Scalable and Efficient Data Management in Distributed Clouds: Service Provisioning and Data Processing

Jad Darrous

Advisors: Shadi Ibrahim and Christian Perez

AVALON/STACK, Inria, ENS de Lyon, LIP

Thesis defense
December 17 2019 ENS de Lyon

INVENTEURS DU MONDE NUMÉRIQUE

UNIVERSITÉ DE LYON

ENS DE LYON

The era of Big Data

350M photos are uploaded every day to Facebook[1]

500 hours of video are uploaded to YouTube every minute[2]

CERN recorded over 300 Petabytes of physics data[3]

[1] Facebook Users Are Uploading 350 Million New Photos Each Day, https://www.businessinsider.fr/us/facebook-350-million-photos-each-day-2013-9

[2] More Than 500 Hours Of Content Are Now Being Uploaded To YouTube Every Minute, https://www.tubefilter.com/2019/05/07/number-hours-video-uploaded-to-youtube-per-minute/

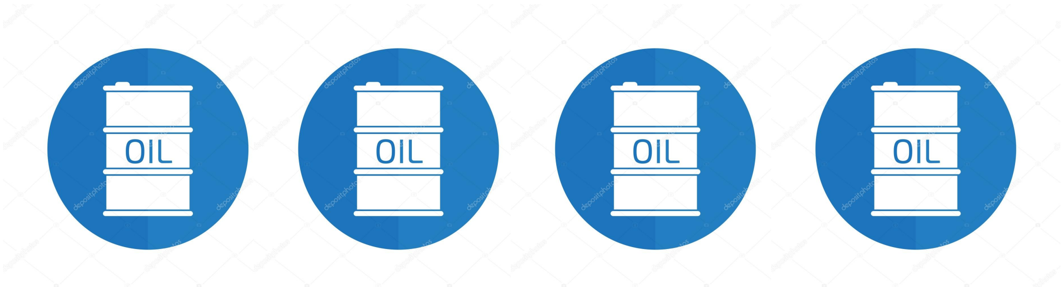
[3] Data preservation at CERN, https://home.cern/science/computing/data-preservation

The era of Big Data

350M photos are uploaded every day to Facebook[1]

500 hours of video are uploaded to YouTube every minute[2]

CERN recorded over 300 Petabytes of physics data[3]



[1] Facebook Users Are Uploading 350 Million New Photos Each Day, https://www.businessinsider.fr/us/facebook-350-million-photos-each-day-2013-9

[2] More Than 500 Hours Of Content Are Now Being Uploaded To YouTube Every Minute, https://www.tubefilter.com/2019/05/07/number-hours-video-uploaded-to-youtube-per-minute/

[3] Data preservation at CERN, https://home.cern/science/computing/data-preservation

The era of Big Data

350M photos are uploaded every day to Facebook[1]

500 hours of video are uploaded to YouTube every minute[2]

CERN recorded over 300 Petabytes of physics data[3]

[1] Facebook Users Are Uploading 350 Million New Photos Each Day, https://www.businessinsider.fr/us/facebook-350-million-photos-each-day-2013-9

[2] More Than 500 Hours Of Content Are Now Being Uploaded To YouTube Every Minute, https://www.tubefilter.com/2019/05/07/number-hours-video-uploaded-to-youtube-per-minute/

[3] Data preservation at CERN, https://home.cern/science/computing/data-preservation

The era of Big Data

350M photos are uploaded every day to Facebook[1]

500 hours of video are uploaded to YouTube every minute[2]

CERN recorded over 300 Petabytes of physics data[3]

Large-scale infrastructures and scalable data management techniques

[1] Facebook Users Are Uploading 350 Million New Photos Each Day, https://www.businessinsider.fr/us/facebook-350-million-photos-each-day-2013-9
[2] More Than 500 Hours Of Content Are Now Being Uploaded To YouTube Every Minute, https://www.tubefilter.com/2019/05/07/number-hours-video-uploaded-to-youtube-per-minute/
[3] Data preservation at CERN, https://home.cern/science/computing/data-preservation

The reign of Clouds

- Scalability
- Ease of use
- Pay-as-you-go
- Elasticity
- No up-front cost

Cloud revenue increased from 26B in 2012 to 138B in 2017[1]

50% of the enterprises has cloud-first policy while 90% use cloud in some way[2]

[1] Public cloud market revenue worldwide from 2012 to 2027, https://www.statista.com/statistics/477702/public-cloud-vendor-revenue-forecast/

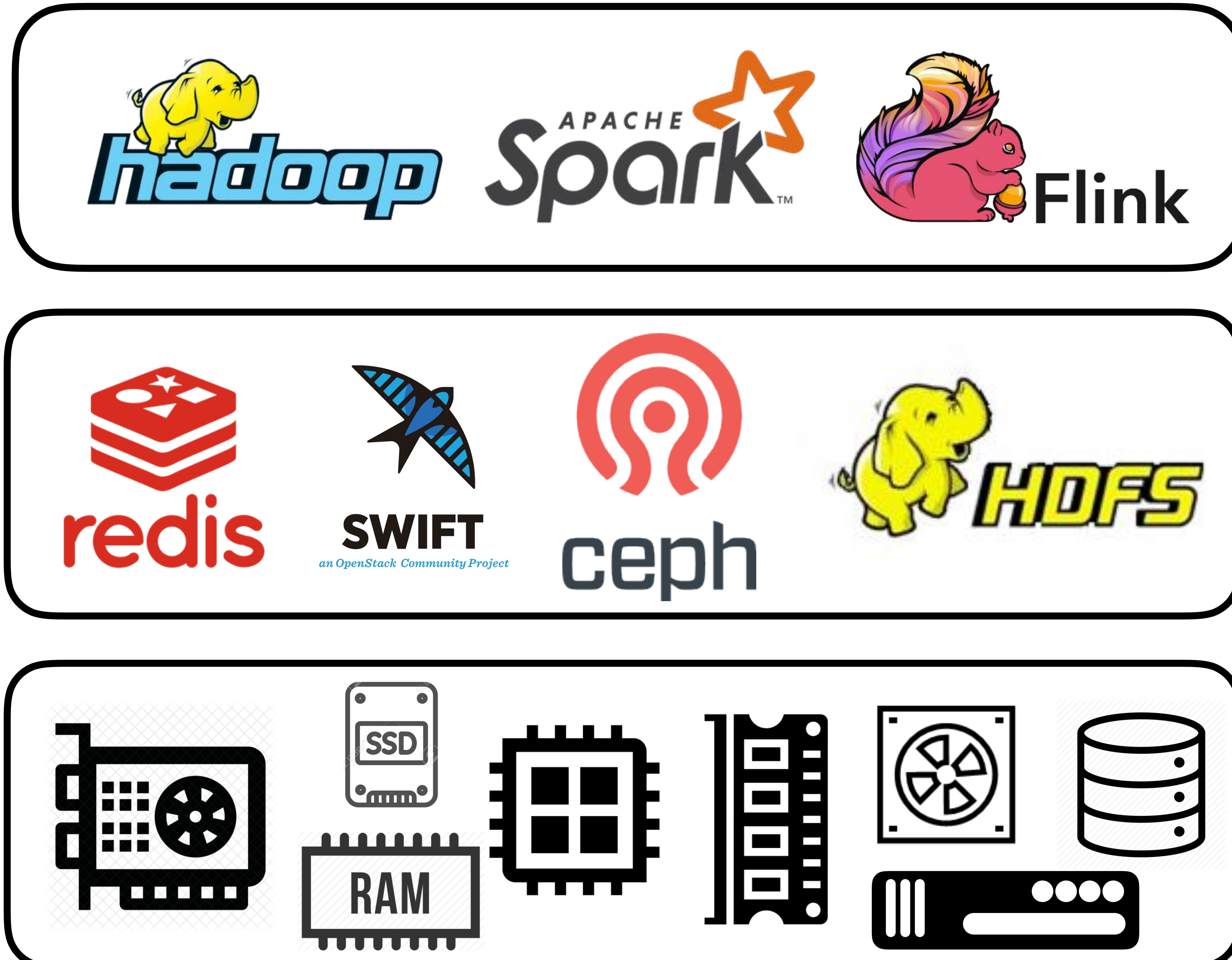
[2] 12 Must-Know Statistics on Cloud Usage, https://www.skyhighnetworks.com/cloud-security-blog/12-must-know-statistics-on-cloud-usage-in-the-enterprise/

The prevalence of Data Management Systems

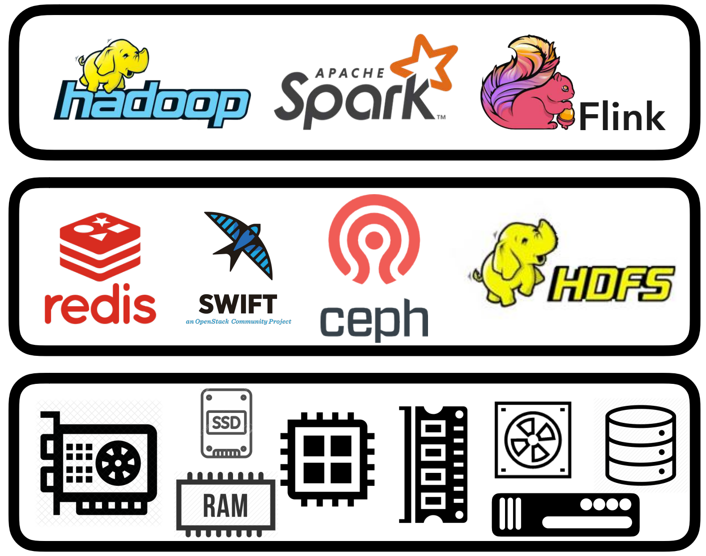
Analytics framework

Storage systems

Hardware



Scalable and Efficient Data Management in Distributed Clouds: Our Contributions



Service provisioning in geo-distributed clouds and Edge

- Network-aware Virtual Machine Image retrieval in geo-distributed clouds
- Network-aware container image placement in Edge

Cost-efficient Big Data processing under erasure coding (EC)

- Performance evaluation of Data-Intensive applications under EC
- Bringing EC-awareness to storage systems

1

Enabling Efficient Service Provisioning in Geo-distributed clouds and Edge environments

- Contribution 1: Optimising VMIs retrieval in heterogeneous WAN
- Contribution 2: Making Container image placement network-aware
- Summary

Clouds go Geo-distributed

wide area network (WAN)

Geo-distributed clouds

- Near the source of the data
- Physically closer to end-users
- Data regulation
- Catastrophic fault-tolerance
- Cheap electricity and free cooling

in 21 regions[1]

in 54 regions[2]

[1] AWS Global Infrastructure, https://aws.amazon.com/about-aws/global-infrastructure
[2] Windows Azure Regions, https://azure.microsoft.com/en-us/regions

Network heterogeneity as a Major Bottleneck for Service provisioning

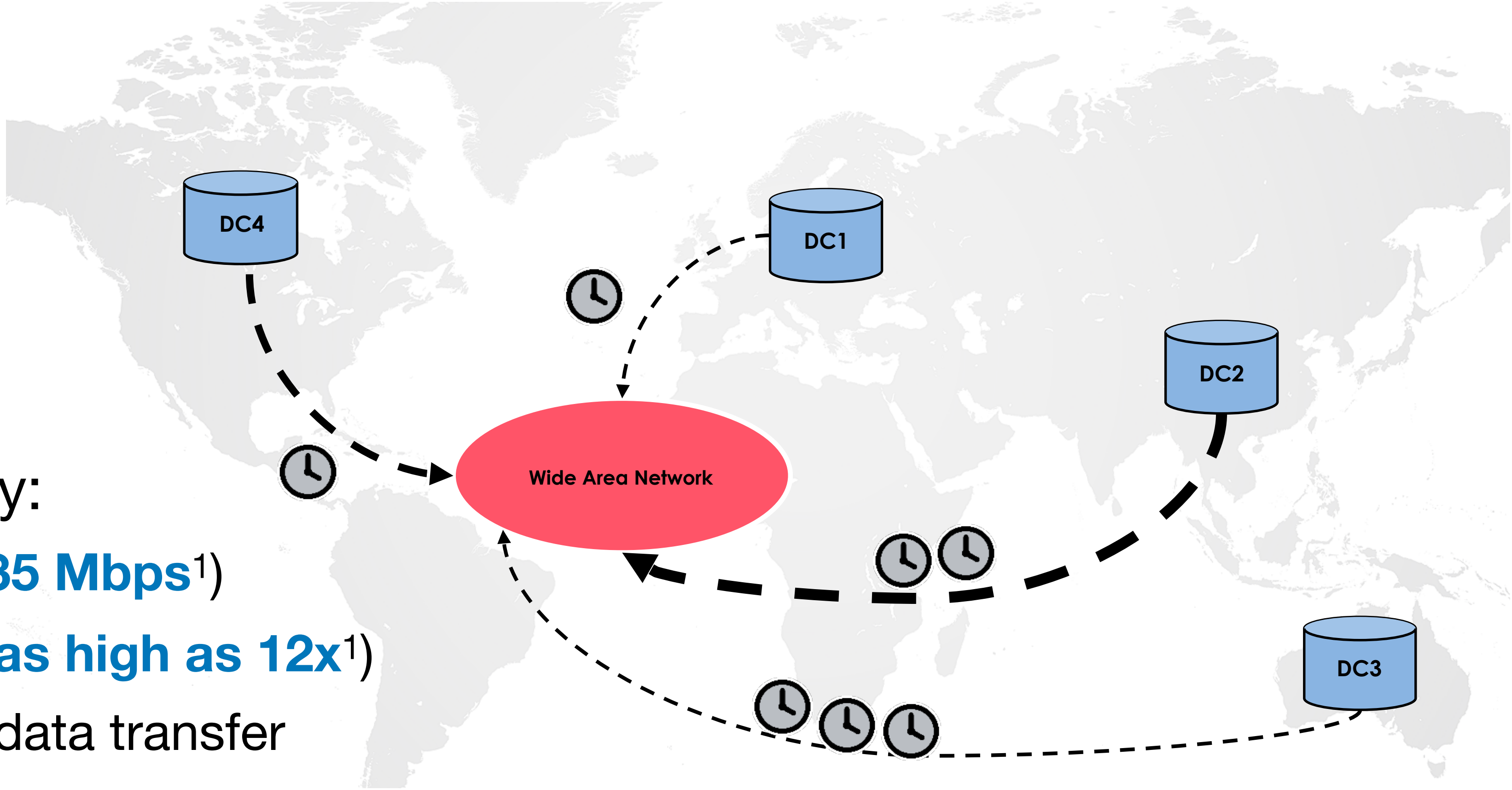
- By a service, we mean simply an application.
- Services in the cloud are deployed as *Virtual Machines* or *Containers*.
- A service image consists of the service program and its dependencies.

[1] Hsieh et al., Gaia: Geo-Distributed Machine Learning Approaching LAN Speeds, NSDI'17

Network heterogeneity as a Major Bottleneck for Service provisioning

- By a service, we mean simply an application.
- Services in the cloud are deployed as *Virtual Machines* or *Containers*.
- A service image consists of the service program and its dependencies.

Large in size (up to **tens of gigabytes**[1]) and increasing constantly in number (**20K** public images are hosted in AWS).



WAN links are characterized by:

- Low bandwidth (**as low as 35 Mbps**[1])
- Heterogeneous bandwidth (**as high as 12x**[1])
- Having a monetary cost for data transfer

[1] Hsieh et al., Gaia: Geo-Distributed Machine Learning Approaching LAN Speeds, NSDI'17

Nitro – Design Goals

Nitro is a VMI management system that focuses on minimizing the transfer time of VMIs over a heterogeneous WAN.

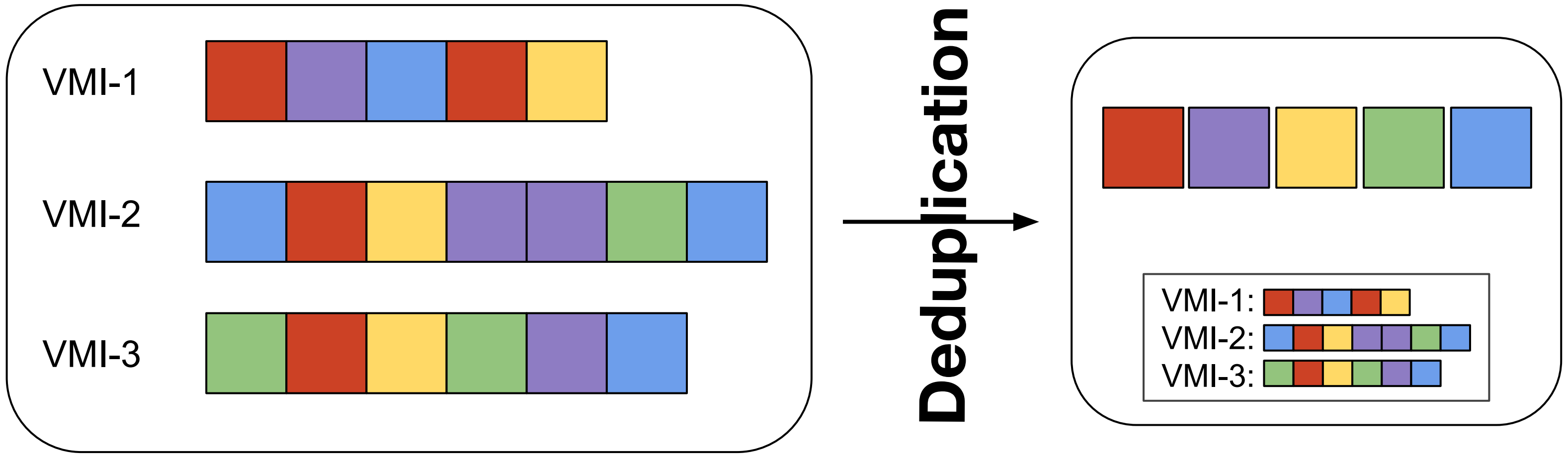
- Reduce network overhead
- Network-aware data retrieval
- Ensure minimal runtime overhead

Nitro – Design Goals

Nitro is a VMI management system that focuses on minimizing the transfer time of VMIs over a heterogeneous WAN.

- Reduce network overhead
  - VMIs are managed in small chunks to employ deduplication thus reducing the storage and network cost.

Original dataset
18 chunks



Deduplicated dataset
5 chunks + metadata

80% reduction in storage cost[1]

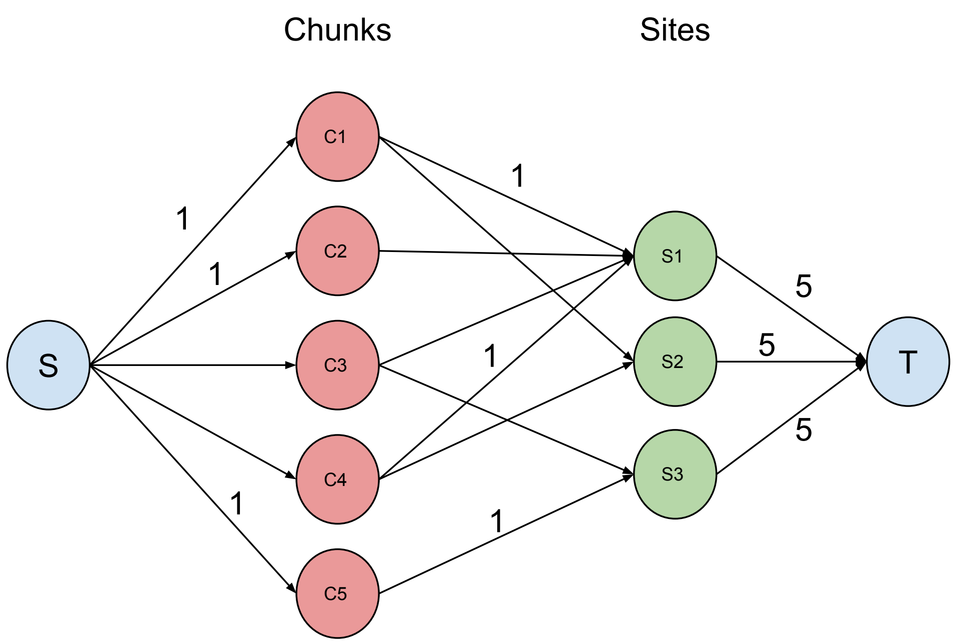
- Network-aware data retrieval
- Ensure minimal runtime overhead

[1] Jayaram et al., An Empirical Analysis of Similarity in Virtual Machine Images, Middleware'11

Nitro – Design Goals

Nitro is a VMI management system that focuses on minimizing the transfer time of VMIs over a heterogeneous WAN.

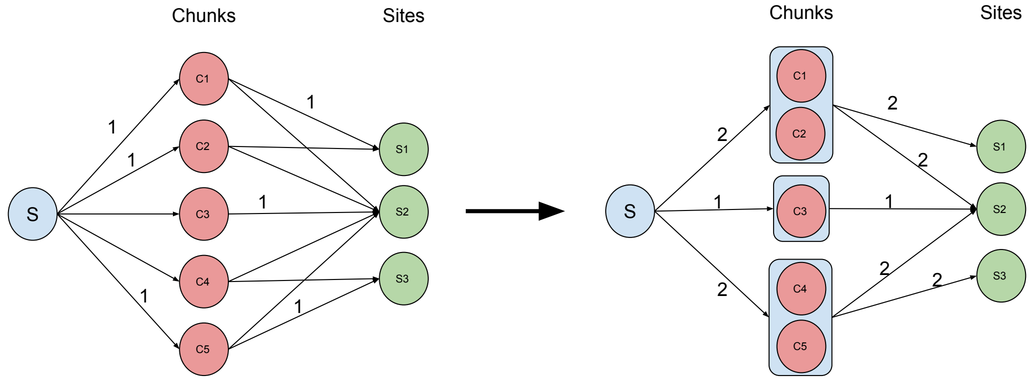
- Reduce network overhead
- Network-aware data retrieval
  - Network-aware data transfer strategy to effectively exploit links with high bandwidth.
  - Based on matching algorithm (max-flow algorithm) in bipartite graph.
  - Produces exact solution in polynomial time.
- Ensure minimal runtime overhead



Nitro – Design Goals

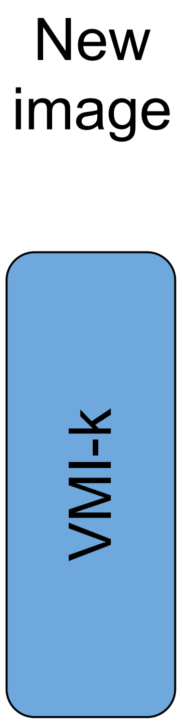
Nitro is a VMI management system that focuses on minimizing the transfer time of VMIs over a heterogeneous WAN.

- Reduce network overhead
- Network-aware data retrieval
- Ensure minimal runtime overhead
  - Optimize the running time of the scheduling algorithm; *Mega Chunks*: Group the chunks that can be found in the same set of sites into one chunk node.
  - Ensure sub-second runtime which allow the algorithm to run online.

