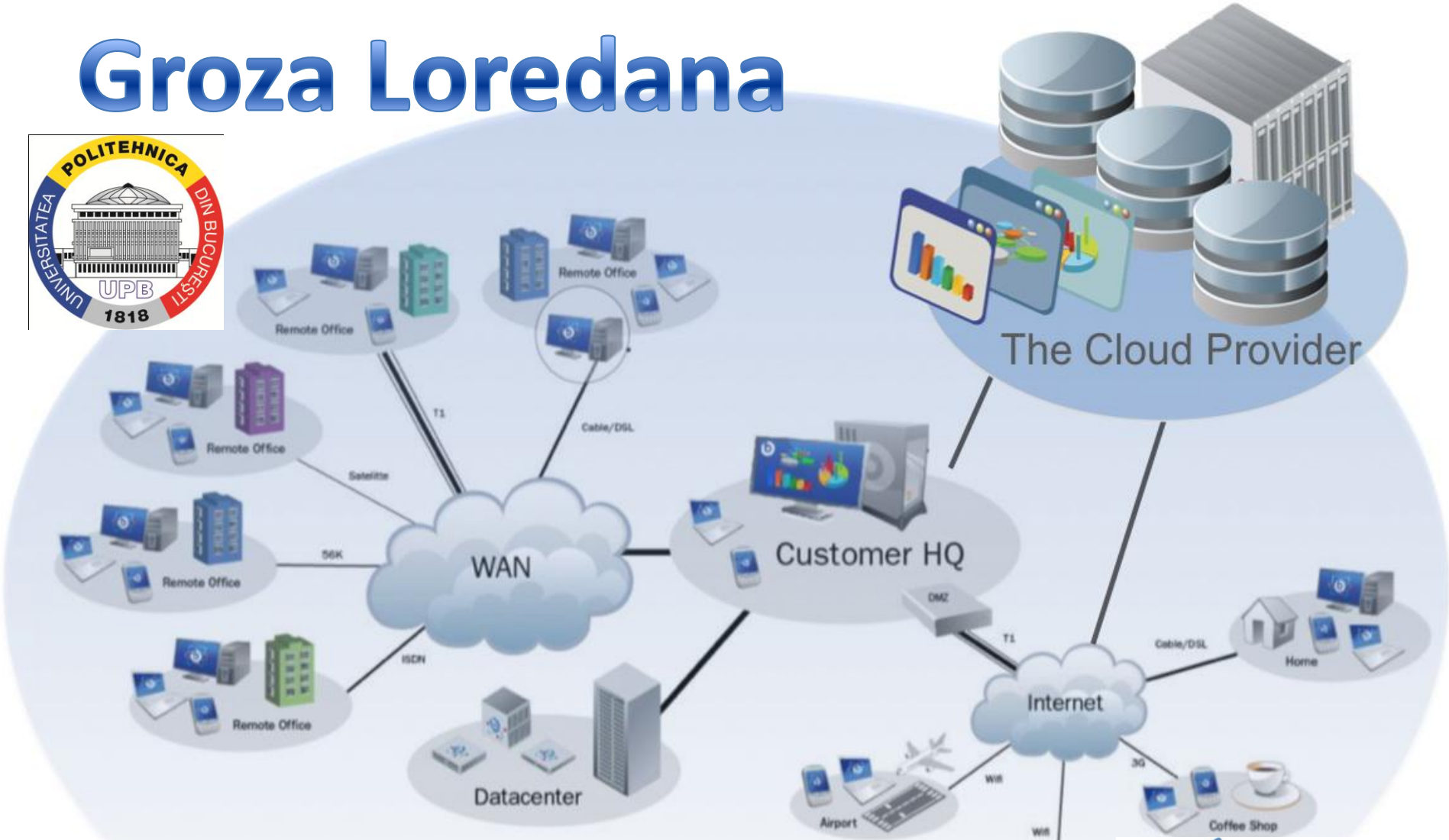
Groza Loredana



Dynamic Load-Balanced Multicast for Data-Intensive Applications on Clouds

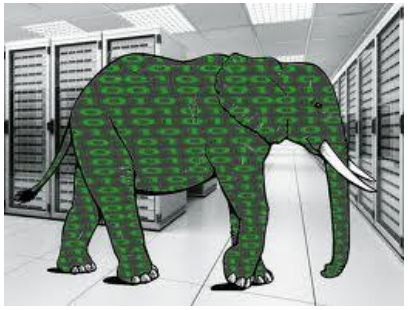
vrije Universiteit amsterdam

Contents:

- Introduction
- Multicast on parallel distributed systems
- Multicast on P2P systems
- Multicast on clouds
- High performance multicast algorithms
- Implementation
- Performance Evaluation
- Summary
- Conclusions

The Problem

- deploy large data sets from the cloud's storage facility to all compute nodes as fast as possible.



- the network performance within clouds is dynamic

- keeping network monitoring data and topology information up-to-date is almost impossible

Common multicast algorithms

- construct one or more spanning trees based on the network topology
- monitoring data in order to maximize available bandwidth and avoid bottleneck links.

We propose two hp multicast algorithms

Combine optimization ideas from **multicast algorithms** used in:

- **parallel distributed systems**
- **P2P systems**

Efficiently transfer large amounts of data stored in Amazon S3 to multiple Amazon EC2 nodes.

Contents:

- Introduction
- Multicast on parallel distributed systems
- Multicast on P2P systems
- Multicast on clouds
- High performance multicast algorithms
- Implementation
- Performance Evaluation
- Summary
- Conclusions

Multicast on parallel distributed systems 1/2

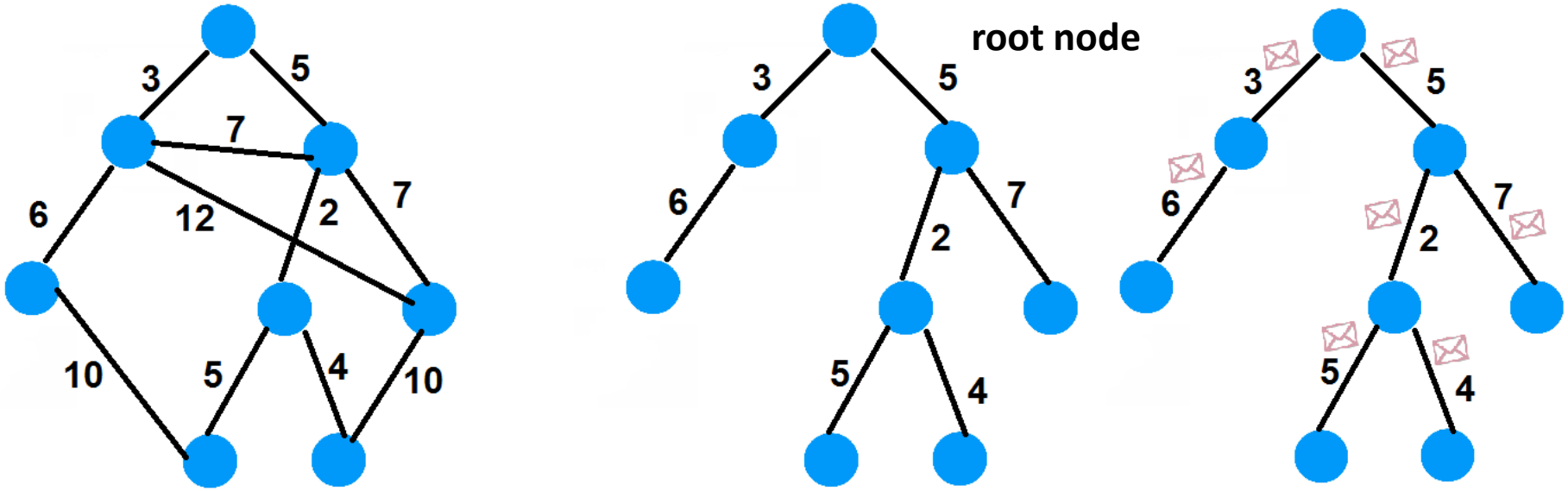
- collective operation algorithms in message passing systems like MPI .

