Reza Sookhtsaraei, Javad Artin, Ali Ghorbani, Ahmad Faraahi, Hadi Adineh, 2016. A locality-based replication manager for data cloud. *Frontiers of Information Technology & Electronic Engineering*, **17**(12):1275-1286. http://dx.doi.org/10.1631/FITEE.1500391



# A Locality-Based Replication Manager for Data Cloud

**Key words:** Data cloud, Replication, Graph, Locality replication manager (LRM)

Corresponding author: Reza Sookhtsaraei
E-mail: reza.sookhtsaraei@gmail.com

ORCID: http://orcid.org/0000-0002-6444-3201

# Motivation

- Cloud computing provides computational resources and scalable storage using the Internet.
- The users can use public services such as water, gas and telephone which are used by the public according to their needs .
- In many sciences, the capacity of data is shown in terabytes and petabytes.
- the replication of data is used as an effective technique for managing the efficiency of such spacious data.

# Motivation (Con’t)

- Advantages of data replication:
  - more access to data
  - less delay in access
  - increased availability
- For efficient replication of data, two important questions should be answered:
  - How many copies should be made from each data to satisfy the needs of the system?
    - The more number of copies, the more space and energy
  - Where the copies are stored?
    - Each copy should be placed to perform the duties faster and to ensure that the load is distributed in a balanced way.

# Main idea

- A locality-based replication manager is presented in this paper for data-based functions in cloud storage.
  - It is named Locality replication manager (LRM)

- LRM takes into account:
  - Quality-of-service (QoS)
  - Physical locality of data blocks in one inquiry

# Main idea (Con’t)

- Main goals of LRM :
  - decreasing the number of copies
  - increasing the availability
  - satisfying the needs of QoS
  - decreasing the operational expenses such as the storage space and energy

