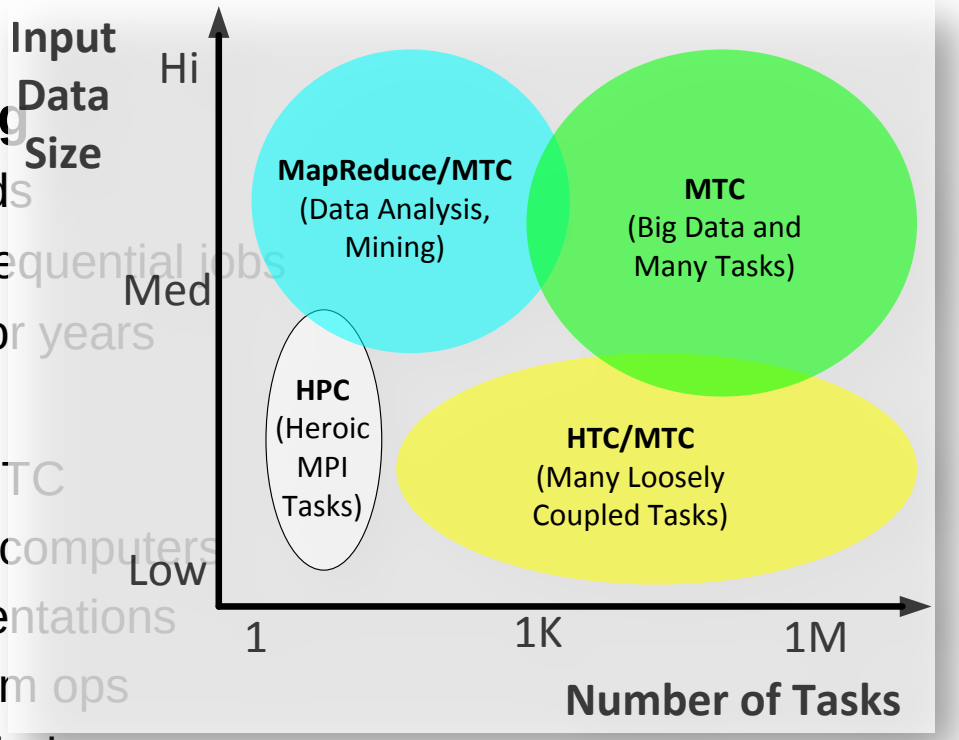
Exascale Many-Task Computing with a Billion Processors

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

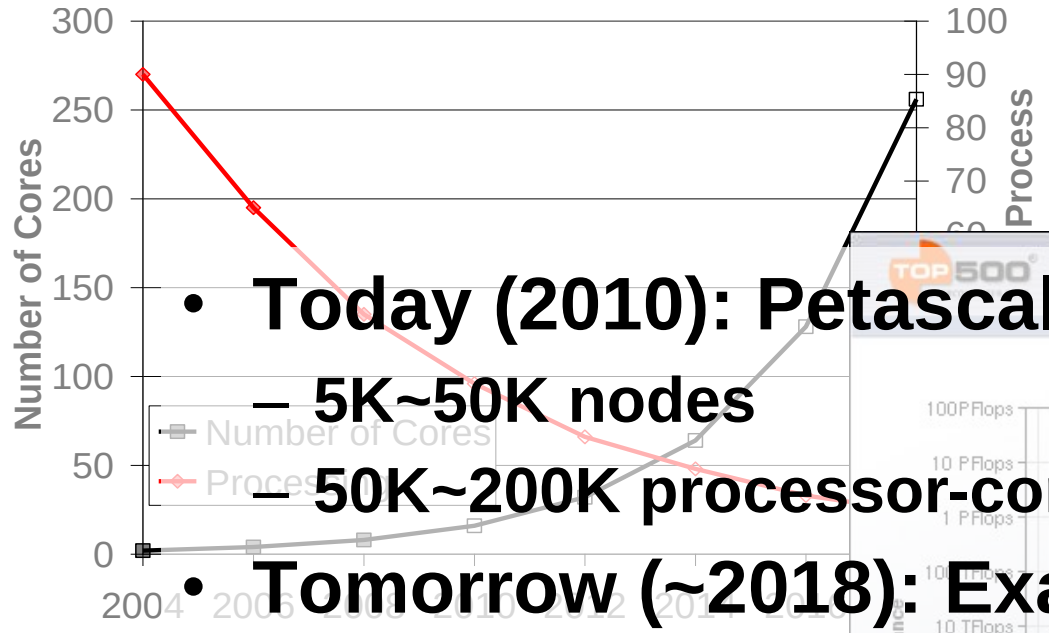
EECS 395 / EECS 495
Hot Topics in Distributed Systems: Data-Intensive Computing
March 11th, 2010

