GeoGrid Project and Experiences with Hadoop

Gong Zhang and Ling Liu

Distributed Data Intensive Systems Lab (DiSL)
Center for Experimental Computer Systems Research (CERCS)
Georgia Institute of Technology

Acknowledgement: This work is sponsored by a grant from NSF CyberTrust program, an IBM faculty award, and an IBM SUR grant. The first author did internship at IBM Almaden on comparison of HDFS with GPFS.

Internet Usage and Content:

Demand for new generation of System Software Technology

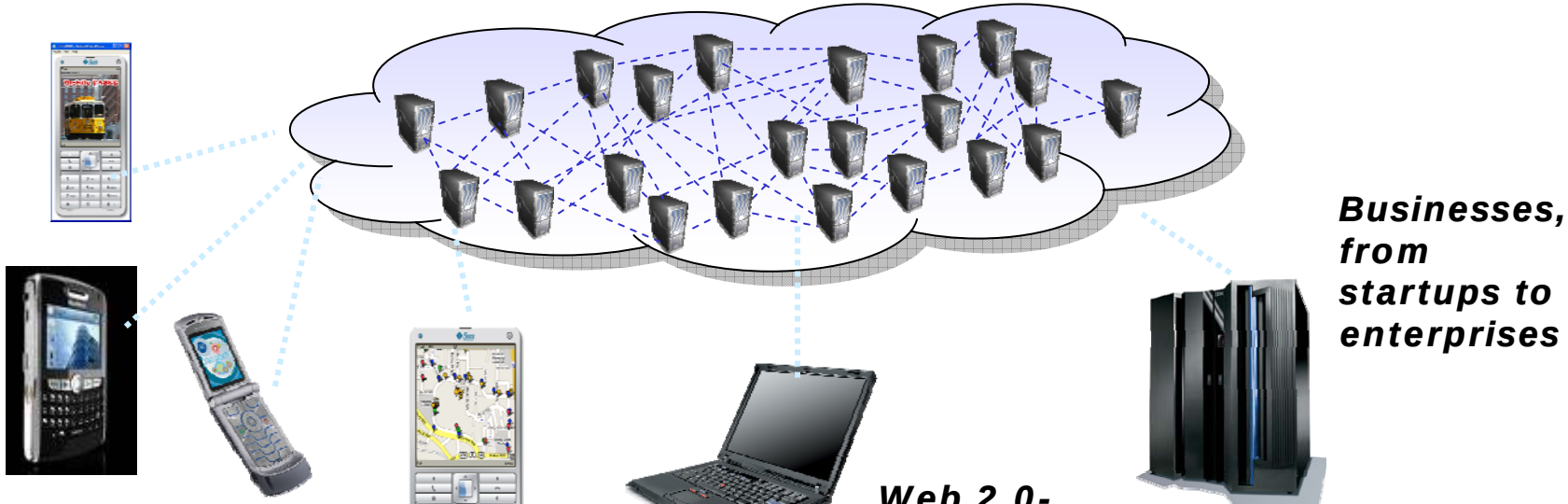
- Explosion and continuous growth fueled by global computing, mobile device access, and user-generated content

- Mass Distribution:
  - Device numbers exploding (cell phones +1B/yr)
  - Edge computing resources exceed those in core

- Mass Centralization:
  - Yield management, optimization, & data analysis
  - Data is the asset
  - Move computation closer to data

Mobile Internet & Cloud Computing

- Vision
  - An emerging computing paradigm where data and services reside in massively scalable data centers and can be ubiquitously accessed from any connected devices over the internet.
  - From one-server, one application model to browser based parallel application model, expanding the Internet to become the compute platform of the future



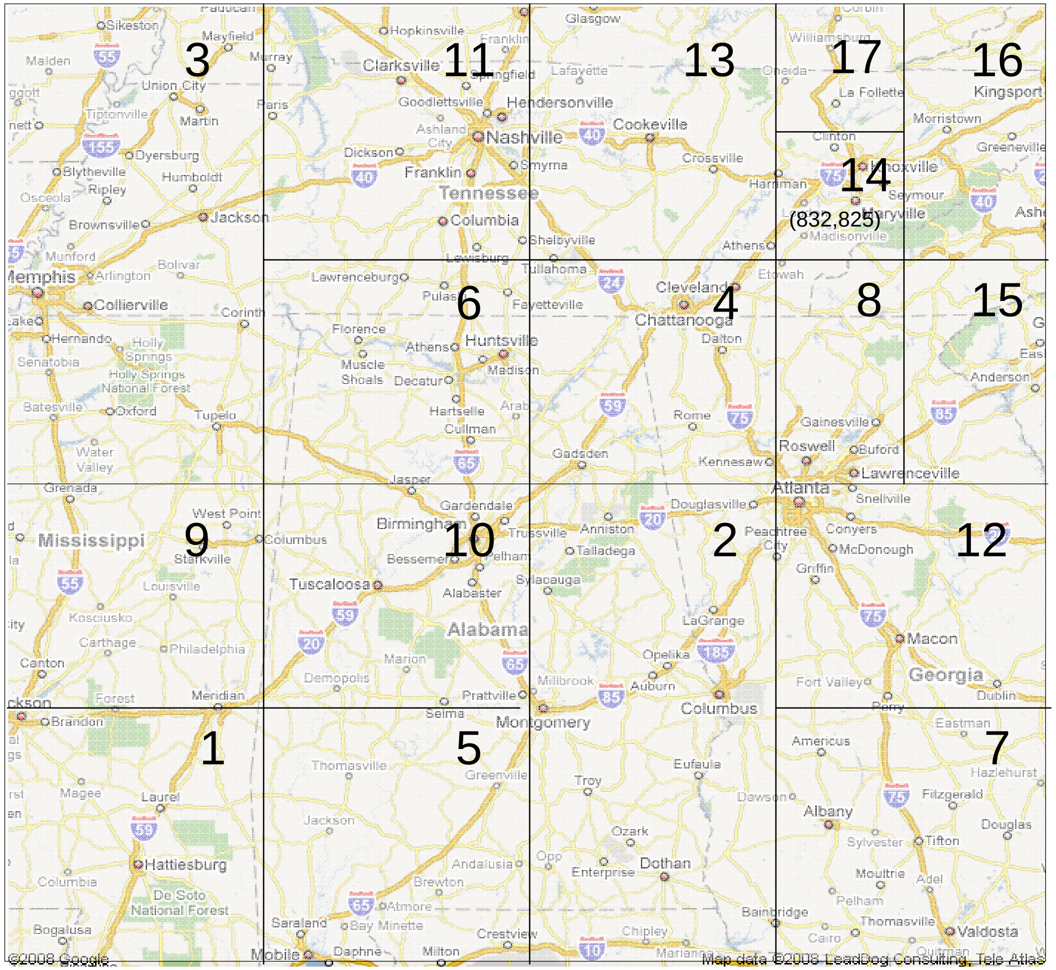
4+ billion phones *by 2010 [Source: Nokia]*

Web 2.0-enabled PCs, TVs, etc.