- The target environment :

1) network performance is high and stable
2) network topology does not change
3) available bandwidth between nodes is symmetric

Multicast on parallel distributed systems 2/2

- construct one or more optimized spanning trees by using network topology information and other monitoring data.



- sender-driven technique

Contents:

- Introduction
- Multicast on parallel distributed systems
- Multicast on P2P systems
- Multicast on clouds
- High performance multicast algorithms
- Implementation
- Performance Evaluation
- Summary
- Conclusions

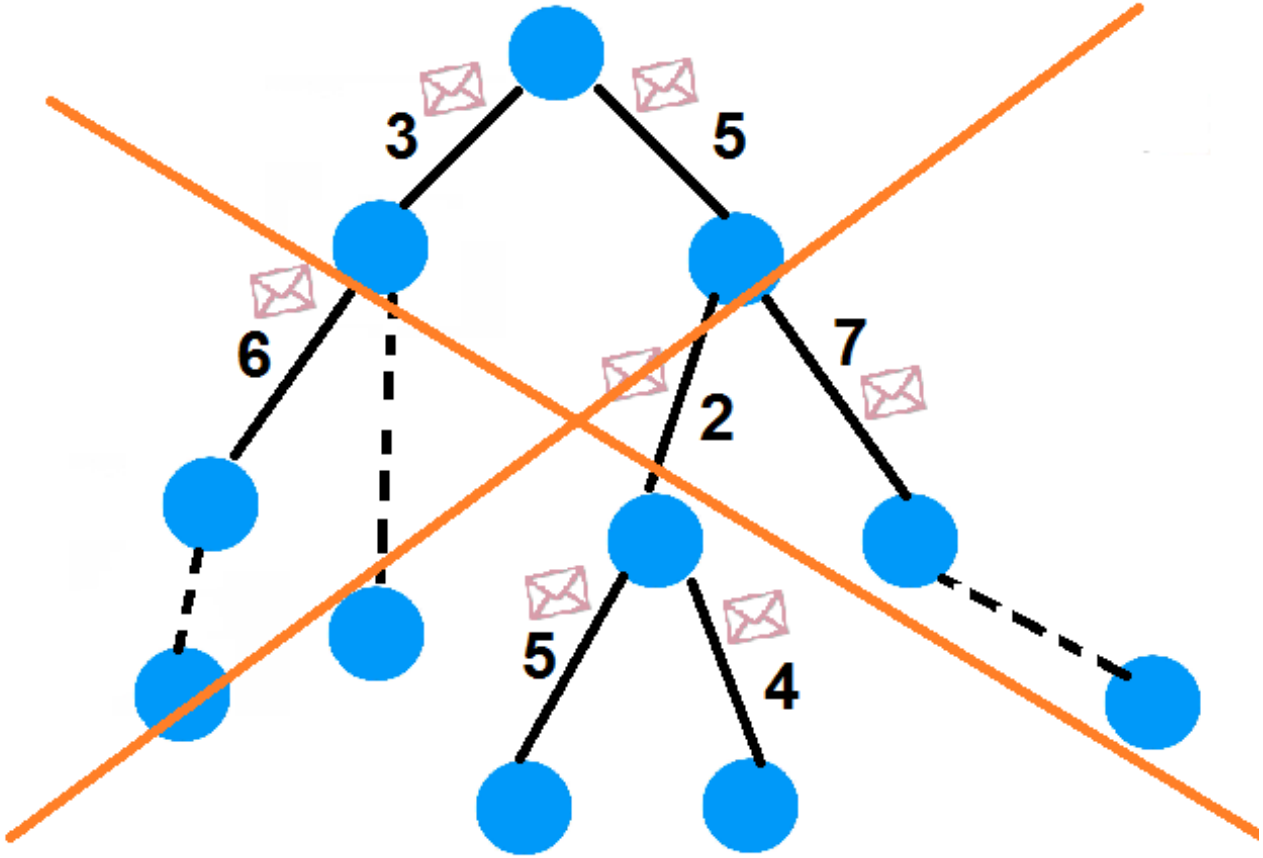
Multicast on P2P systems 1/6

- file sharing applications
- data streaming protocols

- The target environment :

1) network performance is very dynamic
2) nodes can join and leave
3) available bandwidth between nodes can be asymmetric