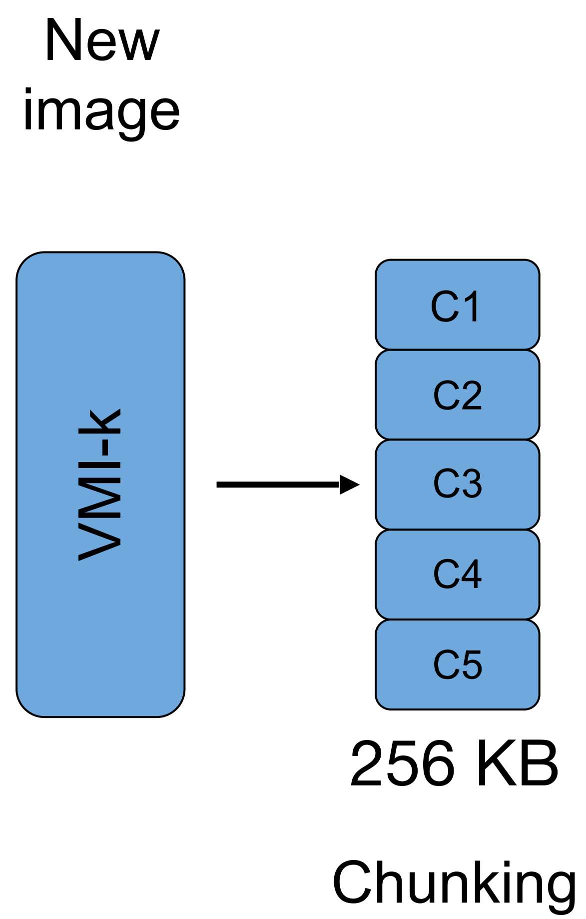


Adding new a VMI in Nitro

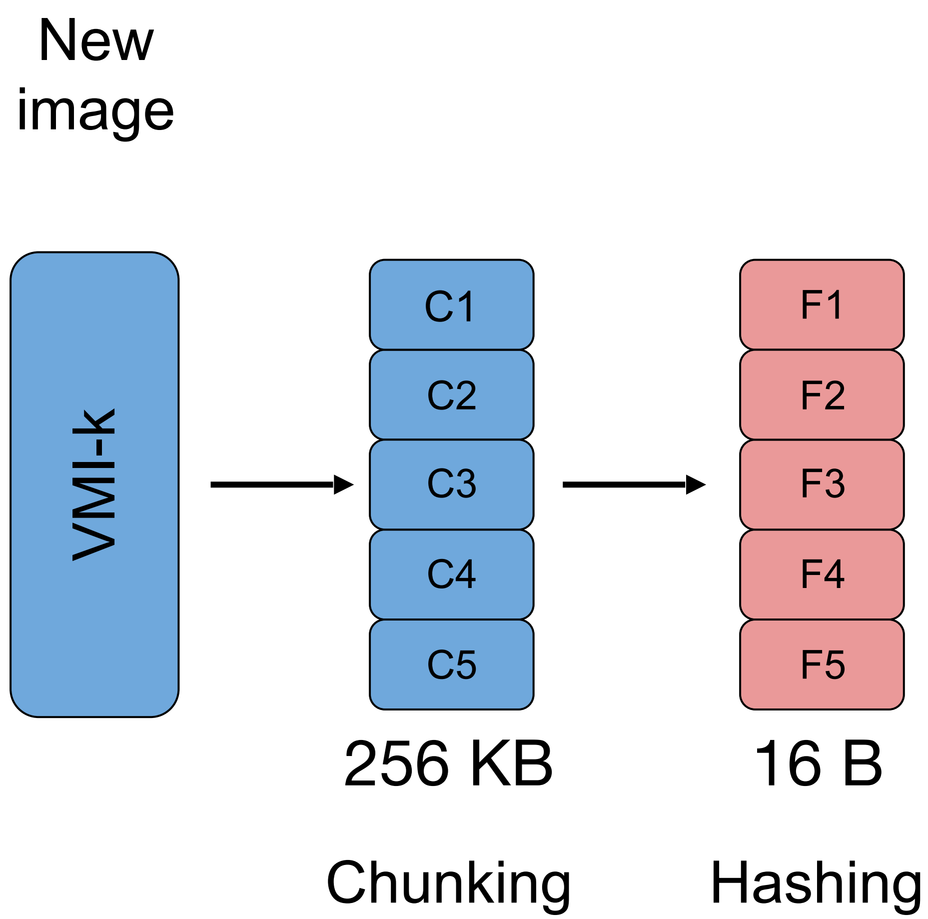
Adding new a VMI in Nitro



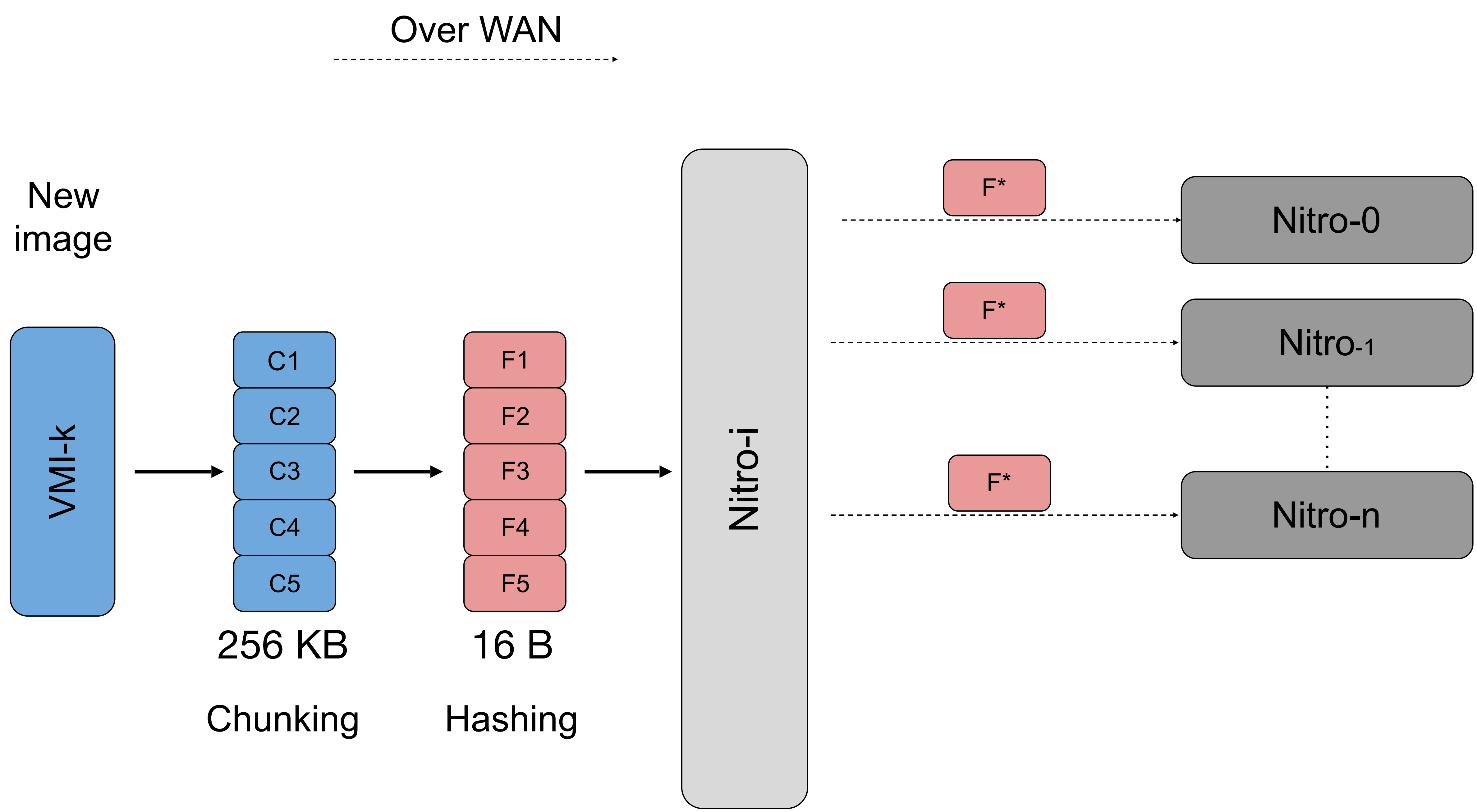
Adding new a VMI in Nitro



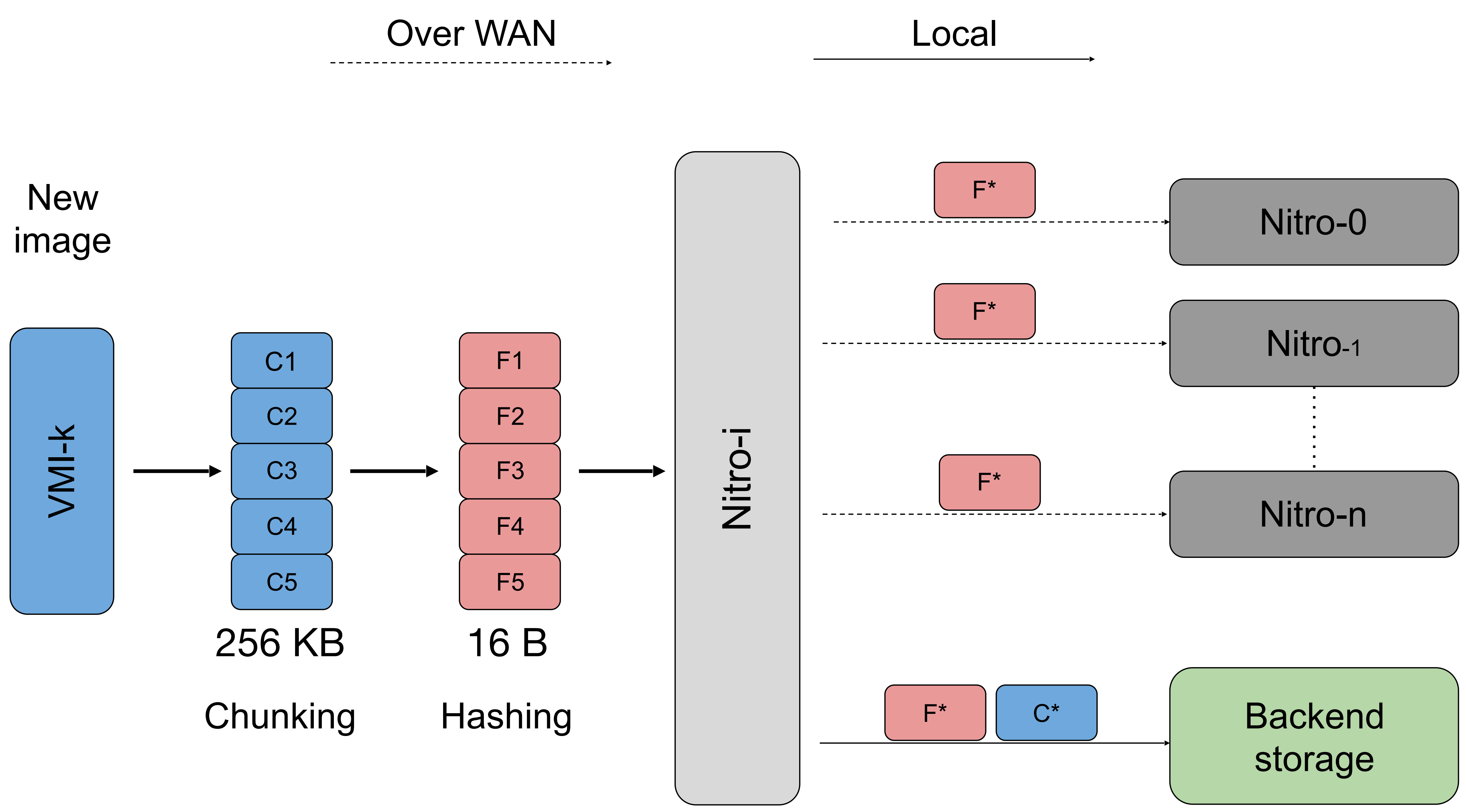
Adding new a VMI in Nitro



Adding new a VMI in Nitro

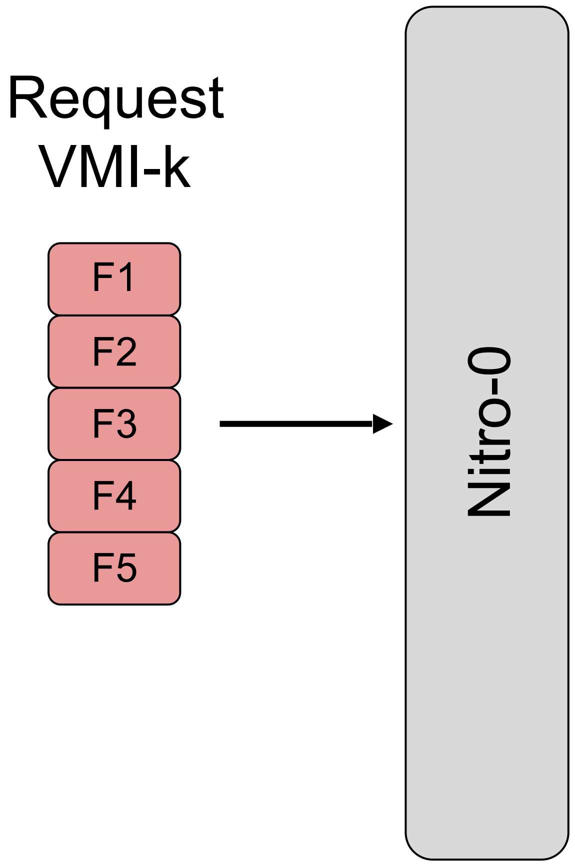


Adding new a VMI in Nitro

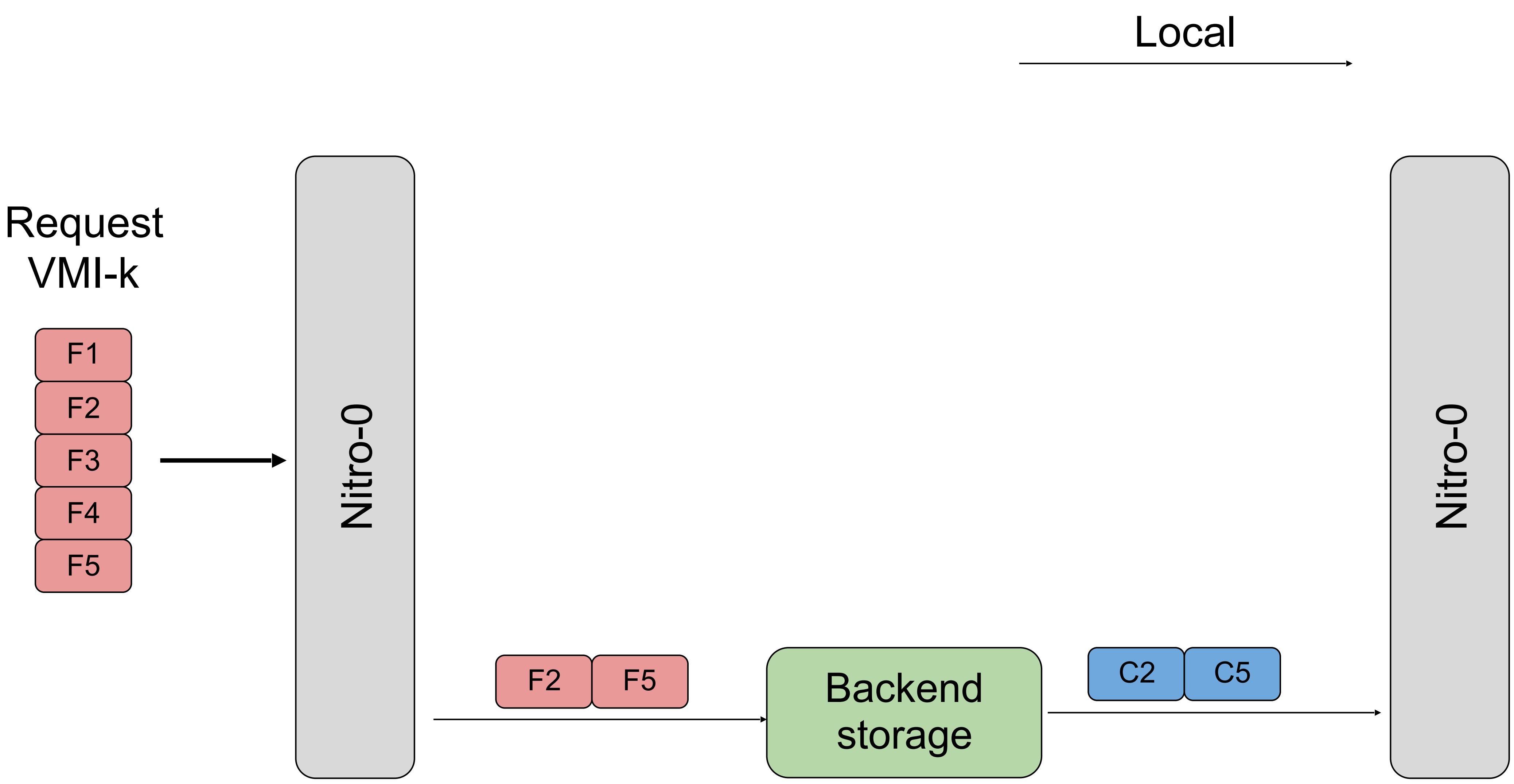


VM Provisioning workflow

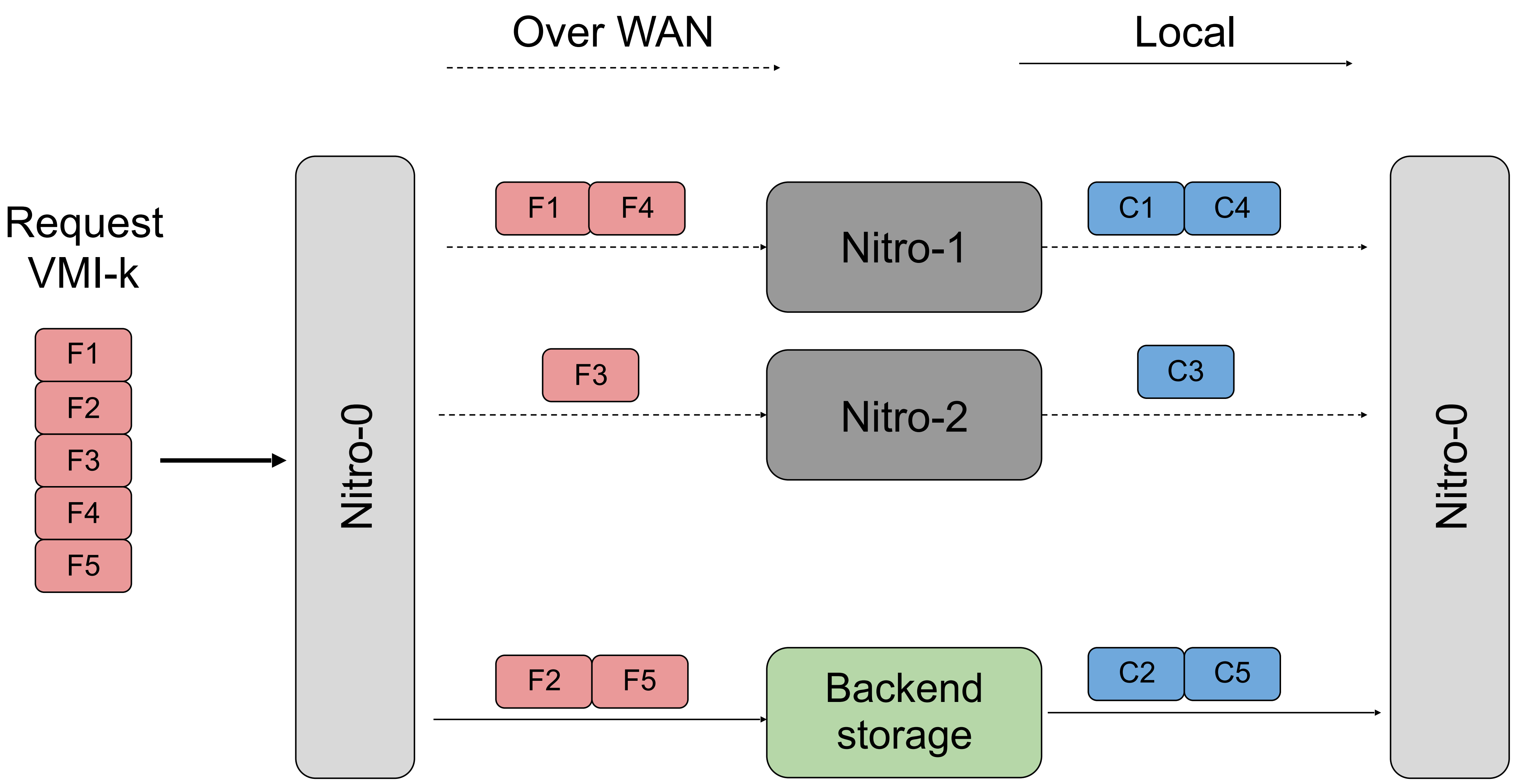
VM Provisioning workflow



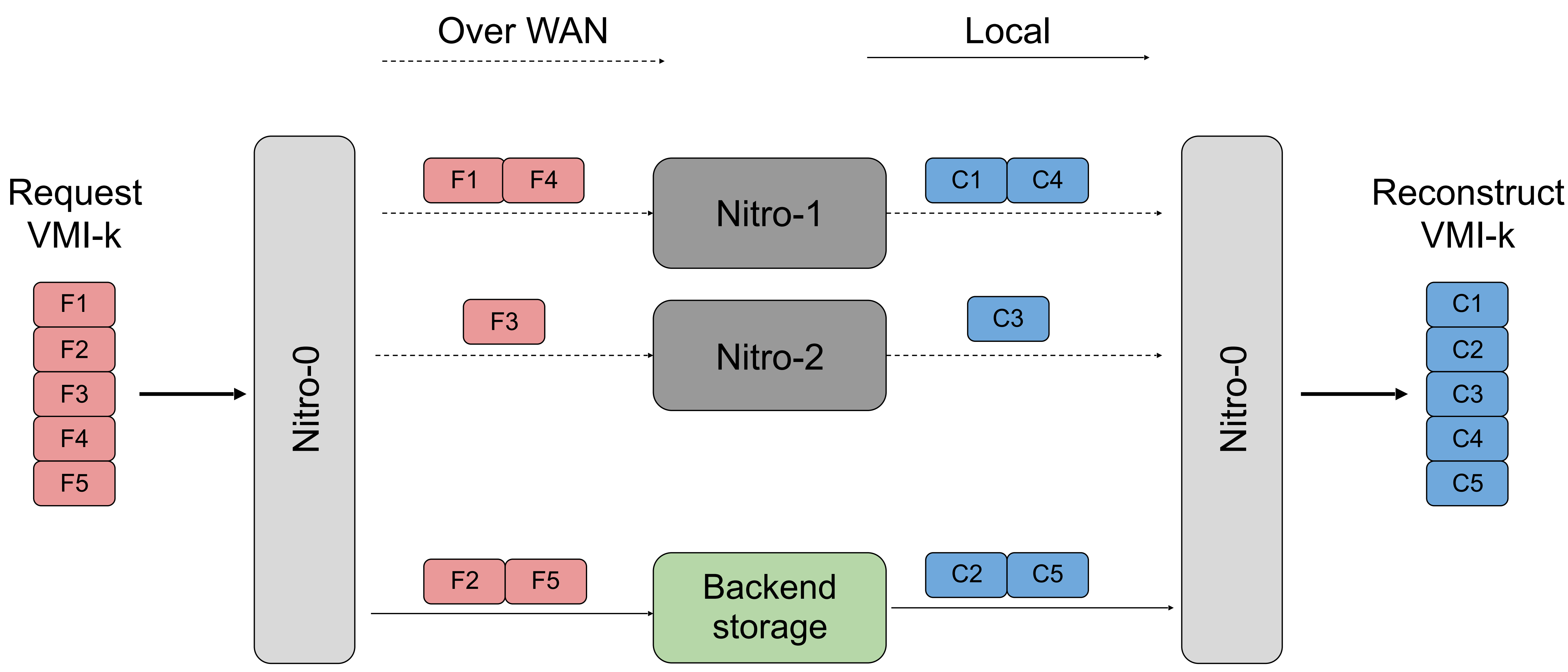
VM Provisioning workflow



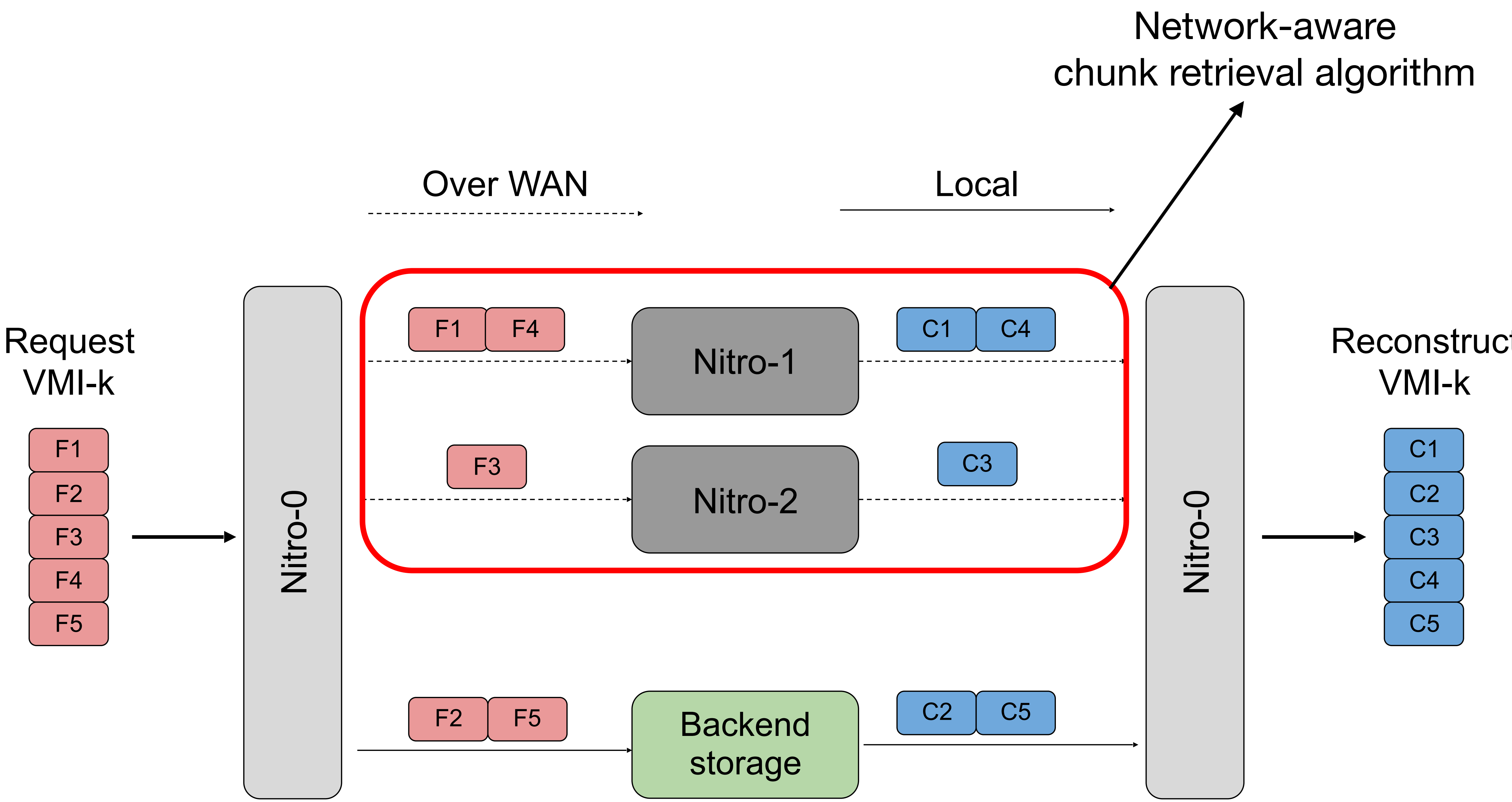
VM Provisioning workflow



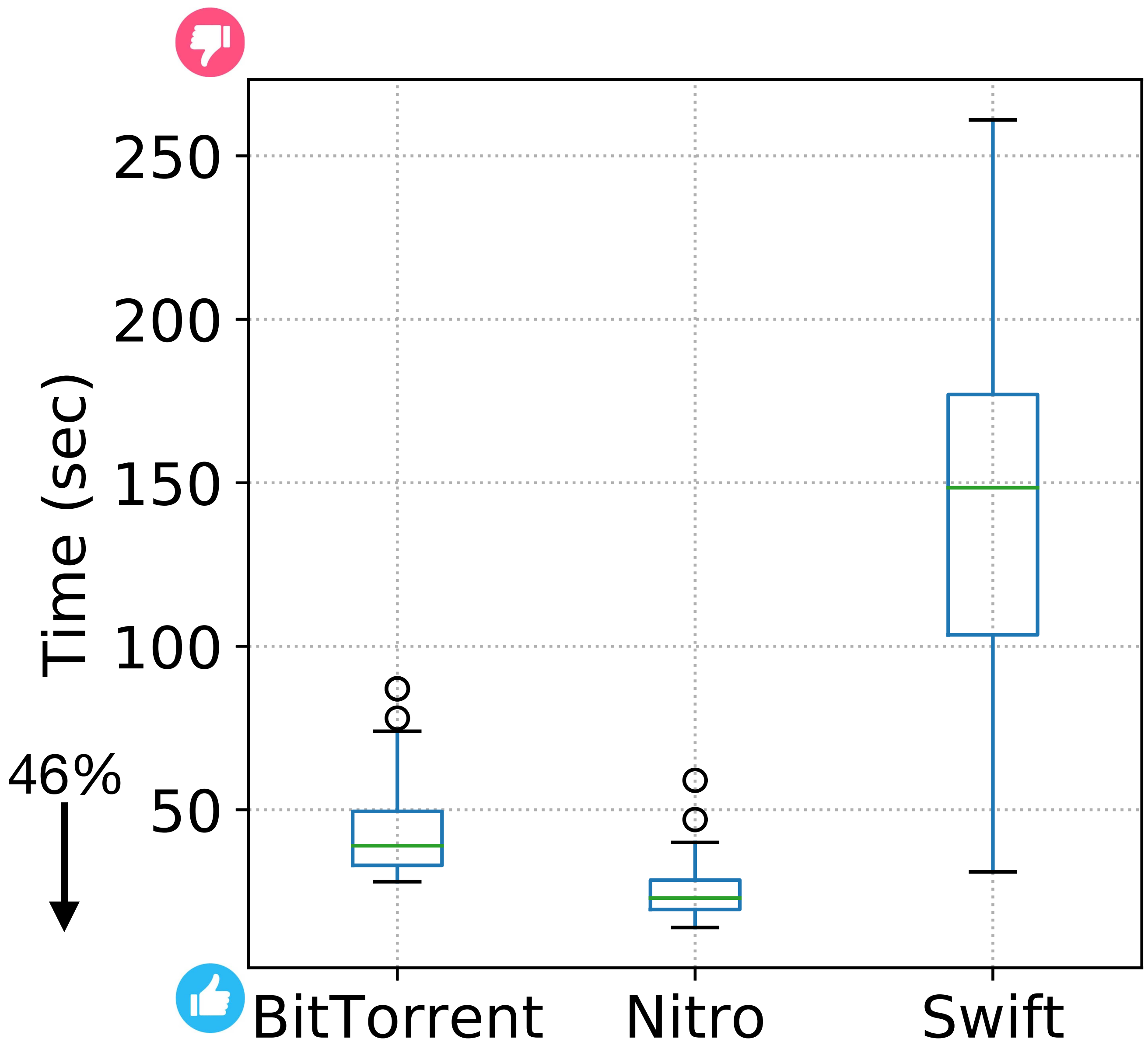
VM Provisioning workflow



VM Provisioning workflow



Results: The effectiveness of Nitro in reducing the VMs provisioning times



Grid'5000

CCGrid'18

Reduces the retrieval time by 77%

Methodology

Testbed: 11 machines of Nova cluster

Emulated network: 11 AWS regions

Dataset: 24 VM images

Scenario: add the images sequentially to 3 sites and then retrieve them from a fourth one

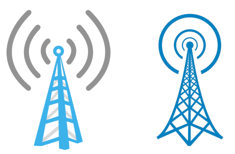
https://gitlab.inria.fr/jdarrous/nitro

From *Retrieval* to *Placement*

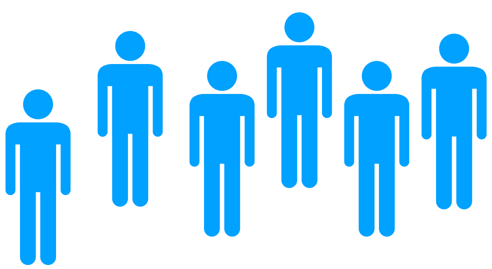
- Given a set of VMIs, we show how <u>network-aware image retrieval</u> can improve service provisioning time..
- We show how <u>network-aware image placement</u> can also contribute to reduce the provisioning time.

The role of containers in the Edge

- Edge-servers are characterised with limited compute and storage capacities. For example: micro-datacenter, Point-of-Presence (PoP), Cloudlet..