Businesses, from startups to enterprises

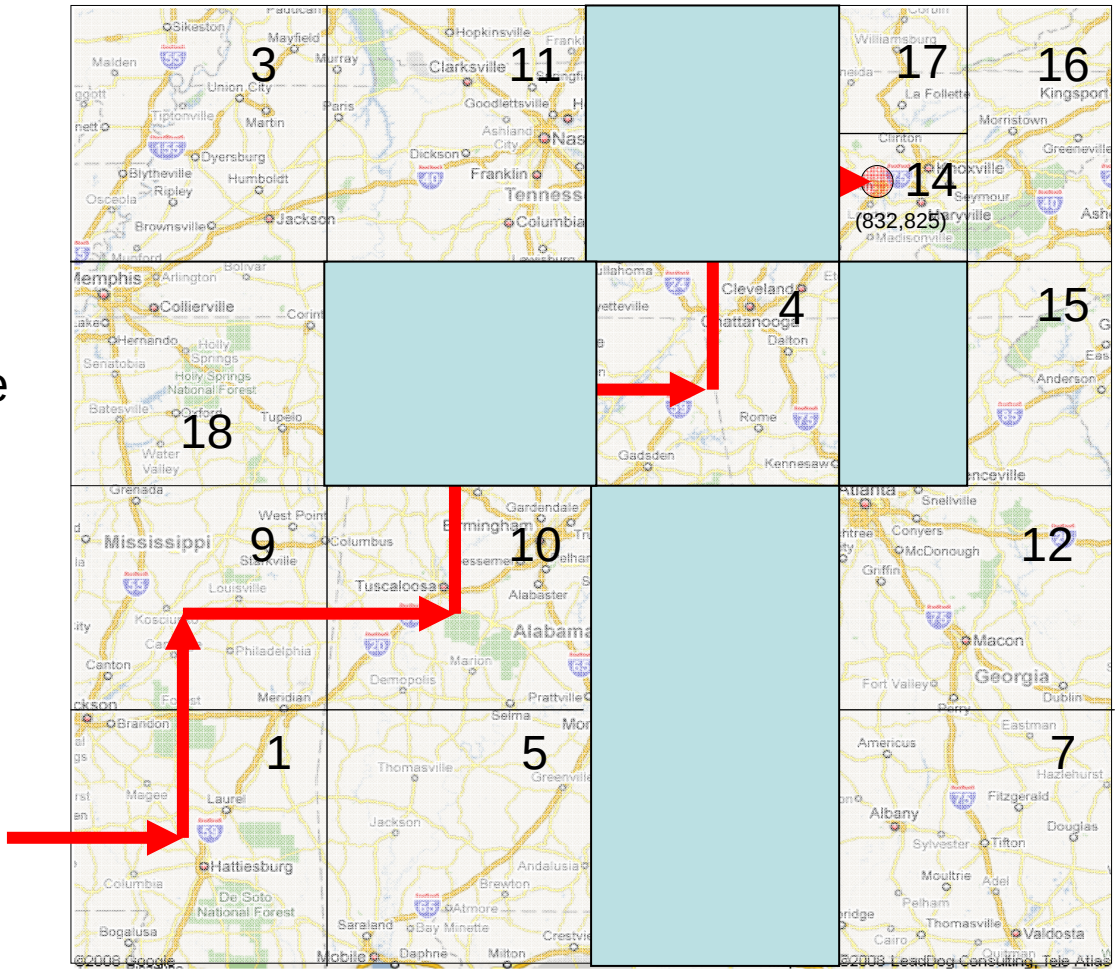
GeoGrid Design and Service Network Management

- A decentralized and geographical location aware overlay network for scalable and reliable delivery of location based service.
- A 2-d rectangular geographical area is divided into a number of rectangular regions.
- Each region is “assigned” to an owner node in the overlay network to process all the service requests mapped to this region
- Each node is a common user PC and can join and leave the network at any time.
- Mobile users connects mobile devices to the GeoGrid node
- Service request message

