



# Performance Analysis of Scheduling and Replication Algorithms on Grid Datafarm Architecture for High-Energy Physics Applications

---

Atsuko Takefusa (Ochanomizu Univ.)

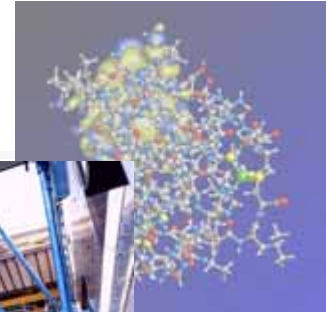
Osamu Tatebe (AIST)

Satoshi Matsuoka (TITECH/NII)

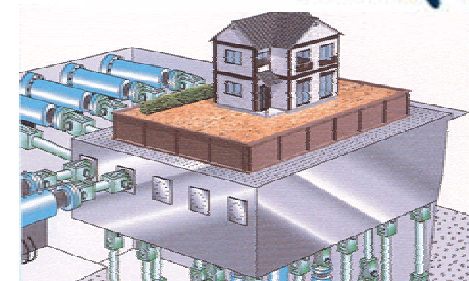
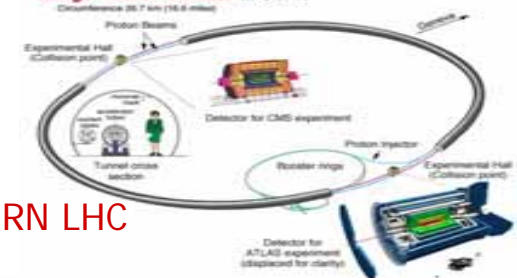
Youhei Morita (KEK)

# Peta-scale Data Intensive Computing

- Large-scale data intensive computer science and data mining
  - HEP, particle physics
    - CERN Large Hadron Collider(LHC) Experiment (2007)
  - Astronomical observatory, life information engineering, etc.
- Large-scale business DB
  - e-Japan, e-commerce
  - Dataware house
  - Search → **Data Grid**



Subaru Telescope  
Large Hadron Collider at CERN



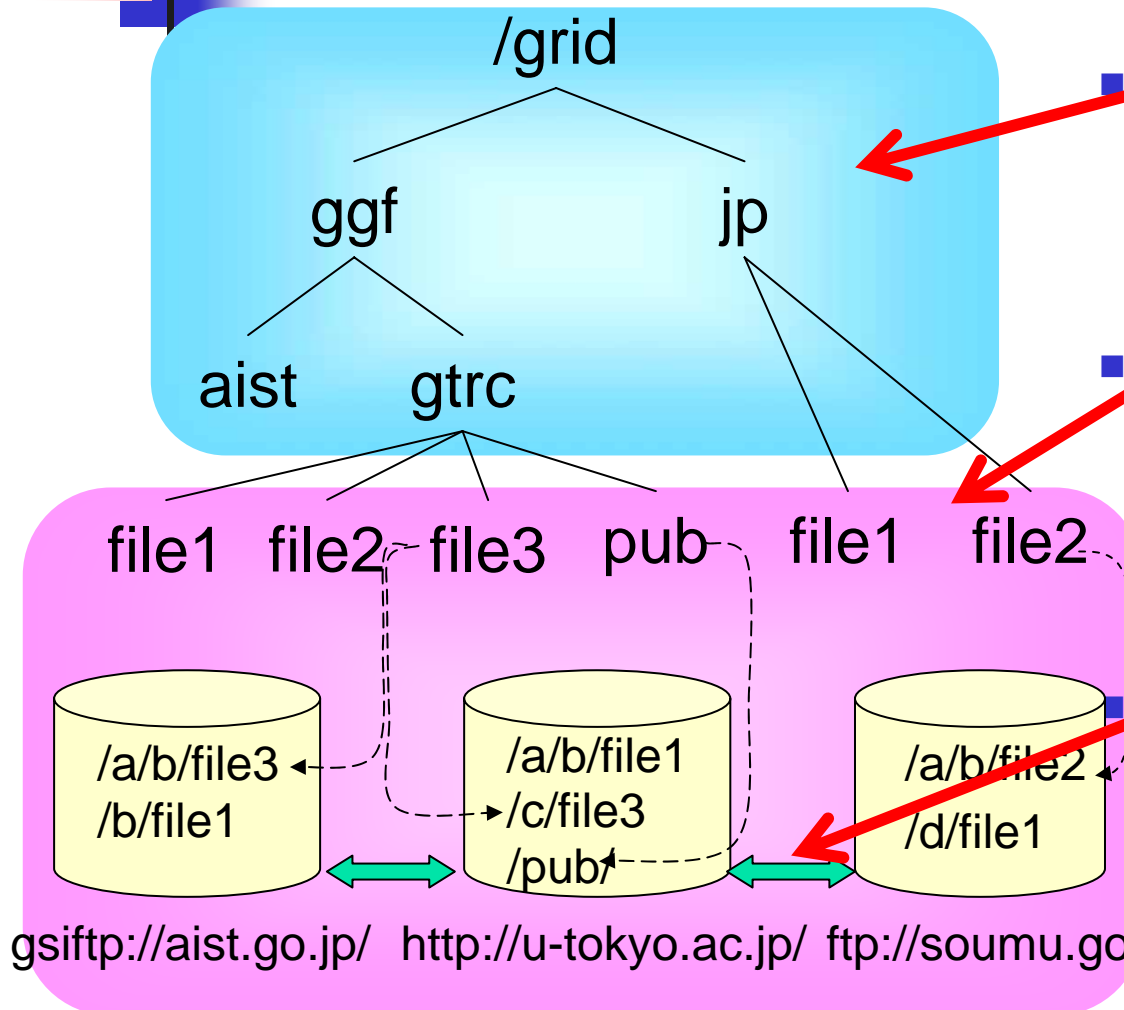
3D Earthquake Simulator

# Technical Issues with the Data Grid



- Techniques of **high speed access** and **data processing**, and **safe sharing** for distributed equipments, machines, humans, or visualization devices
  - Scalable parallel I/O bandwidth
    - > 100GB/s, > 1TB/s (intra-/inter-system)
  - Scalable computing power
    - > 1TFLOPS, > 10TFLOPS
  - Certification, Sharing controlled data/program, limitation of accesses
- System monitoring and management
- Fault tolerance / dynamic re-allocation / recovering data and computing