Multicast on P2P systems 2/6



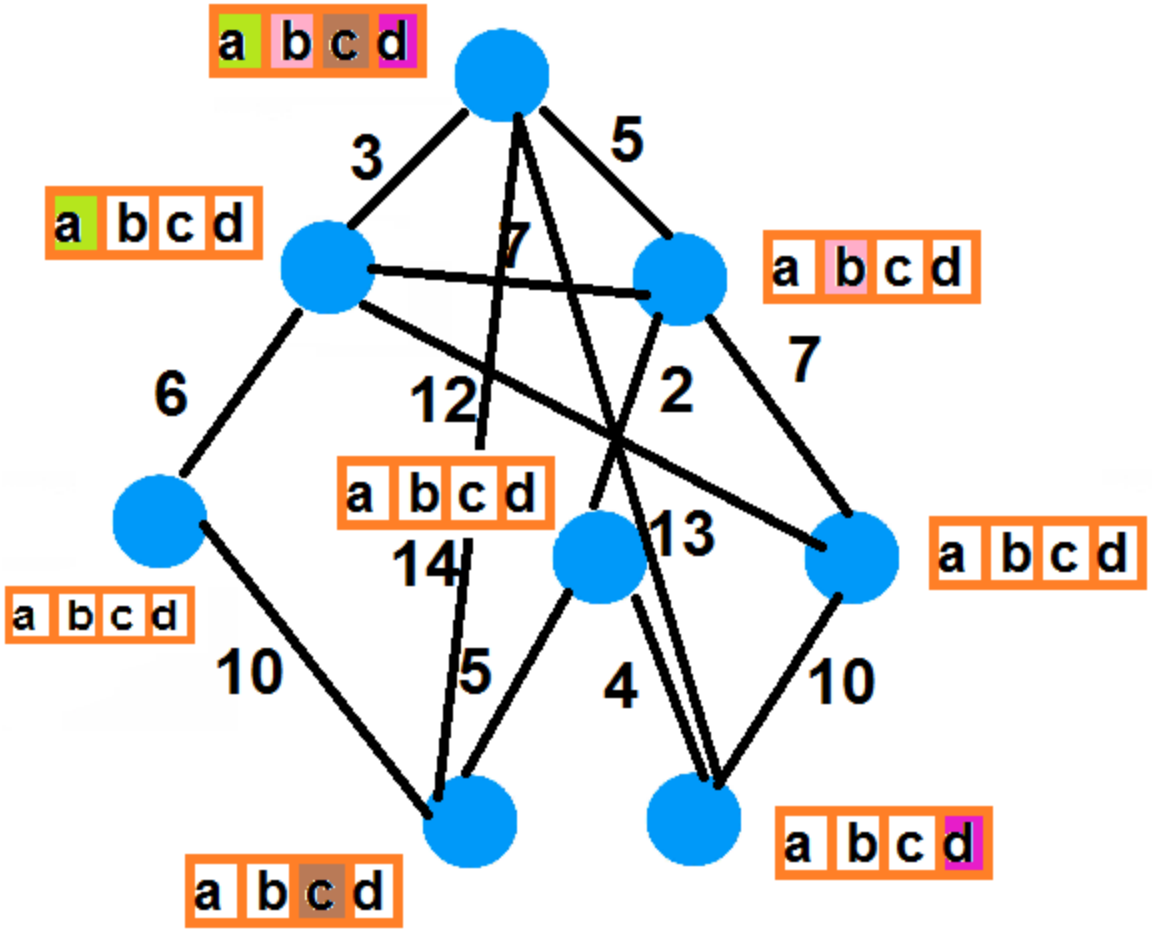
New nodes joined the topology

Using network topology is not a good idea

Multicast on P2P systems 3/6

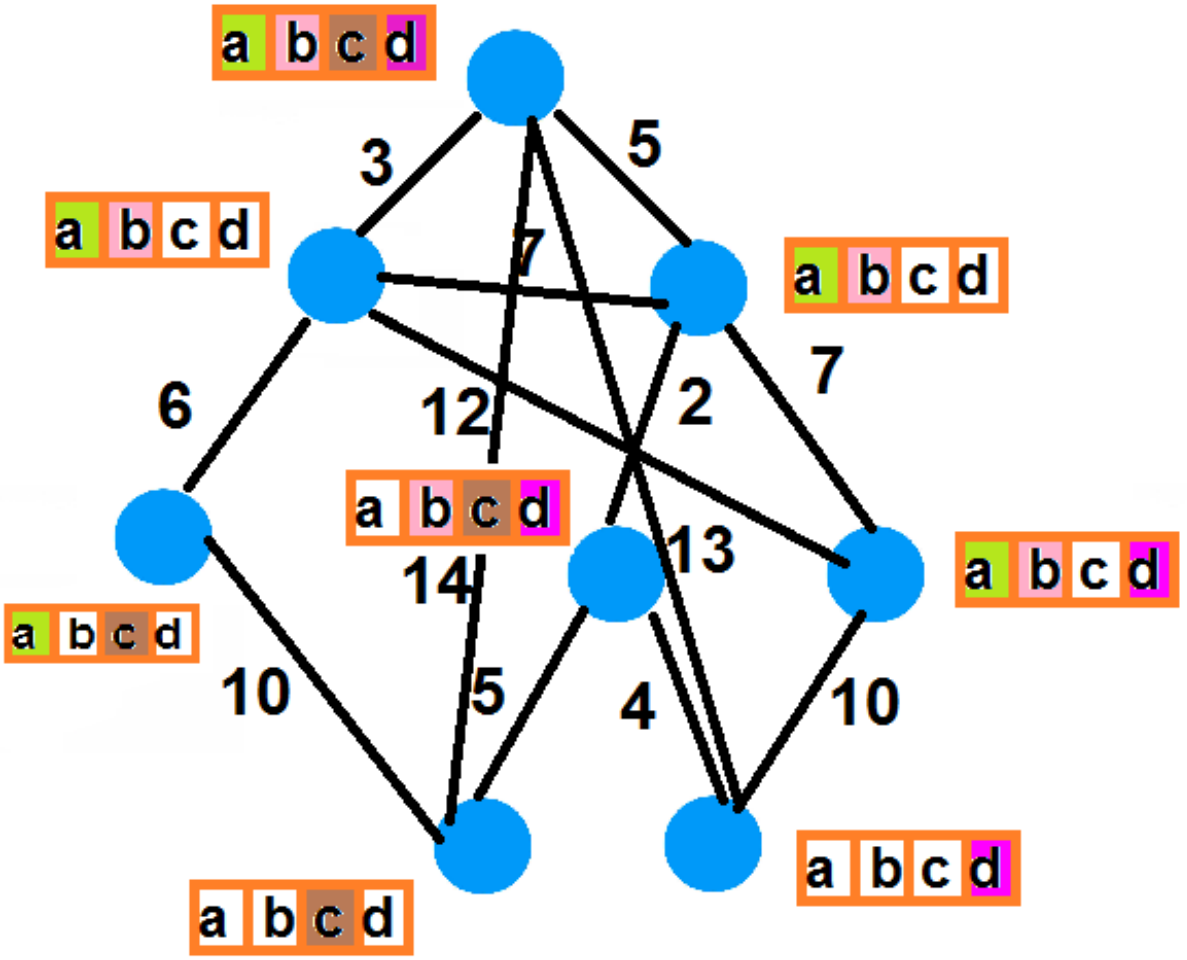
divide data into small pieces

exchanged it with neighbor nodes



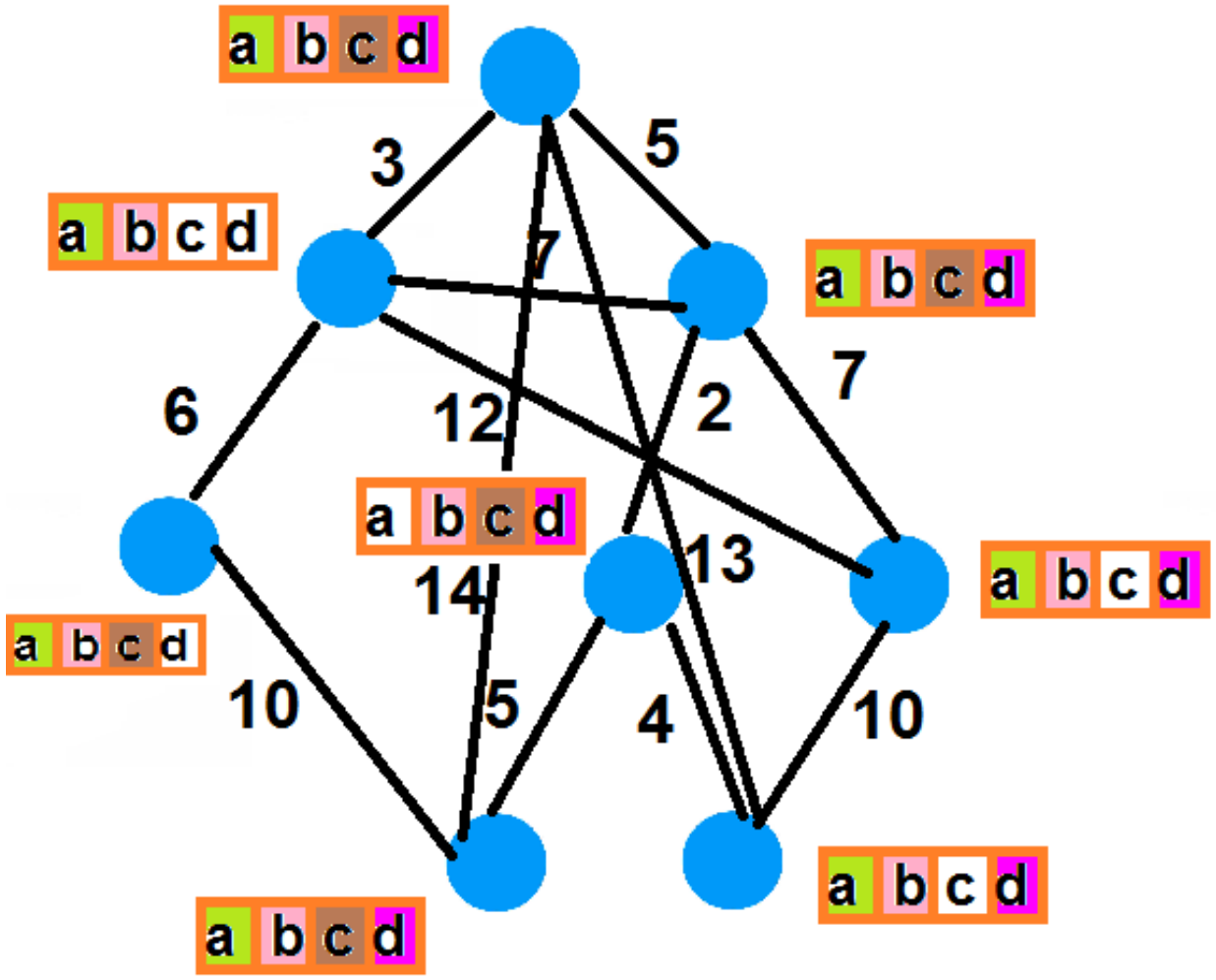
* **receiver-driven technique-> can improve throughput**