# Method

- System Model

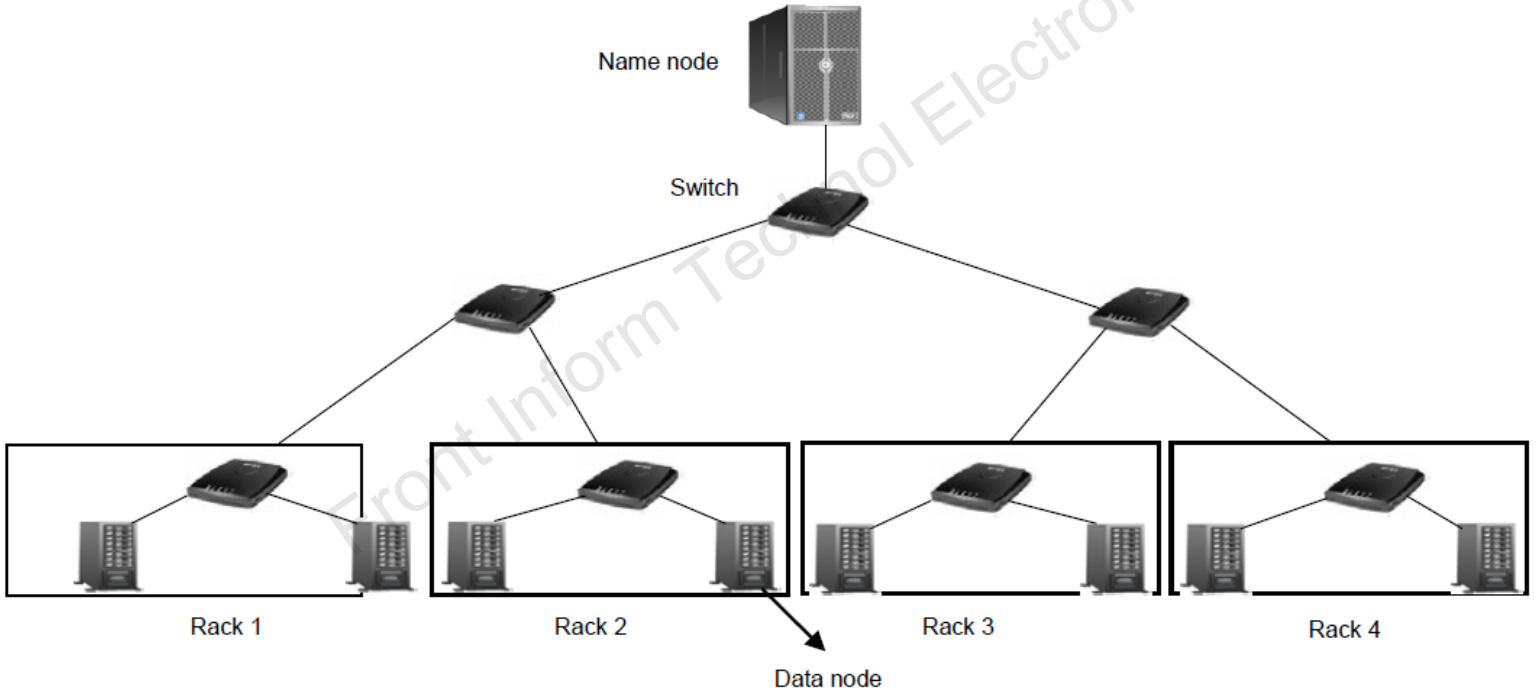



**Fig. 1 Architecture of the Hadoop distributed file system**

# Method (Con’t)

- System Model …

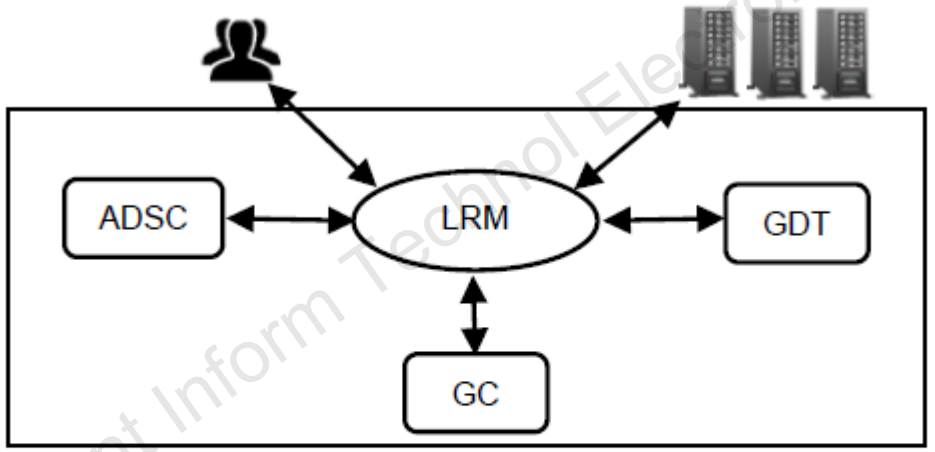



**Fig. 2 Components of the coordinator**

ADSC: availability and delay system care; LRM: locality replication manager; GDT: graph directory table; GC: graph constructor

# Method (Con’t)

- Locality replication manager (LRM)
  - It is the replication manager
  - Receive the inquiries of the users, collect the condition of data nodes in the cluster
  - Decide to place the blocks in the best host
- Graph directory table (GDT)
  - Managed by LRM
  - Includes:
    - very important information from the system
    - Number of accesses to each block in the graph
    - Host of each graph
    - Maximum delay in accessing each graph

# Method (Con’t)

- Graph constructor (GC)
  - Constructs a full graph from among the available blocks in each inquiry
  - Delivers the graph to LRM for placement decisions
- Availability and delay system care (ADSC)
  - Starts When system is not in an appropriate level regarding the delay and availability
  - Receives messages from the LRM
  - Determines the suitable data nodes to place the graphs