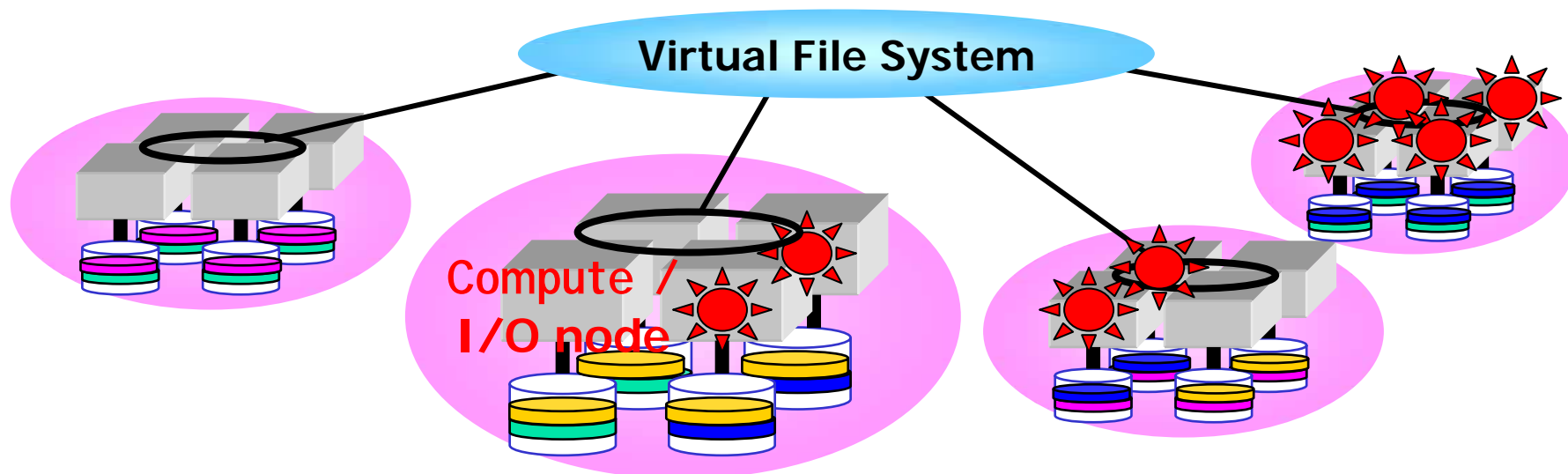
# Grid Datafarm: Grid Virtual File System



- Virtual file system
  - Access using hierarchical virtual path name
  - Hierarchical access control
- Data replica management
  - Fault tolerance and load balancing
  - e.g. OGSA Data Replication Service, Giggle
- High speed file transfer
  - File transfer and remote file access
  - e.g. GridFTP

# Grid Datafarm: Data Parallel Support

- Scalability enabled over TB/sec local I/O Bandwidth
  - File Affinity Scheduling for distributed data over the Grid
  - Local file view – Grid parallel I/O API



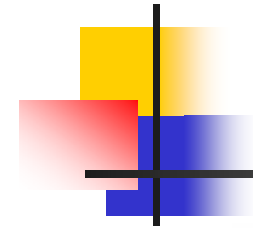


# Performance Analysis on Scheduling and Replication Algorithm on Grid Datafarm

---

- Assuming **Grid Datafarm** architecture
- Realistic assumptions of job processing for CERN LHC experiments starting in 2007
- Experiments on the **Bricks Grid simulator**
- Comparison of Data Grid models
  - **MONARC-style hierarchical model**  
vs. **centralized data storage model**
- Comparison of scheduling/replication algorithms
  - **Owner Computes + background replication**  
vs. **MCT + on-demand replication**

# Data Grid Application: CERN LHC Experiment

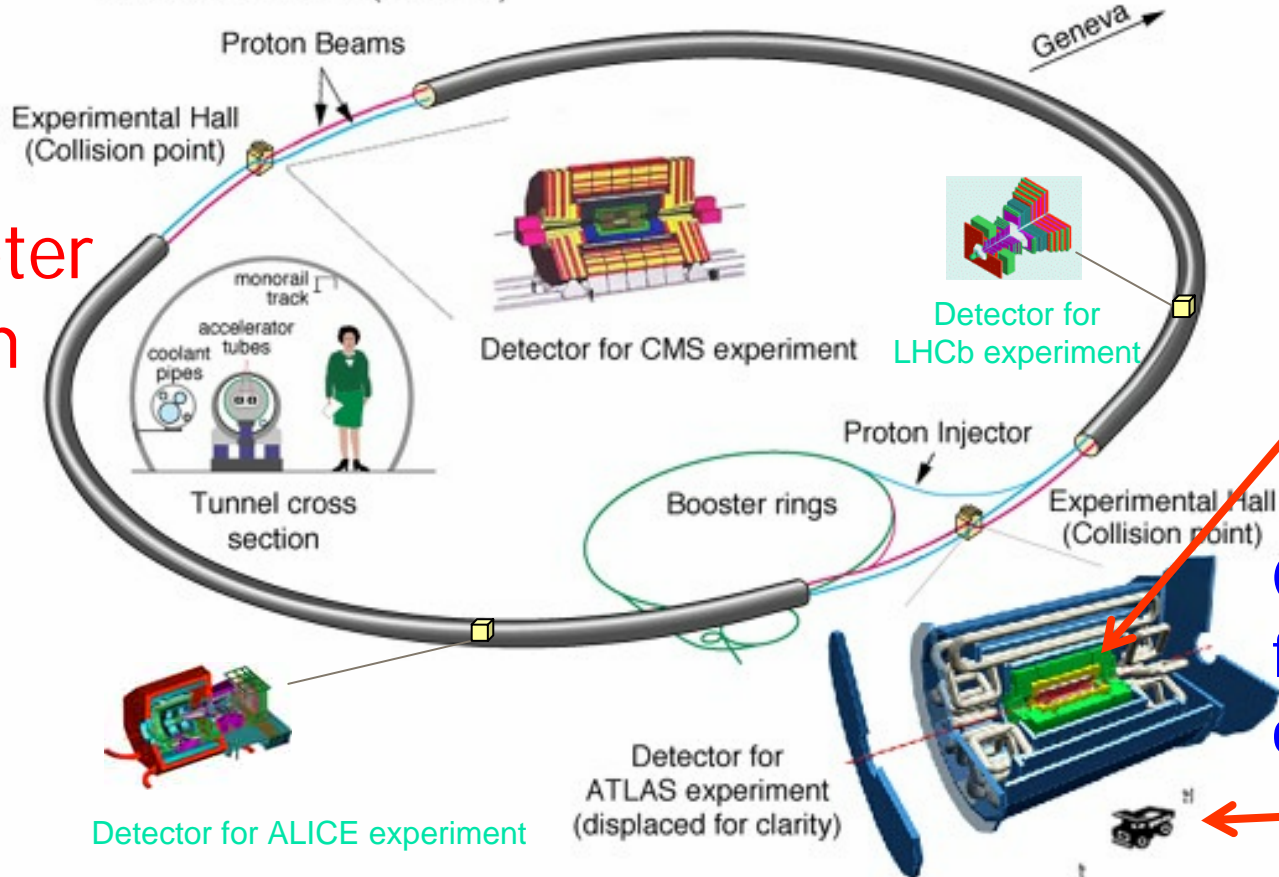


~2000 physicists from  
35 countries

LHC  
perimeter  
26.7km

## Large Hadron Collider at CERN

Circumference 26.7 km (16.6 miles)