Multicast on P2P systems 4/6

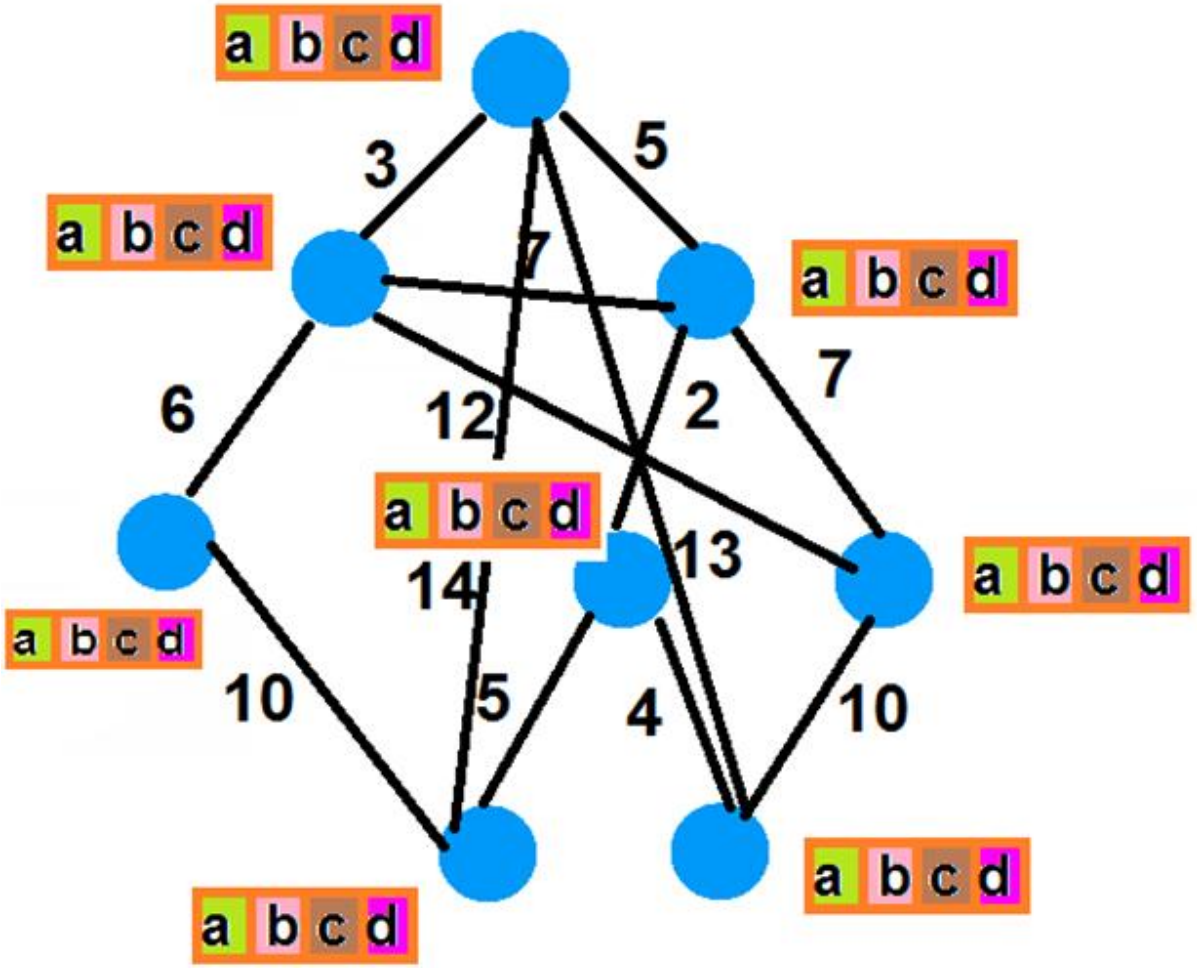


all nodes tell their neighbors which pieces they have and request pieces they lack