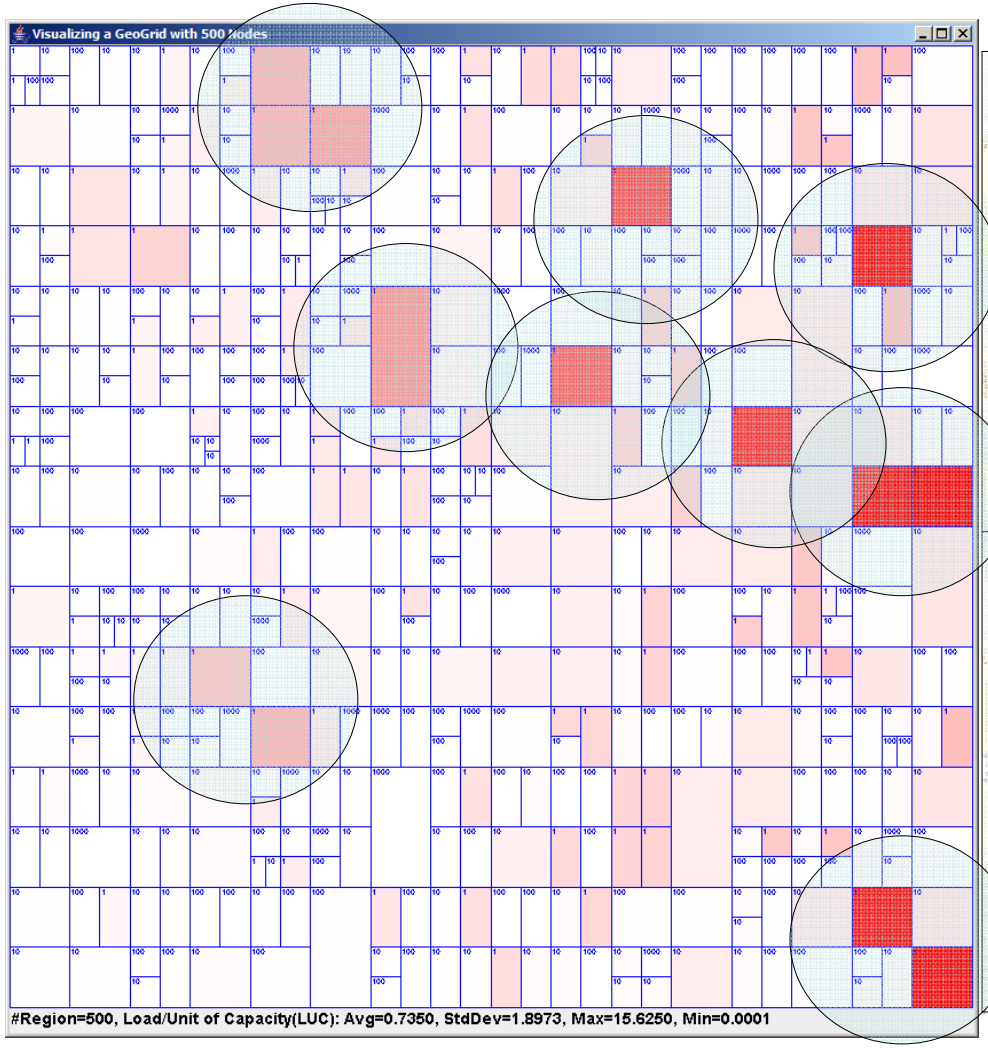


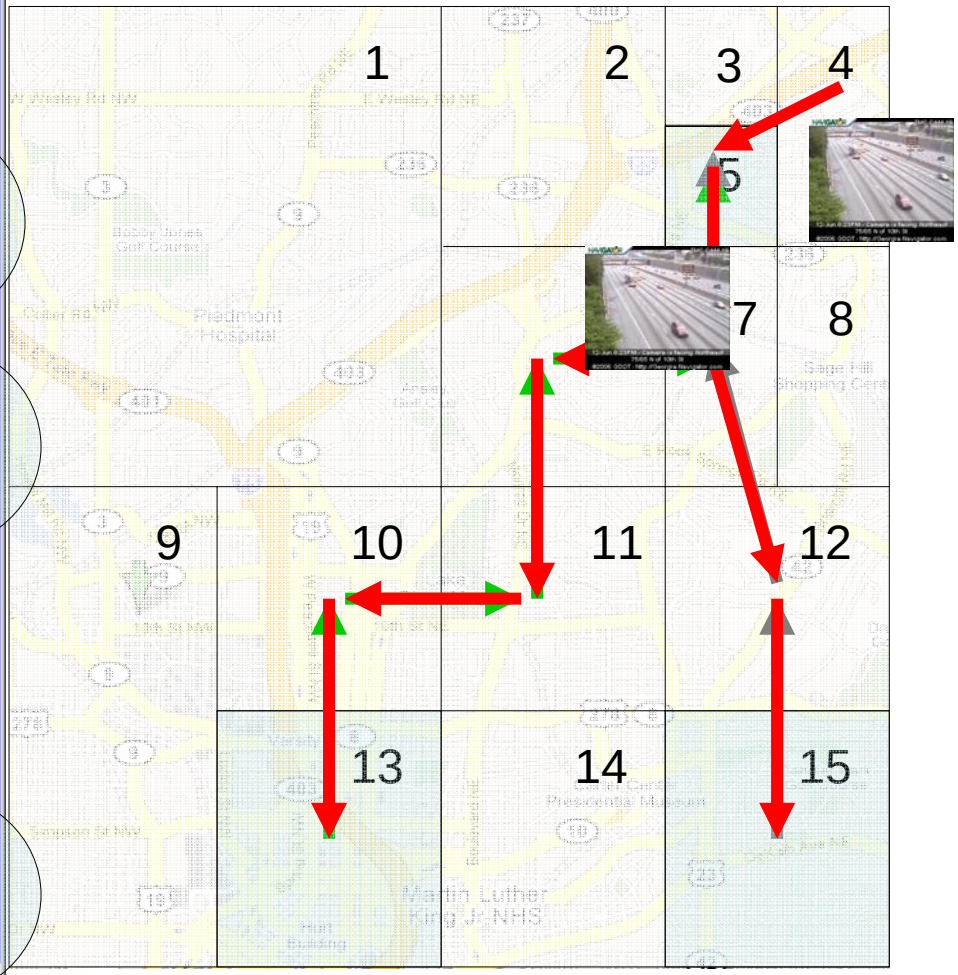
GeoGrid Design and Service Network Management

- Each node maintains a neighbor List
- Greedy forwarding routing
- Node joining
  - New node splits the region with the existing node
- Node Departure
  - a neighbor node takes over by merging two rectangular regions



GeoGrid: Scaling LBSs through Runtime Adaptation, Replication, and Load Balance





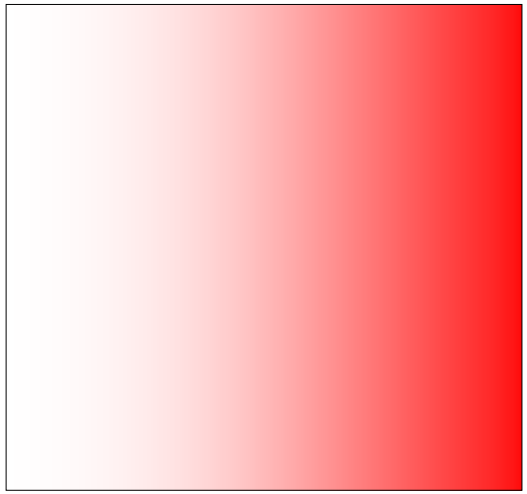
Basic GeoGrid Network with 500 nodes

GeoGrid Research:

Enhancing Reliability and Scalability of Pervasive Computing Applications

- Reliability/Fault Tolerance
  - Node failure
  - Network partition failure
- Scalability/Performance
  - CPU, Disk I/O, Network Bandwidth, Virtualization
  - Energy Efficiency (Mobile clients)
- Availability
  - 24 by 7 uptime
  - Distributed and Parallel Computing Infrastructure
  - Virtualization
- Security/Trust/
  - Pervasive security
  - Cross-Layer Trust Management

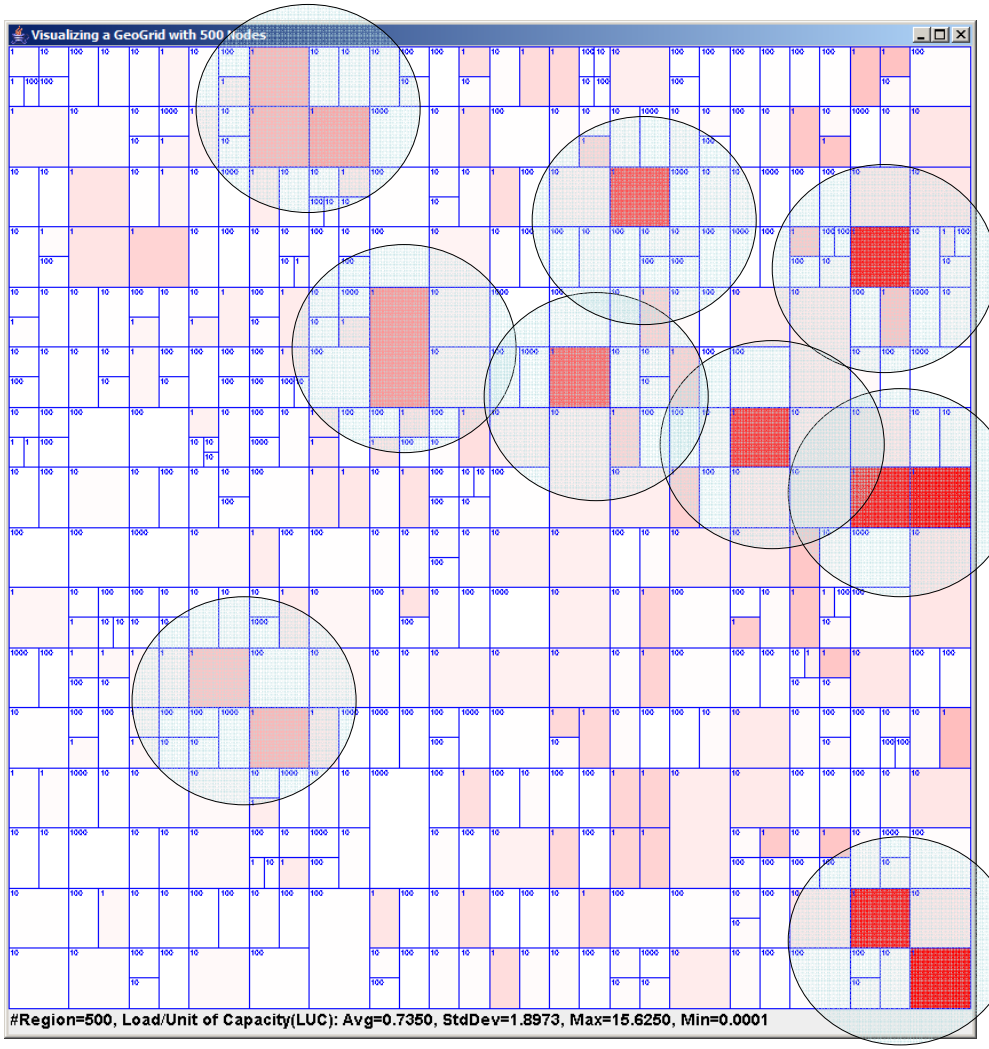
Highly skewed node workload distribution