# Method (Con't)

- Symbols and the Functions of the Used Goal…
  - Possibility of block availability in system is determined by

$$P(B_i) = 1 - \prod_{j=1}^{M} \theta(i,j) \cdot P_j ,$$

  - Availability of one graph inquiry is defined by

$$P(G_k) = \prod_{i=1}^{|G_k|} P(B_i).$$

  - Delay of $B_i$ is shown by

$$\overline{L_{B_i}} = \frac{1}{r_i} \sum_{j=1}^{M} \theta(i,j) \cdot A(i,j),$$

# Method (Con't)

- Symbols and the Functions of the Used Goal…
  - Delay of a group of blocks

$$\overline{L_{G_k}} = \frac{1}{|G_k|}\sum_{i=1}^{|G_k|}\overline{L_{B_i}}.$$

  - LRM selects a node from among other nodes that can satisfy

$$\max[(\mathrm{Qos}(\mathrm{req}_i) - \overline{L_{G_{\mathrm{new}}}}) + P(G_{\mathrm{new}}) + |S(G_{\mathrm{new}}) \cap S(m_j)| + \mathrm{storage}_j \cdot \alpha - \mathrm{load}_j].$$

# Method (Con't)

- Finding the Appropriate Node(s) to Send the Inquiry (QRep Algorithm)

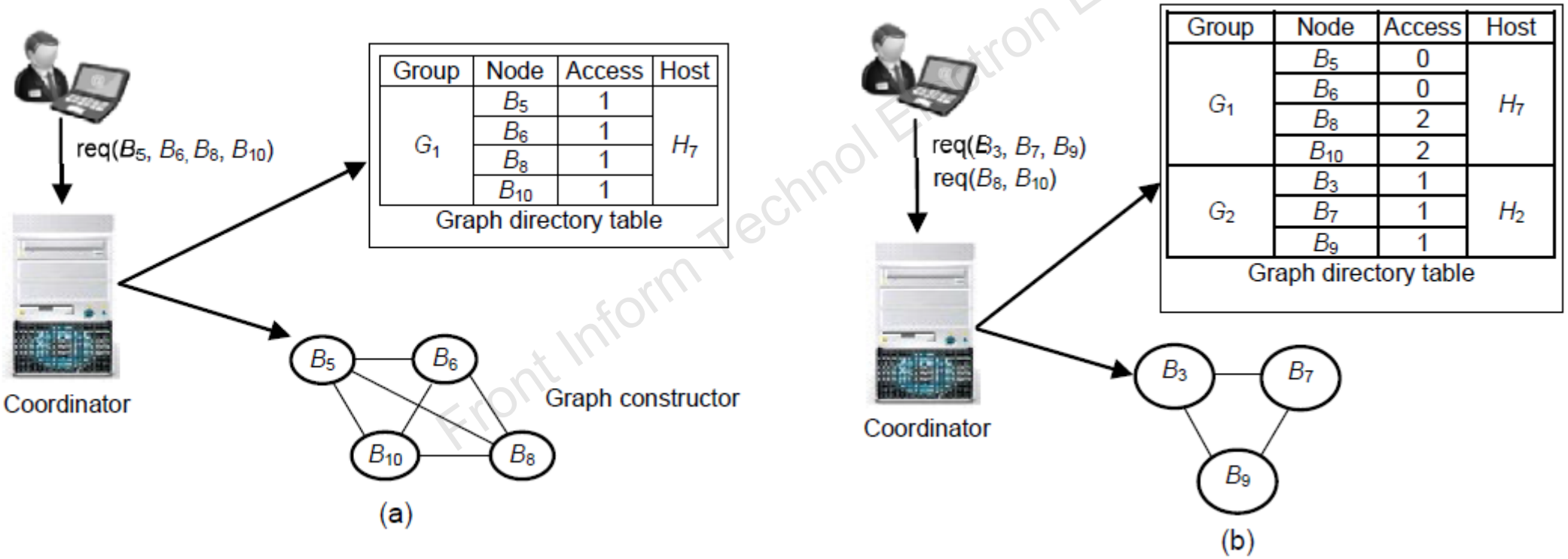
**Algorithm 1** Finding the appropriate node(s) to send the inquiry