Multicast on P2P systems 5/6



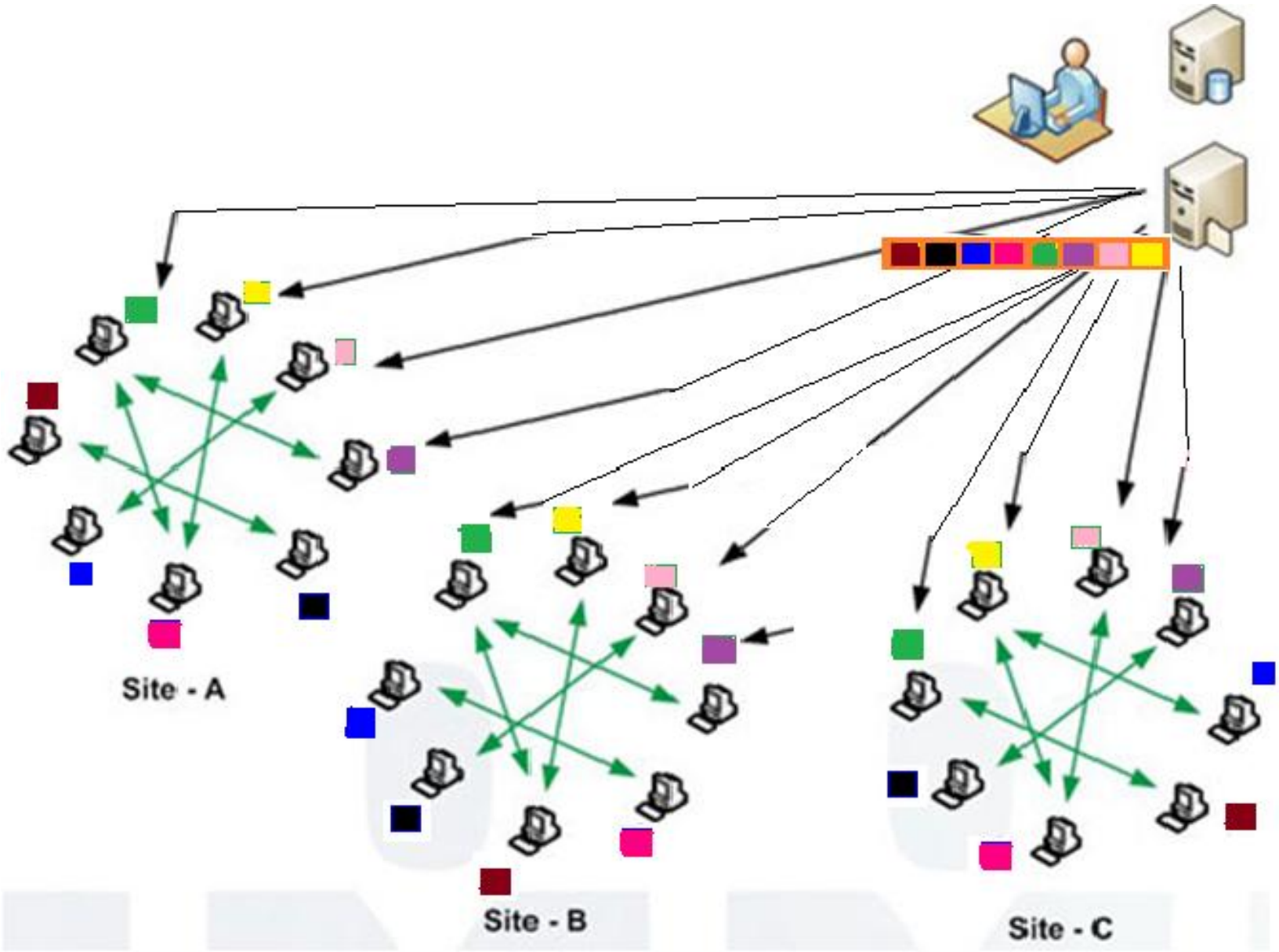
Multicast on P2P systems 6/6



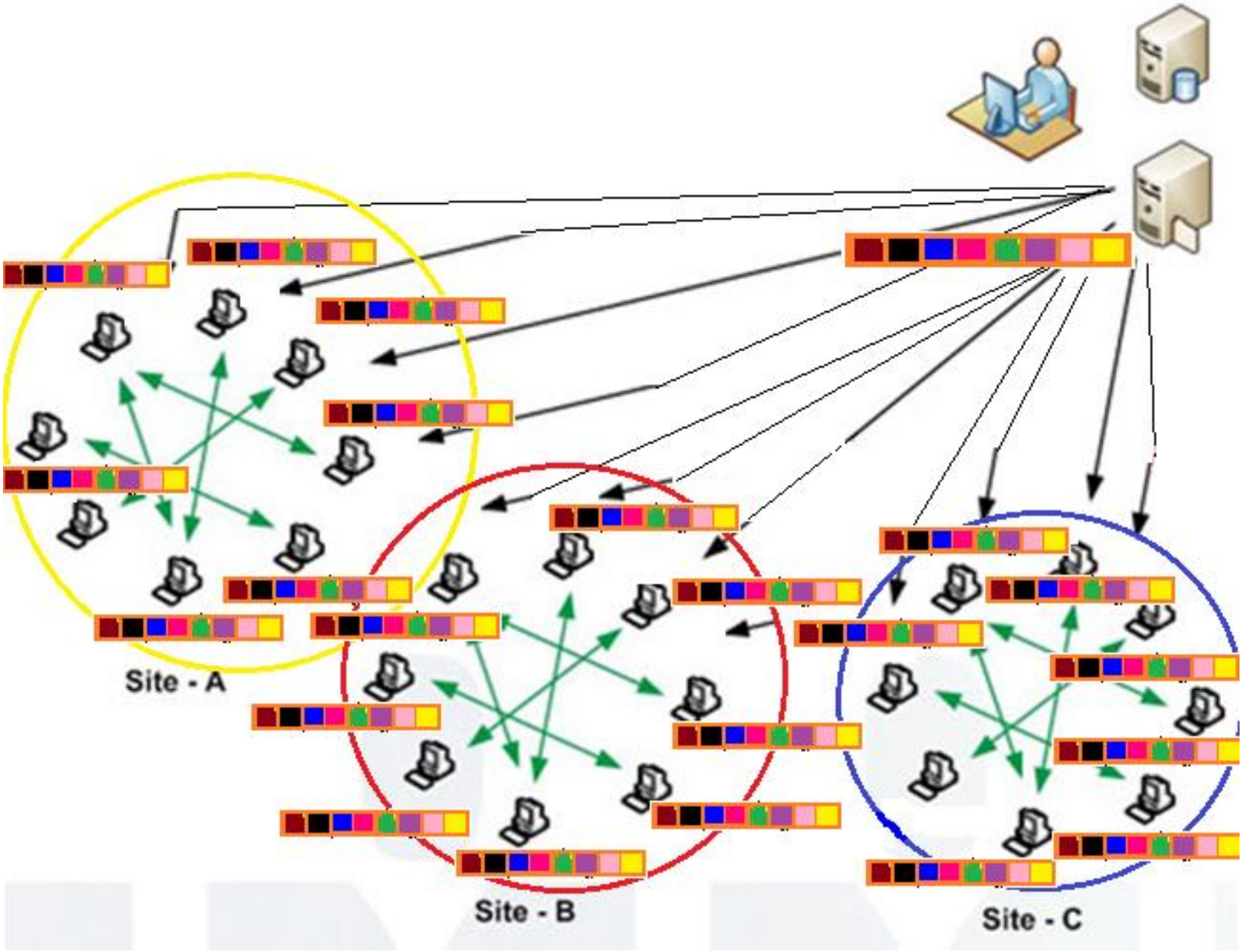
Example : Mob

- optimizes multicast communication between multiple homogeneous clusters connected via a WAN.

- based on BitTorrent, and takes node locality into account to reduce the number of wide-area transfers.



Each node in a mob steals an equal part of all data from peers in remote clusters



Each node in a mob distributes the stolen pieces locally

Contents:

- Introduction
- Multicast on parallel distributed systems
- Multicast on P2P systems
- Multicast on clouds
- High performance multicast algorithms
- Implementation
- Performance Evaluation
- Summary
- Conclusions

Cloud systems 1/2

- are based on large clusters in which nodes are densely connected
- storage resources provided by clouds are fully or partly virtualized.
- -> A multicast algorithm for clouds can therefore not assume anything about the exact physical infrastructure

Cloud systems 2/2

- the uplink and downlink of a virtual compute node can be affected by other virtual compute nodes that are running on the same physical host.

- routing changes and load balancing will also affect network performance.

- the data to distribute is often located in the cloud's storage service, which introduces another potential shared bottleneck

Cloud platform used in our experiments
Amazon EC2/S3

- Amazon **Elastic Compute Cloud** (EC2)

- **Simple Storage Service** (S3)

- EC2 provides **virtualized computational resources** that can range from one to hundreds of nodes as needed.

- **virtual machines** on top of **Xen are** called **instances**

- Xen hypervisor- open source standard for virtualization

Amazon EC2/S3

- **S3** stores files as **objects** in a **unique name space**, called a **bucket**.

- Buckets have to be created before putting objects into S3. They have a location and an arbitrary but globally unique name.

- The size of objects in a bucket can currently range from 1 byte to 5 gigabytes.

