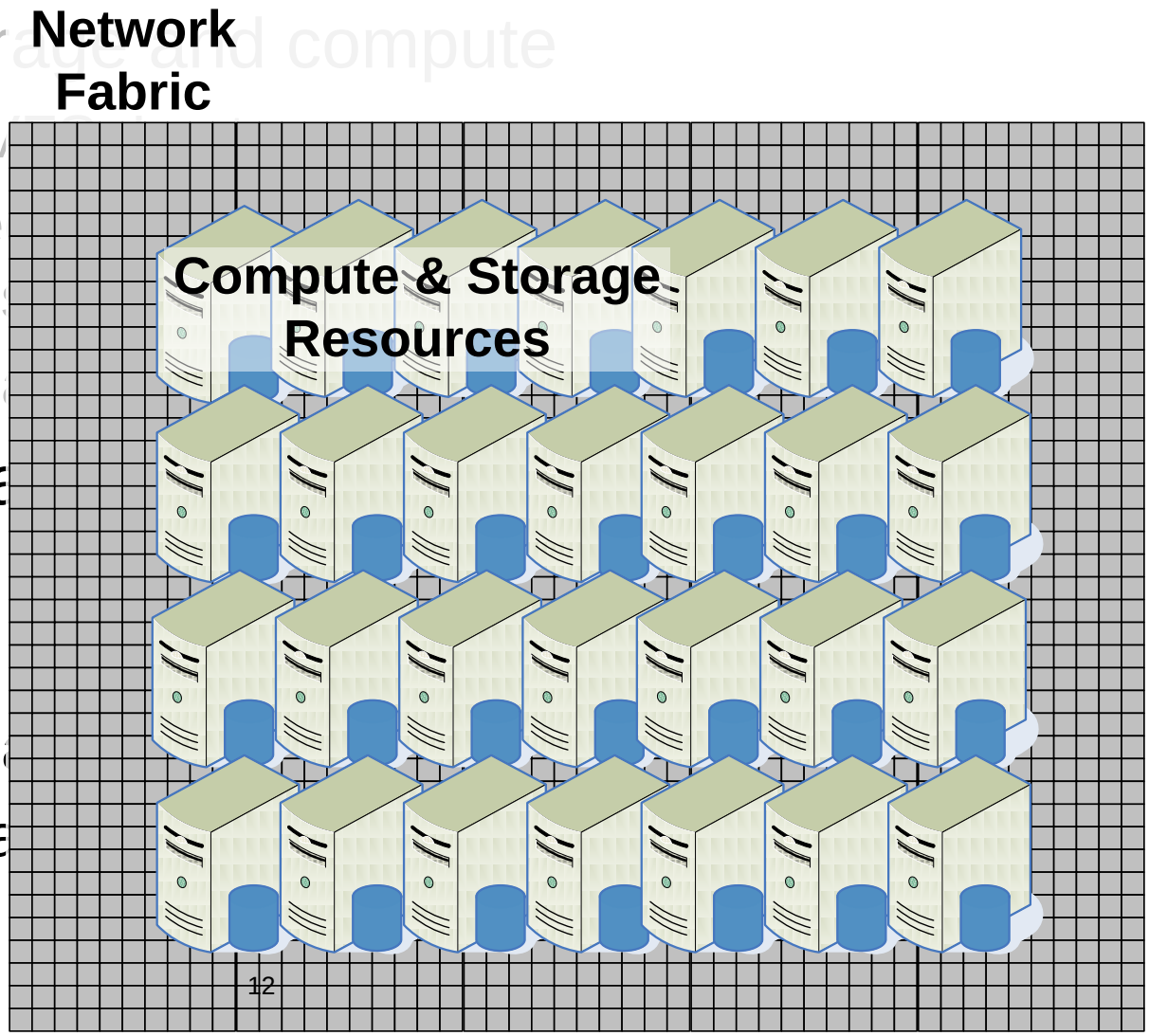
State of the Art Storage Systems: Parallel File Systems

- Segregated storage and compute
  - NFS, GPFS, PVFS, Lustre
  - Batch-scheduled systems: Clusters, Grids, and Supercomputers
  - Programming pa
- located stora
  - HDFS, GFS
- ata centers at
  - Programming pa
  - Others from aca