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

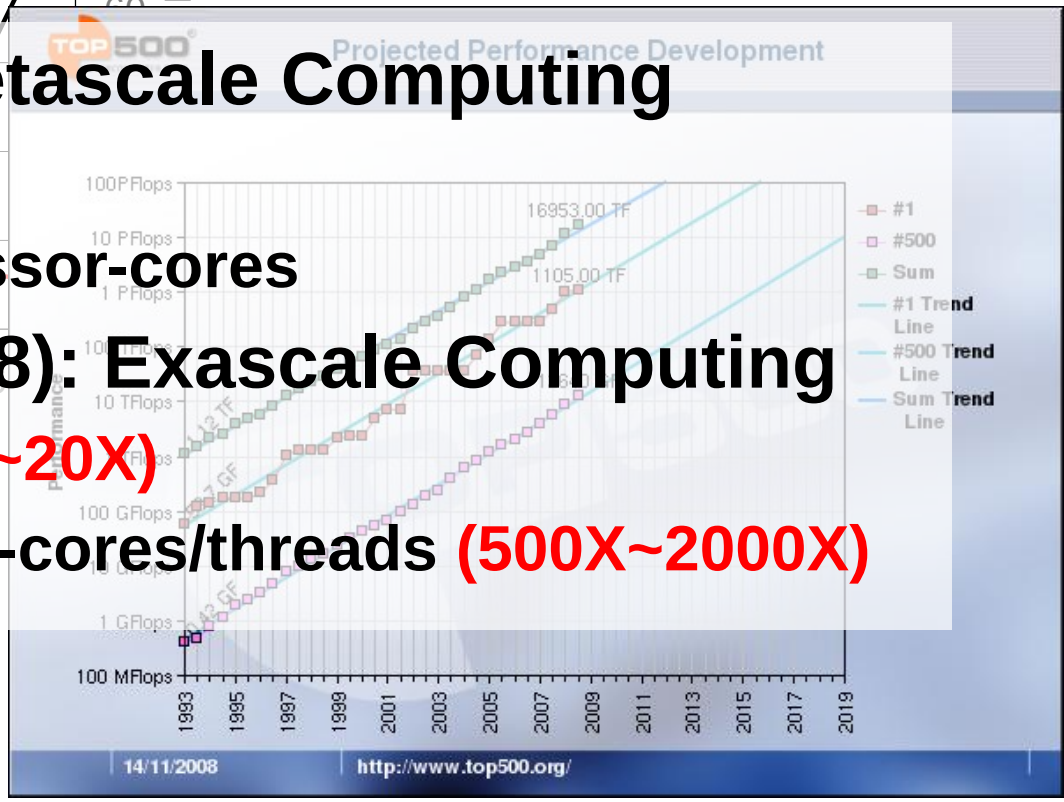


Projected Growth Trends



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