- Containers are widely accepted as the virtualization technology for Edge, due to their lightweight overhead.



- Retrieving images from a central (remote) repository is time consuming:
  - Downloading a 500 MB image over 5 MB/s link takes **100s**



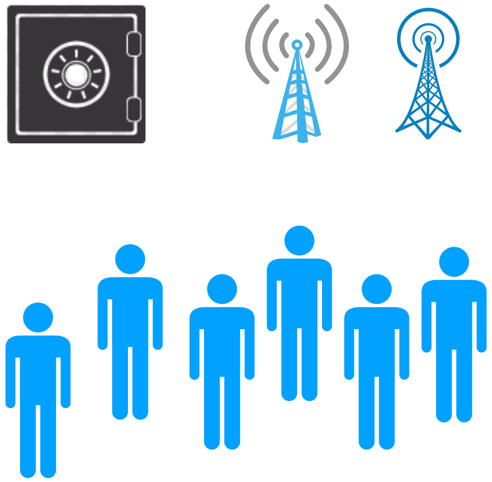
- Storing all the images locally is not possible
  - **2.5 million** images are hosted in Docker Hub.

The role of containers in the Edge

- Edge-servers are characterised with limited compute and storage capacities. For example: micro-datacenter, Point-of-Presence (PoP), Cloudlet..

- Containers are widely accepted as the virtualization technology for Edge, due to their lightweight overhead.

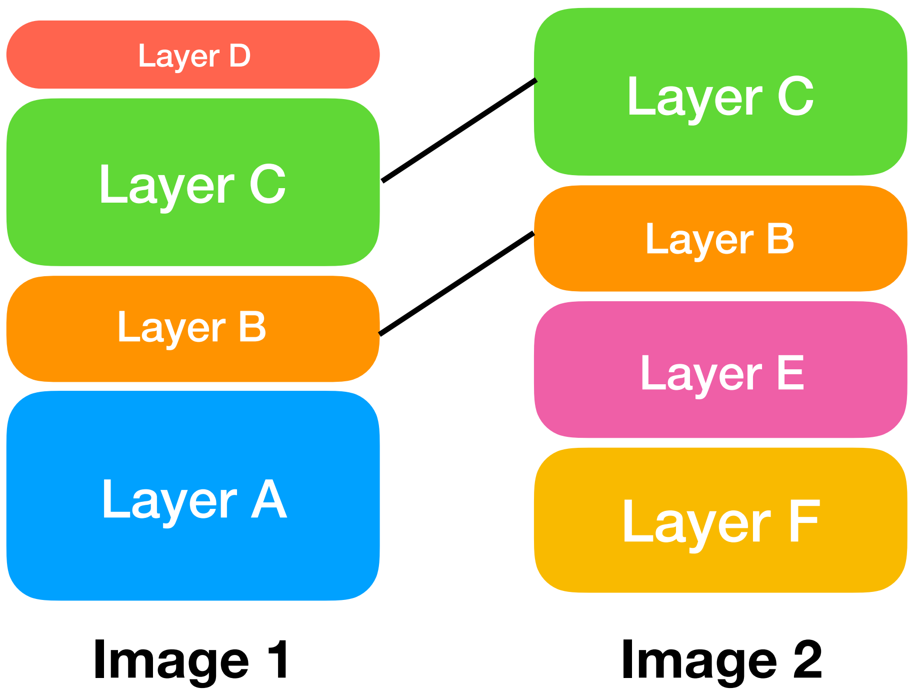
- Retrieving images from a central (remote) repository is time consuming:
    - Downloading a 500 MB image over 5 MB/s link takes **100s**



- Storing all the images locally is not possible
    - **2.5 million** images are hosted in Docker Hub.

Provide *fast* and *predictable* provisioning times for a set of containers by placing their images across the Edge-servers

How to place container images across Edge-Servers to provide fast and predictable retrieving time?



How to place container images across Edge-Servers to provide fast and predictable retrieving time?

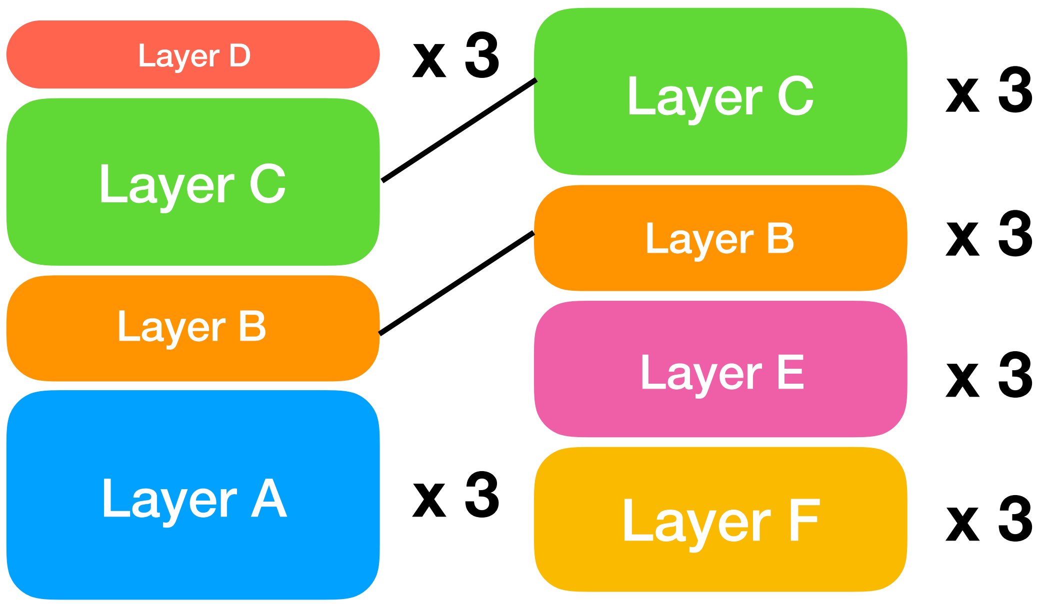
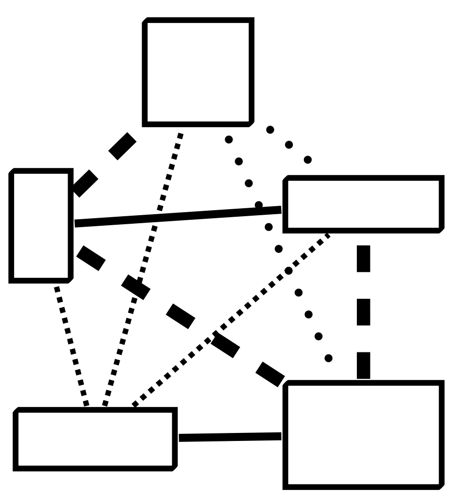


Image 1 **Image 2**

Edge infrastructure



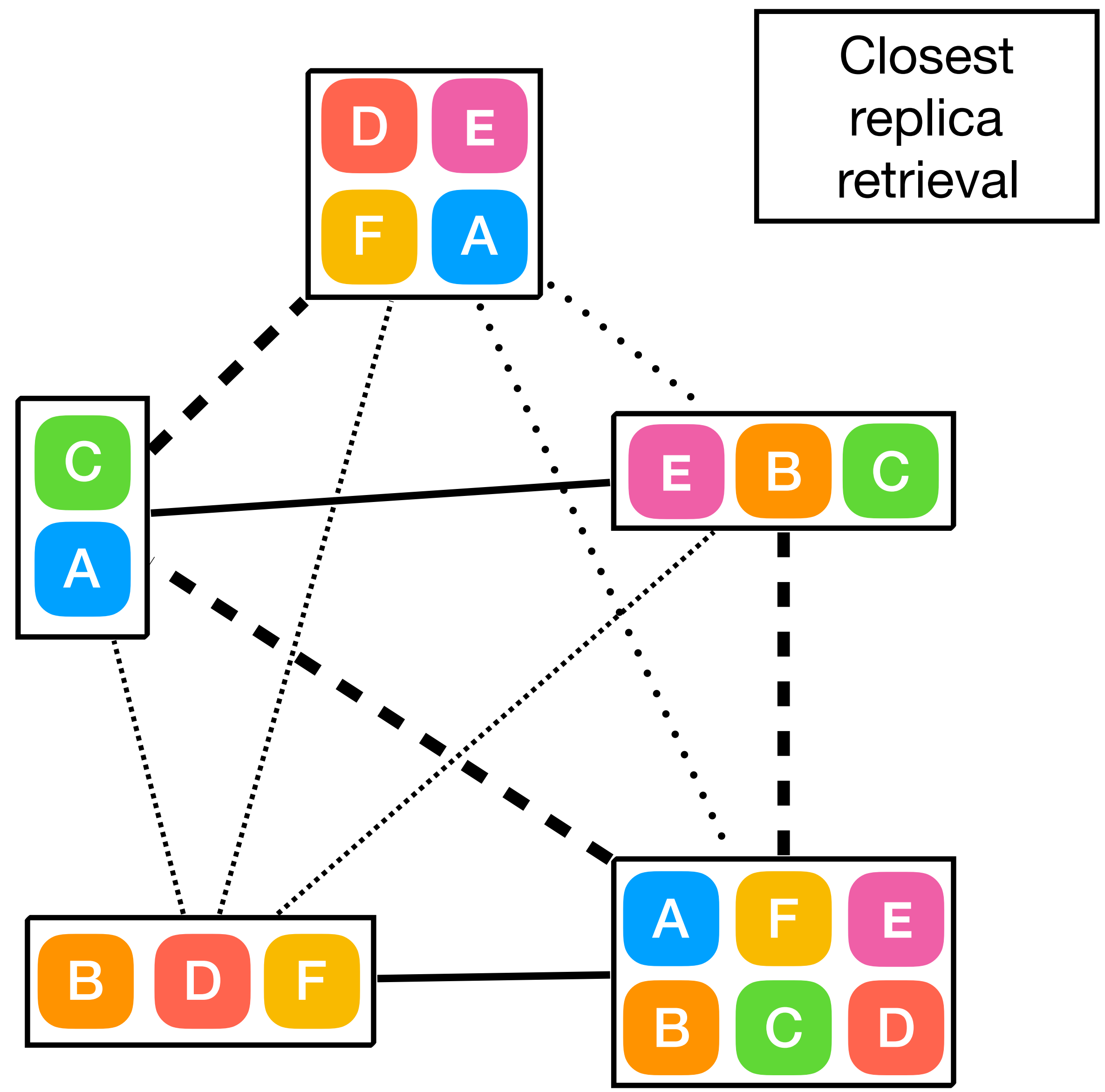
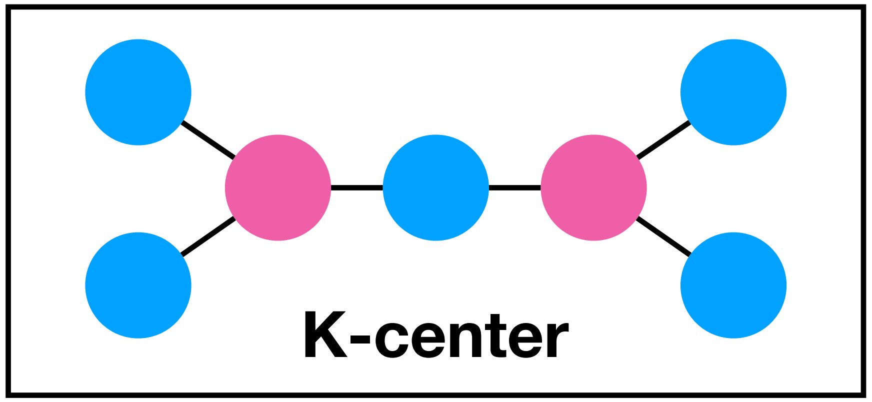


Image placement in Edge

Network-aware placement strategies

Placement algorithms:

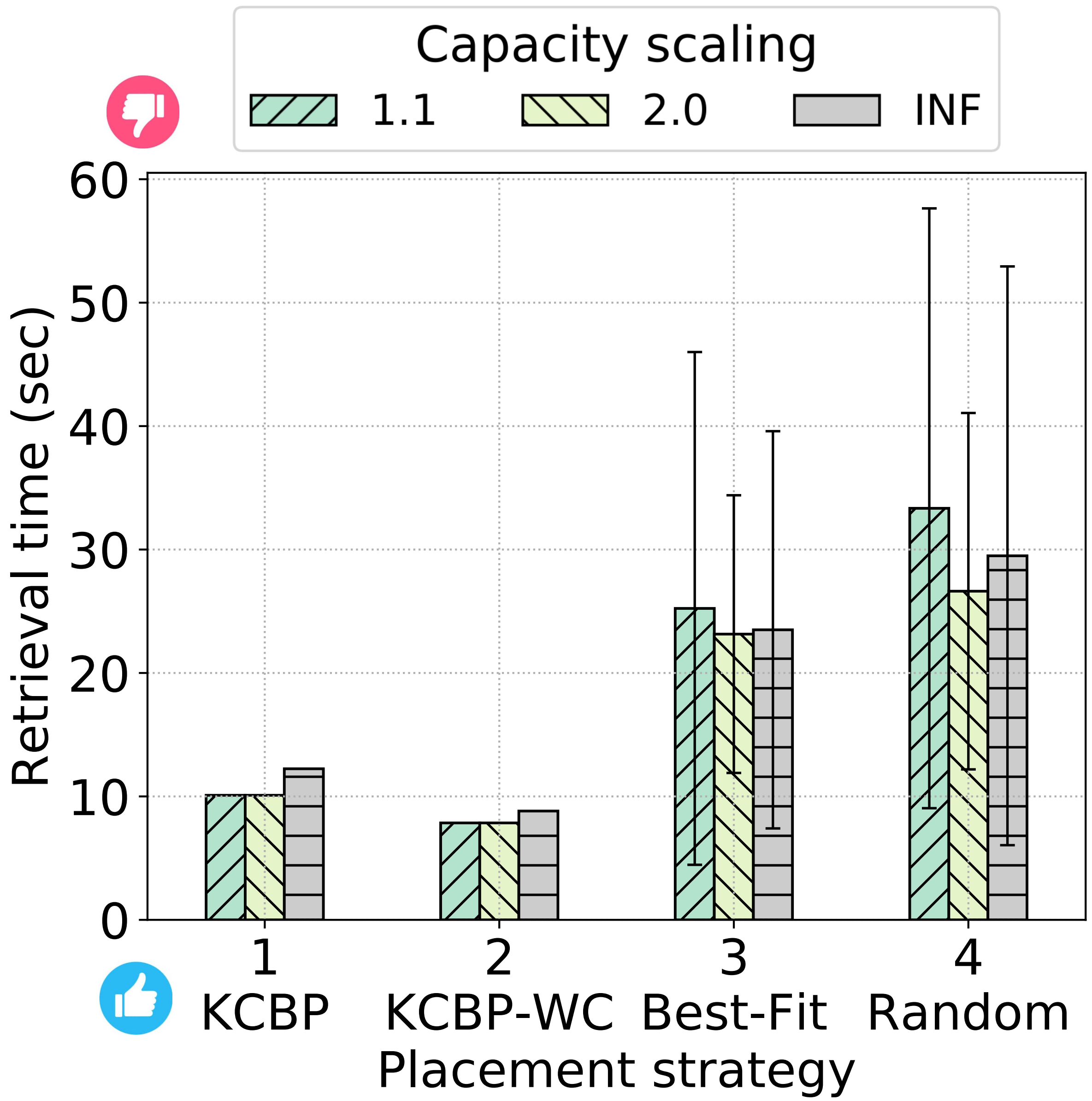


Assumptions:

- Complete graph network.
- Retrieval policy: Closest replica.
- Fixed dataset of images.