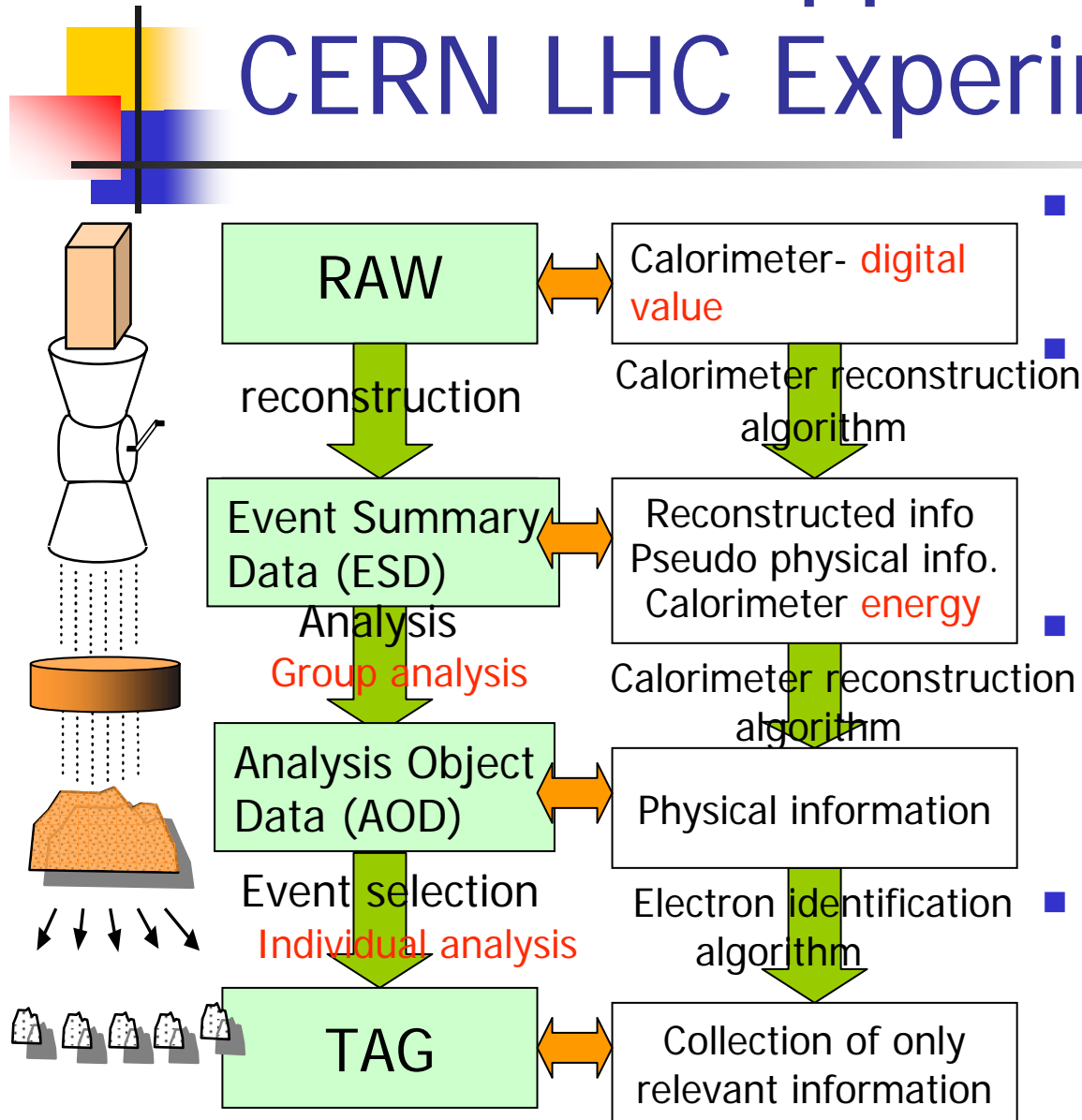


ATLAS  
detector  
40m x 20m  
7000 tons

Collect data  
from collisions  
of particles

← truck

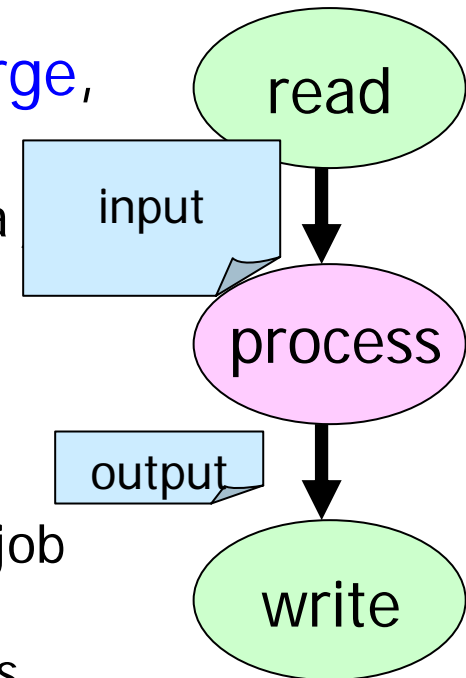
# Data Grid Application: CERN LHC Experiment