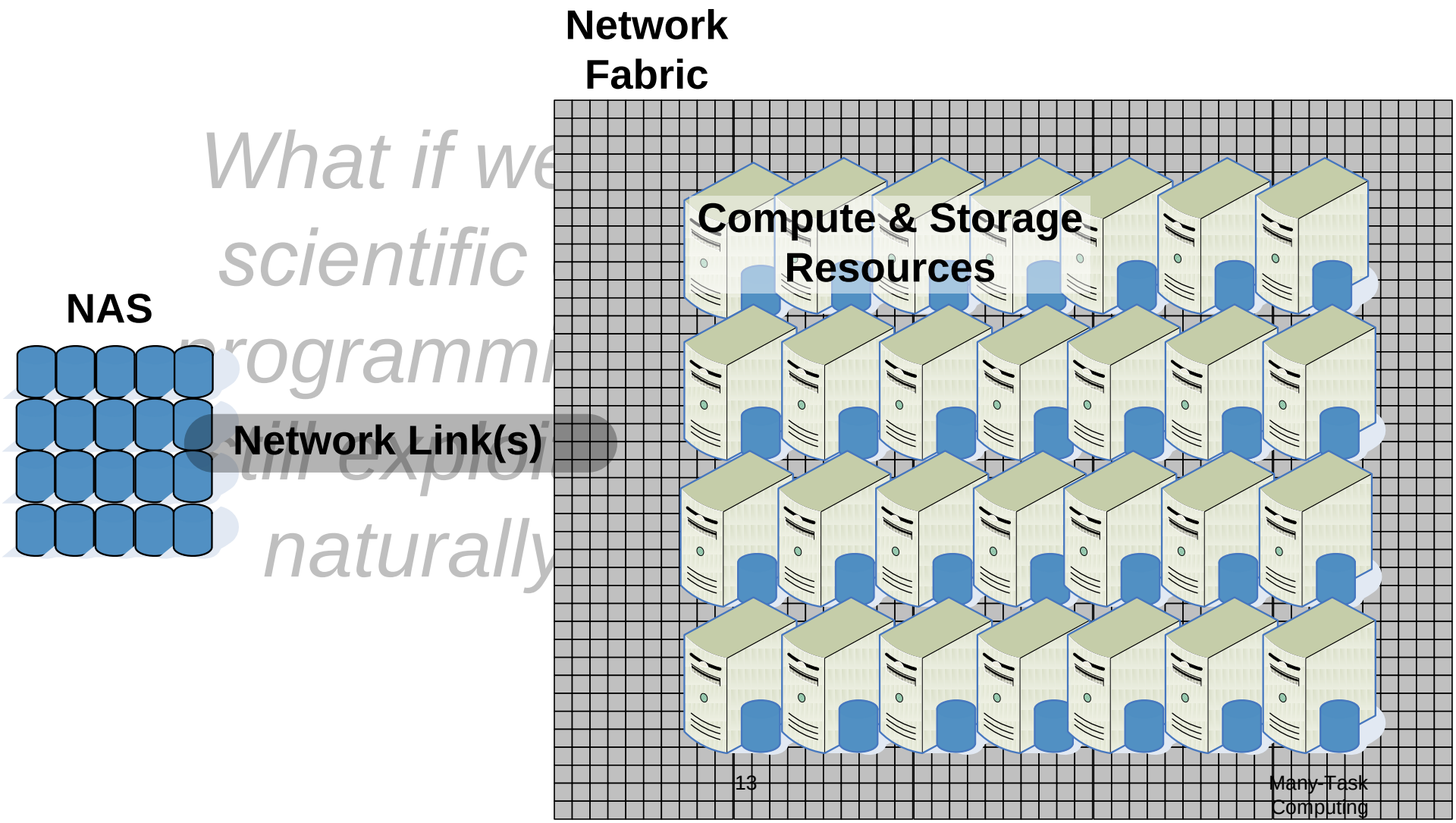


State of the Art Storage Systems: Distributed File Systems

- Segregated storage and compute
  - NFS, GPFS, PV
  - Batch-schedule
    Supercomputer
  - Programming p
- Co-located stora
  - HDFS, GFS
  - Data centers at
  - Programming p
  - Others from aca



Combine State of the Art Storage Systems



Why is all this important?