**Input:** file *f* with *b* blocks.
**Output:** QoS-based distributed replicas on physical nodes.

1 Distribute *b* blocks of file *f* on physical nodes randomly;
2 **for** each input inquiry $inq_i$ **do**
3 Coordinator receives $inq_i$;
4 Form a graph $G_{new}$ for $inq_i$;
5 Search *t*=a node (or a set of nodes) in *S*(*M*) which $G_{new}$ (part of $G_{new}$) hosts;
6 Meet QoS($inq_i$) in the graph directory table;
7 **if** *t*!=∅ **then**
8 **if** *t* is a node **then**
9 Coordinator redirects $inq_i$ to *t*;
10 **else** *t* is several nodes **then**
11 Coordinator sorts *t* based on the supported QoS non-descendingly;
12 *S*=∅;
13 **while** nodes in *S* do not host $G_{new}$ **do**
14 *S*=*S*∪(the first node of *t*);
15 **end while**
16 **if** average QoS(*s*) meets QoS($inq_i$) **then**
17 Register *s* in the coordinator;
18 **end if**
19 **end if**
20 Coordinator changes the access field for each node (block) in *G* in the directory group table;
21 Coordinator removes each node in *G* and *t* whose access field reaches zero;
22 (Replica evaporation) except for the replica which is the last from the main version;
23 **else**
24 Coordinator lies in a new replica for each node in $G_{new}$ in a physical node;
25 Calculate the minimum value by function (5);
26 Coordinator registers a $G_{new}$ in the graph directory table;
27 **end if**
28 **end for**

# Method (Con't)

- Sending the inquiry to LRM, making the graph and recording it in GDT
- Updating the GDT by entering new inquiries into LRM




| Group | Node | Access | Host |
|---|---|---|---|
| $G_1$ | $B_5$ | 1 | $H_7$ |
| | $B_6$ | 1 | |
| | $B_8$ | 1 | |
| | $B_{10}$ | 1 | |

| Group | Node | Access | Host |
|---|---|---|---|
| $G_1$ | $B_5$ | 0 | $H_7$ |
| | $B_6$ | 0 | |
| | $B_8$ | 2 | |
| | $B_{10}$ | 2 | |
| $G_2$ | $B_3$ | 1 | $H_2$ |
| | $B_7$ | 1 | |
| | $B_9$ | 1 | |

**Fig. 3 Managing the graph directory table (GDT)**

# Method (Con't)

- Taking Care of Availability and Delay of System (NPMapping Algorithm)
- This algorithm uses the genetic complementary algorithm

**Algorithm 2** Keeping availability and delay of system at a desired level

**Input:** number of groups of requested blocks $G$, number of physical nodes $N$, maximum generation number Gens, and current generation $t$.

**Output:** near optimal mapping $G \rightarrow N$.

1 $\Phi_t$=Generate initial population randomly;
2 **while** ($t$<Gens) **do**
3 Evaluate the fitness of each individual in $\Phi_t$ based on function (6);
4 $\Phi_{temp}$=$Q$ highest individuals based on their fitness;
5 $\Phi_{t+1}$=$Q-L$ highest individuals based on their fitness from $Q$;
6 $\Phi_{t+1}$=$\Phi_{t+1} \cup$($L$ remaining individuals crossed from $Q$ with a two-point cross-over);
7 $\Phi_{t+1}$=$\Phi_{t+1} \cup$($K$ individuals generated randomly);
8 $\Phi_{t+1}$=Mutation($\Phi_{t+1}$);
9 $t$=$t$+1;
10 **end while**