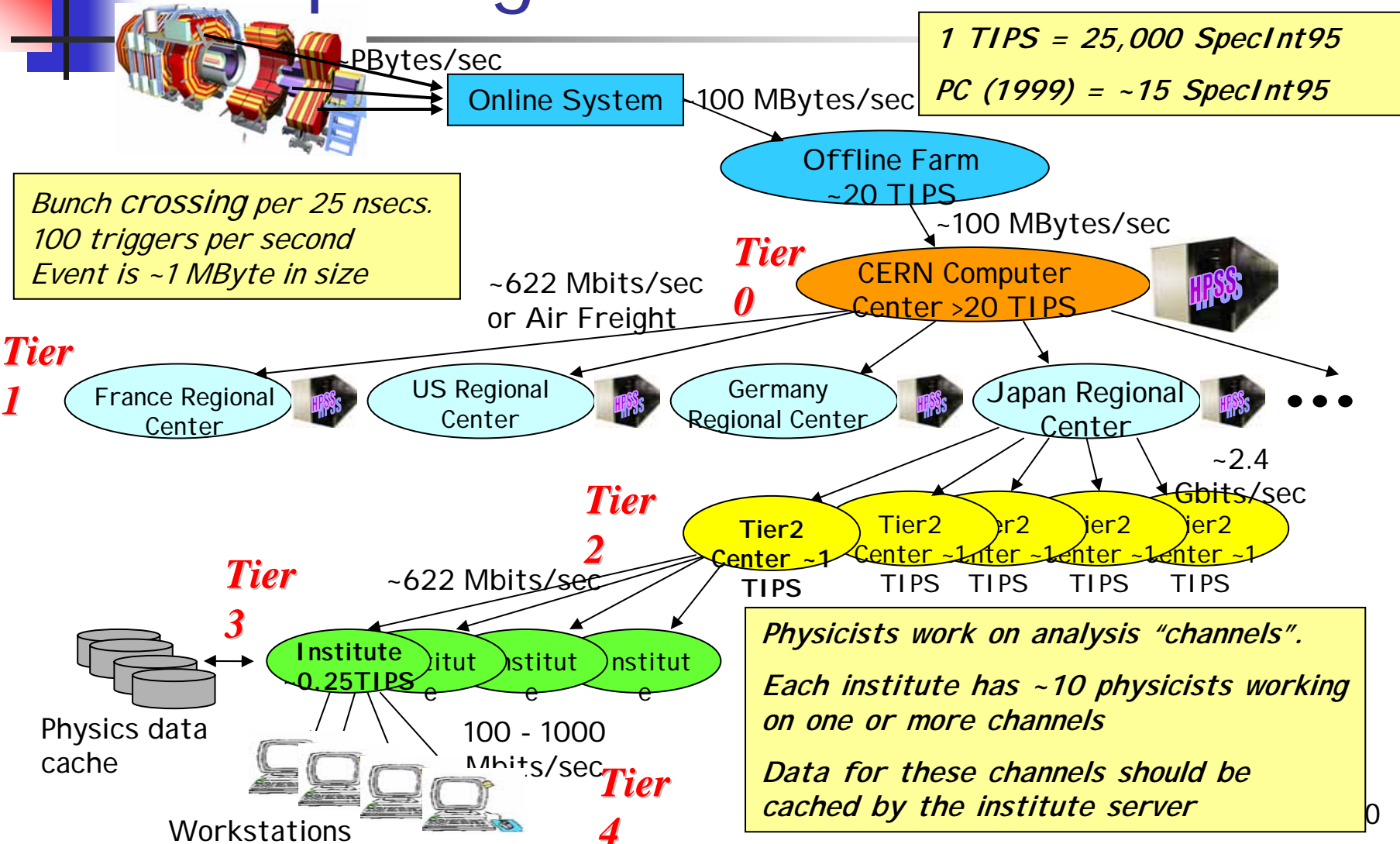
- Analyze data in different levels of hierarchy
- RAW→ESD (Large)**  
Frequency: 2-4/year  
Input: 1PB, Output: 100TB  
Comp: 1000GSpecInt95\*sec
- ESD→AOD (Medium)**  
Frequency: 1/month  
Input: 100TB, Output: 10TB  
Comp: 25GSpecInt95\*sec
- AOD→TAG (Small)**  
Frequency: 1/4 hours  
Input: 10TB, Output: 0.1TB  
Comp: 5GSpecInt95\*sec

# Simulation Model: Job Processing

- A typical job for each of the job classes, **Large**, **Medium**, or **Small** is handled as follows:
  - A user (physicist) at the client machine invokes a
  - The DataGrid scheduler selects a suitable set of servers
  - Each server loads the data fragment required by the job, over the Grid if non-local
  - Each server processes individual portions of the job that the server is assigned to
  - The servers send the output to specified storages
  - (The client receives only statistical data → negligible)
- The time duration to process a job is given as:  
*ResponseTime = Read + Process + Write*



# MONARC-style Multi-tier Center Computing Model for LHC



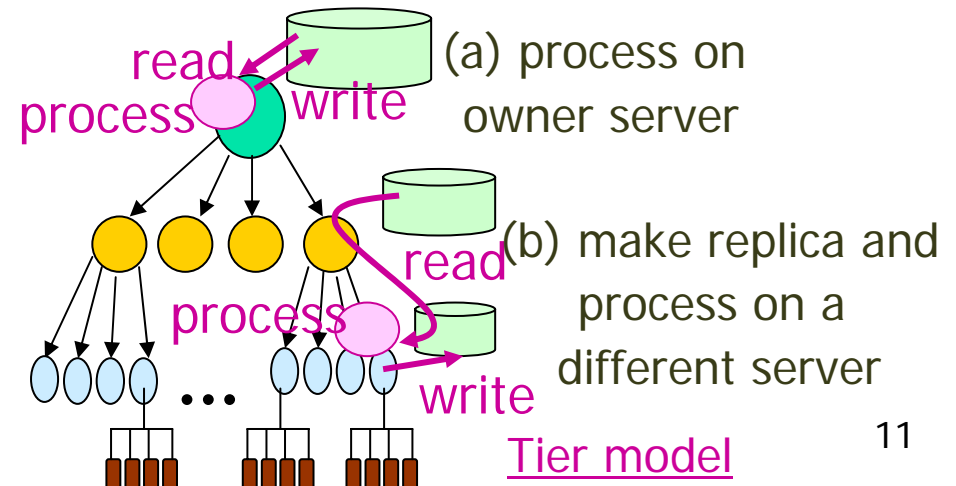
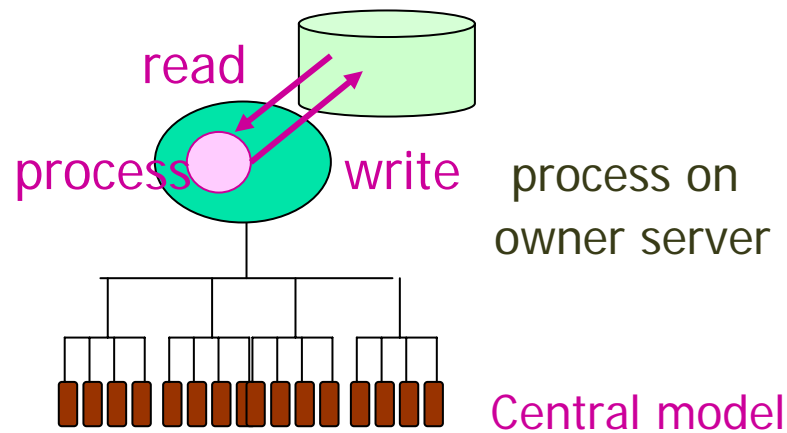
# Simulation Model: Data Grid Architectures

## ■ Central Model

- All the jobs are processed at a single site
- Advantages: performance, manageability

## ■ Tier Model (MONARC-style multi-tier center model)

- Jobs are processed in different levels of the hierarchy
  - Must facilitate suitable scheduling and replication policies
- Advantages: lower power consumption and cost per site





# Scheduling and Replication Algorithms for Tier Model

---

- Central Model: file affinity scheduling (disk **owner-computes** rule)
- Tier Model: must facilitate scheduling and replication policies
  - On-line scheduling algorithms  
→ **On-demand replication**
  - Replication algorithms  
→ **Background replication**



# On-line Scheduling Algorithms

---

On-line Data Grid Scheduler determines DataSourceHost (I/O host), ComputeHost (computation host), DataDestinationHost (I/O host)

- **Greedy(MCT: Minimum Completion Time)**  
Scheduler assigns the job to the host that completes it the earliest
- **OwnerComputes**  
Scheduler selects a *ComputeHost* that owns the input data and completes the job earliest
- **LoadBound-Read/-Write**  
Scheduler select a ComputeHost with MCT from the host group which satisfies:

$$Performance_{estimated} > Performance_{specified}$$

$$Performance_{estimated} = ProcessorPerformance / (LoadAverage + 1)$$

(Scheduler generates replica at the Read/Write time  
if I/O host != computation host)



# Background Replication Algorithm

---

Replica Manager periodically collect status of hosts and trigger replica creation and migration at its discretion

- **LoadBound-Replication:**

- Compute *Perfestimated* for all the hosts

$$Perfestimated = Perf / (LoadAvg + 1)$$

- If  $Perfspecified > Perfestimated$ ,

the Replica Manager creates a replica with the largest *AccessRate* and migrate from the host with the min.