- MTTF is likely to decrease with system size
  - Loosely coupled asynchronous programming models (MTC as opposed to HPC) are more realistic to scale to exascales with high parallelism and concurrency
- Support for data intensive applications/operations
  - Fueled by more complex questions, larger datasets, and the many-core computing era
  - ***HPC***: OS booting, application loading, check-pointing
  - ***HTC***: Inter-process communication
  - ***MTC***: Metadata intensive workloads, inter-process communication

Outline

- Overview
- Past Work
- **Future Work**
  - Motivation
  - **Proposal**
- Work-in-Progress

Exascale Supercomputing Architecture

- Compute
  - 100K nodes, with ~1K threads/cores per node
  - ~10GF per thread/core
- Networking
  - N-dimensional torus
  - Meshes
- Storage
  - SANs with spinning disks will replace today's tape
  - SANs with SSDs might exist, replacing today's spinning disk SANs
  - SSDs will exist at every node

Some Challenges to Overcome at Exascale Computing

- Programming paradigms
  - HPC is dominated by MPI today
  - Will MPI scale another 4 orders of magnitude?
  - MTC has better scaling properties (due to its asynchronous nature)
- Network topology must be used in job management, data management, compilers, etc
- Storage systems will need to become more distributed to scale

Proposed Work Directions

- ***Decentralization is critical***
  - Computational resource management (e.g. LRMs)
  - Storage systems (e.g. parallel file systems)
- ***Data locality must be maximized, while preserving I/O interfaces***
  - 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

Proposed Work

- Develop theoretical and practical aspects of building efficient and scalable support for MTC
- Build a new distributed data-aware execution fabric that will support HPC, MTC, and HTC
  - Scale to at least millions of processors with 1-core granularity
  - Scale to petabytes of storage with billions of files
  - Verify through simulations scalability to a exascale levels (billions of processors, exabytes of storage)

Proposed Work (cont)

- ***Building on research results of Swift and Falkon***
- Library-based APIs to support a variety of languages
- Clients submit computational jobs into the execution fabric to any compute node
- The fabric will:
  - Guarantee jobs will execute at least once
  - Optimize data movement
  - Be elastic
  - Support job dependencies
  - Load-balance via work stealing

Proposed Work (cont)

- ***Building on research results of data-diffusion***
- Data will be automatically replicated
- Data access semantic
  - POSIX-like compliance for generality
  - Relaxed semantics to increase scalability
    - Eventual consistency on data modifications
    - Write-once read-many data access patterns
- Distributed metadata management
  - Employ structured distributed hash tables
  - Should scale logarithmically with system size
  - Can leverage network-aware topology overlays

Outline

- Overview
- Past Work
- Future Work
  - Motivation
  - Proposal
- **Work-in-Progress**

Work-in-Progress

- Distributed execution fabric
  - In the design phase
    - Linux, library-based APIs, C/C++, ysSSL, work stealing for load-balancing, support for simple workflow specification
- Distributed file system
  - Functional prototype
    - Implementation in C/C++
    - Supports open, read, write, close, and stat (FUSE support soon)
    - Distributed meta-data management via a structured DHT (Chord)
- Prototype ETA: ~July 2010

More Information

- More information: http://www.eecs.northwestern.edu/~iraicu/
- Related Projects:
  - Falkon: http://dev.globus.org/wiki/Incubator/Falkon
  - Swift: http://www.ci.uchicago.edu/swift/index.php
- People contributing ideas, slides, source code, applications, results, etc
  - Ian Foster, Alex Szalay, Rick Stevens, Mike Wilde, Jim Gray, Catalin Dumitrescu, Yong Zhao, Zhao Zhang, Gabriela Turcu, Ben Clifford, Mihael Hategan, Allan Espinosa, Kamil Iskra, Pete Beckman, Philip Little, Christopher Moretti, Amitabh Chaudhary, Douglas Thain, Quan Pham, Atilla Balkir, Jing Tie, Veronika Nefedova, Sarah Kenny, Gregor von Laszewski, Tiberiu Stef-Praun, Julian Bunn, Andrew Binkowski , Glen Hocky, Donald Hanson, Matthew Cohoon, Fangfang Xia, Mike Kubal, Alok Choudhary…
- Funding:
  - **NASA**: Ames Research Center, Graduate Student Research Program
  - **DOE**: Office of Advanced Scientific Computing Research
  - **NSF**: TeragGrid and Computing Research Innovation Fellow Program