0.0 ⟶ 1.0

Workload distribution

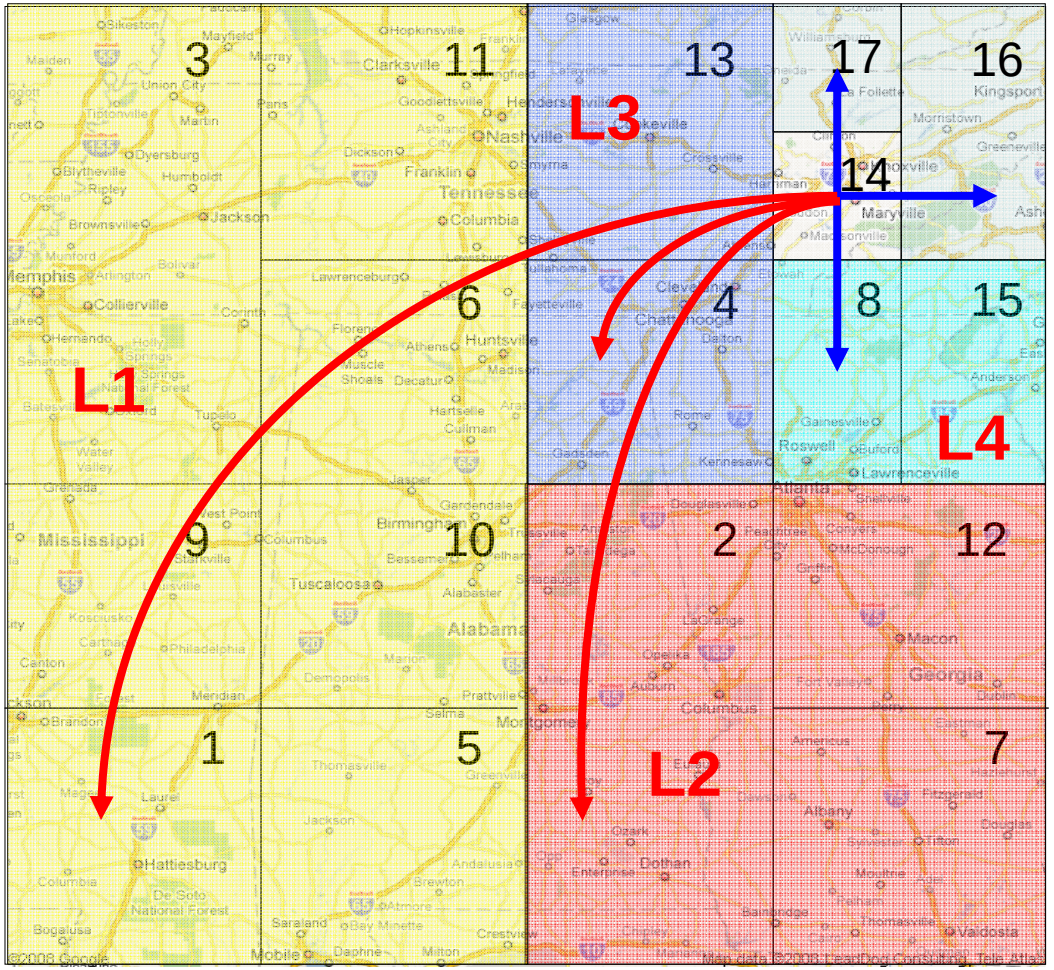
Capacity level	Percentage of peers
1x	20%
10x	45%
100x	30%
1000x	4.9%
10000x	0.1%



Reliable GeoGrid Service Network Load Visualization

Replication and Replica based Load Balancing

- Three Basic Components
  - Service Owner and Executor model
  - Executor selection from neighbor + shortcut replica
  - Executor selection criteria take into account multiple performance factors
    - Load per node – workload index
    - Cache factor
    - Proximity factor



Reducing the deadly failures

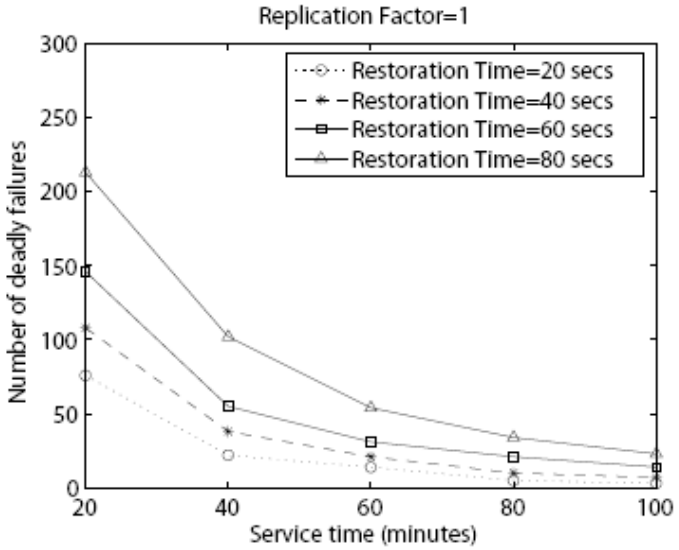


Figure 1. Deadly failures, $rf = 1$

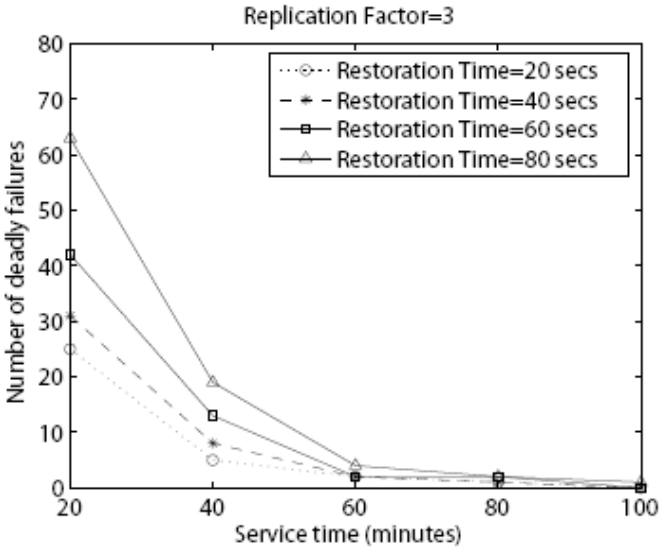
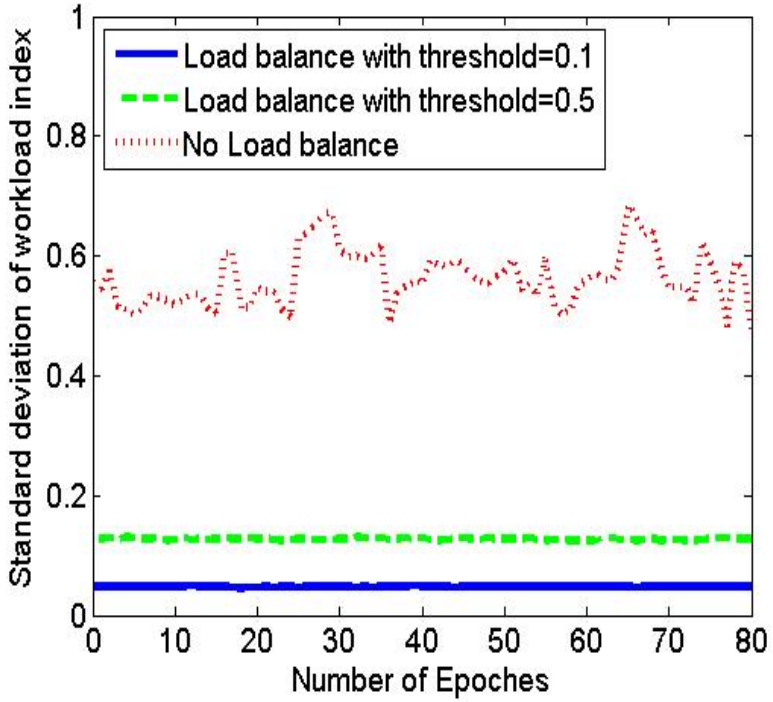
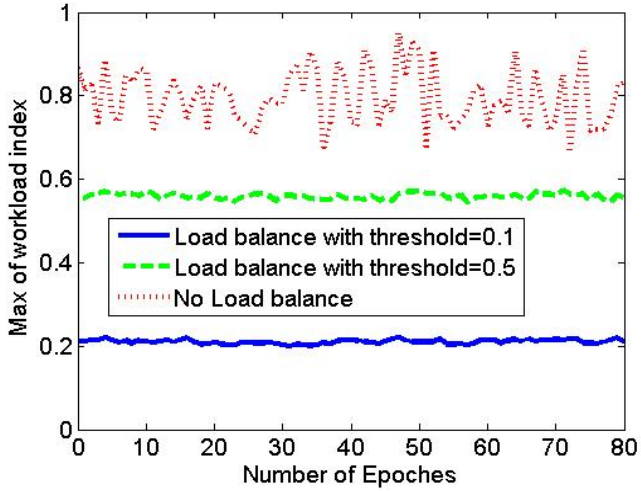