*Perfestimated* to the host with the max.

$$AccessRate = Naccesses / (Tcurrent - Tstored)$$

- **Aggressive-Replication:**

The Replica Manager always generates a replica of all the data generated by each job to the host with the max. *Perfestimated*



# Replica Elimination Algorithm

If the disk space for a job turns out to be insufficient, or some  $x$  % of hosts do not embody some  $y$  % of available disk space within the entire DataGrid, “replica Elimination” is performed

1. Select data that have replicas within the Grid
2. Sort all the selected data by the last recently used (LRU) time
3. Compute Access Rate  $A_{Relim}$  for the first  $N$  data on the list  
$$A_{Relim} = N_{accesses} / (T_{current} - T_{stored}) / N_{copies}$$

( $N_{copies}$  : # of replicas)
4. Select and eliminate replica for data with min.  $A_{Relim}$  while maintaining the following condition:  
$$TotalDiskSize \times Compactness > AvailableDiskSize$$

( $Compactness$  determines the frequency of elimination)
5. If the condition 4 is not satisfied, return to step 3 for the next  $N$  data

( $N = 10$ ,  $x=80$ ,  $y=90$ ,  $N_{accesses}=10$ ,  $Compactness=0.9$  in our simulations)



# Evaluation on Bricks

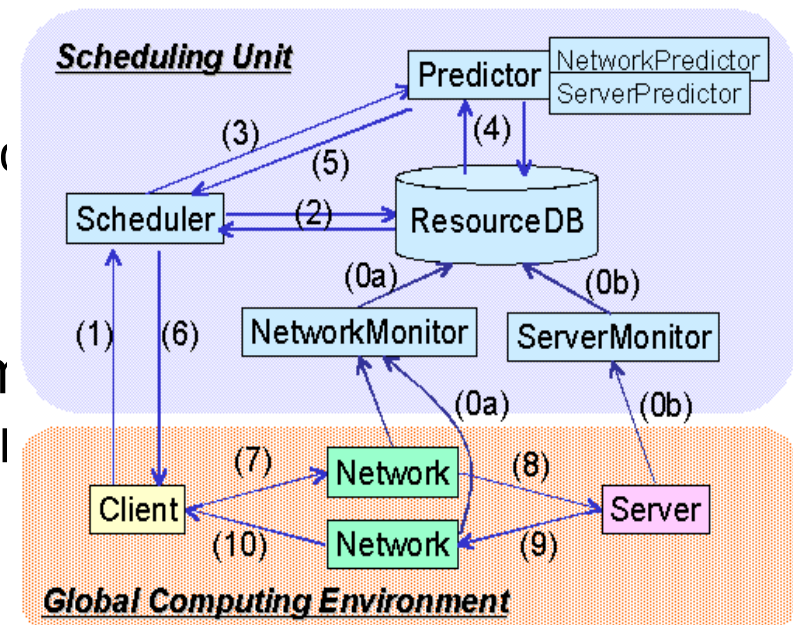
---

- Evaluation on the Bricks Grid simulator
- Investigate the performance of our simulation scenarios for the LHC experiment:
  - Grid Datafarm Architecture
  - Central Model vs. Tier Model
  - 12 different policies: 4 scheduling algorithms x 3 replication algorithms
  - Data access localities (random, [time locality](#))
- Compare
  - [Response time](#)

# Bricks Grid Simulator

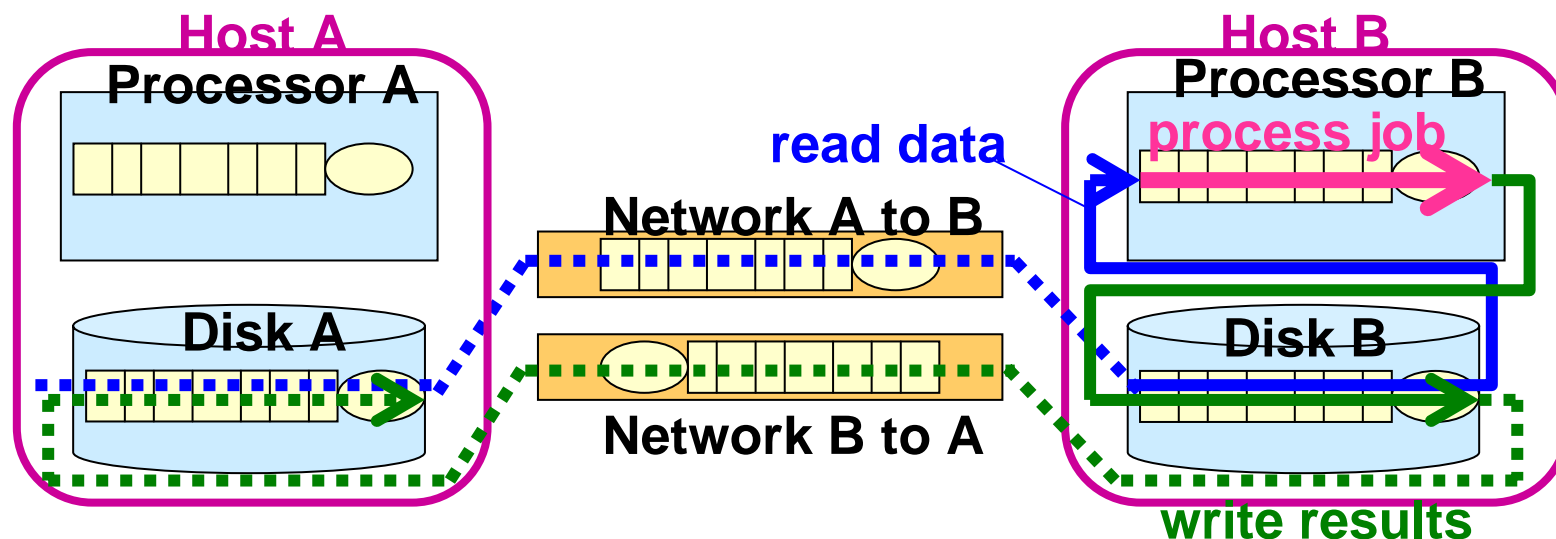
- Java based discrete event simulator
- Represents the behaviors of network and server using queues
- Provides a typical Grid scheduling modules
- Flexible settings of simulation environment
  - Grid topologies (e.g. hierarchical network)
  - Behaviors of Server and network
  - Client model
- Allows to evaluate various scheduling algorithm on dynamic Grid environment