- k-Center Based Placement (KCBP)
  - Focuses on individual layer placement.
  - Iterative k-Center algorithm.
- k-Center Based Placement-Without Conflict (KCBP-WC)
  - Considers the images when placing the layers.

The importance of container image placement



Images Retrieval Times (Low Network)

ICCCN'19

Methodology

Simulator: written in Python (~1500 LoC)

Network: Synthetic and real-world topologies

Dataset: IBM traces (1000 images / 5600 layers)

Metric: maximal retrieval time of all images from any Edge-server.

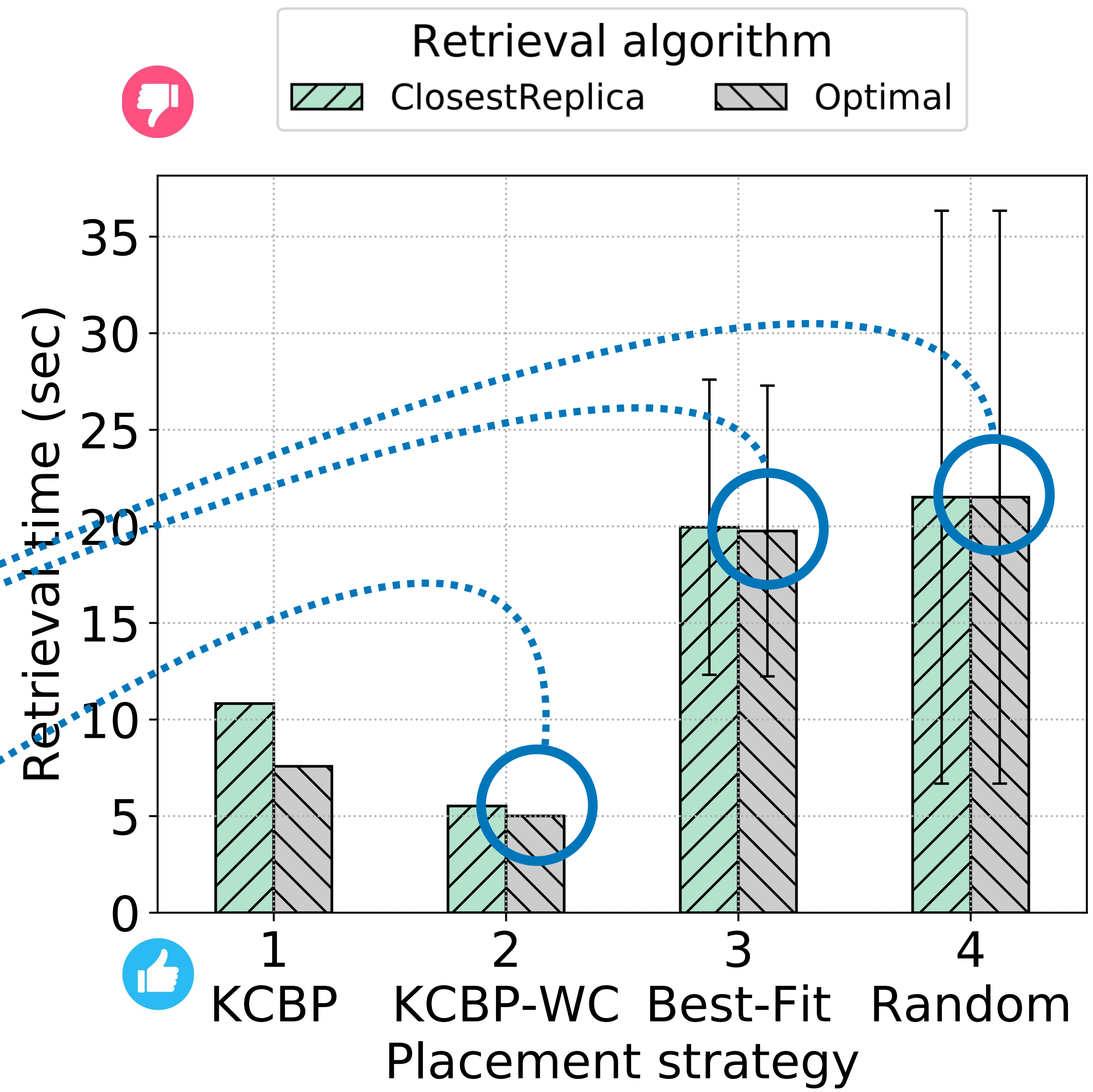
gitlab.inria.fr/jdarrous/image-placement-edge

Summary: Retrieval and placement should be jointly considered when provisioning a service

- Network-aware data retrieval is important to achieve fast service provisioning in geo-distributed clouds.
- Moreover, we also show that data placement can also contribute to reduce the provisioning time.

Optimal retrieval alone is not sufficient.

Both the **placement** and the **retrieval** are equally important.



Images Retrieval Times (Low Network)

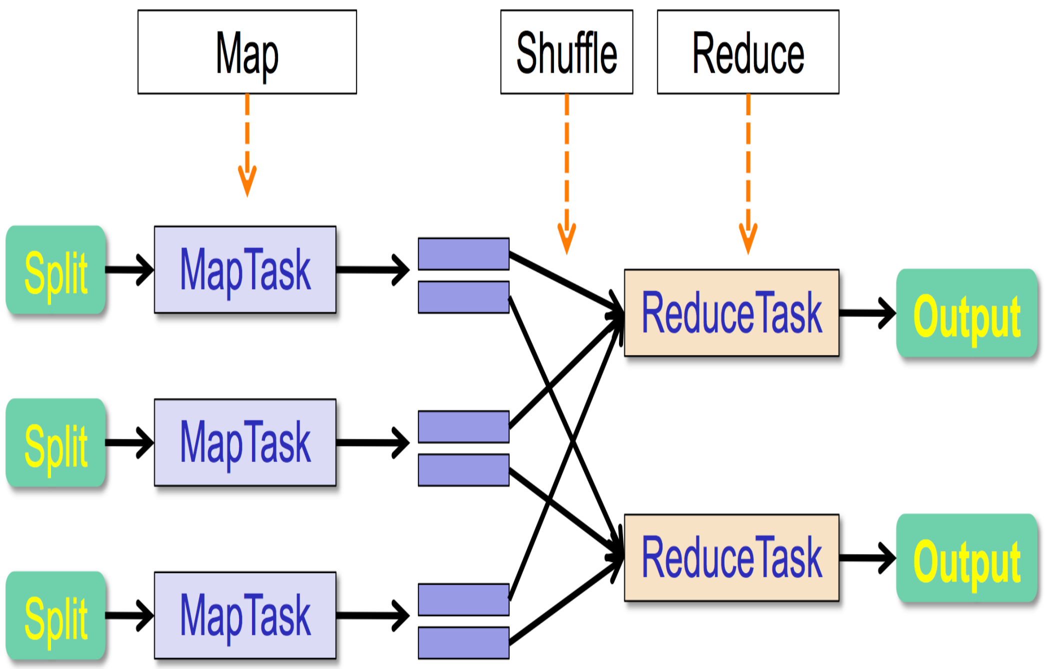
2

~~Replication~~

On the Efficiency of Erasure Coding for Data-Intensive Applications

- Data analytics in the Cloud
- Erasure coding
- Contribution 3: Understanding the performance data-intensive applications under erasure coding
- Contribution 4: Balancing data read under erasure coding

The prevalence of Data Analytics Frameworks



- MapReduce: a de-facto processing model
- Main features: scalability and simplicity
- Hadoop is widely adopted by industry and academia[1,2]
- Available as cloud-based solution by major providers[3,4]

- Emergence of new applications:
  - Iterative applications, interactive analytics..
- Brings richer transformations: FlatMap, Join..

▸ These frameworks rely on distributed file systems (DFSs) to store and access their jobs's input and output data.

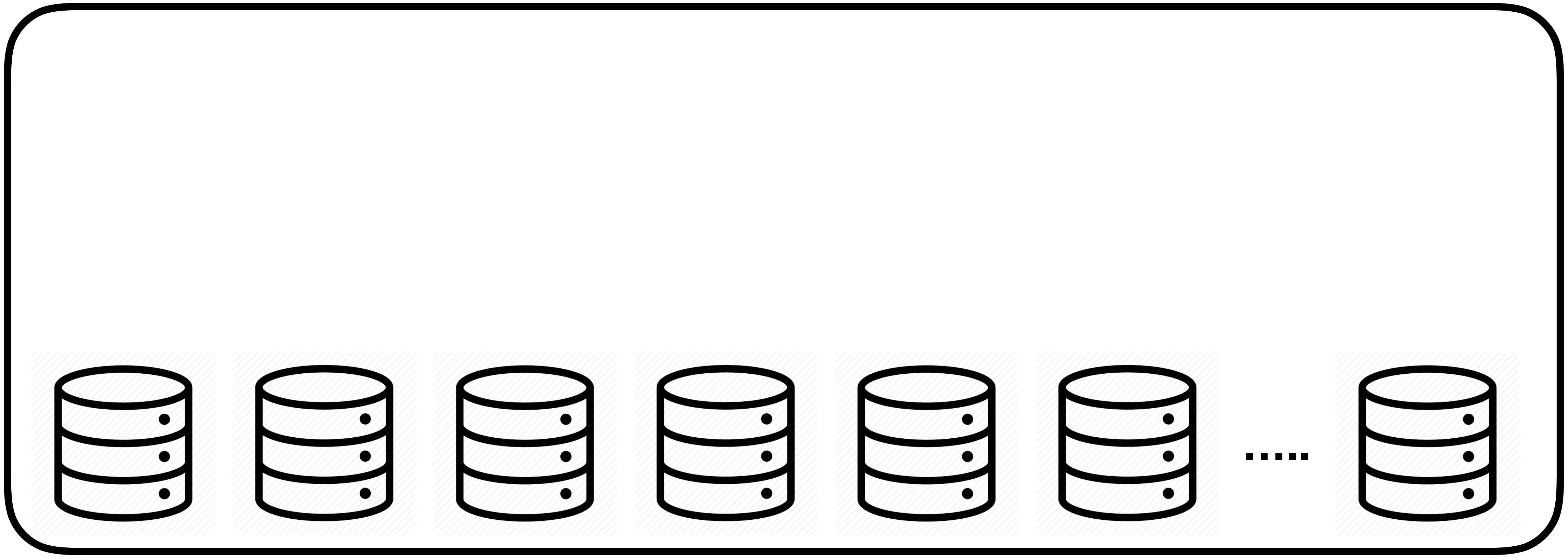
[1] Powered by Apache Hadoop, https://cwiki.apache.org/confluence/display/HADOOP2/PoweredBy
[2] MapReduce and Hadoop Algorithms in Academic Papers http://atbrox.com/2011/05/16/mapreduce-hadoop-algorithms-in-academic-papers-4th-update-may-2011
[3] Amazon EMR, https://aws.amazon.com/emr/
[4] Azure HDInsight, https://azure.microsoft.com/en-us/services/hdinsight

Data analytics in the Cloud

The role of replication

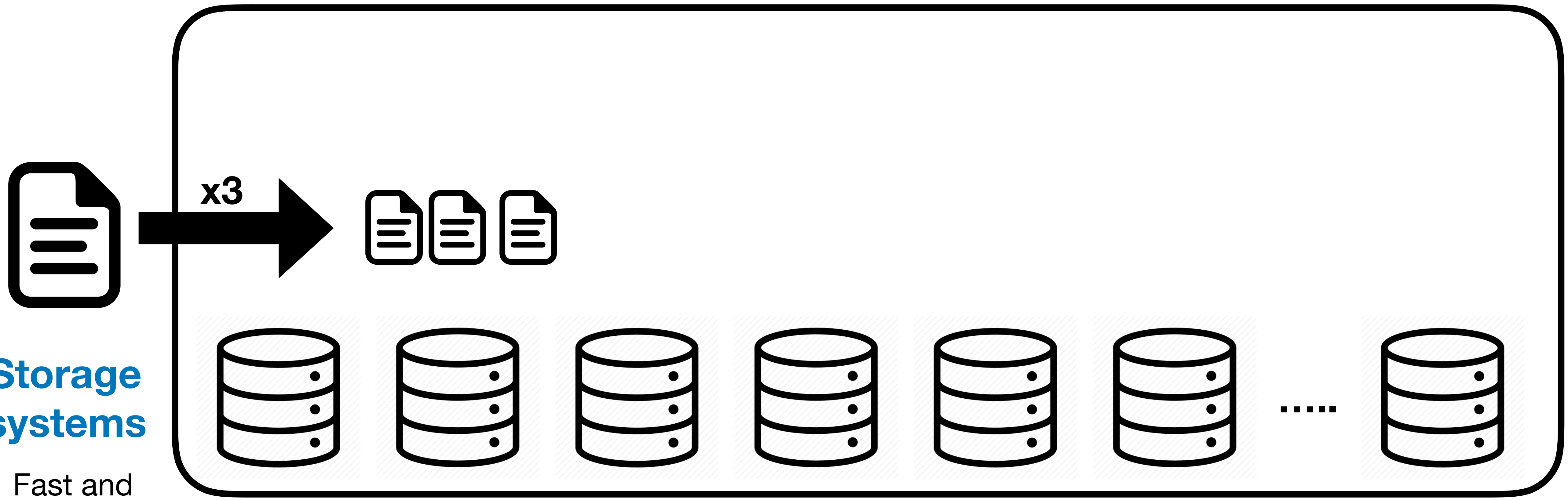
Storage systems

Fast and reliable access



Data analytics in the Cloud

The role of replication



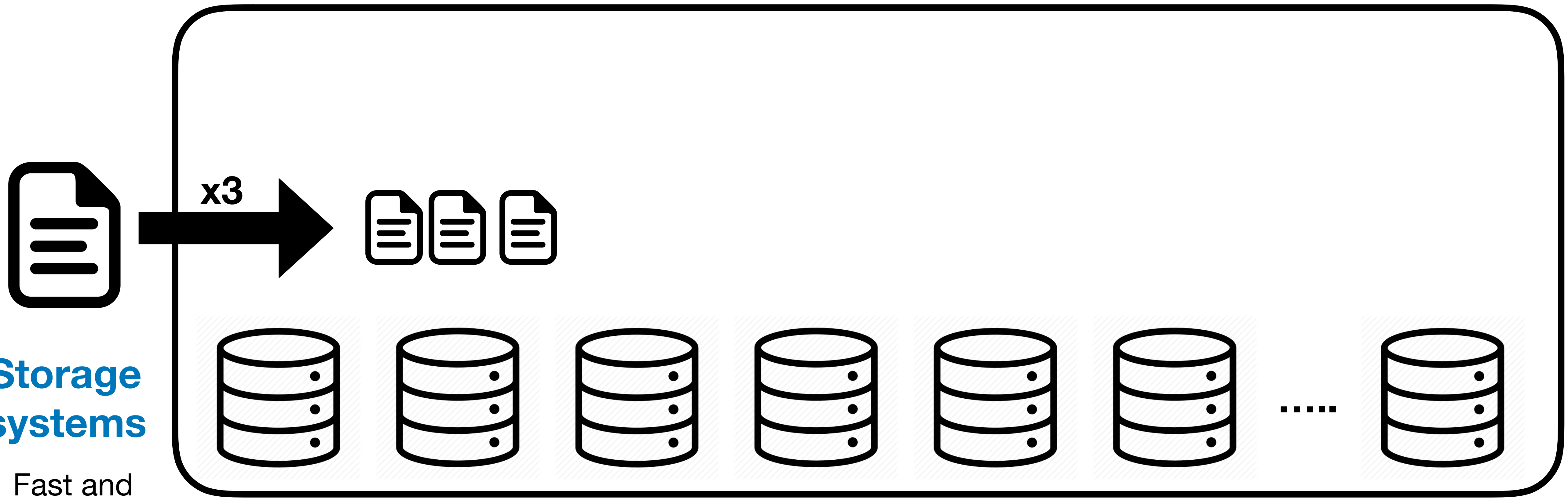
Storage systems

Fast and
reliable access

Data analytics in the Cloud
The role of replication

Analytics frameworks

Data locality
Fault tolerance



Storage systems

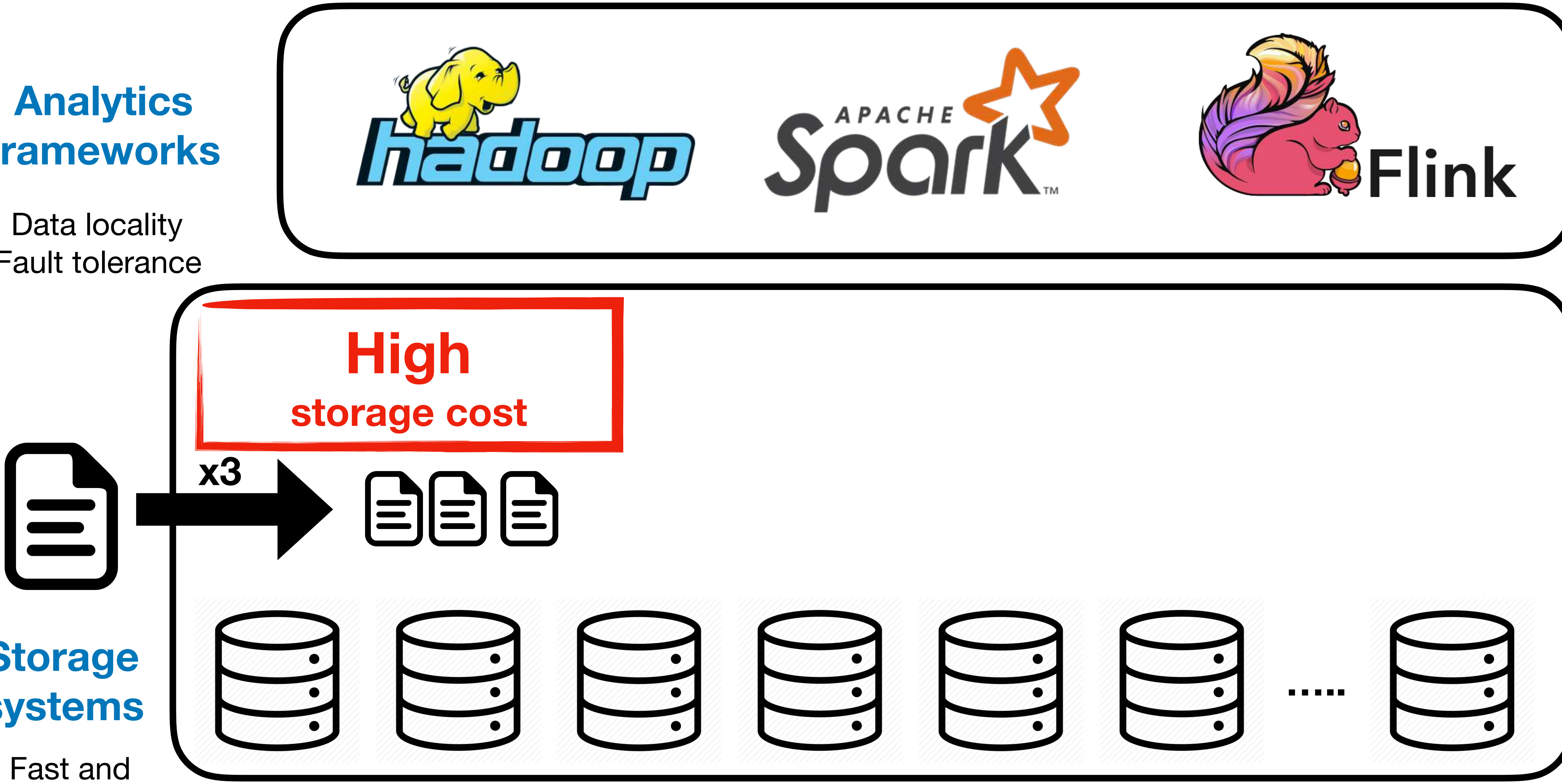
Fast and
reliable access

Data analytics in the Cloud

The role of replication

Analytics frameworks

Data locality
Fault tolerance



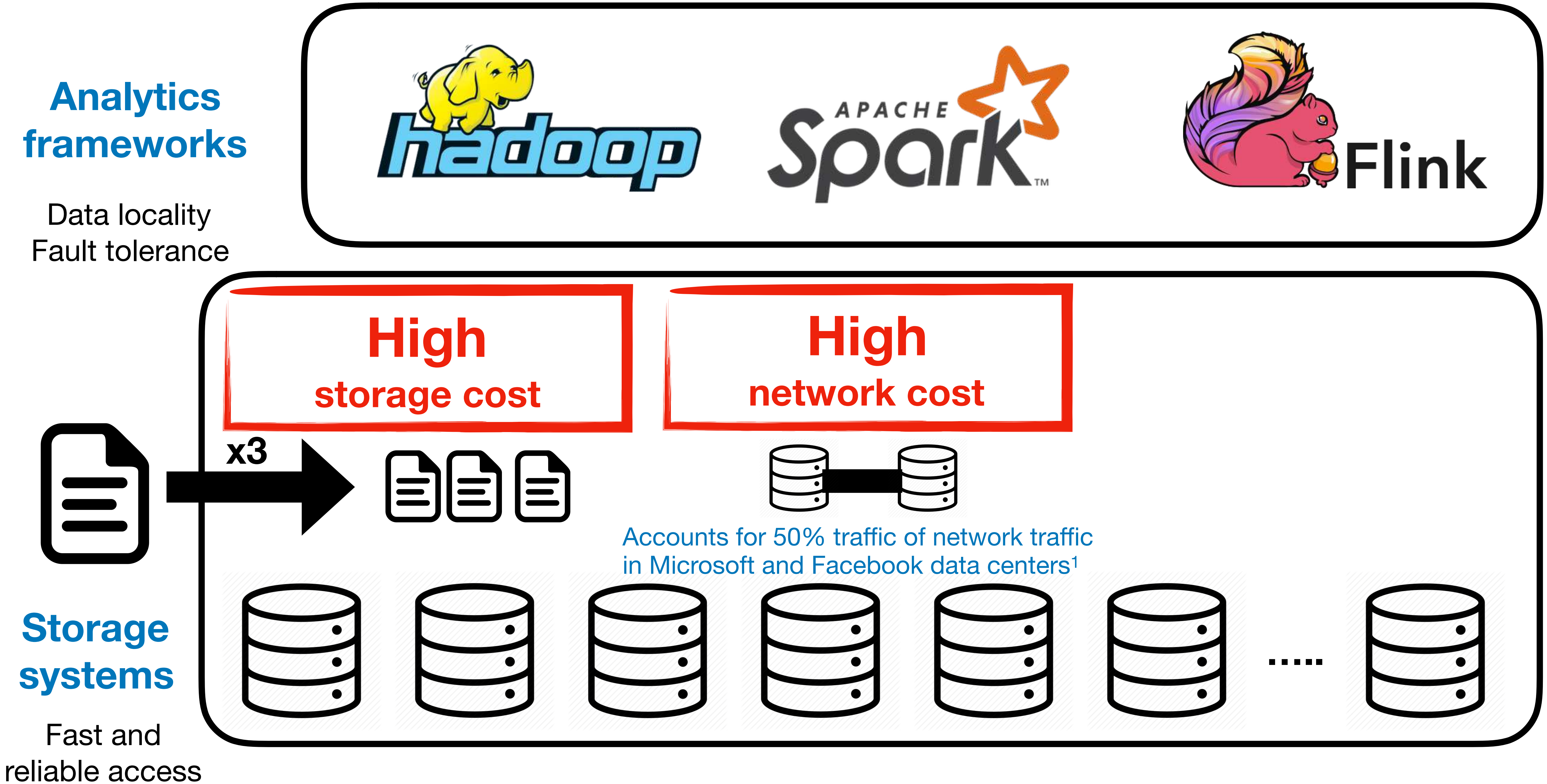
Storage systems

Fast and reliable access

[1] Chowdhury et al., Leveraging Endpoint Flexibility in Data-Intensive Clusters, ACM SIGCOMM 2013