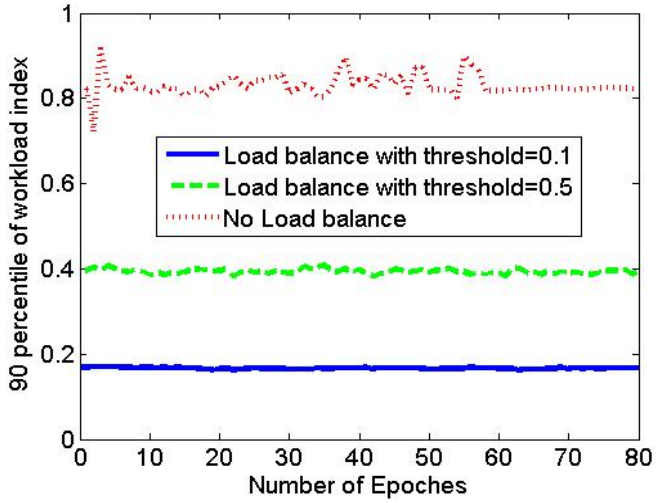
Figure 2. Deadly failures, $rf = 3$

Replication scheme helps to reduce the deadly failures

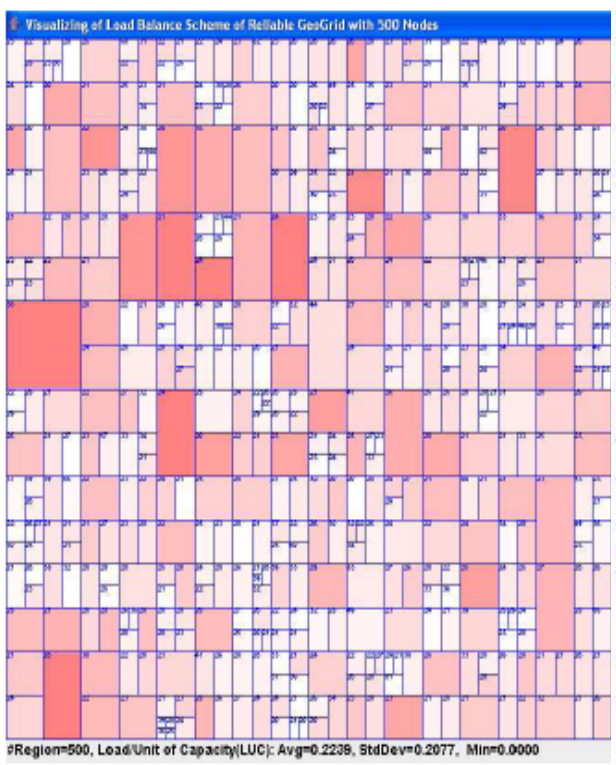
Hot spot diameter on workload index adaptation



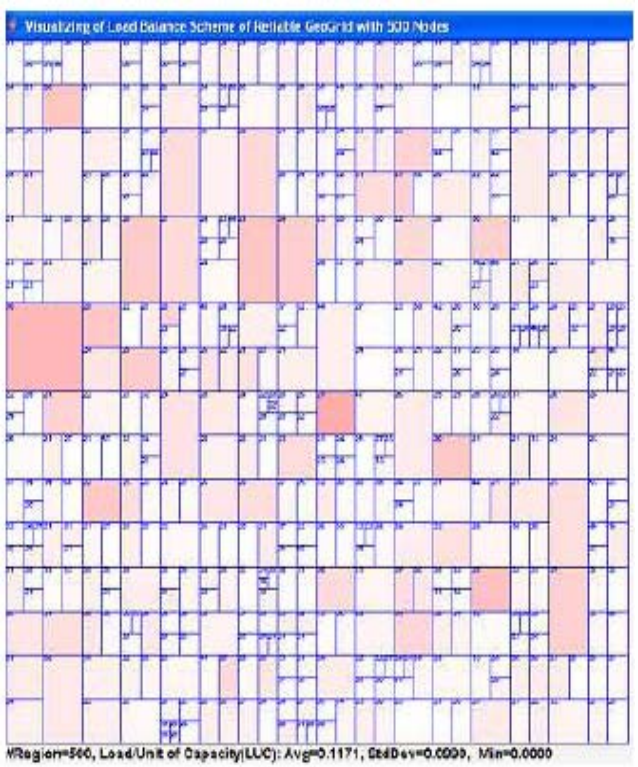




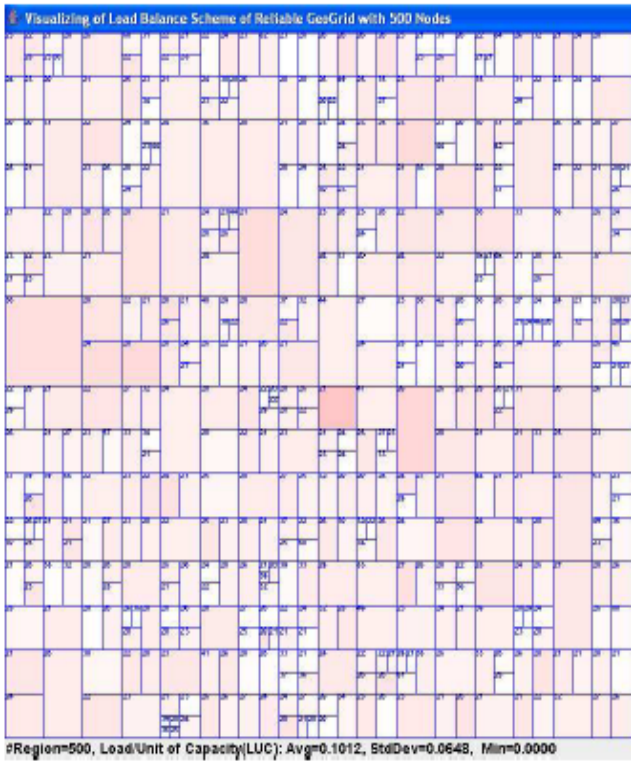
Workload Visualization



Workload distribution without load balance at time epoch 95



Workload distribution with load balance at time epoch 105



Workload distribution with load balance at time epoch 110

500 nodes, 39871 regular location query event, 35910 hot spot location query event

Experiences with Hadoop

Based on summer intern project at IBM Almaden Storage Systems Division

Hadoop: Designed for Map-reduce Applications

- Hadoop: Framework for running applications on large scale commodity hardware
  - Core: HDFS as Storage, Map-reduce as Processing programming model
  - Ship computation to data
  - Scalable, reliable, ease of use
  - Tolerates node/disk failures without impact on applications
  - Large scale data: gigabytes to terabytes of single file
  - Large scale distributed systems: thousands of nodes

- Why is Hadoop so popular?
  - The first open source distributed file system for Clouding Computing
  - The designated storage management support Map-reduce applications.