<http://grid-team.is.titech.ac.jp/bricks/>



# Data Grid Extension of the Bricks Grid Simulator

- Provides **Replica Manager** as a Scheduling Unit module
- Provides **Replica Catalog**
- Provides **disk management mechanism**
- Represents **local disk I/O overheads** using queues:





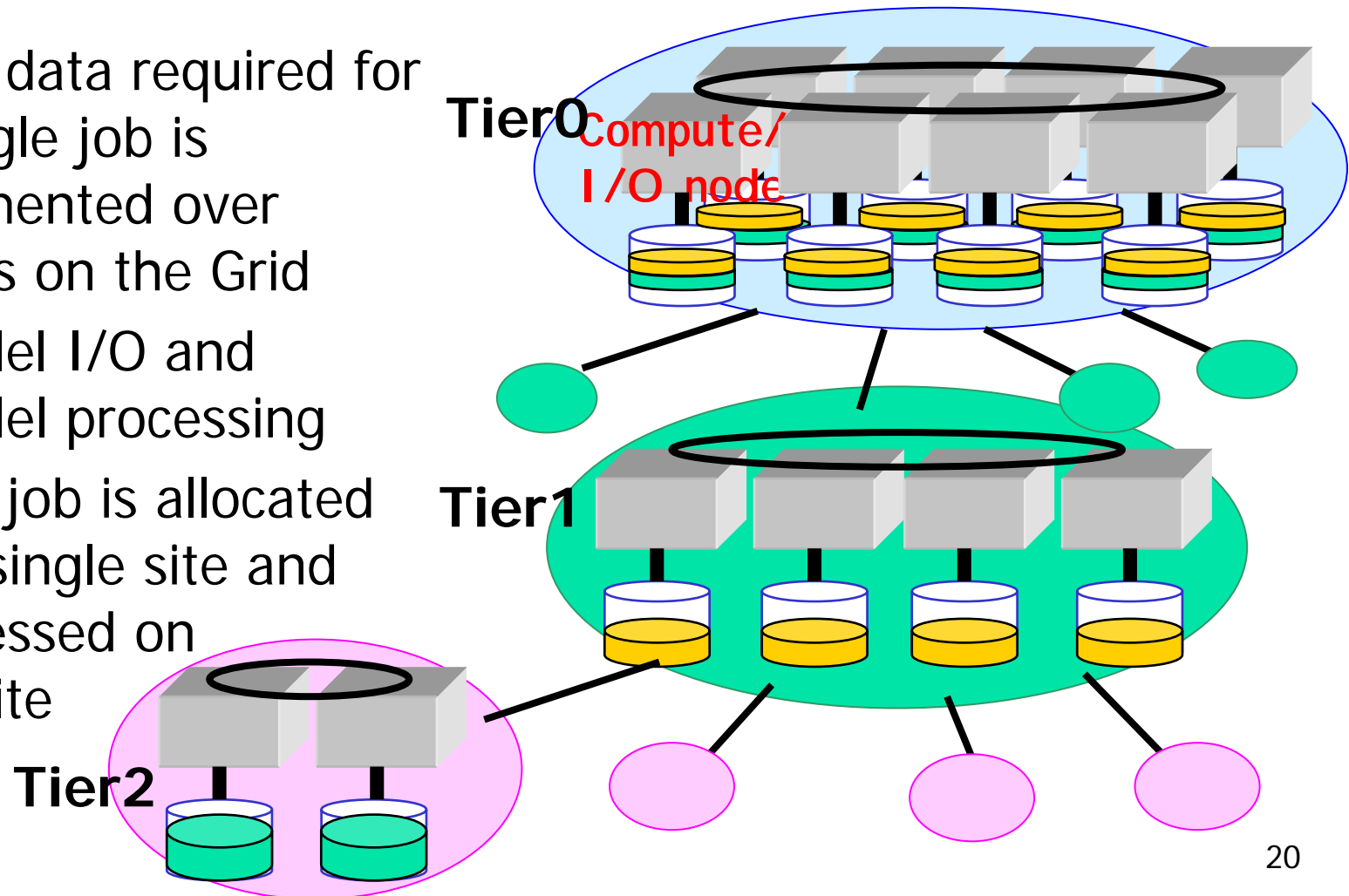
# Evaluation on Bricks

---

- Evaluation on the Bricks Grid simulator
- Investigate the performance of our simulation scenarios for the LHC experiment:
  - Grid Datafarm Architecture
  - Central Model vs. Tier Model
  - 12 different policies: 4 scheduling algorithms x 3 replication algorithms
  - Data access localities (random, [time locality](#))
- Compare
  - [Response time](#)

# Experimental Environment for the Grid Datafarm System

- Each data required for a single job is fragmented over nodes on the Grid
- Parallel I/O and parallel processing
- Each job is allocated to a single site and processed on the site



# Simulation Settings: Environment

Model	Disk [PB]	Performance [MSpecInt95]	# of Nodes on the Site	Total I/O Bandwidth
Central	2	0.5-1.8	10000	1[TB/sec]
Tier	T0 (x1): 2	0.6/0.5/ <b>0.4</b>	10000	1[TB/sec]
	T1 (x4): 1	0.3/0.25/ <b>0.2</b>	5000	500[GB/sec]
	T2 (x16): 0.1	0.03/0.025/ <b>0.02</b>	500	50[GB/sec]

- Similar in settings to the GriPhyN simulation[Grid2001] for Tier Model
- Queuing theory indicates
  - Est. avg. response time in Central Model is 38.575-1.337 [hours]
  - Central Model with under 0.453318[MSpecInt95] saturates and cannot process LHC jobs
- WAN and Local I/O bandwidth are set to 10[Gbps] and 100[MB/sec]
- On Grid Datafarm architecture aggregation local I/O bandwidth is:

**Total I/O Bandwidth = Local I/O Bandwidth × # of nodes on the site**

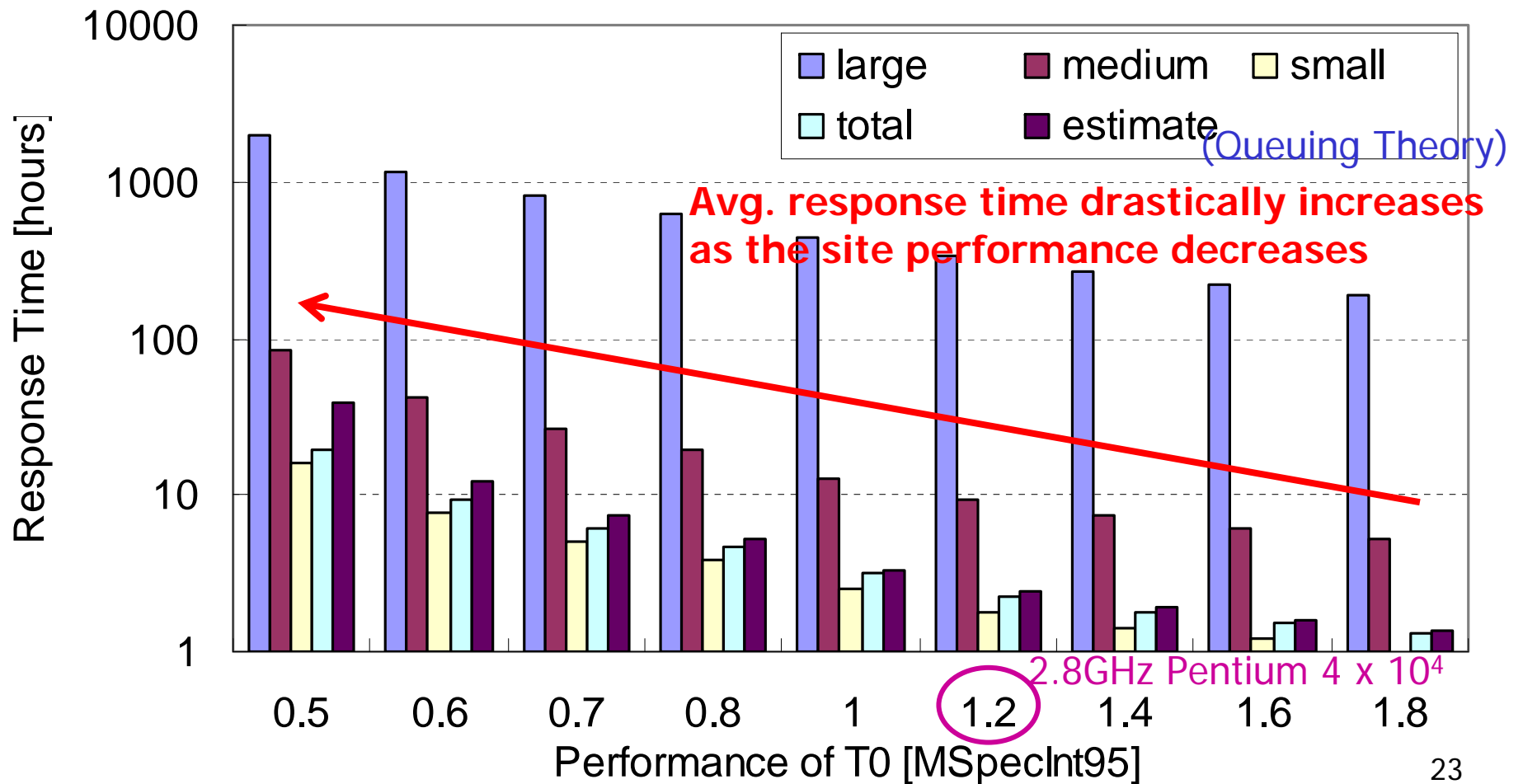


# Simulation Settings: Job

Job	# events / Job	Comp. Size [GSI95*sec]	Frequency (Avg.)	Input [TB]	Output [TB]
<b>Large:</b> RAW→ESD	1G	1000	1/4 [月]	1000	100
<b>Medium:</b> ESD→AOD		25	1/1 [月]	100	10
<b>Small:</b> AOD→TAG		5	1/4[時間]	10	0.1

- Actual parameters for LHC in the MONARC Report2
- All the initial data(1PBx1, 100TBx2, 10TBx4) are stored in the Tier0 site and the total # of data is increasing during the simulations
- 1 year simulation x 10 for each algorithms
- Executed simulation on the Presto III cluster (Dual Athlon MP 1900+, 768MB memory, 256 nodes) at TITECH