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



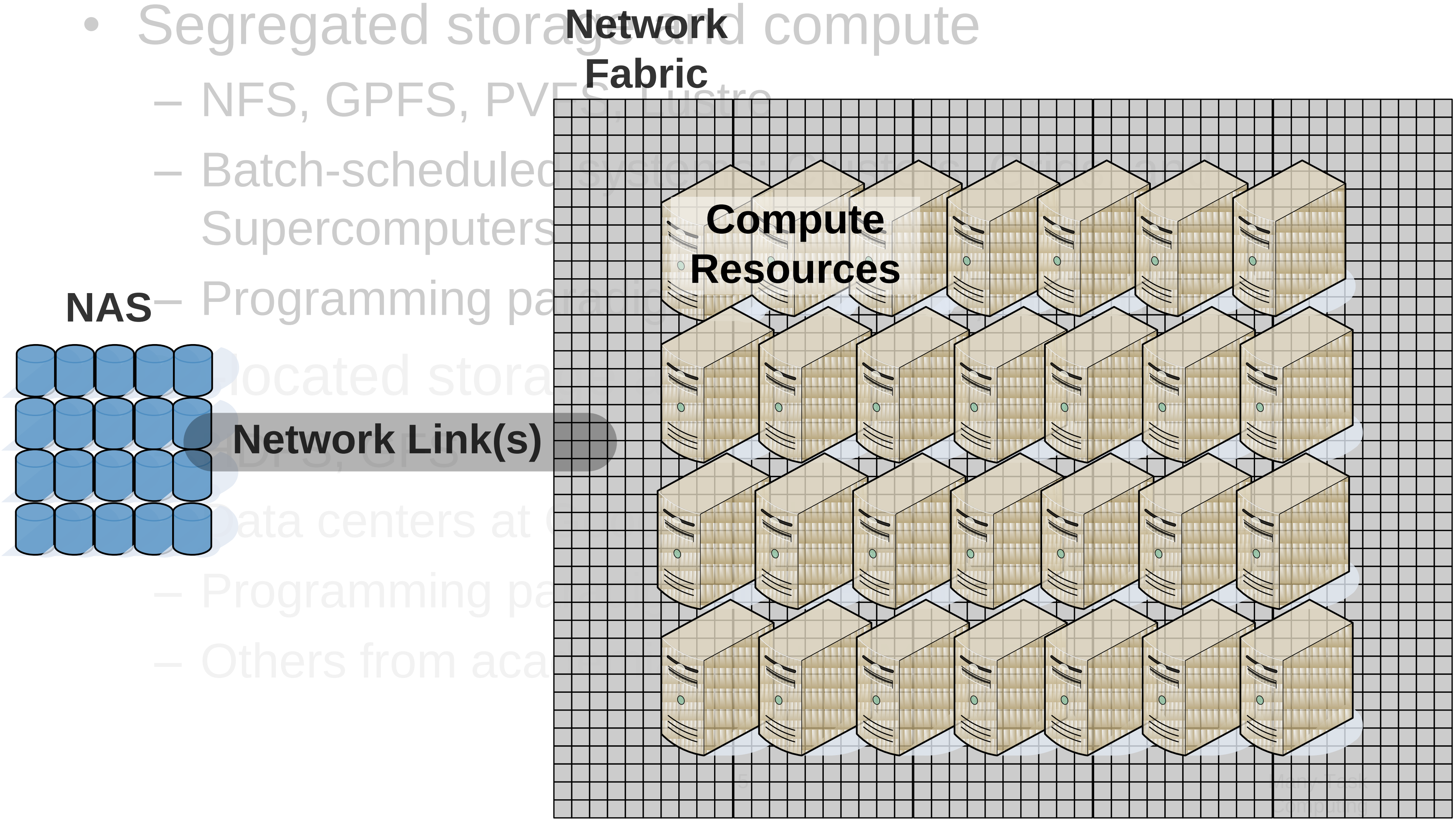
Top500 Projected Development,
http://www.top500.org/lists/2008/11/performance_development