Data analytics in the Cloud

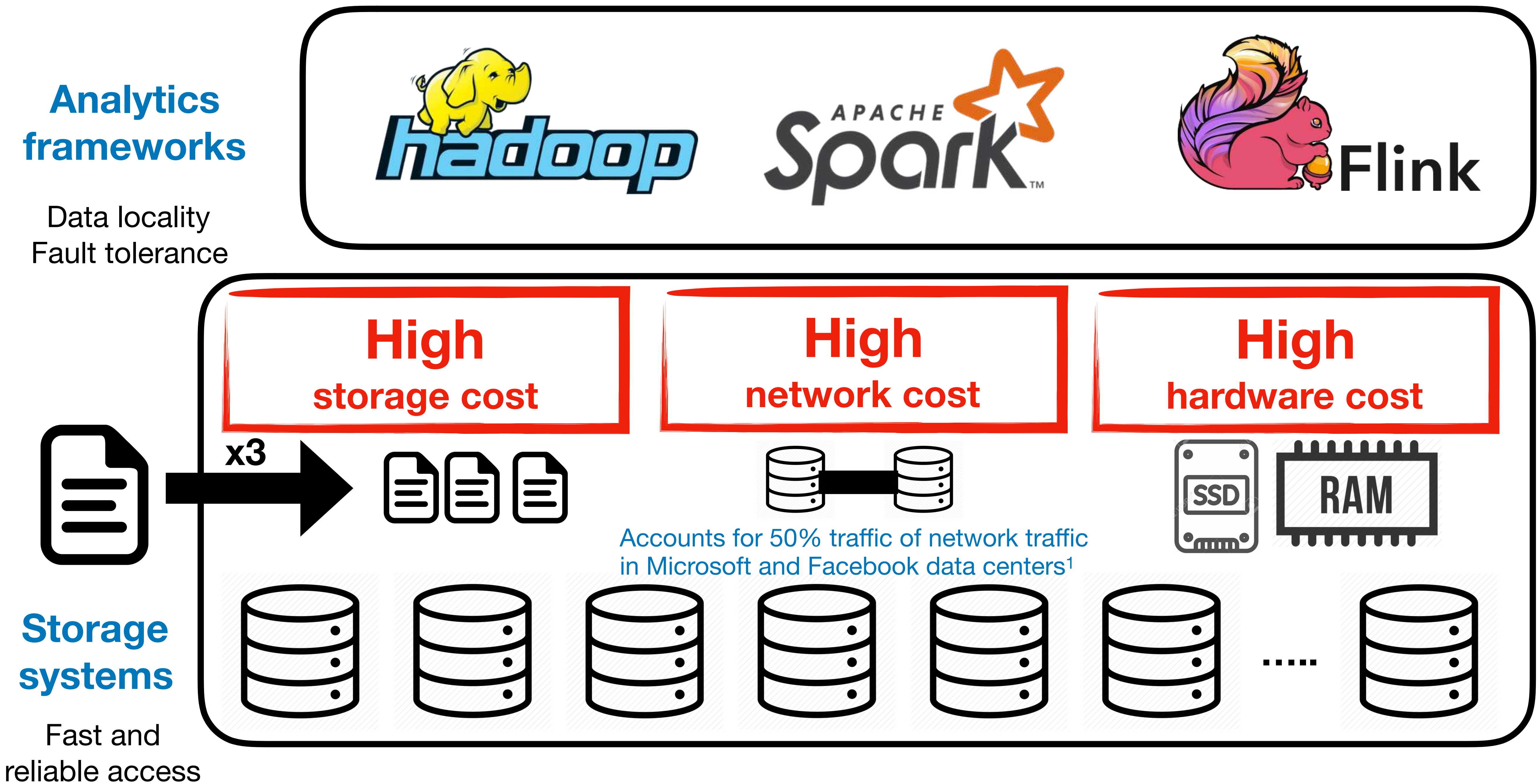
The role of replication



[1] Chowdhury et al., Leveraging Endpoint Flexibility in Data-Intensive Clusters, ACM SIGCOMM 2013

Data analytics in the Cloud

The role of replication



[1] Chowdhury et al., Leveraging Endpoint Flexibility in Data-Intensive Clusters, ACM SIGCOMM 2013

Erasure coding

The case of Reed-Solomon RS(n, k)

<u>RS is employed in</u>: HDFS, Ceph, Swift, EC-Cache, Windows Azure Storage, Microsoft Giza, Facebook's f4, Google Colossus..

Erasure coding

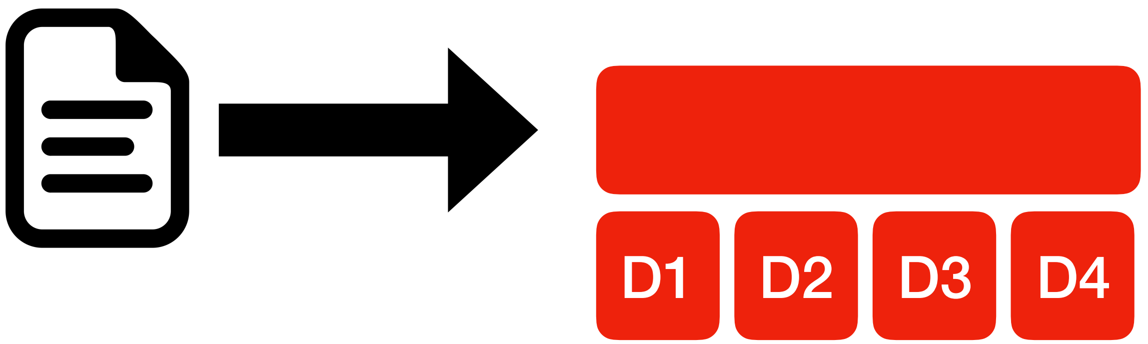
The case of Reed-Solomon RS(n, k)

<u>RS is employed in</u>: HDFS, Ceph, Swift, EC-Cache, Windows Azure Storage, Microsoft Giza, Facebook's f4, Google Colossus..

Erasure coding

The case of Reed-Solomon RS(n, k)

RS is employed in: HDFS, Ceph, Swift, EC-Cache, Windows Azure Storage, Microsoft Giza, Facebook's f4, Google Colossus..

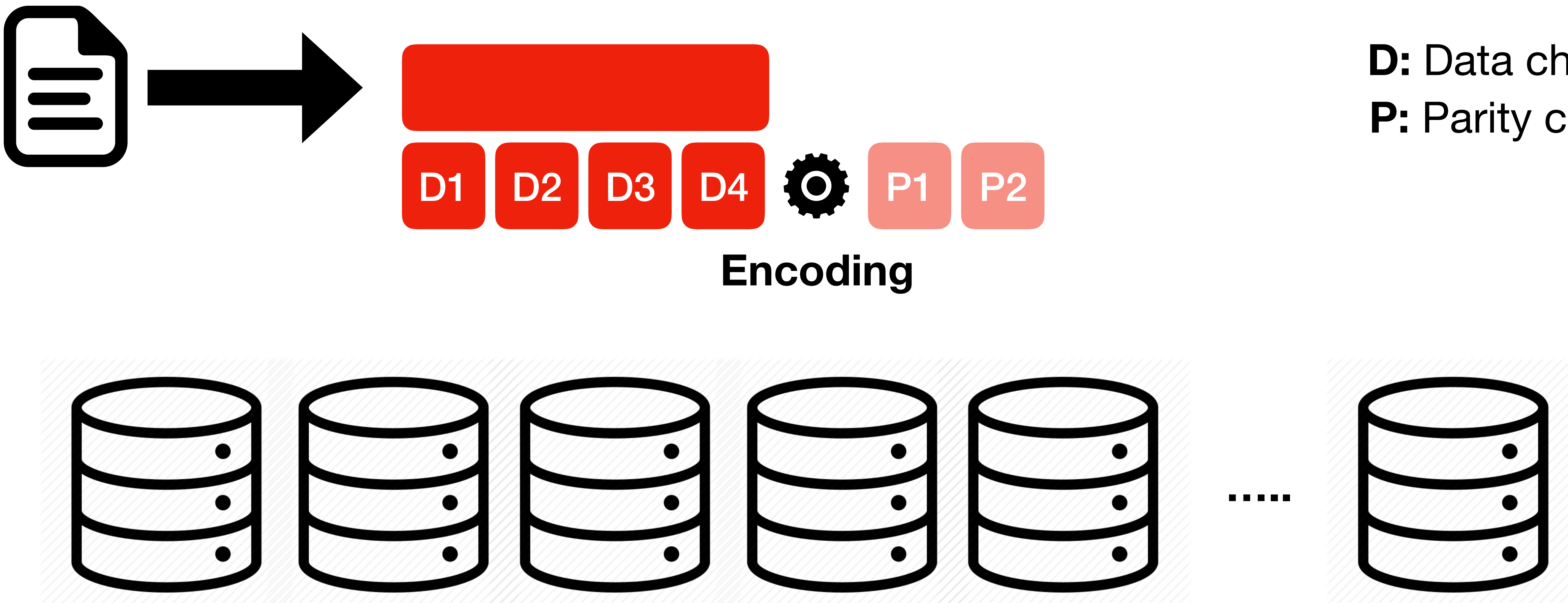


D: Data chunk
P: Parity chunk

Erasure coding

The case of Reed-Solomon RS(n, k)

<u>RS is employed in</u>: HDFS, Ceph, Swift, EC-Cache, Windows Azure Storage, Microsoft Giza, Facebook's f4, Google Colossus..



D: Data chunk

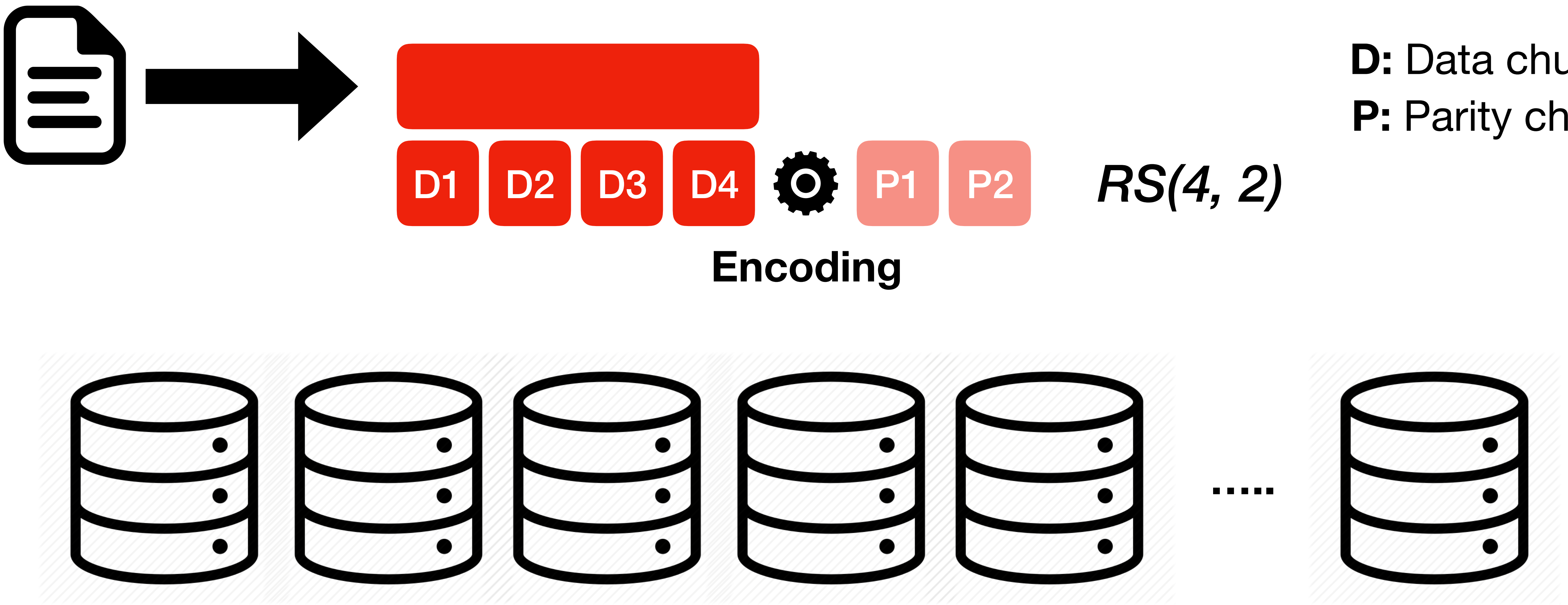
P: Parity chunk

Erasure coding

The case of Reed-Solomon RS(n, k)

RS is employed in: HDFS, Ceph, Swift, EC-Cache, Windows Azure Storage, Microsoft Giza, Facebook's f4, Google Colossus..

D: Data chunk
P: Parity chunk

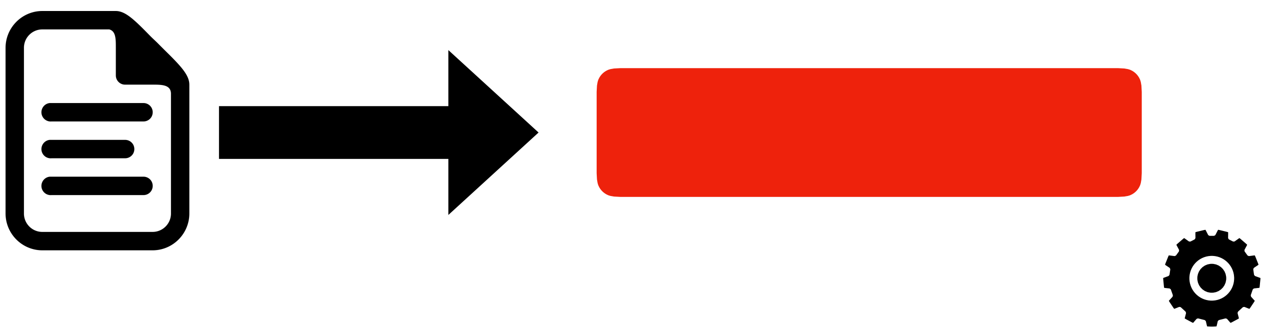


Erasure coding

The case of Reed-Solomon RS(n, k)

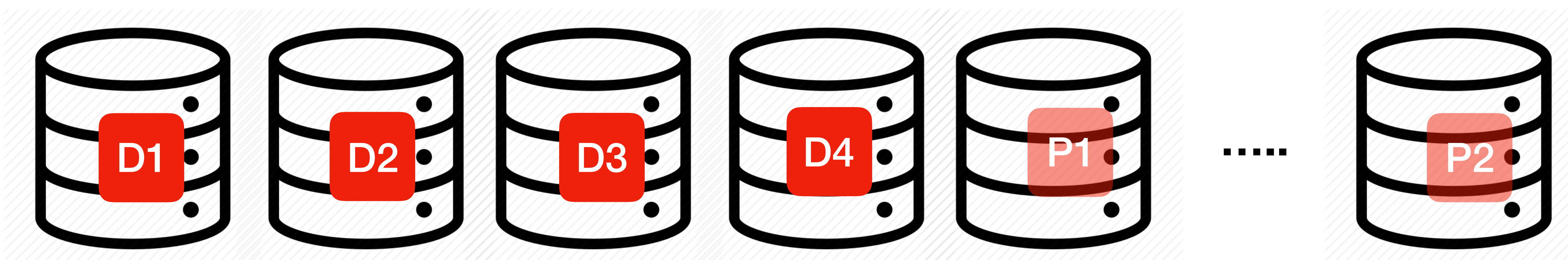
<u>RS is employed in</u>: HDFS, Ceph, Swift, EC-Cache, Windows Azure Storage, Microsoft Giza, Facebook's f4, Google Colossus..

D: Data chunk
P: Parity chunk



RS(4, 2)

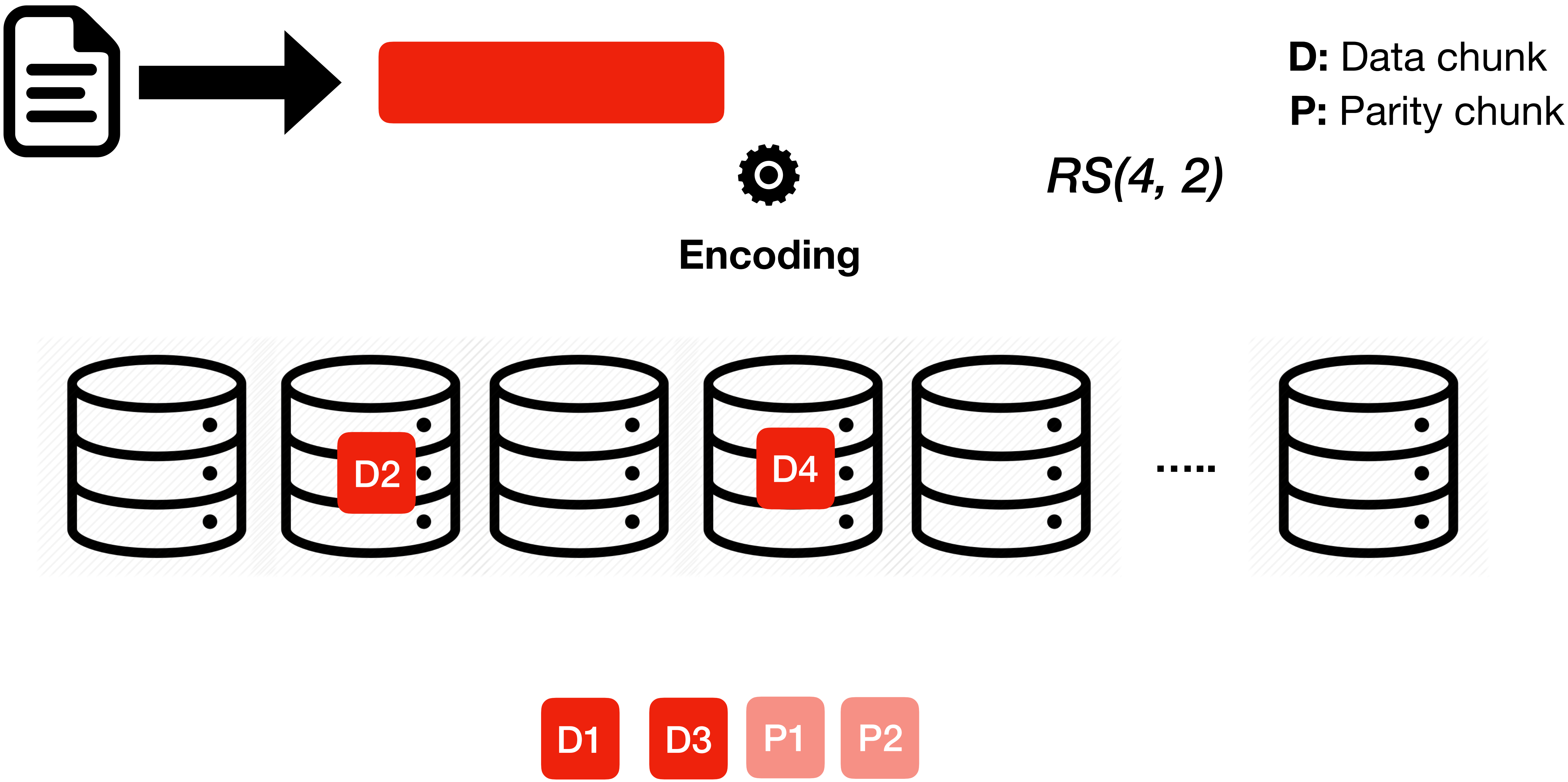
Encoding



Erasure coding

The case of Reed-Solomon RS(n, k)

<u>RS is employed in</u>: HDFS, Ceph, Swift, EC-Cache, Windows Azure Storage, Microsoft Giza, Facebook's f4, Google Colossus..

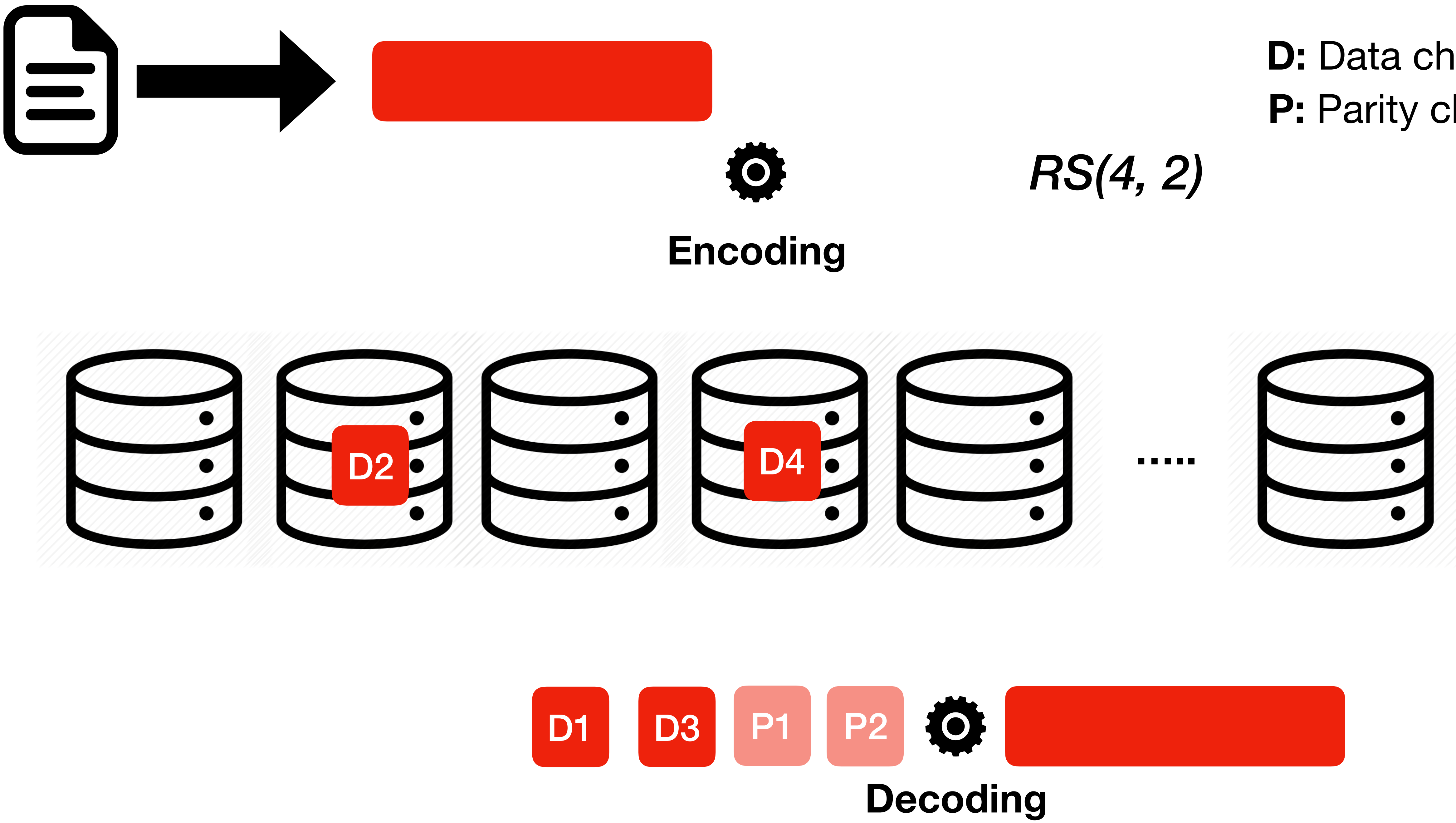


Erasure coding

The case of Reed-Solomon RS(n, k)

<u>RS is employed in</u>: HDFS, Ceph, Swift, EC-Cache, Windows Azure Storage, Microsoft Giza, Facebook's f4, Google Colossus..