Distributed File System

- **Single Namespace for entire cluster**
- **Data Coherency**
  - Write-once-read-many access model
  - Client can only append to existing files
- **Files are broken up into blocks**
  - Typically 64 MB block size
  - Each block replicated on multiple DataNodes
- **Intelligent Client**
  - Client can find location of blocks
  - Client accesses data directly from DataNode

Comparing Hadoop with IBM GPFS

- Motivation
  - Both Hadoop and GPFS are designed as a framework for running applications on large scale commodity hardware (thousands of nodes)
  - Understanding the key properties for distributed file system for map-reduce applications

- HDFS as Storage + Map-reduce as Processing programming model
  - Ship computation to data
  - Scalable, reliable, ease of use
  - Tolerates node/disk failures without impact on applications
  - Large scale data: gigabytes to terabytes of single file
  - Large scale distributed systems: thousands of nodes

GPFS Basics

- Provides High-performance I/O
  - Divides files into blocks and stripes the blocks across disks (on multiple storage devices)
    - Reads/Writes the blocks in parallel
    - Tuneable block sizes (depends on your data)
  - Block-level locking mechanism
    - Multiple applications can access the same file concurrently
    - “multiple editors can work on different parts of a single file simultaneously. This eliminates the additional storage, merging and management overhead typically required to maintain multiple copies”
  - Client-side data-caching
    - Where is data cached?

Comparing Features of HDFS and GPFS

HDFS

- Simple coherency model: Write-once-read-many
- Splitting file into uniform sized blocks and distributed across cluster nodes (64MB)
- No locking
- Block replication to handle hardware failure
- Master-Slave architecture
- Block location exposed to apps

GPFS

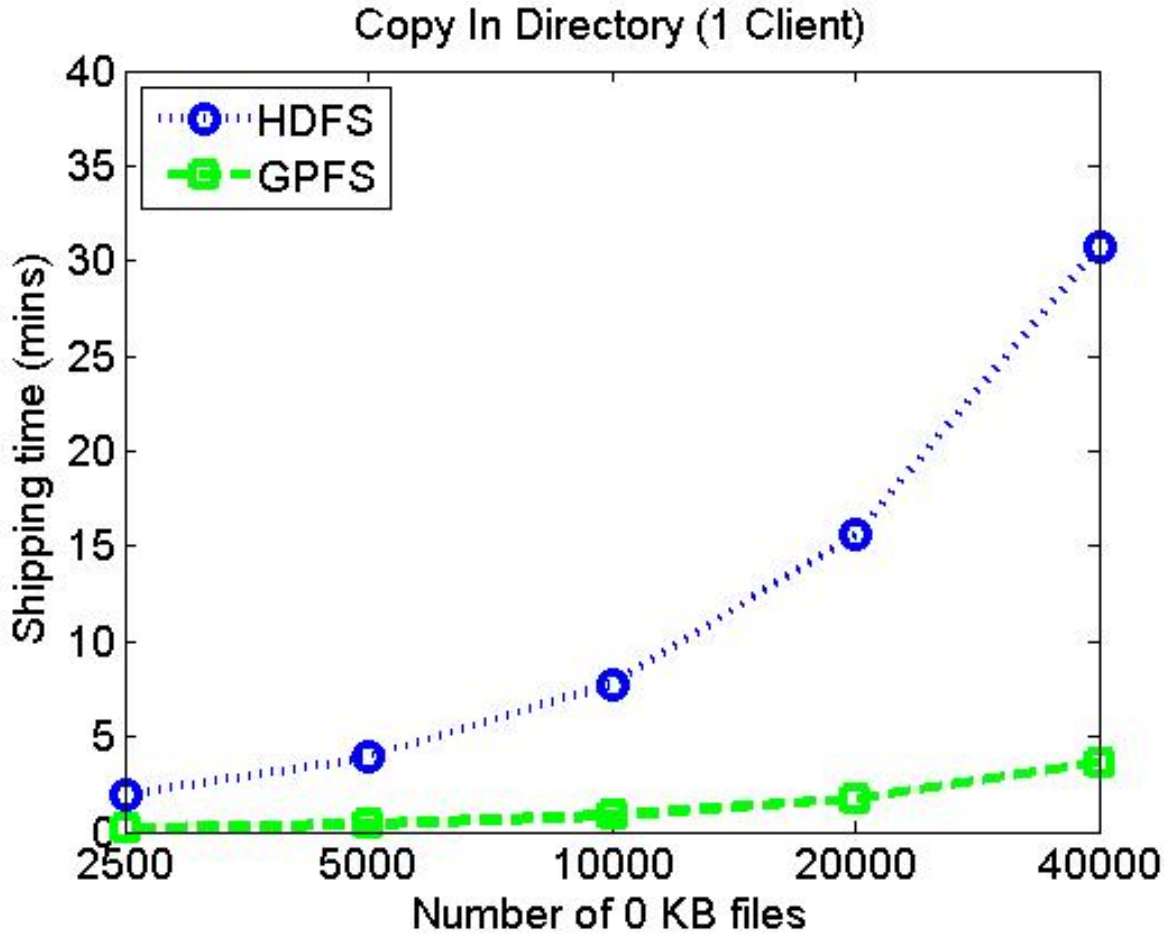
- POSIX semantics
- High performance through striping blocks across multiple nodes and disks
- Distributed locking
- Default Replication factor 2
- Shared disk architecture
- Block location not exposed to apps