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

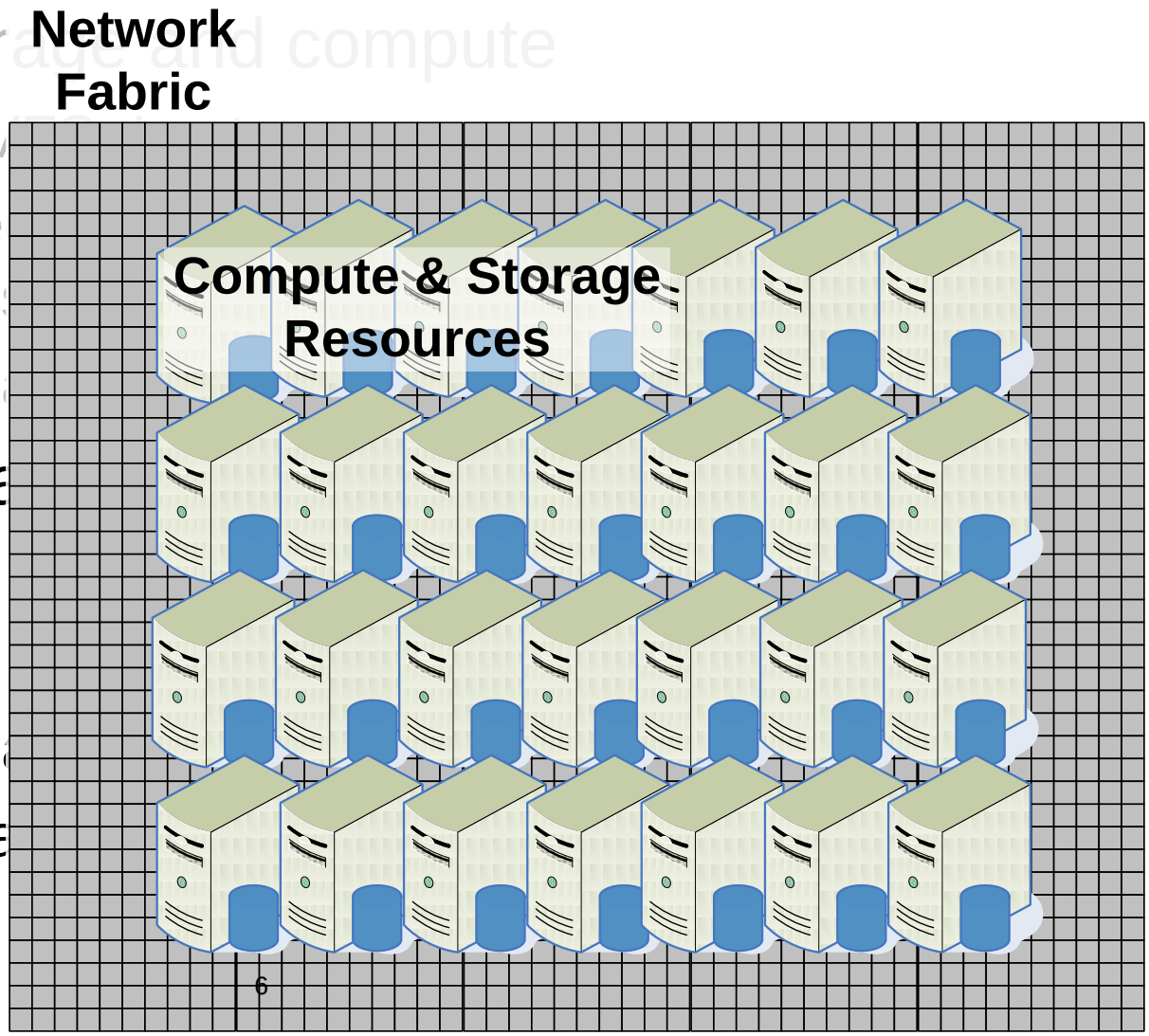
State of the Art Storage Systems: Parallel File Systems