# Method (Con't)

- Create a chromosome from physical nodes and graphs

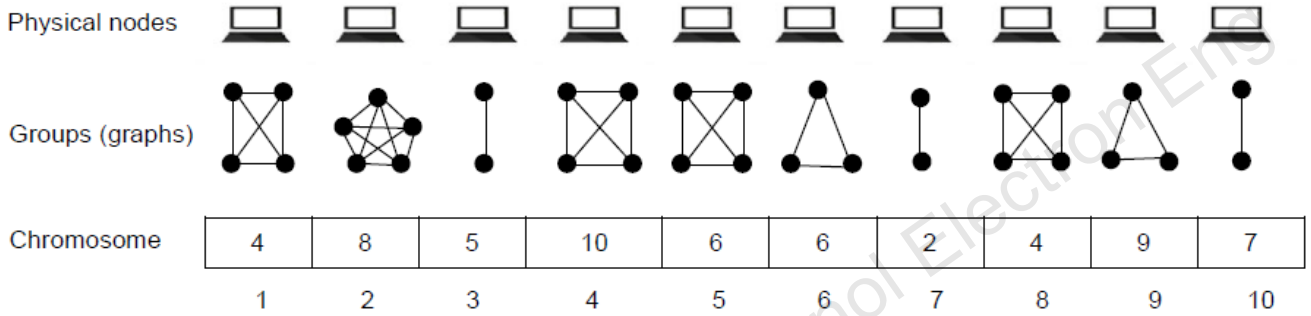



**Fig. 4 Creation of a chromosome from physical nodes and graphs**

- Proportion of delay to the availability of the whole cloud system

$$P(S)=\prod_{i=1}^{|G|}P(G_i)$$

$$\overline{L_S}=\frac{1}{|G|}\sum_{i=1}^{|G|}\overline{L_{G_i}}.$$

$$\min\{\overline{L_S}/P(S)\},$$

# Method (Con't)

- Two-point cross over

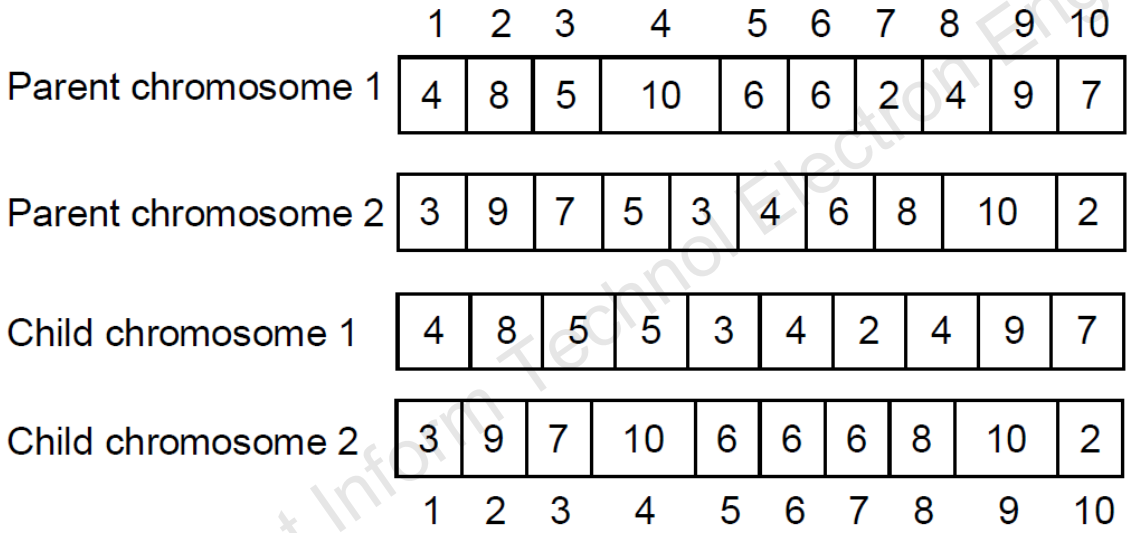



**Fig. 5 Two-point cross-over**



- Mutation
  - The mutation step is done by 0.5 transfer rate

# Simulation Environment

- We made a simulator called LRMsim using the java language.
- This simulator makes a modular simulation framework that helps us to model different structures of data cloud and other resources.
- Simulation lets us determine
  - Different structures from network
  - Different types of nodes
  - Different resources for nodes
  - A method for replication