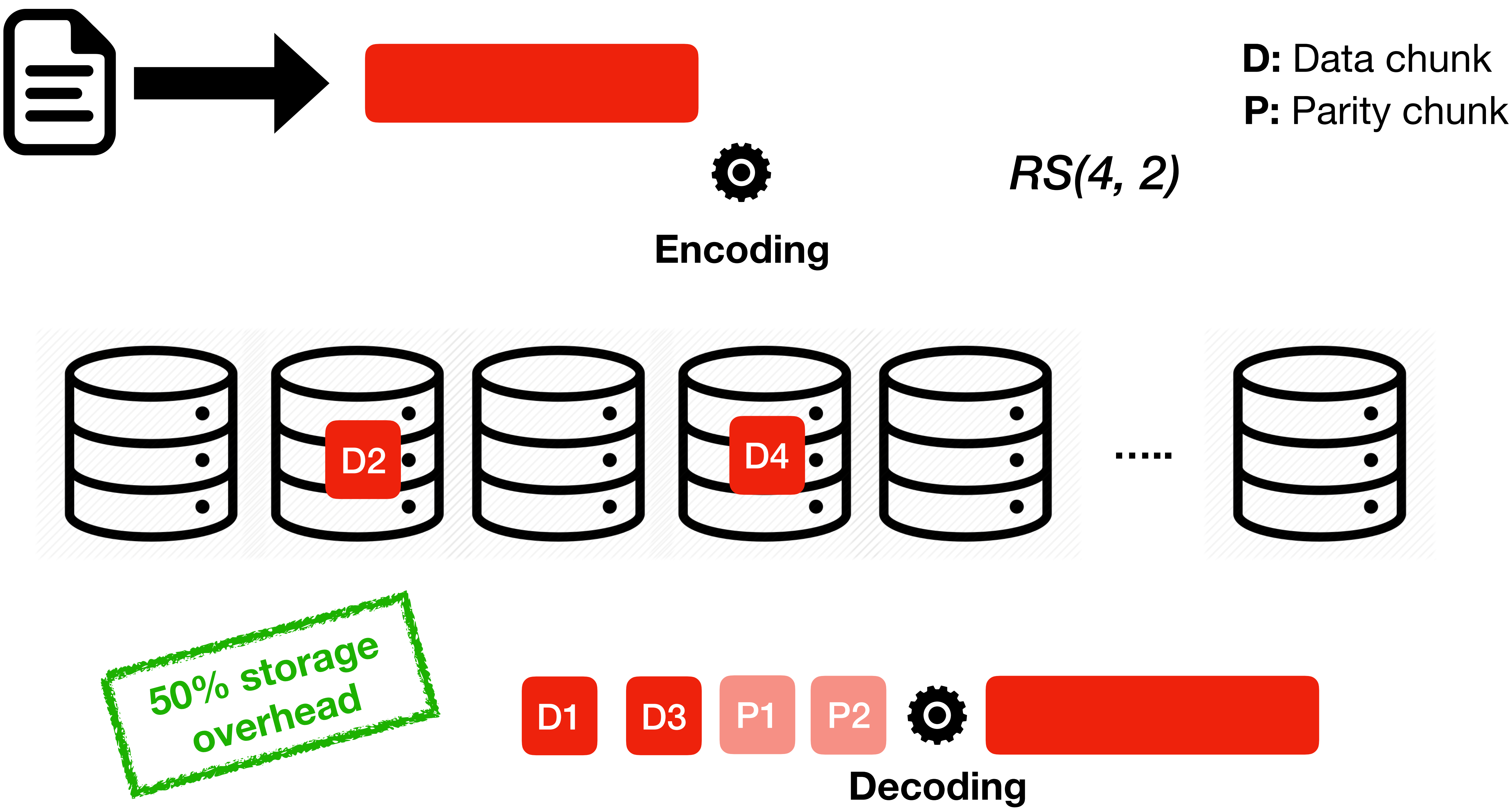
D: Data chunk
P: Parity chunk



Erasure coding

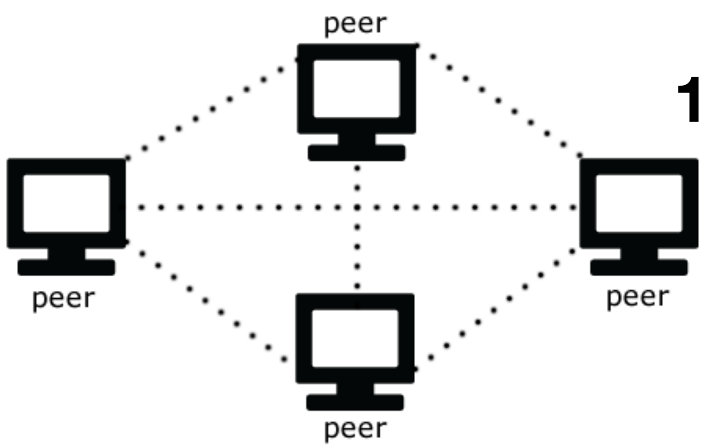
The case of Reed-Solomon RS(n, k)

RS is employed in: HDFS, Ceph, Swift, EC-Cache, Windows Azure Storage, Microsoft Giza, Facebook's f4, Google Colossus..



Where we can find EC?

Before



[1]

Cold (archived) Data[2]

2014 ---------------------- **ISA-L** ------------------------------

After

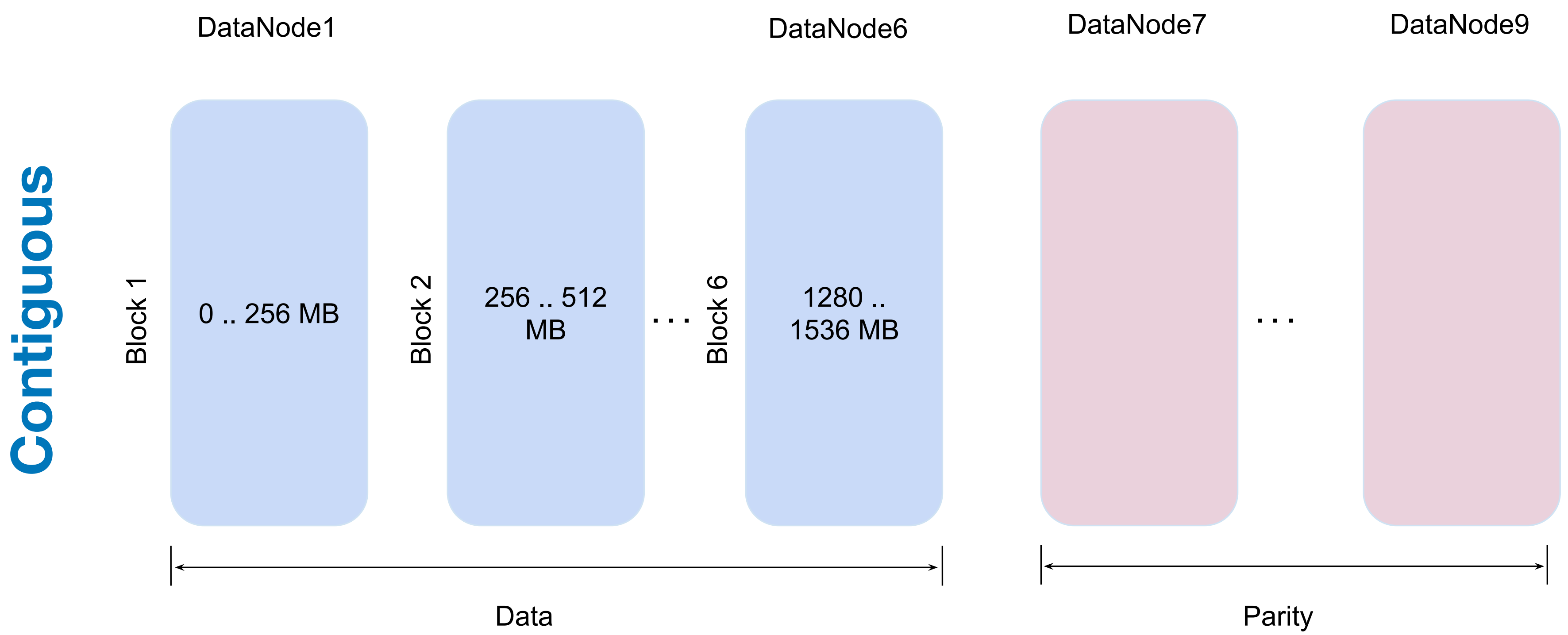
HDFS 3.0.0 (2017)

Hot (cached) Data[3] (2016)

[1] Kubiatowicz et al., OceanStore: An Architecture for Global-scale Persistent Storage, ASPLOS'00.
[2] Haeberlen et al., Glacier: Highly durable, decentralized storage despite massive correlated failures, NSDI'05.
[3] Rashmi et al., EC-Cache: Load-Balanced, Low-Latency Cluster Caching with Online Erasure Coding, OSDI'16.

Block layout under EC

Block layout under EC



Allows local reads (Data locality)

- Equals to one replica (as parity blocks cannot be used)[1,2,3]

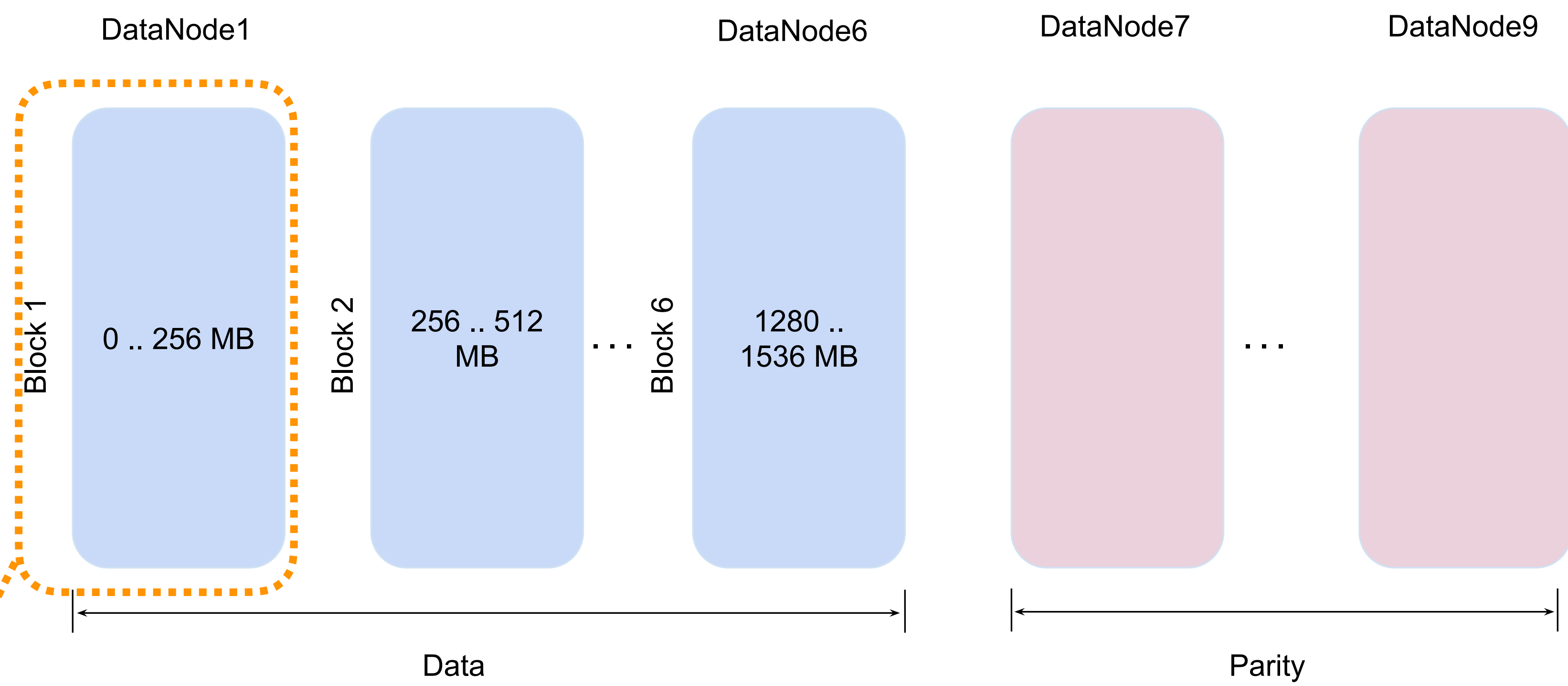
High memory overhead

- As all the blocks should be in memory for encoding and decoding[3]

[1] Fan et al., DiskReduce: RAID for Data-intensive Scalable Computing, WPDC'09.
[2] Zhang et al., Does erasure coding have a role to play in my data center?, Microsoft research 2010.
[3] Li et al., Degraded-First Scheduling for MapReduce in Erasure-Coded Storage Clusters, DSN'14.

Block layout under EC

Contiguous



Allows local reads (Data locality)

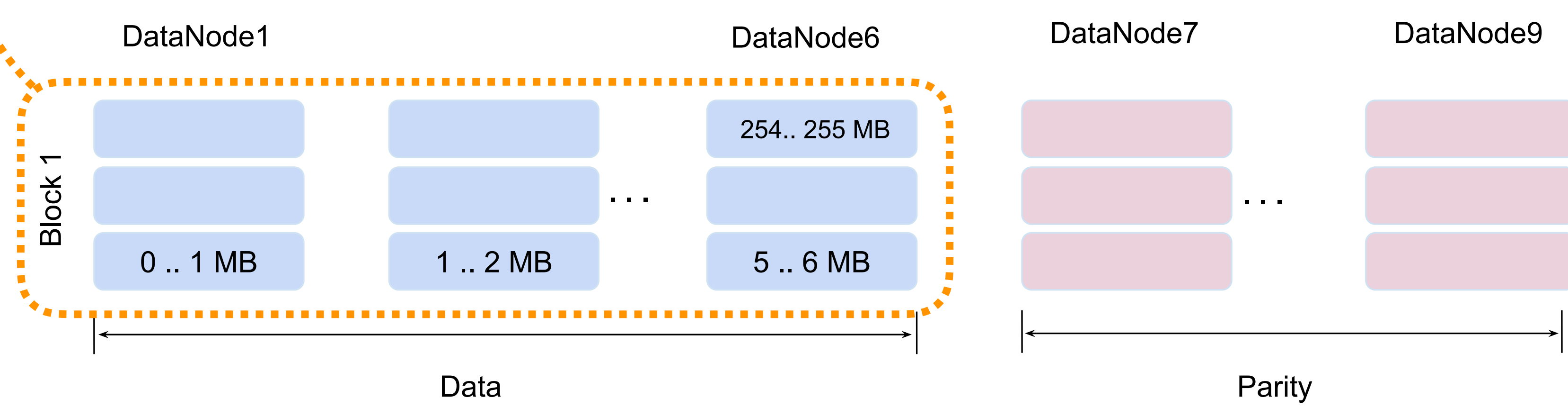
- Equals to one replica (as parity blocks cannot be used)[1,2,3]

High memory overhead

- As all the blocks should be in memory for encoding and decoding[3]

[1] Fan et al., DiskReduce: RAID for Data-intensive Scalable Computing, WPDC'09.
[2] Zhang et al., Does erasure coding have a role to play in my data center?, Microsoft research 2010.
[3] Li et al., Degraded-First Scheduling for MapReduce in Erasure-Coded Storage Clusters, DSN'14.

Striped



- Efficient for small files
- Low memory overhead
- Allows parallel I/O

➡ Employed in caching systems[4] and HDFS

[4] Rashmi et al., EC-Cache: Load-Balanced, Low-Latency Cluster Caching with Online Erasure Coding, OSDI'16.

What are the performance characteristics of analysis jobs under striped EC data?

✗ Data locality can not be achieved

✓ Faster networks with low over-subscription factor are more common[1]

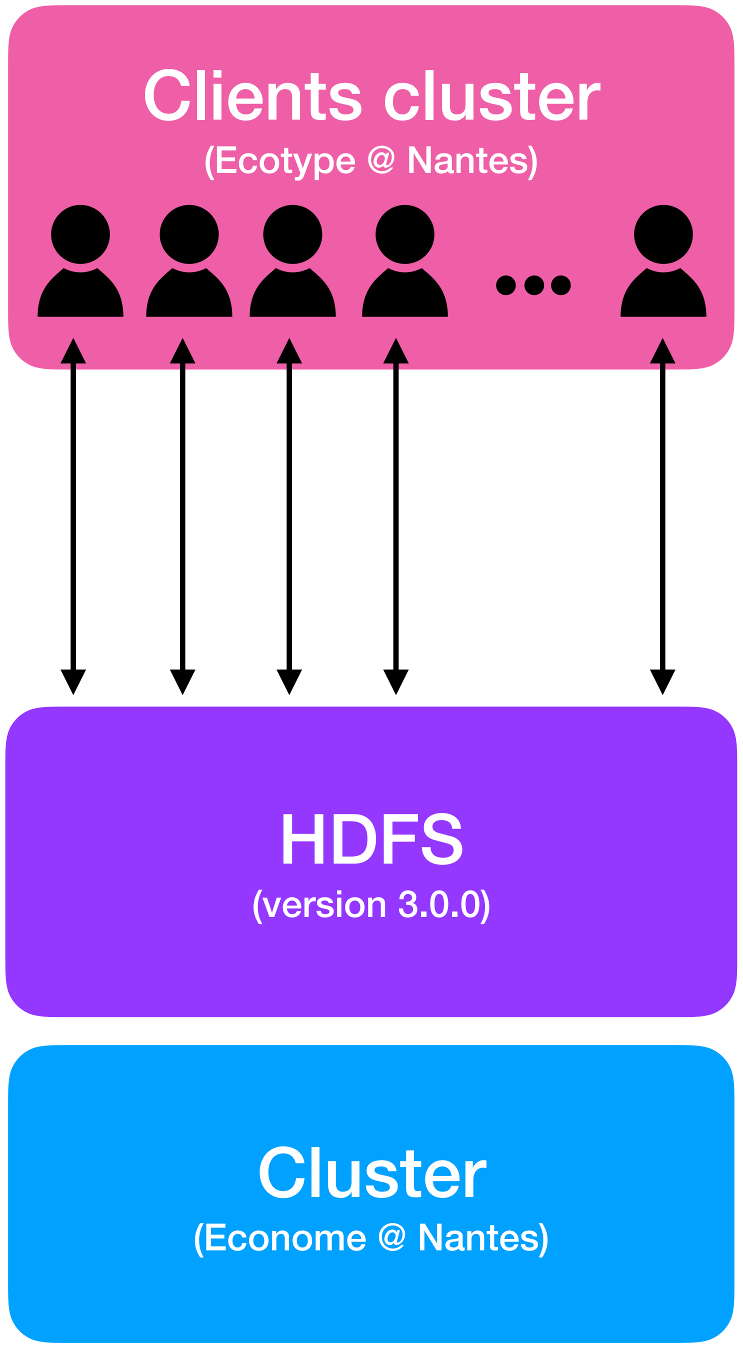
✓ In some infrastructures, disk locality becomes irrelevant as the bottleneck is shifting to storage I/O[2,3]

The **first** to answer this question through an in-depth experimental evaluation

- On the storage level: HDFS
- On the processing level: MapReduce

[1] Gao et al., Network Requirements for Resource Disaggregation, OSDI'16.
[2] Rashmi et al., Having Your Cake and Eating It Too: Jointly Optimal Erasure Codes for I/O, Storage and Network-bandwidth, FAST'15.
[3] Ananthanarayanan et al., Disk-locality in Datacenter Computing Considered Irrelevant, HotOS'11
[4] Jonas et al., Occupy the Cloud: Distributed Computing for the 99%, SoCC'17

Methodology (HDFS)



Micro-benchmarks of <u>read</u> and <u>write</u>

<u>Metric</u>: Average throughput per client (MB/sec)

Each client runs on a separate machine and stores its data in memory.