Testbed Configuration

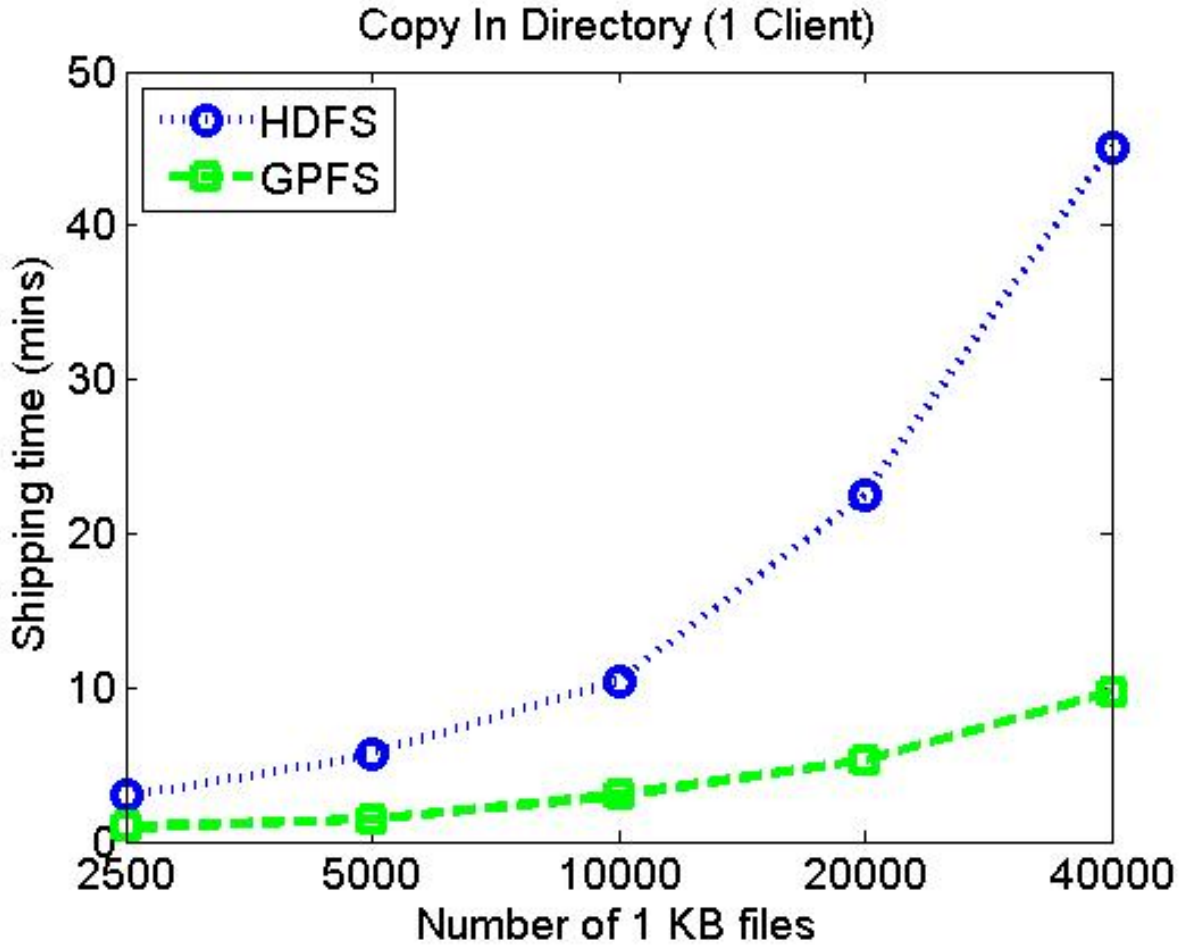
- HDFS 0.17.0
- GPFS 3.2
- 8 nodes
  - 2.8 GHz Pentium IV
  - 5 x 36 GB 10K RM SCSI disk
  - 1 GB Memory
  - 1 Gbps network bandwidth
  - Each node has 5 disks.
    - One disk for OS image
    - Two disks are for HDFS
    - Two disks are for GPFS

- Tools
  - **Hadoop logs**
  - **Ganglia: cluster performance monitoring tool**
- Methodology
  - Creating I/O workload to run on Hadoop
  - Metrics categories: execution time, I/O throughput, network utilization, CPU utilization

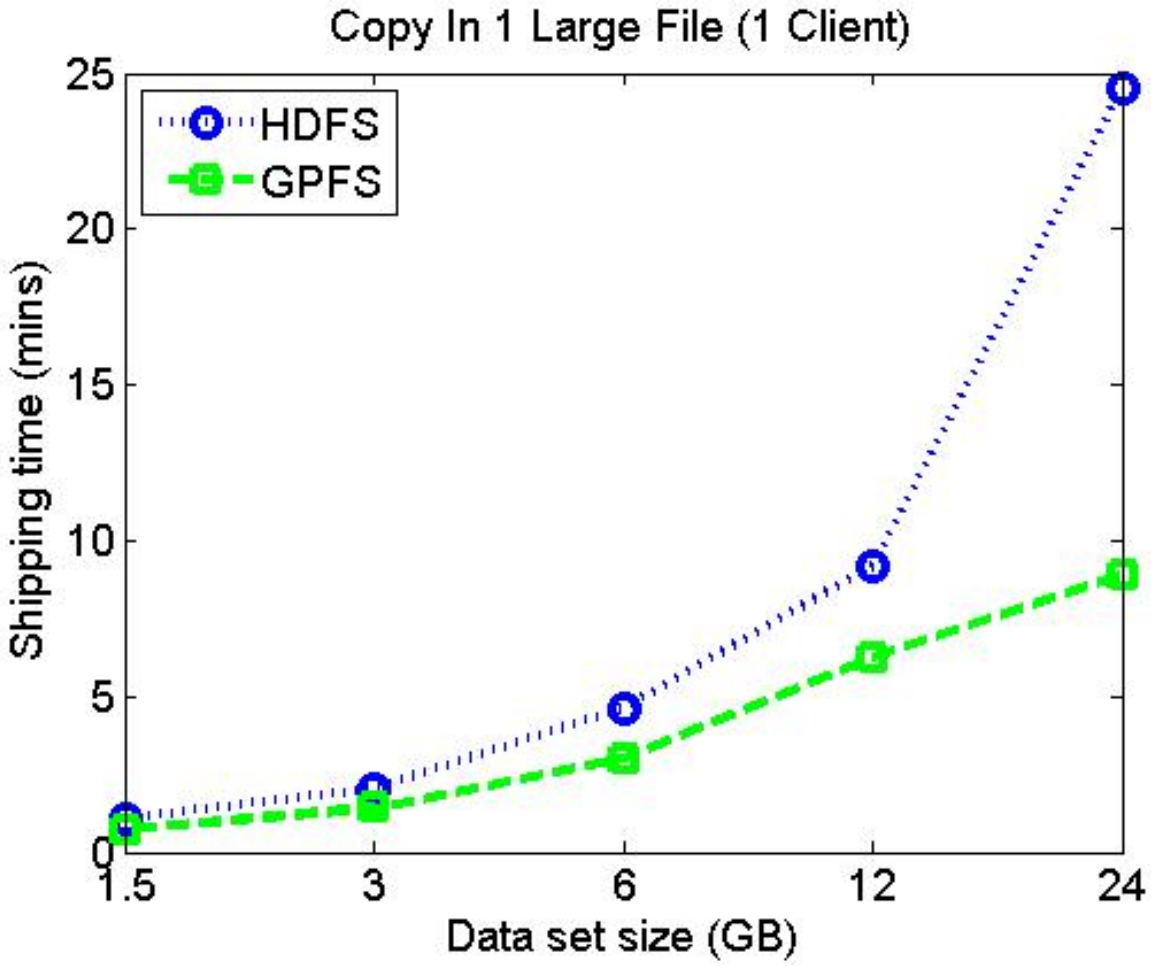
Metadata Efficiency (0 KB files)