Multicast on clouds

Requirements that multicast operations on clouds should fulfill:

- maximized utilization of available aggregate download throughput from cloud storage
- minimization of multicast completion time of each node
- non-dependence on monitoring network throughput nor estimation of network topology

Contents:

- Introduction
- Multicast on parallel distributed systems
- Multicast on P2P systems
- Multicast on clouds
- High performance multicast algorithms
- Implementation
- Performance Evaluation
- Summary
- Conclusions

High performance multicast algorithms

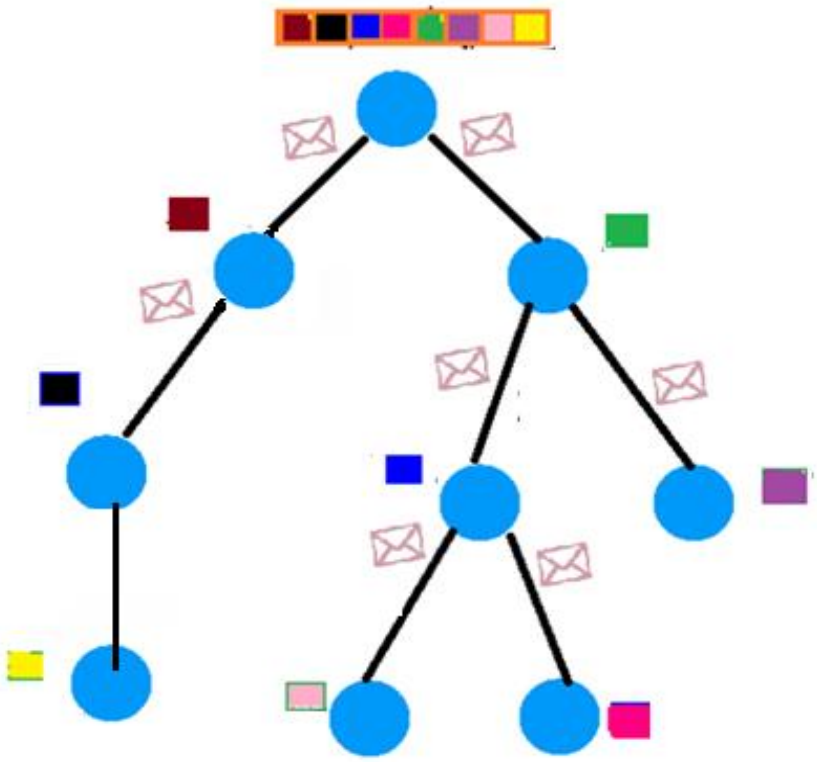
Features of our algorithms :

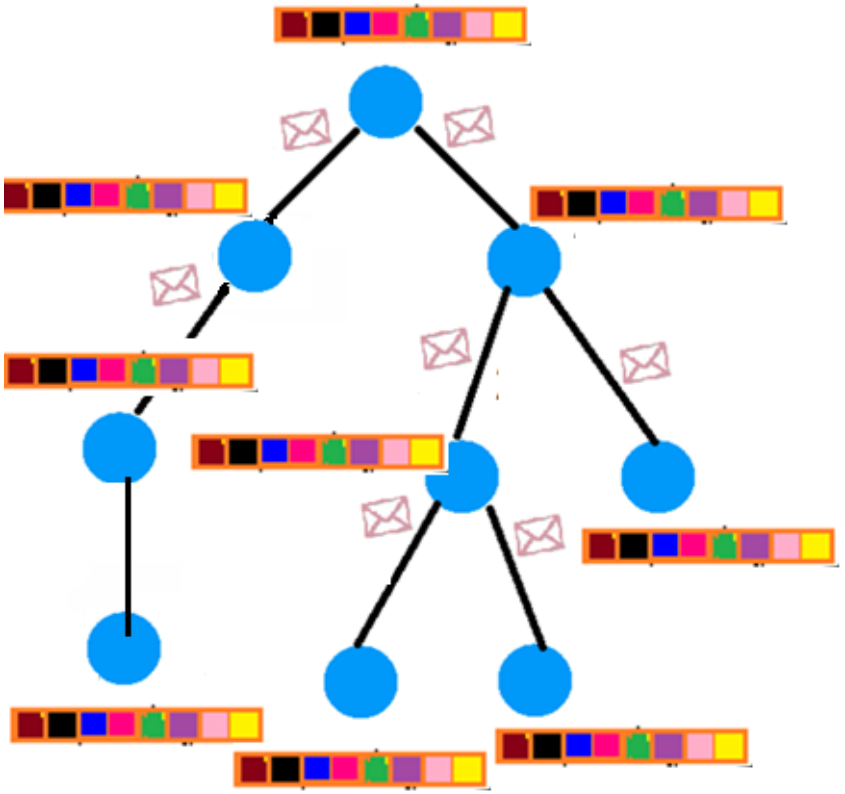
- construct an overlay network on clouds without network topology information;
- optimize the total throughput dynamically;
- increase the download throughput by letting nodes cooperate with each other.

How they work ?

- divide the data to download from the cloud storage service over all nodes
- exchanges the data via a mesh overlay network
- the first ‘non-steal’ algorithm lets each node download an equal share of all data
- the second ‘steal’ algorithm uses work stealing to counter the effect of heterogeneous download bandwidth.