Block size: **256 MB**[1,2]

Replication factor: **3**

EC policy: **RS(6, 3)** with 1 MB cell size

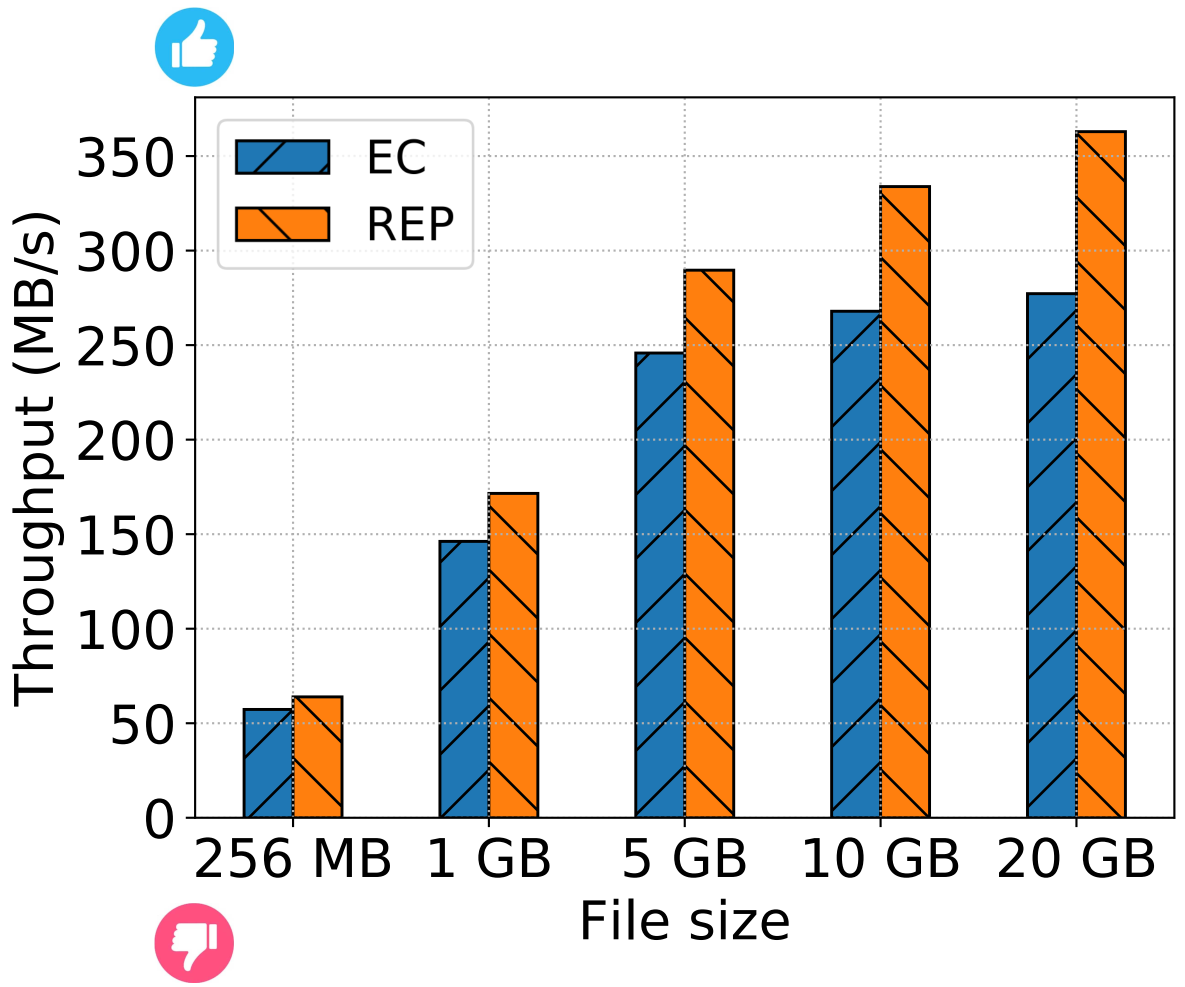
21 machines with 8-core processor, 64GB of main memory, and one HDD at 7.2k RPM

10 Gigabit Ethernet network

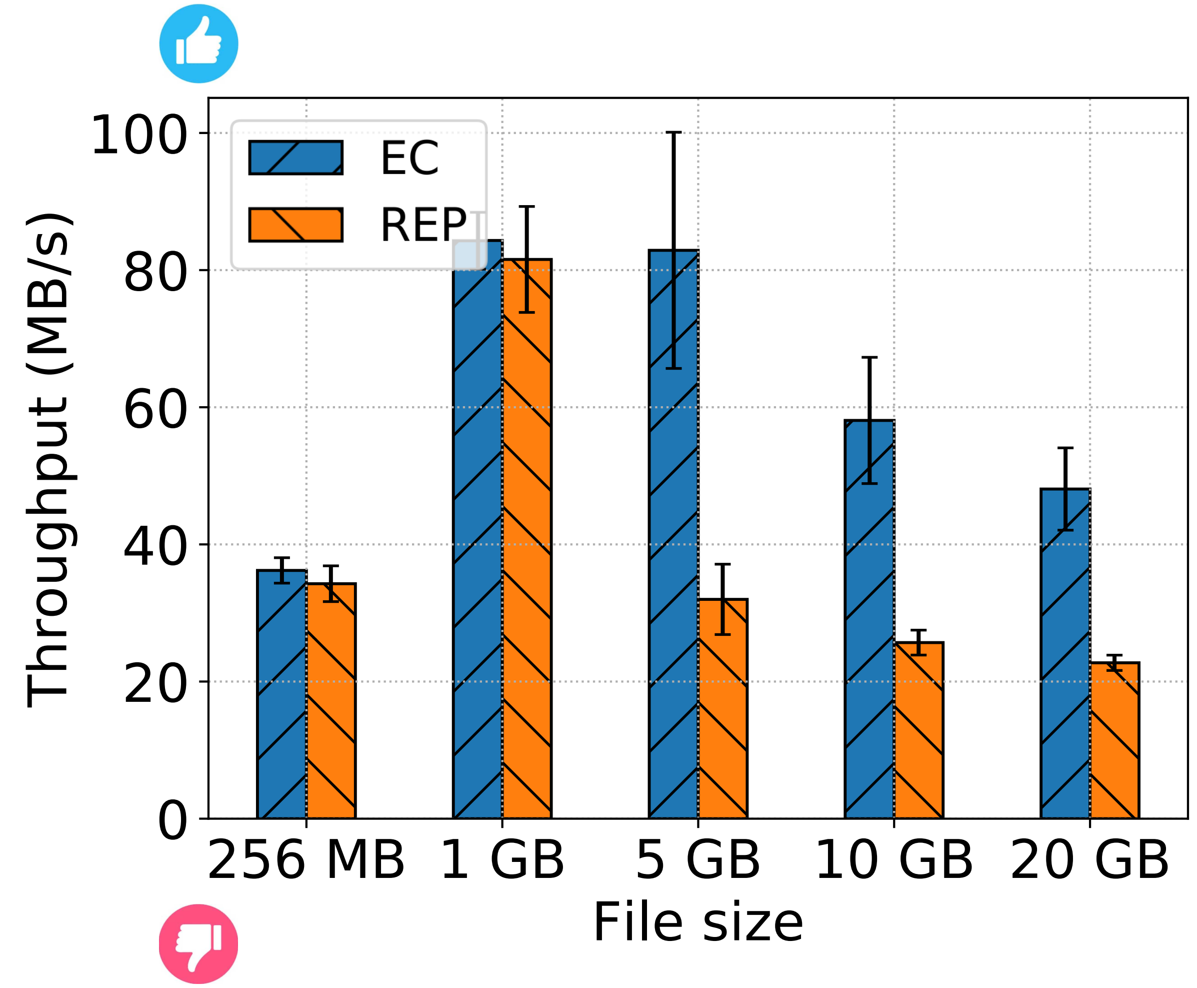
[1] Dinu et al., RCMP: Enabling Efficient Recomputation Based Failure Resilience for Big Data Analytics, IPDPS'14
[2] Yildiz et al., Enabling Fast Failure Recovery in Shared Hadoop Clusters: Towards Failure-Aware Scheduling, FGCS'16

The cost of adding data

```
./hadoop fs -put
```



Single client

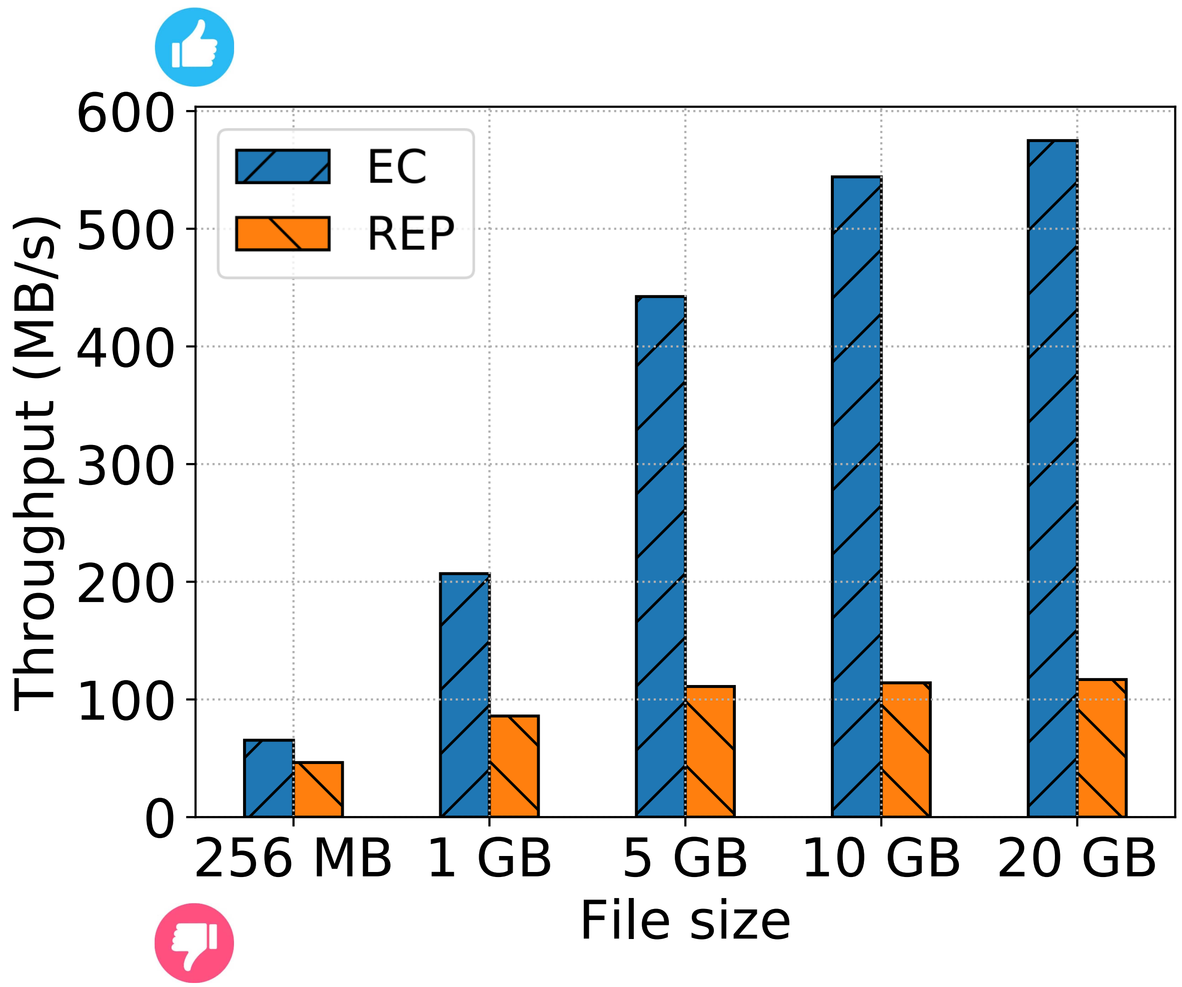


40 concurrent clients

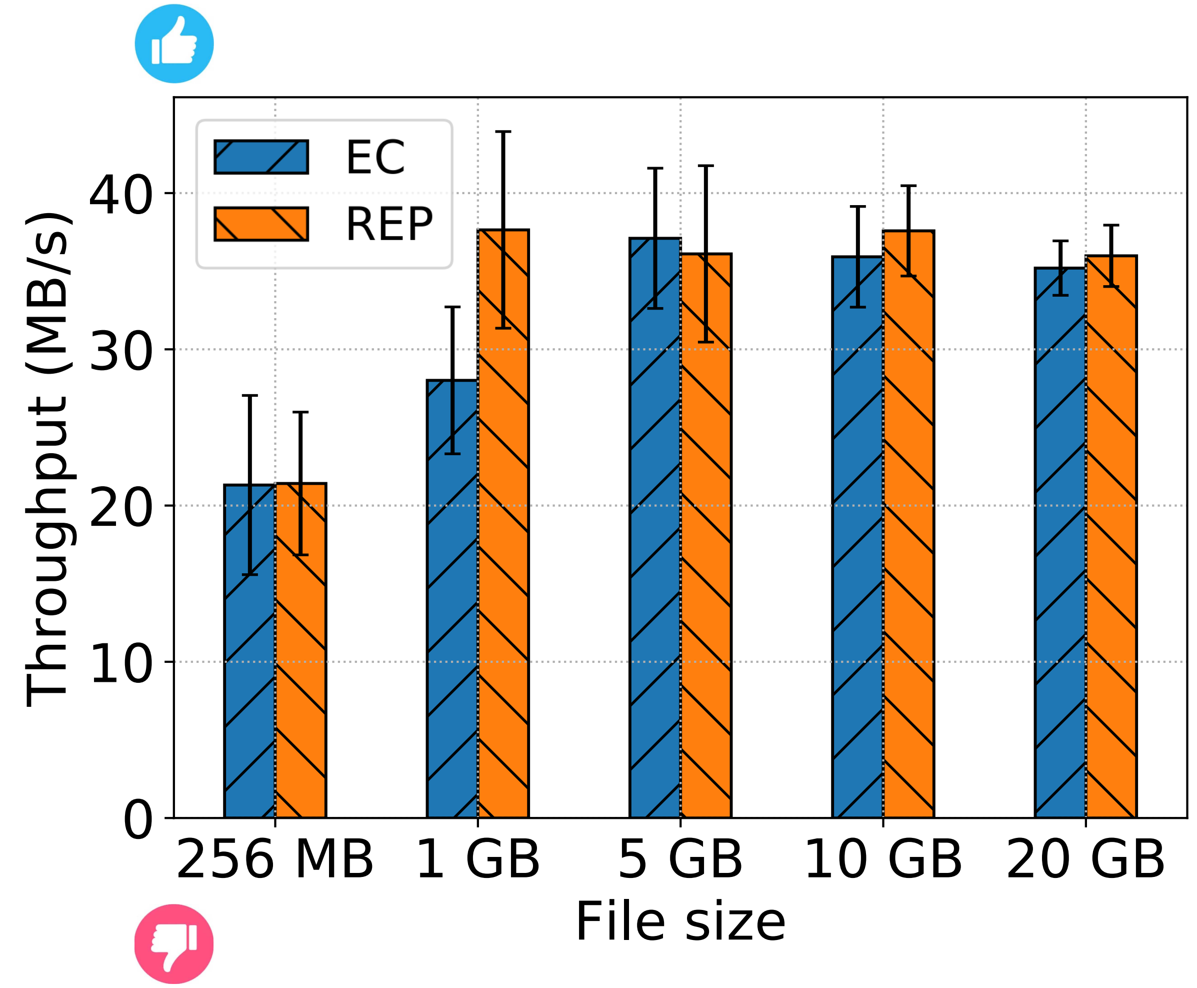
Under RS(6,3), **50%** more data goes from the client to cluster but **50%** less data is written to disk.

Reading data under EC - distinct files

```
./hadoop fs -get
```



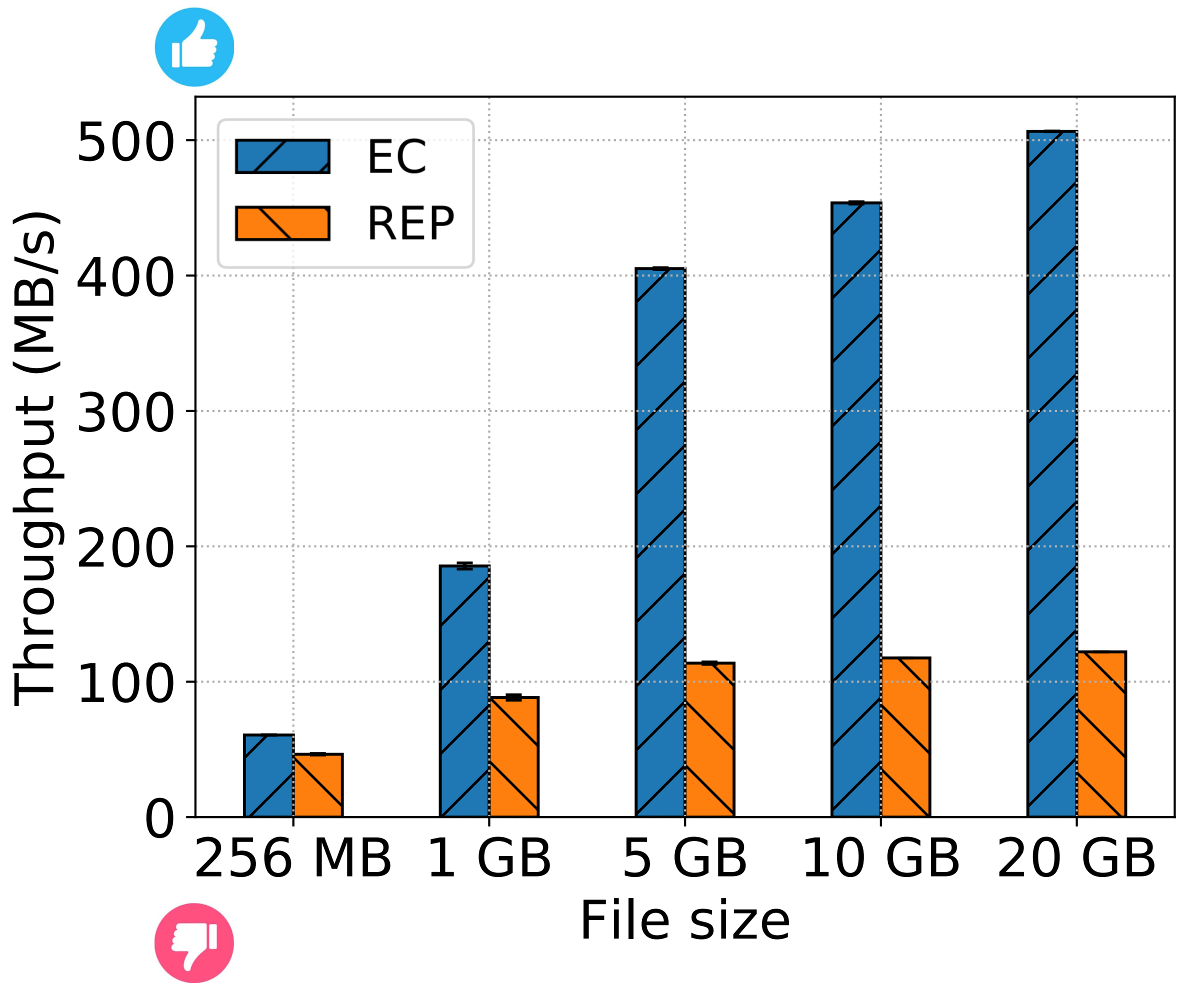
Single client



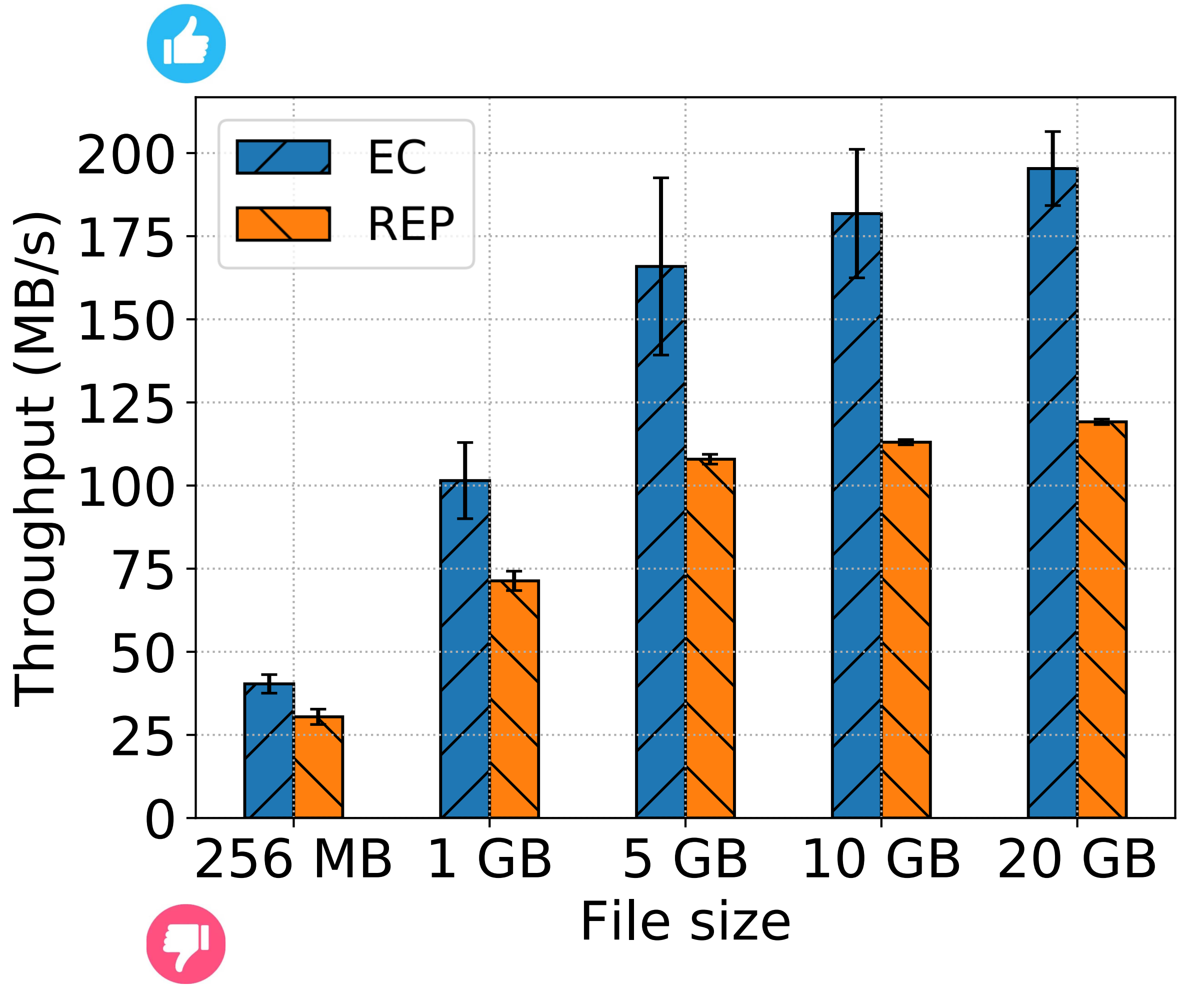
40 concurrent clients

When reading under EC, multiple disks are leveraged in parallel. However, concurrent reads cause stragglers.

Reading data under EC - same file



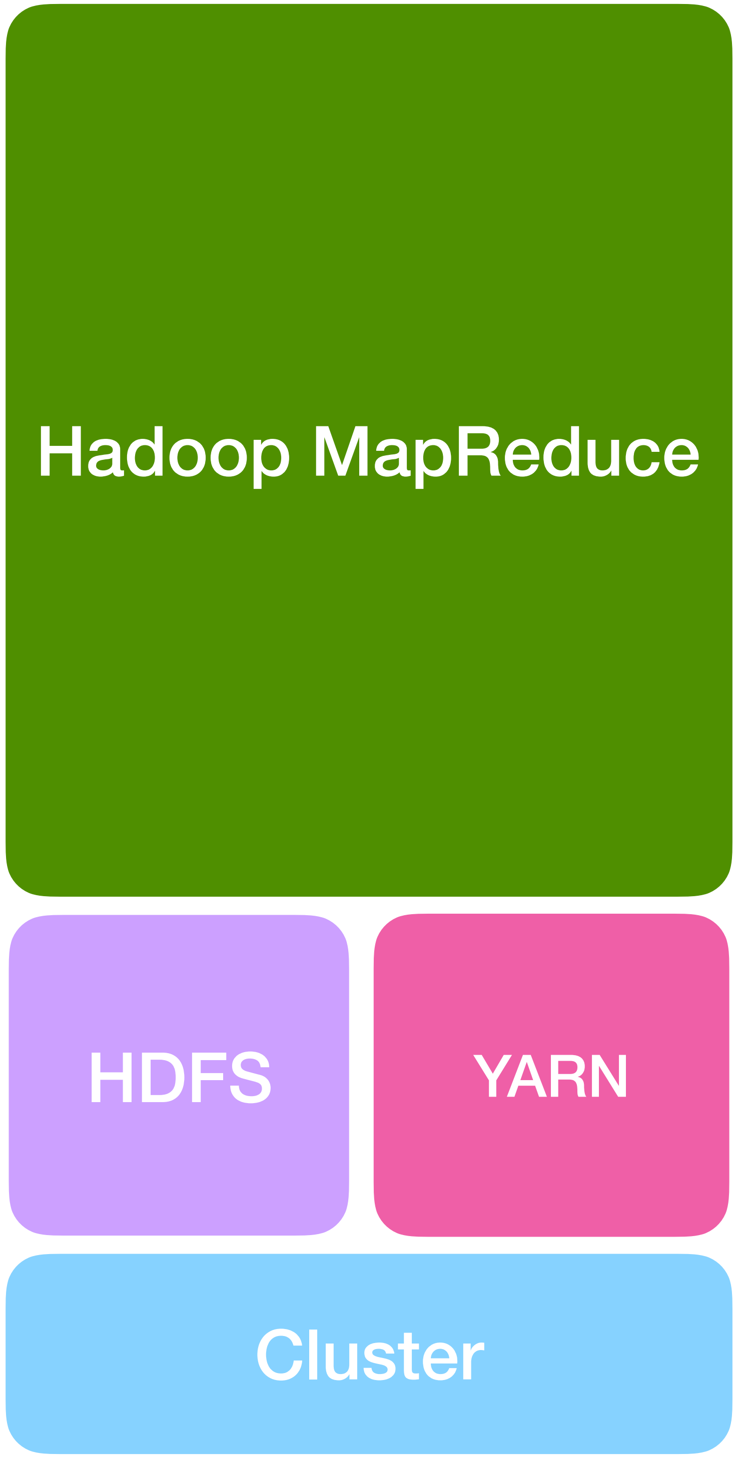
5 concurrent clients



40 concurrent clients

Stragglers can still be seen when reading the same file. EC benefit more from OS caches than replication as data chunks are always read from the same node.