# Major results

- Replication factors obtained by different replication managers

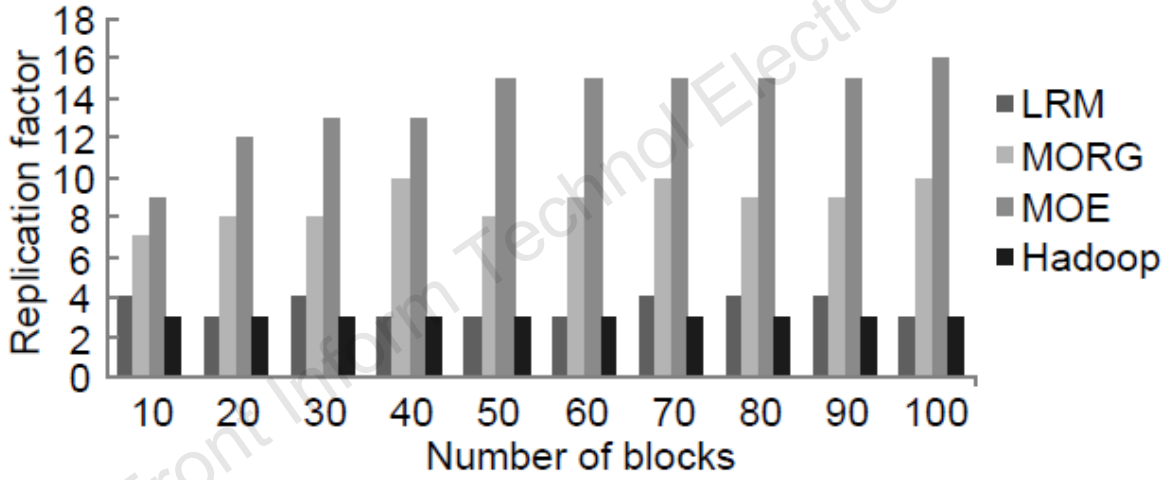



**Fig. 6 Replication factor with each method**

LRM: locality replication manager; MORG: multi-objective randomized greedy; MOE: multi-objective evolutionary

# Major results (Con't)

- Number of accessed physical nodes by different methods

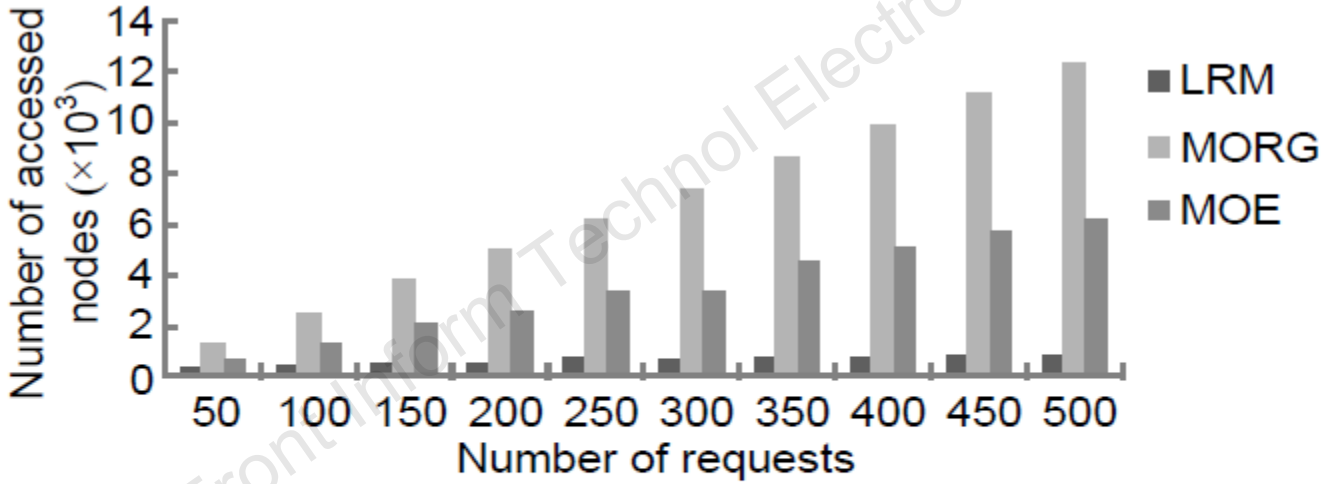





**Fig. 7 Number of accessed physical nodes with each method**

LRM: locality replication manager; MORG: multi-objective randomized greedy; MOE: multi-objective evolutionary