# Central Model: Avg. Response Time



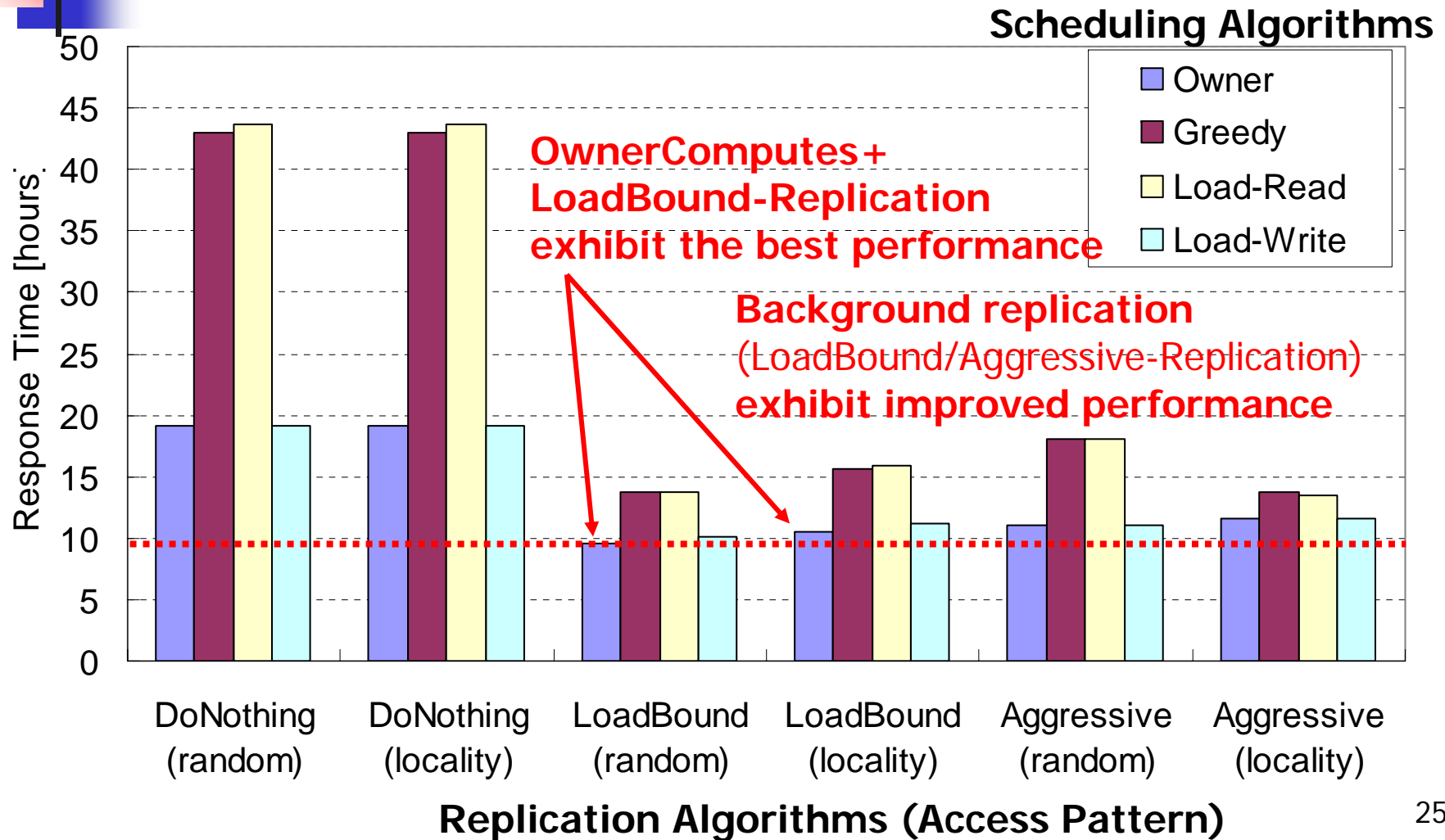


# Combinations of Scheduling and Replication Algorithms for Tier Model

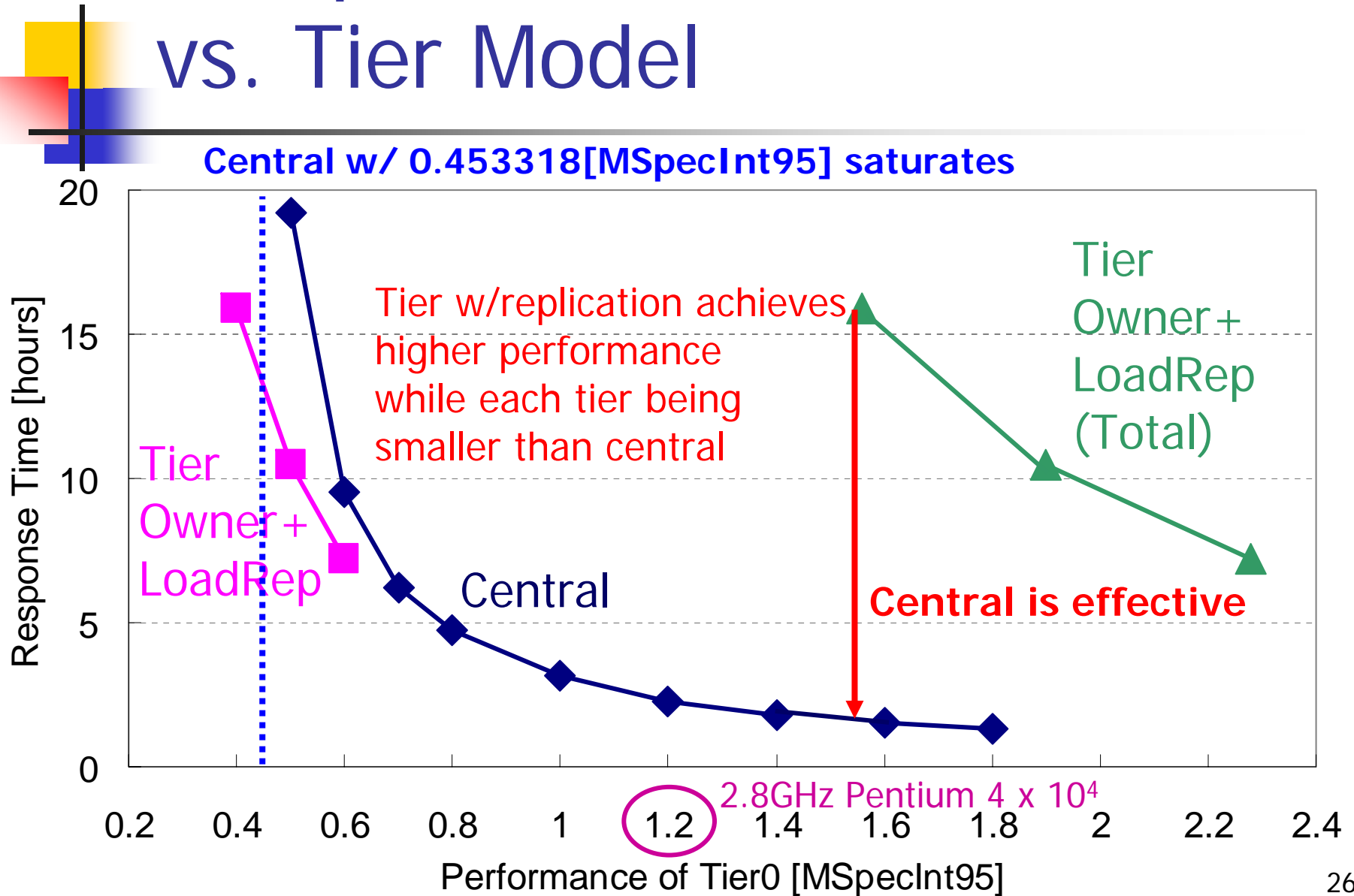
---

- Scheduling Algorithms (on-demand replication)
  - Greedy
  - OwnerComputes
  - LoadBound-Read
  - LoadBound-Write
- Replication Algorithms (Background replication)
  - LoadBound-Replication
  - Aggressive-Replication
  - DoNothing (no background replication)
- 4 Scheduling x 3 Replication = 12


# Tier Model: Avg. Response Time (T0, T1, T2) = (0.5, 0.25, 0.025)



# Comparison of Central Model vs. Tier Model



# Related Work: A Study of GriPhyN Simulation[HPDC-11, '02]



- GriPhyN [Univ. of Chicago et al]
  - Targets the CERN LHC experiment
  - Construct Globus-based peta-scale virtual Data Grid
- Simulation study for scheduling and replication for DataGrid
  - Propose External Scheduler, Local Scheduler, and Dataset Scheduler for DataGrid scheduling and replication
  - Evaluated External and Dataset Scheduling Algorithms
    - JobDataPresent (OwnerComputes) + replication shows improved performance
  - Not assume the actual parameters for the LHC experiment (small job granularity, shorter duration, # of "original" data is NOT increasing)

↔ Our study: Experiments on Grid Datafarm architecture  
w/ actual parameters for the CERN LHC



# Conclusions

---

- Evaluate the performance of DataGrid system models by using the Bricks Grid simulator, newly enhanced with Data Grid
  - Affinity scheduling enabled over **TB/sec** local I/O bandwidth on the Grid Datafarm architecture
- Compare Central vs. MONARC-style Tier Model, assuming the Grid Datafarm architecture:
  - Central Model is effective and could be constructed in a feasible fashion at this point
  - In Tier Model, Background replication algorithms proves to be effective
  - Tier Model achieves higher performance while each tier being smaller than Central



# Future Work

---

- Propose effective replica elimination algorithm
- Propose suitable scheduling and replication algorithms and evaluate on large-scale actual environments