Van de Geijn ’non-steal’ algorithm

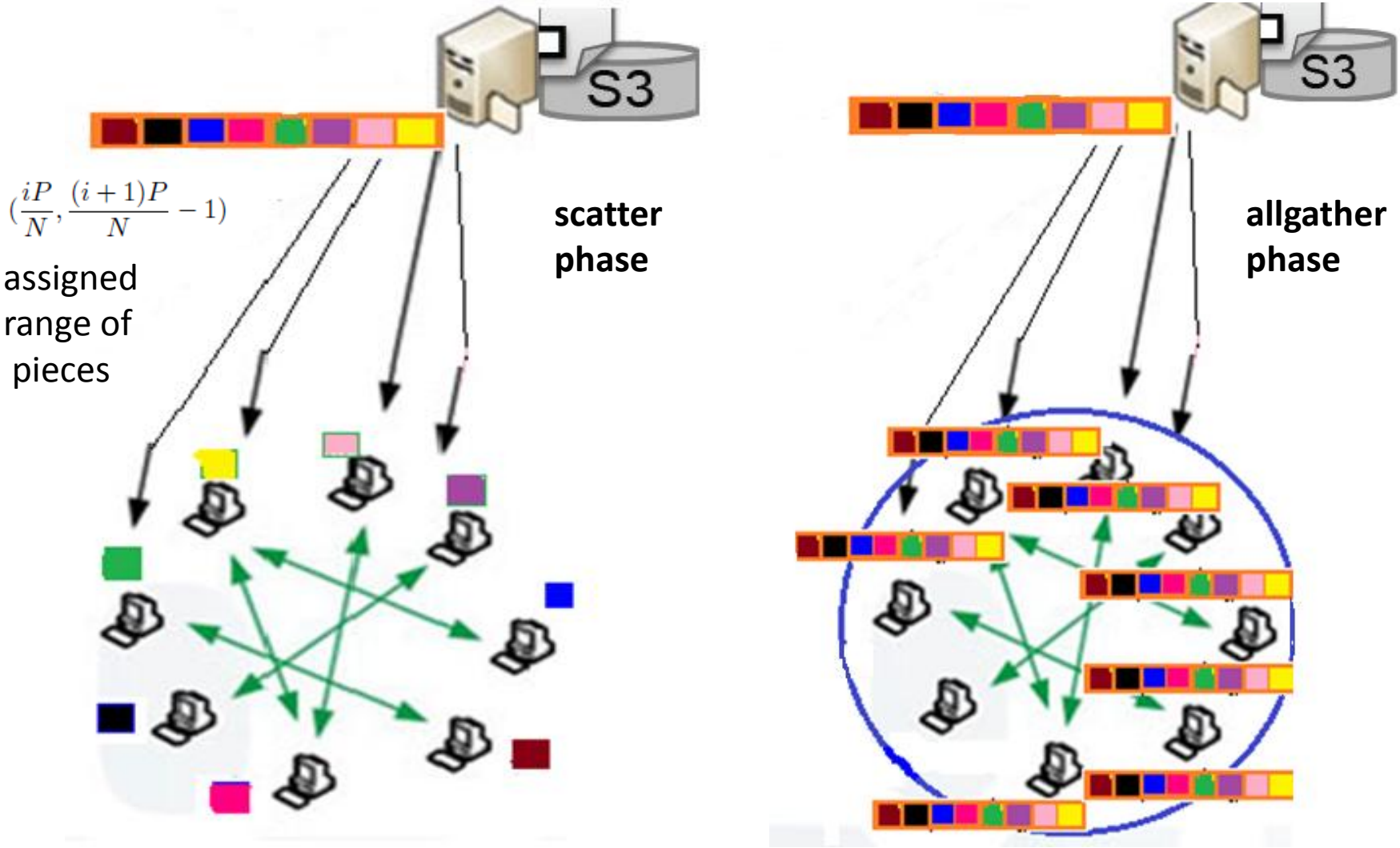




Scatter phase: the root node divides the data to be multicast into blocks of equal size depending on the number of nodes. Blocks are then send to each corresponding node using a binomial tree.

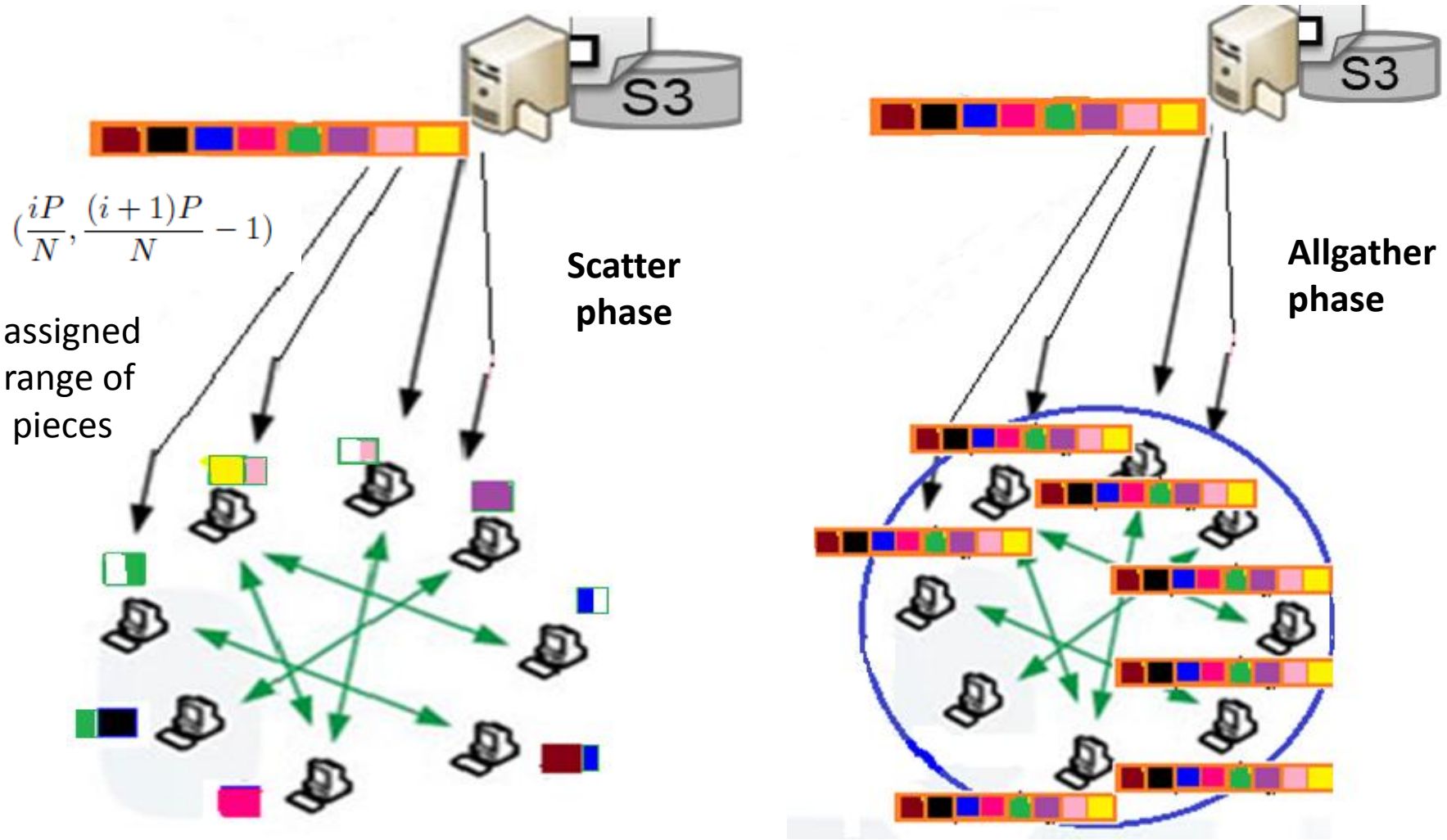
Allgather phase : the missing blocks are exchanged and collected by using the recursive doubling technique

The ’non-steal’ algorithm



Once a node finishes downloading the assigned range of pieces, it waits until all the other nodes have finished too and then exchanges information with its neighbors in the mesh.

The ’steal’ algorithm

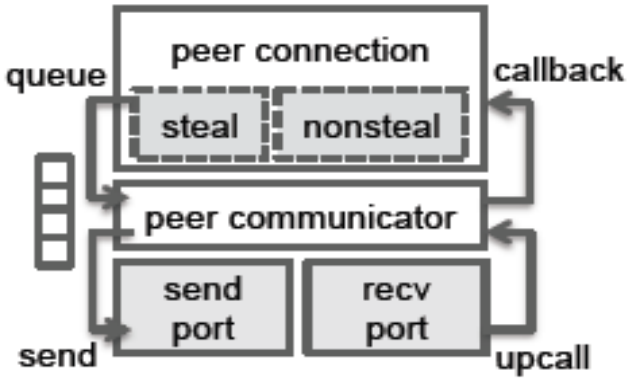


When a node has finished downloading its pieces, it asks other nodes if they have any work remaining. The amount of work download is proportional to download bandwidth.

Contents:

- Introduction
- Multicast on parallel distributed systems
- Multicast on P2P systems
- Multicast on clouds
- High performance multicast algorithms
- Implementation
- Performance Evaluation
- Summary
- Conclusions

Implementation



- On top of **Ibis** , a Java-based grid programming environment.
- Each node is notified of the identity of new nodes that join the Ibis network, as well as its unique rank assigned by Ibis.
- The steal and non-steal algorithms are implemented as an extension to the **S3MulticastChannel object**.
- S3MulticastChannel manages an S3Connection object and a number of PeerConnection objects.
- The S3Connection object downloads pieces from S3, while the PeerConnection objects send and receive messages to and from other nodes.

Contents:

- Introduction
- Multicast on parallel distributed systems
- Multicast on P2P systems
- Multicast on clouds
- High performance multicast algorithms
- Implementation
- Performance Evaluation
- Summary
- Conclusions

PERFORMANCE EVALUATION 1/3

- We compared the multicast performance of the **flat tree**, **nonsteal** and **steal** algorithms on **Amazon EC2/S3**.

- We only use EC2/S3 sites located in Europe, with a small EC2 instance type.