# Major results (Con't)

- Number of active physical nodes to manage the replicas

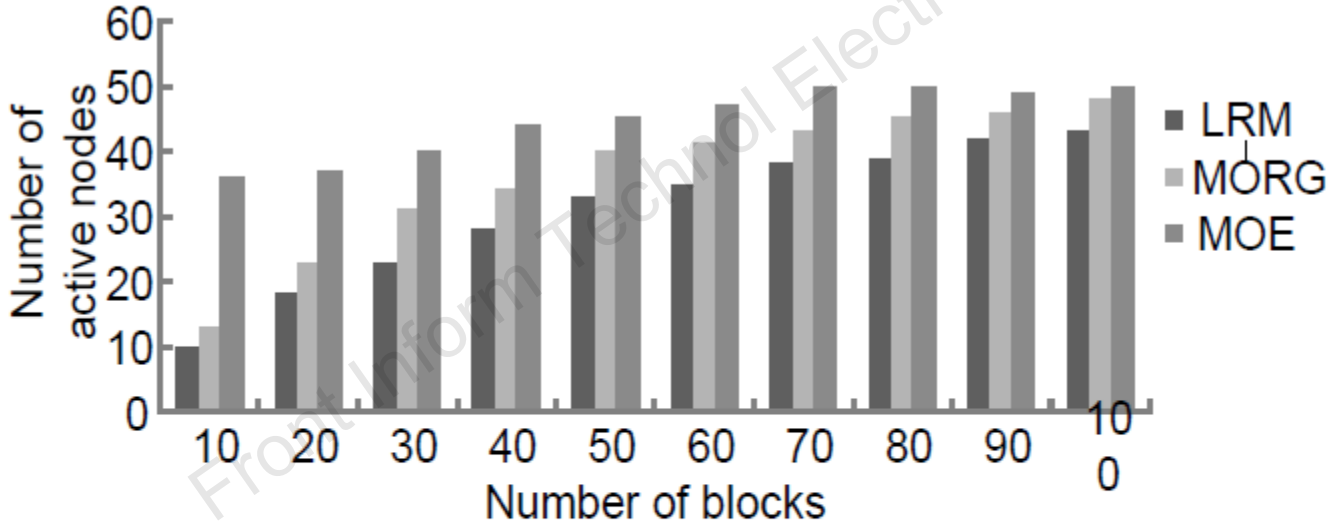





**Fig. 8 Number of active physical nodes to manage the replicas with each method**

LRM: locality replication manager; MORG: multi-objective randomized greedy; MOE: multi-objective evolutionary

# Major results (Con't)

- Distribution of load in physical nodes by different replication methods

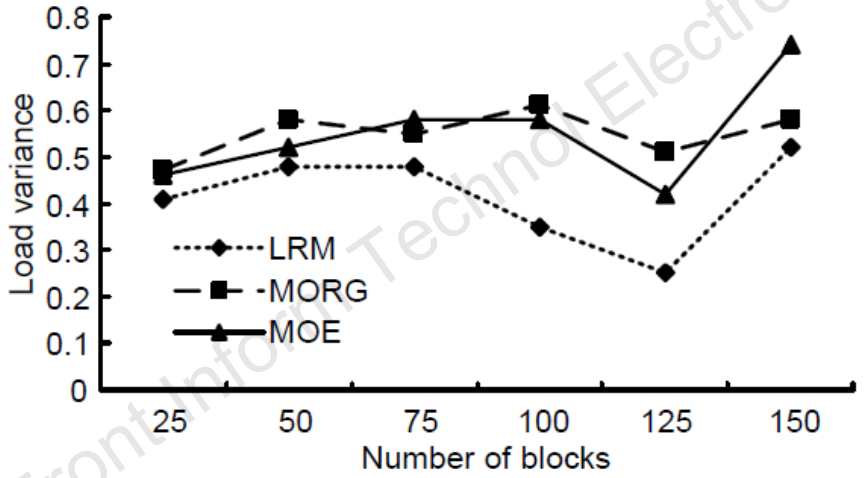



**Fig. 9 Distribution of load in physical nodes with each method**

LRM: locality replication manager; MORG: multi-objective randomized greedy; MOE: multi-objective evolutionary