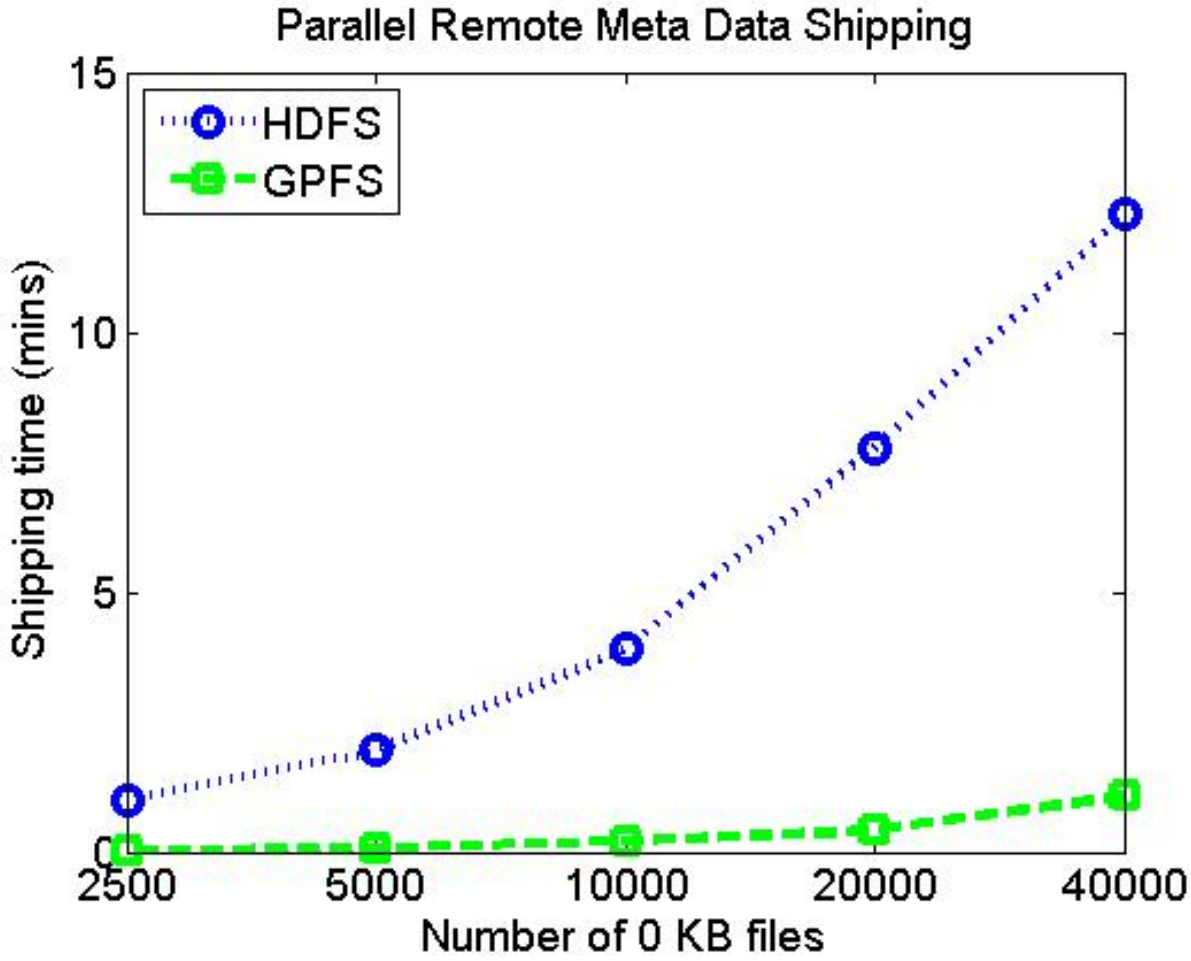
Metadata Efficiency (1 KB files)



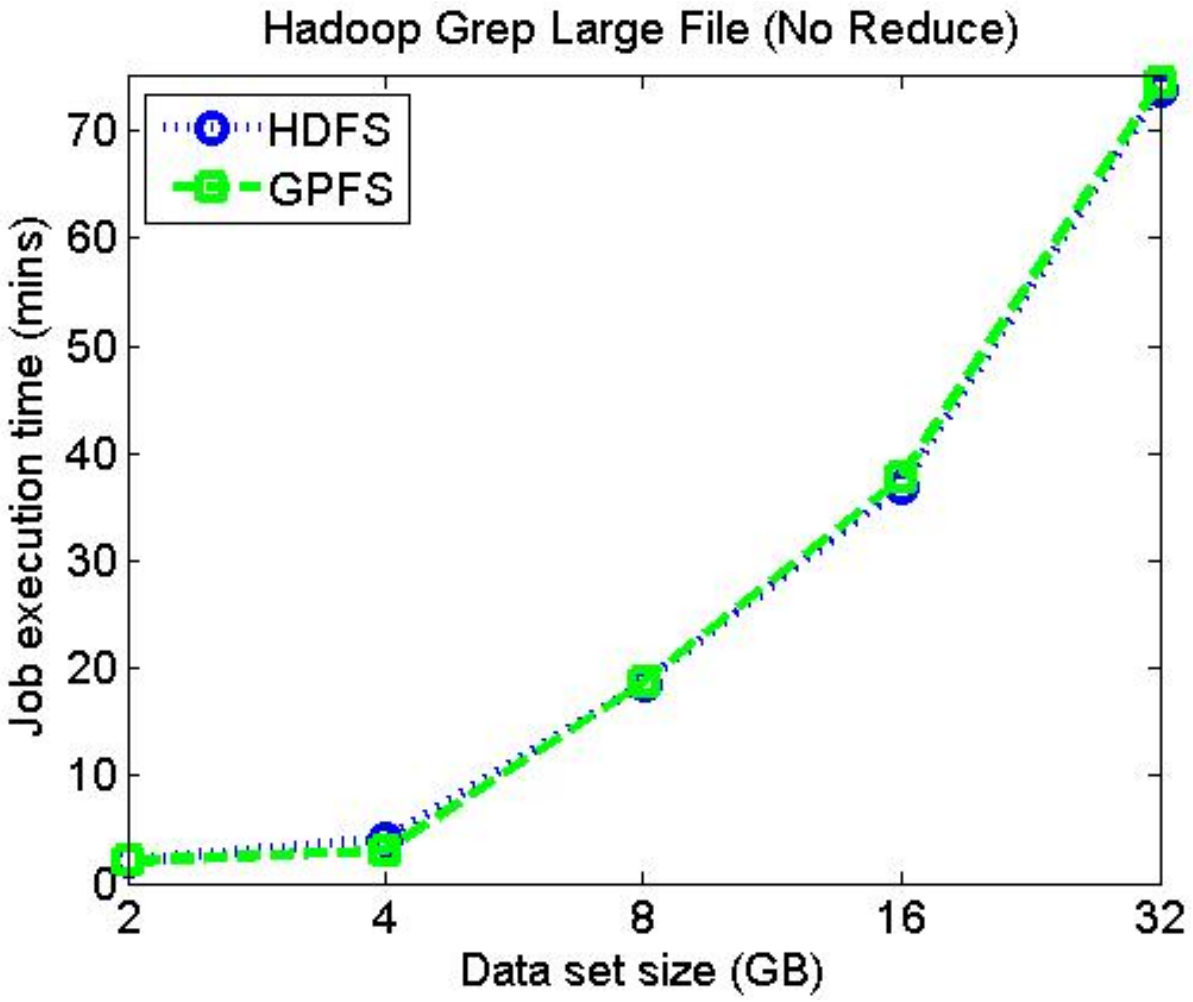
Data Efficiency



Metadata Efficiency (0 KB files)



Function Shipping



Interplay between GPFS and HDFS

- HDFS could benefit from:
  - Metadata scalability
  - Data transfer efficiency
- GPFS could benefit from:
  - Function shipping
  - Location metadata

Integrating GeoGrid with Hadoop

- Goal:
  - Supporting Storage as Service Model
  - Supporting Location based Content Delivery and Dissemination

- Target Applications
  - Internet Movie, Music, and Photo Sharing through Geo-tagging and GeoGrid Overlay Network

Thanks!

- Questions?