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

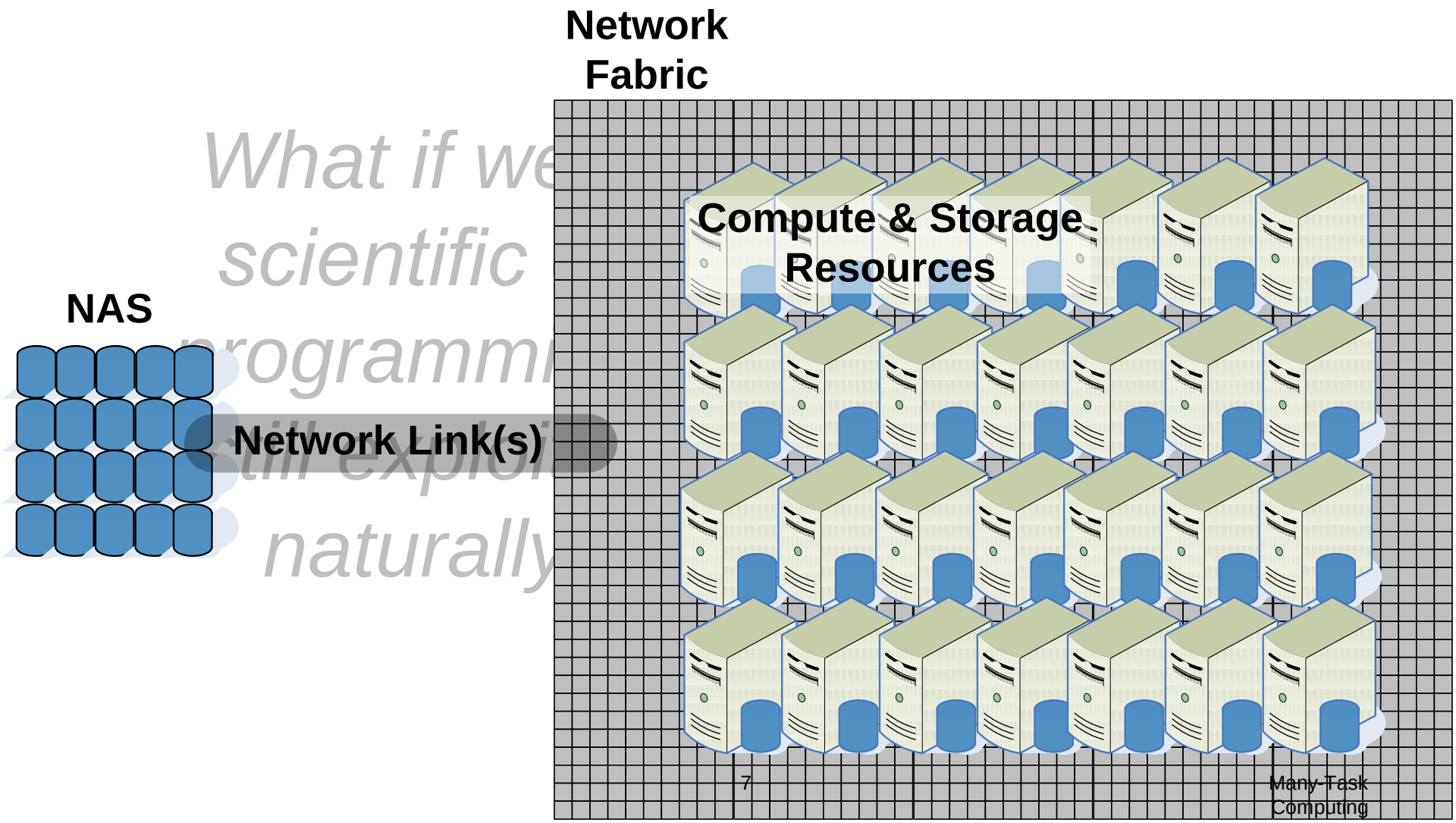


State of the Art Storage Systems: Distributed File Systems

- 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

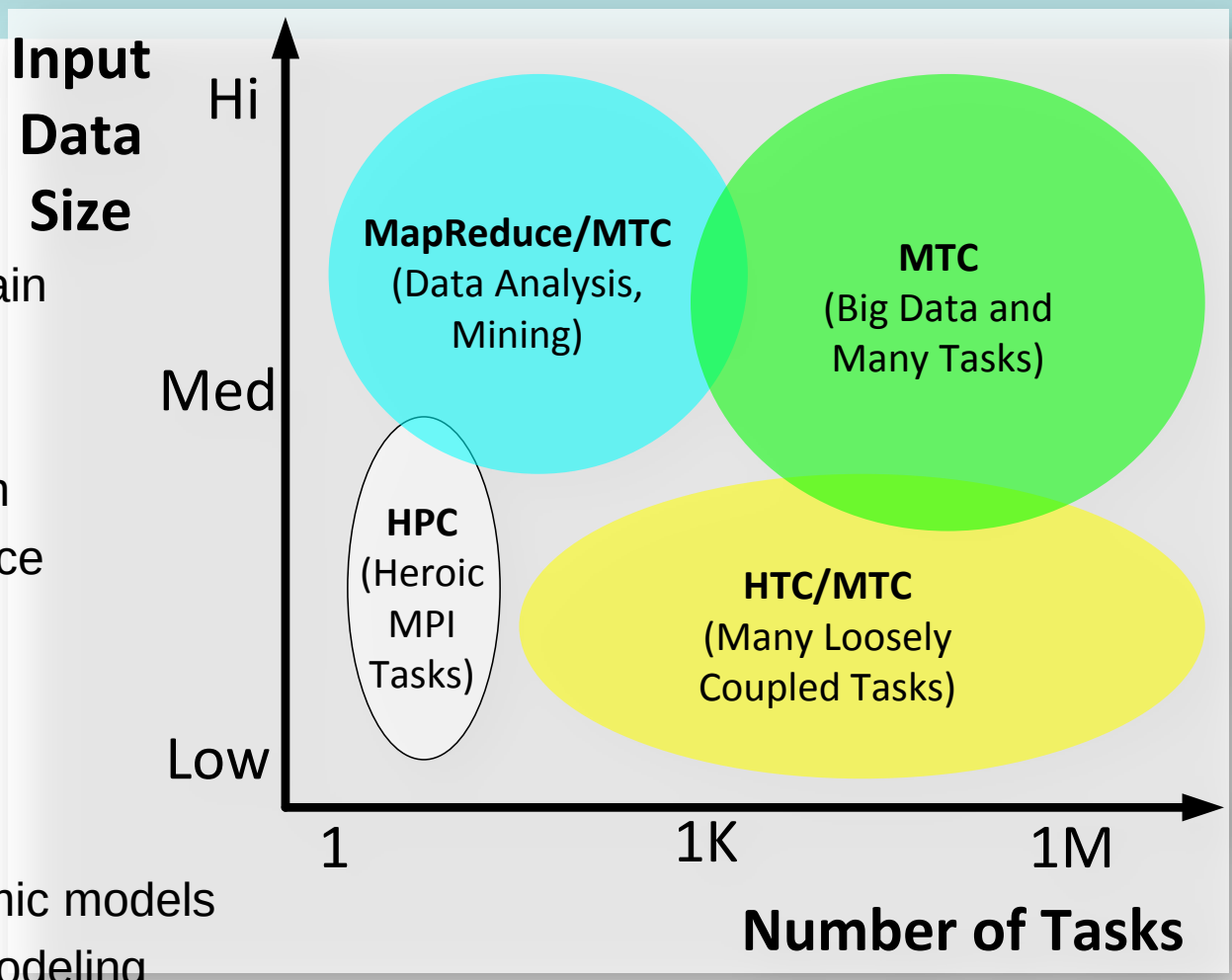


Why is all this important?

- In general
  - Support for data intensive applications
- HPC
  - OS booting
  - Application loading
  - Check-pointing
- HTC
  - Inter-process communication
- MTC
  - Metadata operations
  - Inter-process communication

MTC 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



Prior Work

- Falkon
  - Centralized
    - Scales to O(10K cores), O(1M running tasks)
  - Naïve decentralization
    - Scales to O(160K cores), O(1M running tasks)
    - Issues with load balancing
- Data Diffusion
  - Centralized in-memory metadata and scheduling
  - Scales to O(4K cores), 20GB/s, TB of data
- Swift
  - Centralized, in memory
  - Scales to O(16K cores) and O(500K task graphs)

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 and petabytes of storage
  - Verify through simulations scalability to a billion processors
- Support interactive HPC applications

Proposed Work (cont)

- 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
  - Employ work stealing for low cost scalable load balancing

Proposed Work (cont)

- Data will be automatically replicated
- Data access semantic
  - POSIX 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
  - Can scale logarithmically with system size
  - Can create network topology aware overlays

Questions