# Major results (Con't)

- System's average of not being available in different replication methods

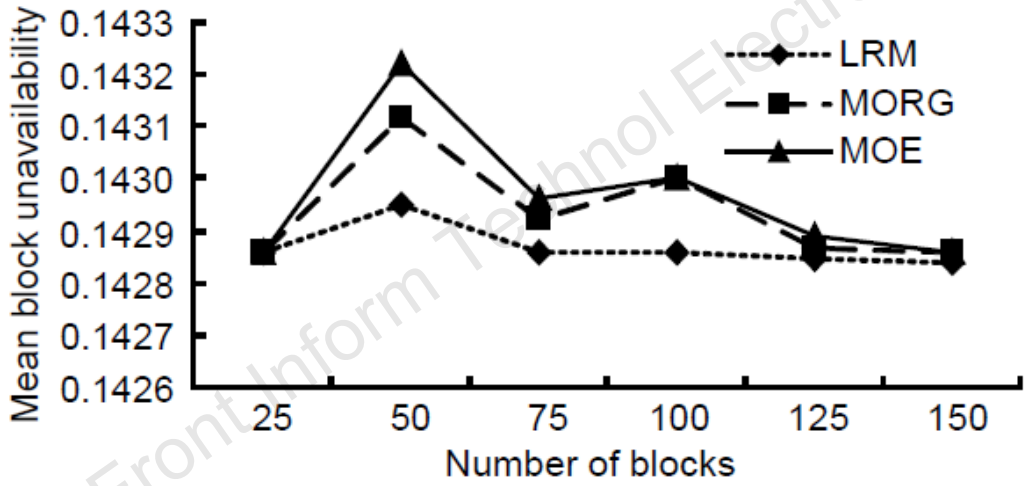



**Fig. 10 System's average rate of not being available with each method**

LRM: locality replication manager; MORG: multi-objective randomized greedy; MOE: multi-objective evolutionary

# Major results (Con't)

- Average delay in satisfying the requests by different replication methods

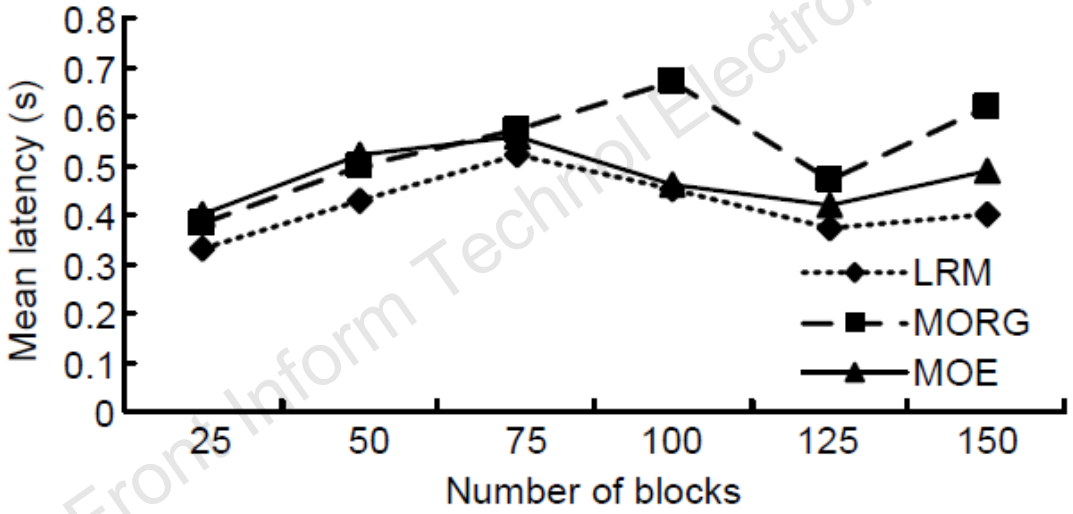





**Fig. 11 Average delay in satisfying the requests with each method**

LRM: locality replication manager; MORG: multi-objective randomized greedy; MOE: multi-objective evolutionary

# Conclusion and Future Studies

- In this paper
  - we dealt with the issue of replication management of blocks in one file for the data cloud to select a suitable replica factor and find a good place for them.
  - we suggested LRM using the concept of blocks' locality in one request and the complementary GA to solve the problem.
  - To assess the success of LRM, we compared it to two other algorithms named MOE and MORG in a simulated environment written in Java.
  - LRM performs better than the other two algorithms and Hadoop replication subsystem.
- In future
  - we intend to assess the LRM on a real cluster of the data cloud.
  - we would like to examine other algorithms of replication and different models of cost.
  - we intend to study the methods of online data movement to face the dynamic changes in the pattern of accessing blocks.