- piece size = 32KB

PERFORMANCE EVALUATION 2/3

The best completion time is similar across all algorithms: about **100 s**

The slowest completion time:
*at least one node in the flat tree takes approximately 200s
*but none of the nodes in either the non-steal or steal algorithms takes more than 140s

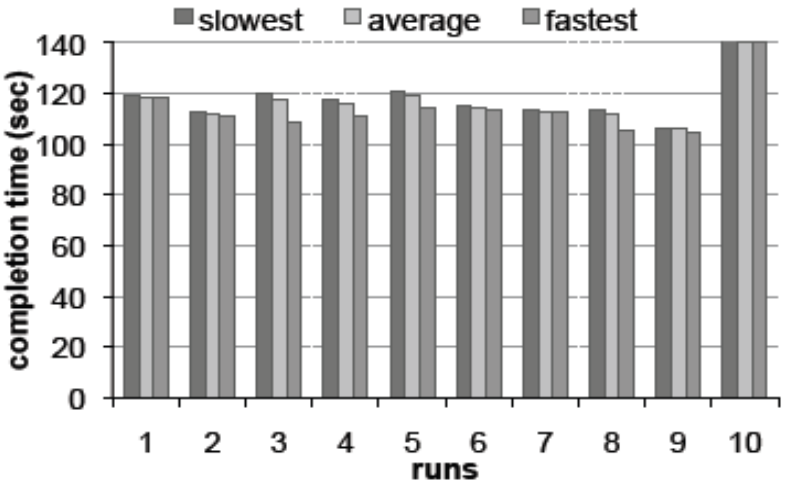


Fig. 8. completion time with non-steal algorithm (1GB, 8 nodes)

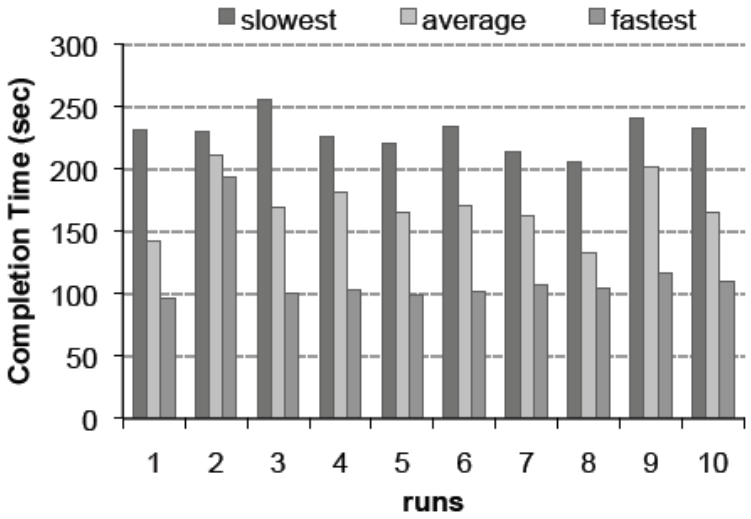


Fig. 2. download completion time with flat tree algorithm (8 nodes, 1GB)

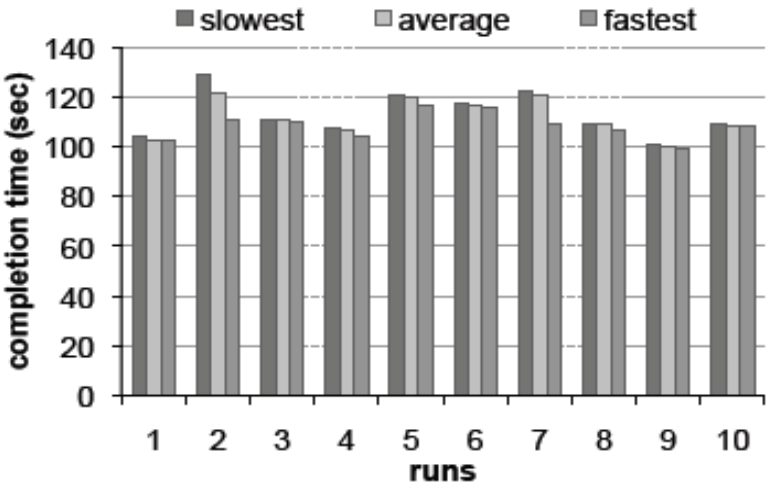
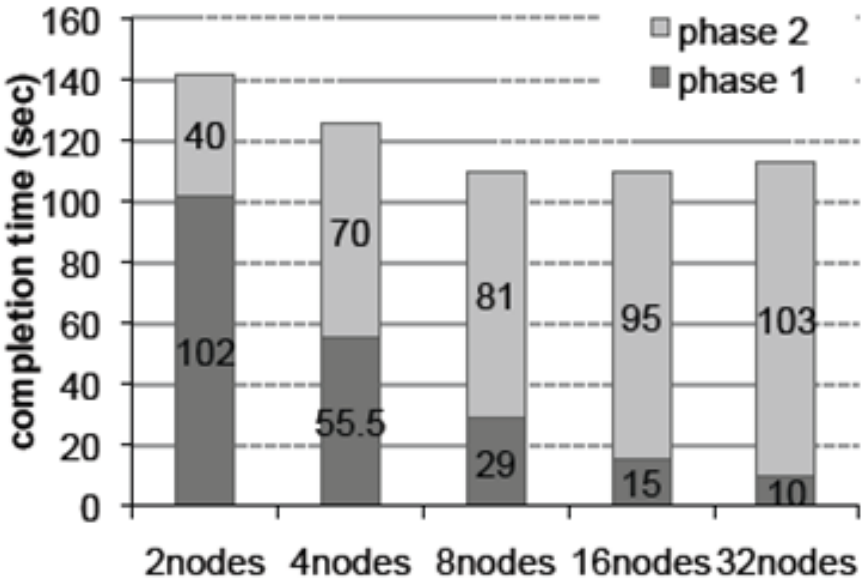


Fig. 9. completion time with steal algorithm (1GB, 8 nodes)

PERFORMANCE EVALUATION 3/3



12. completion time ratio of phase 1 and phase 2 (steal, 1GB)

->number of nodes increases
-> the time spent in phase 1 decrease
->time spent in phase 2 increases
->total time spent in phase 1 and phase 2 remains nearly constant

-> 32 nodes take slightly much time than 16 nodes.

Contents:

- Introduction
- Multicast on parallel distributed systems
- Multicast on P2P systems
- Multicast on clouds
- High performance multicast algorithms
- Implementation
- Performance Evaluation
- Summary
- Conclusions

Summary

- different multicast algorithms for distributed parallel and P2P systems.

- we have presented two high performance multicast algorithms.

- we have focused on Amazon EC2/S3

Contents:

- Introduction
- Multicast on parallel distributed systems
- Multicast on P2P systems
- Multicast on clouds
- High performance multicast algorithms
- Implementation
- Performance Evaluation
- Summary
- Conclusions

CONCLUSIONS

We evaluate our algorithms on EC2/S3, and show that they are scalable and consistently achieve high throughput.

As a result, all nodes can download files from S3 quickly, even when the network performance changes while the algorithm is running.

Take home message

As a best practice :

Use **High performance multicast algorithms** because the nodes cooperate with each other for getting the data from the cloud storage and the performance is increased.


Cloud
computing
service
software
edit
data
Google
Architecture
Amazon
Storage
systems
Provider
also
typically
computer
category
utility
Internet
including
delivery
users
Components
Platform
See
Web
Application
services
infrastructure
example
applications
Characteristics

Thank you !