Methodology (MapReduce)



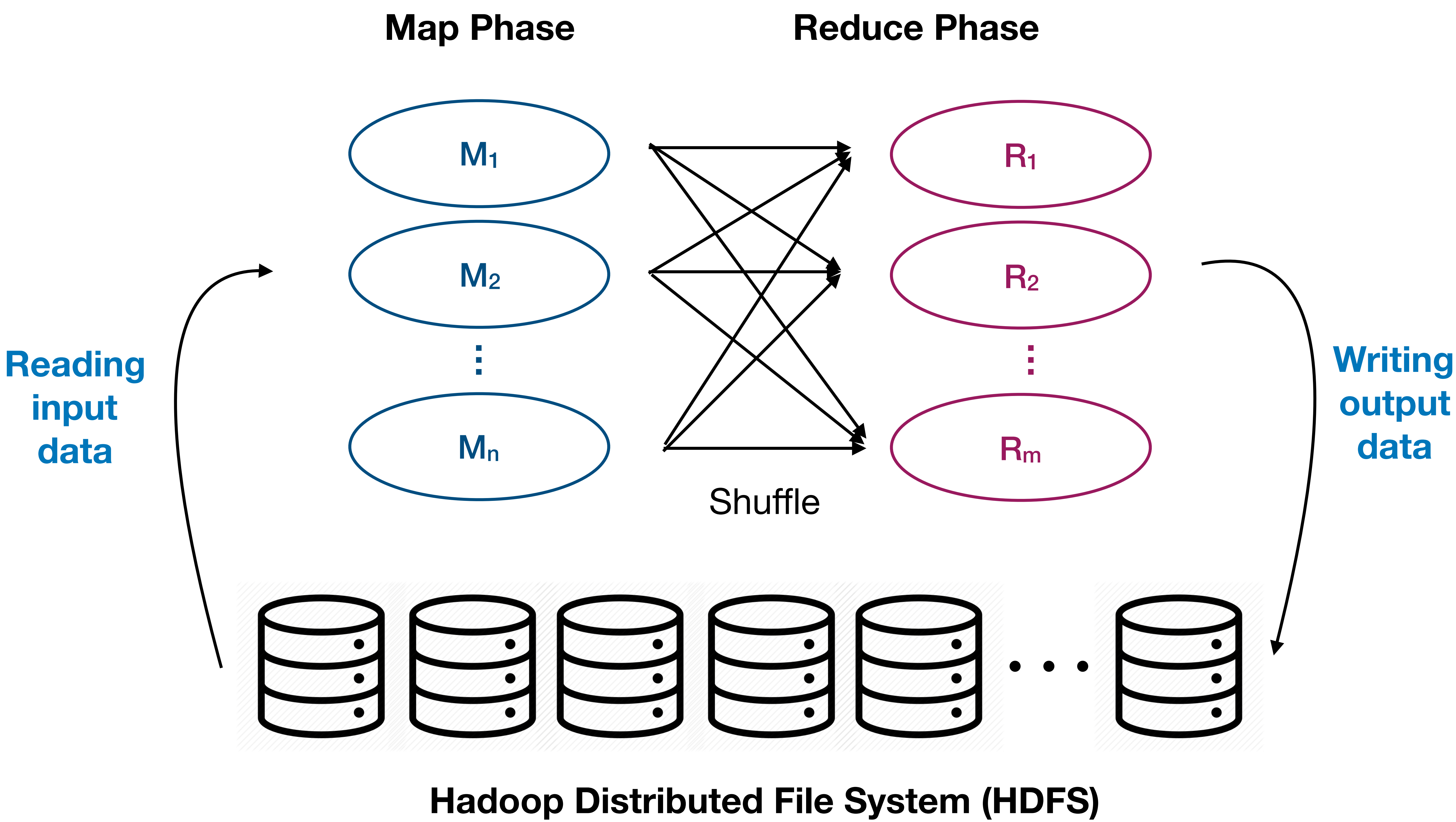
- Benchmarks
  - **Sort** (shuffle intensive)
  - **Wordcount** (map intensive)
  - **Kmeans** (Machine Learning application)
- Software configurations: Overlapping and non-overlapping shuffle, failures, RS schemes and disk persistence.
- Hardware configurations: HDD and MEM / 1 and 10 Gbps
- Performance metric: Job execution time (seconds)

8 containers per node (one per core with 1GB memory)

Block size: **256 MB** - Replication factor: **3** - EC policy: **RS(6, 3)**

21 machines with 8-cores processor, 64GB of main memory, and one HDD at 7.2k RPM. 10 Gigabit Ethernet network.

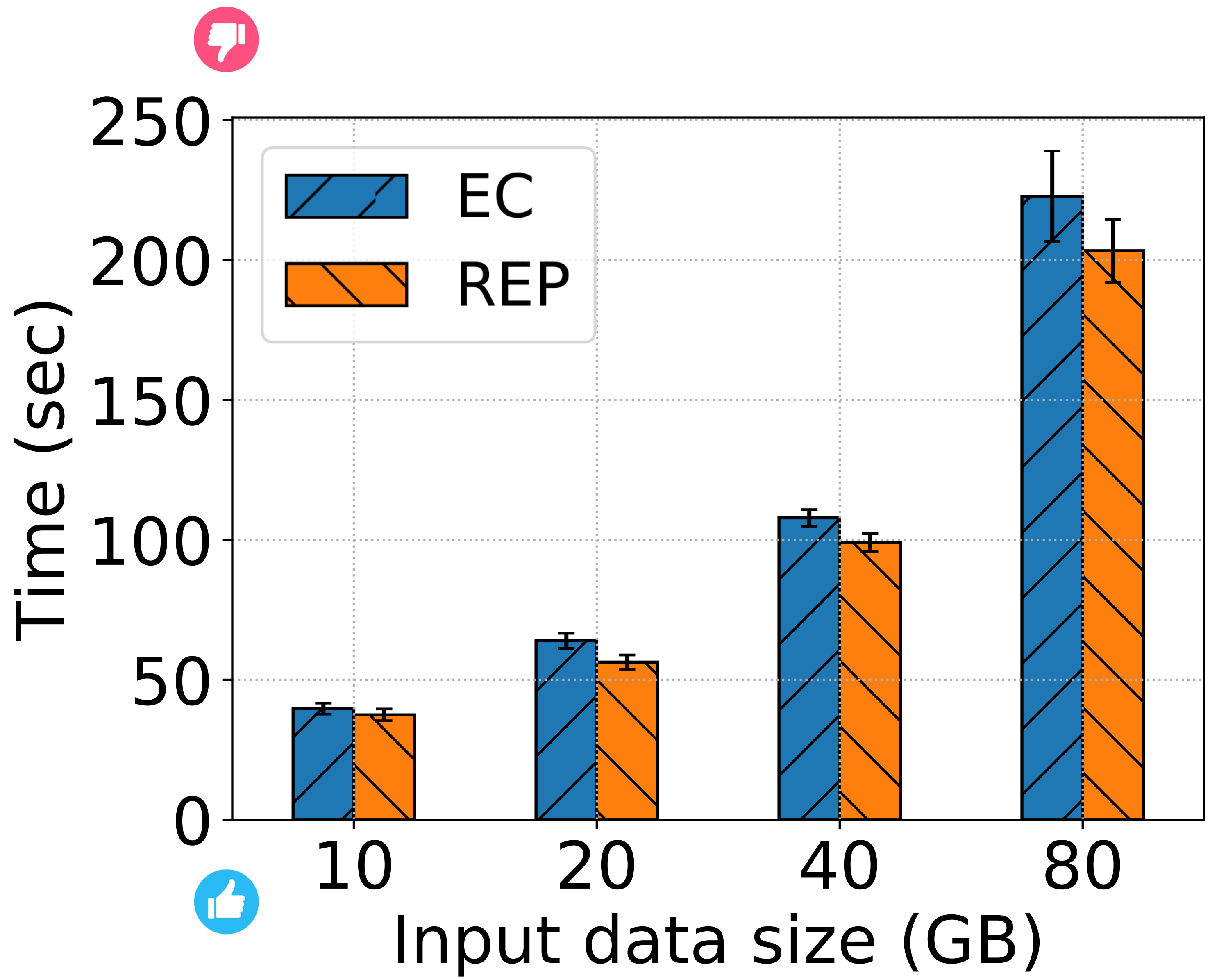
Hadoop MapReduce: Execution overview



Data processing under EC

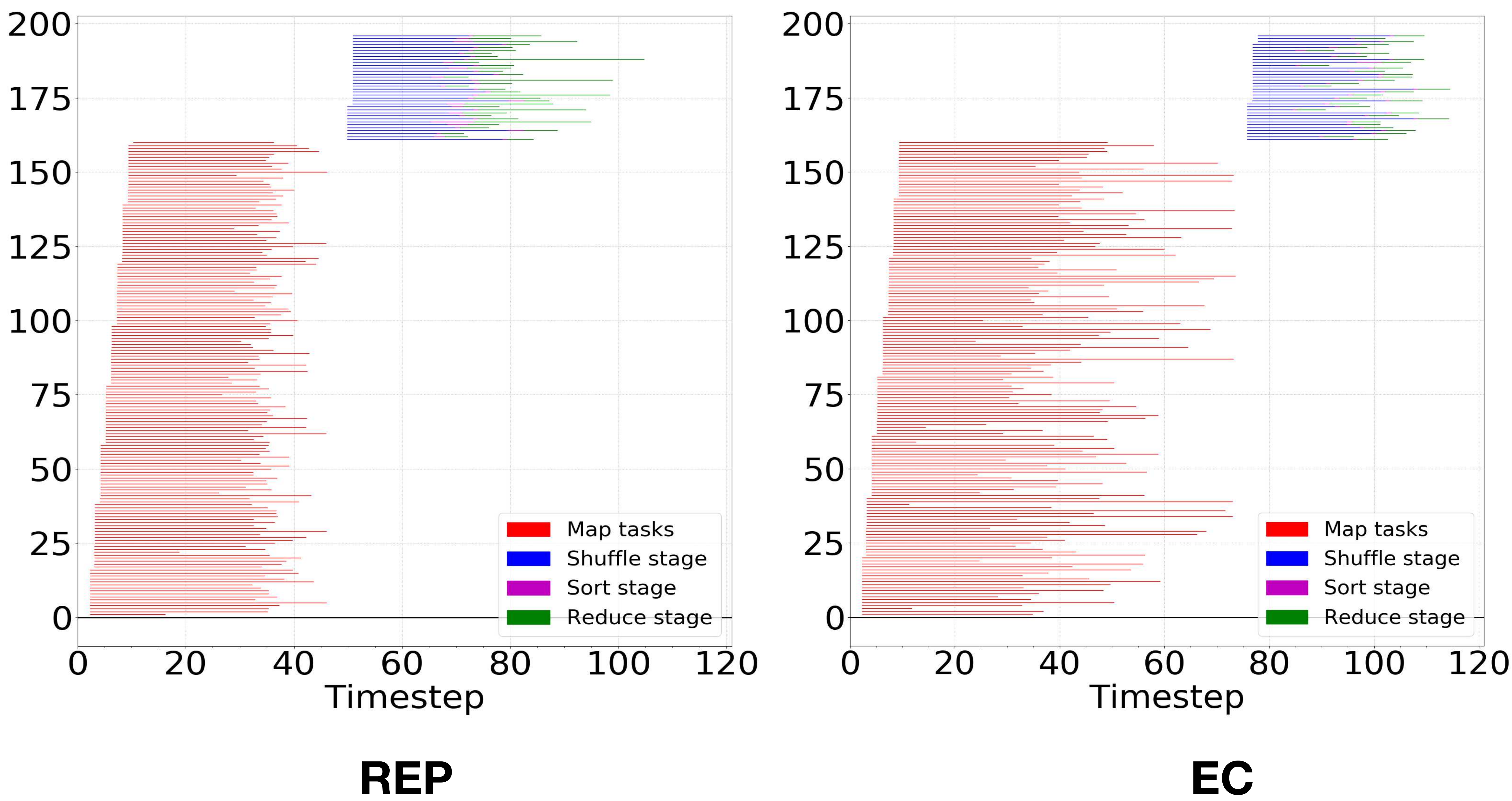
Job execution time of Sort application

Non-overlapping Shuffle,
HDD,
10 Gbps



Zoom-in on tasks runtimes distribution

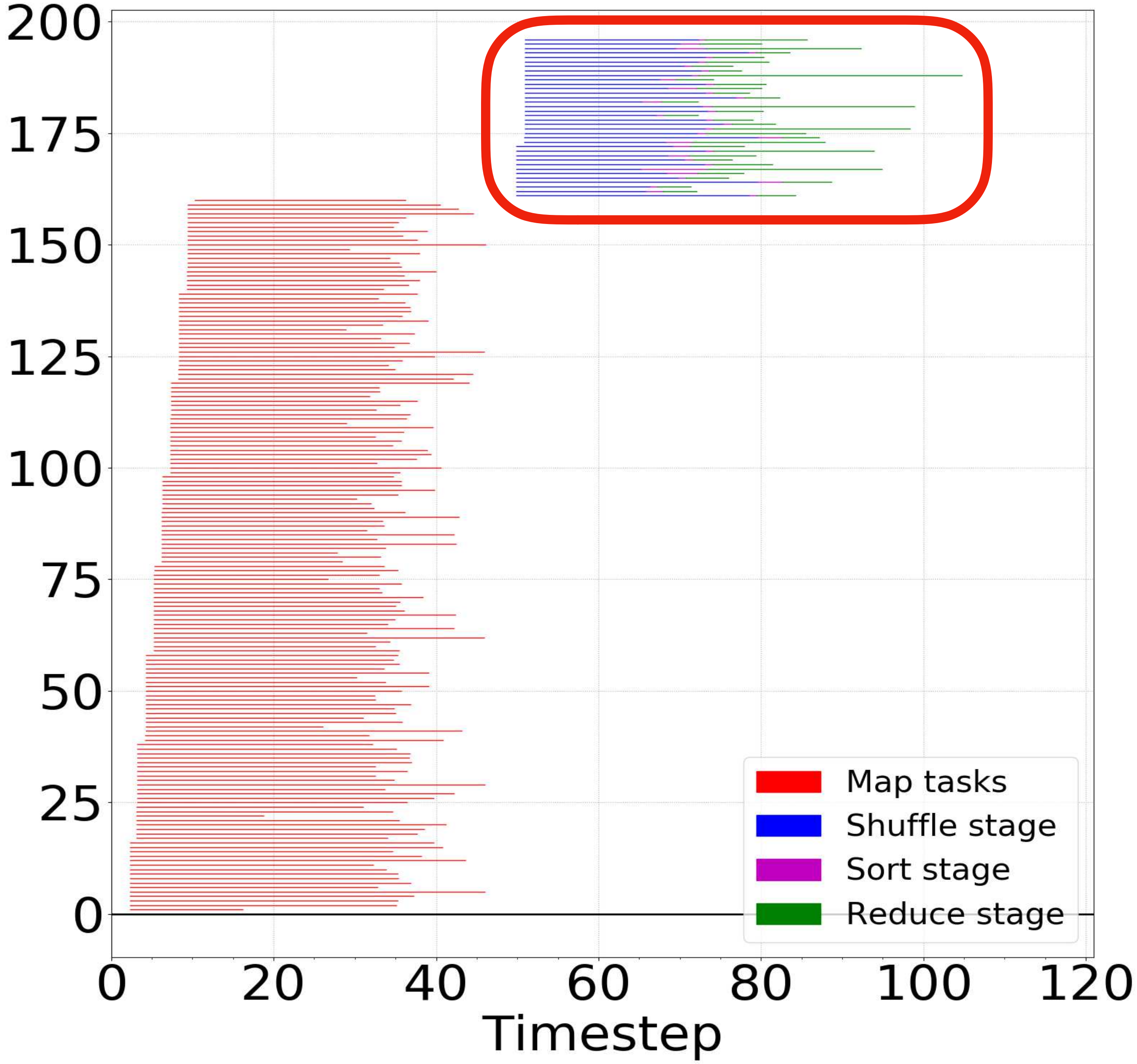
Sorting 40 GB



Zoom-in on tasks runtimes distribution

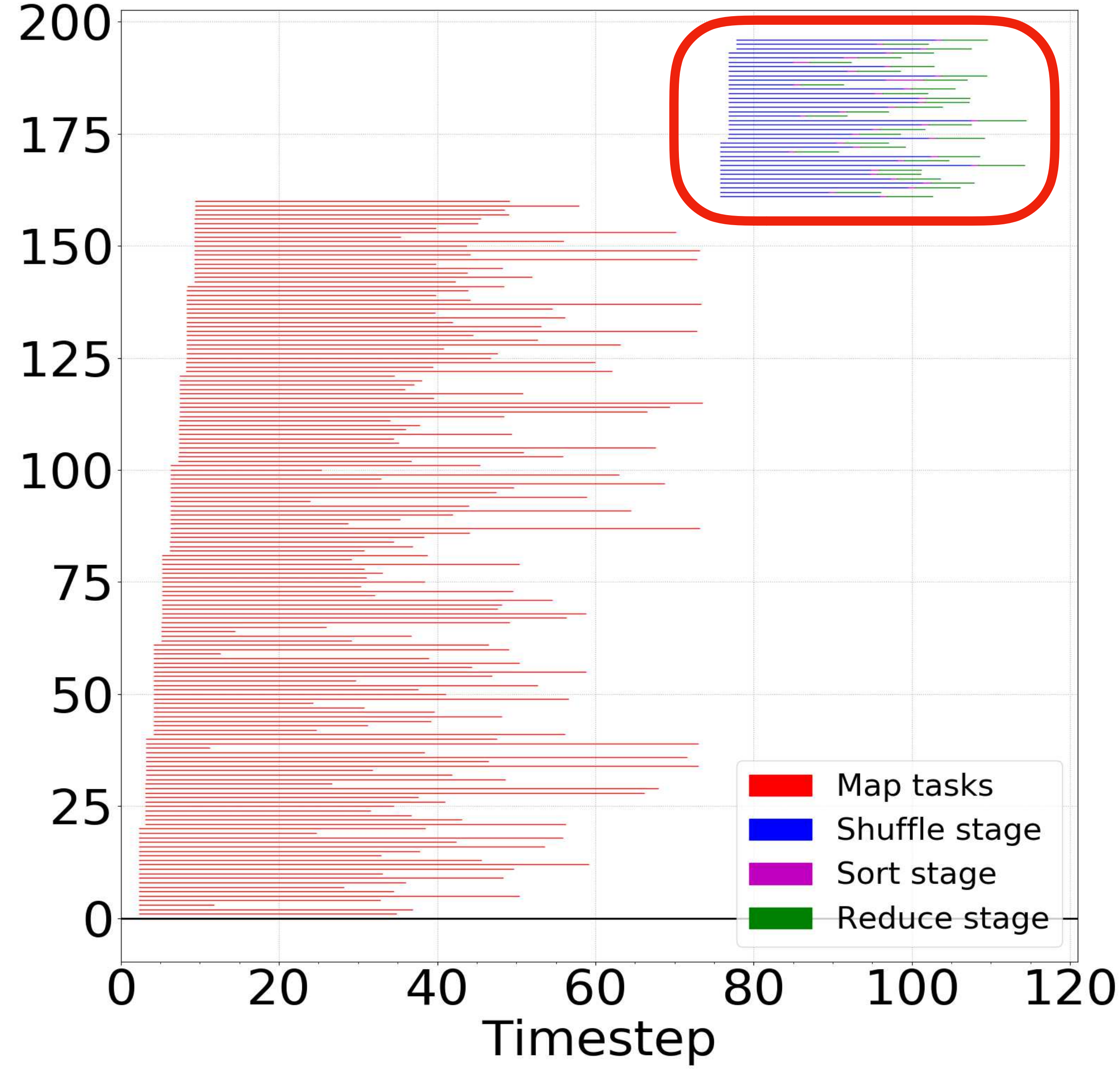
Sorting 40 GB

54s



REP

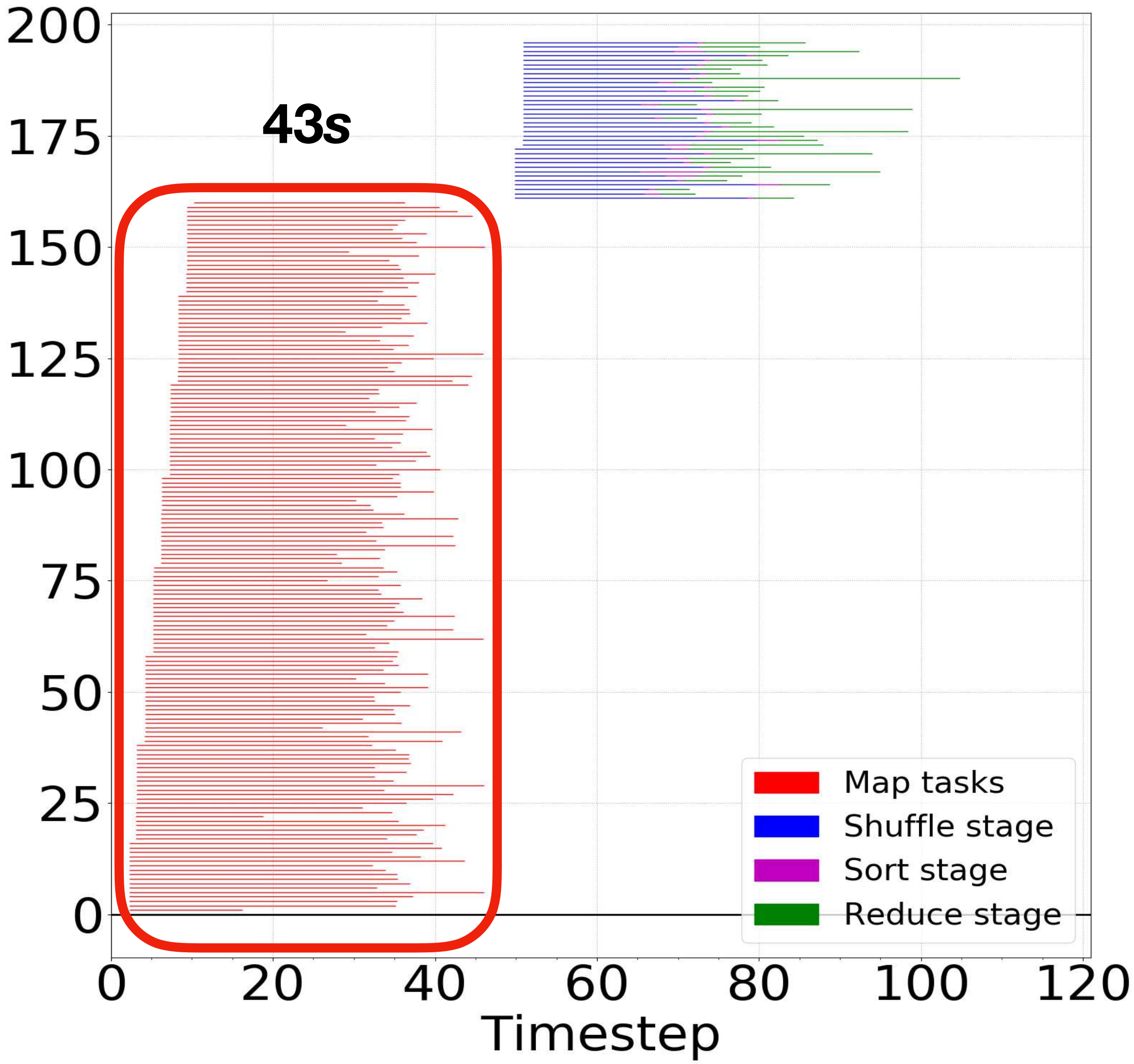
38s



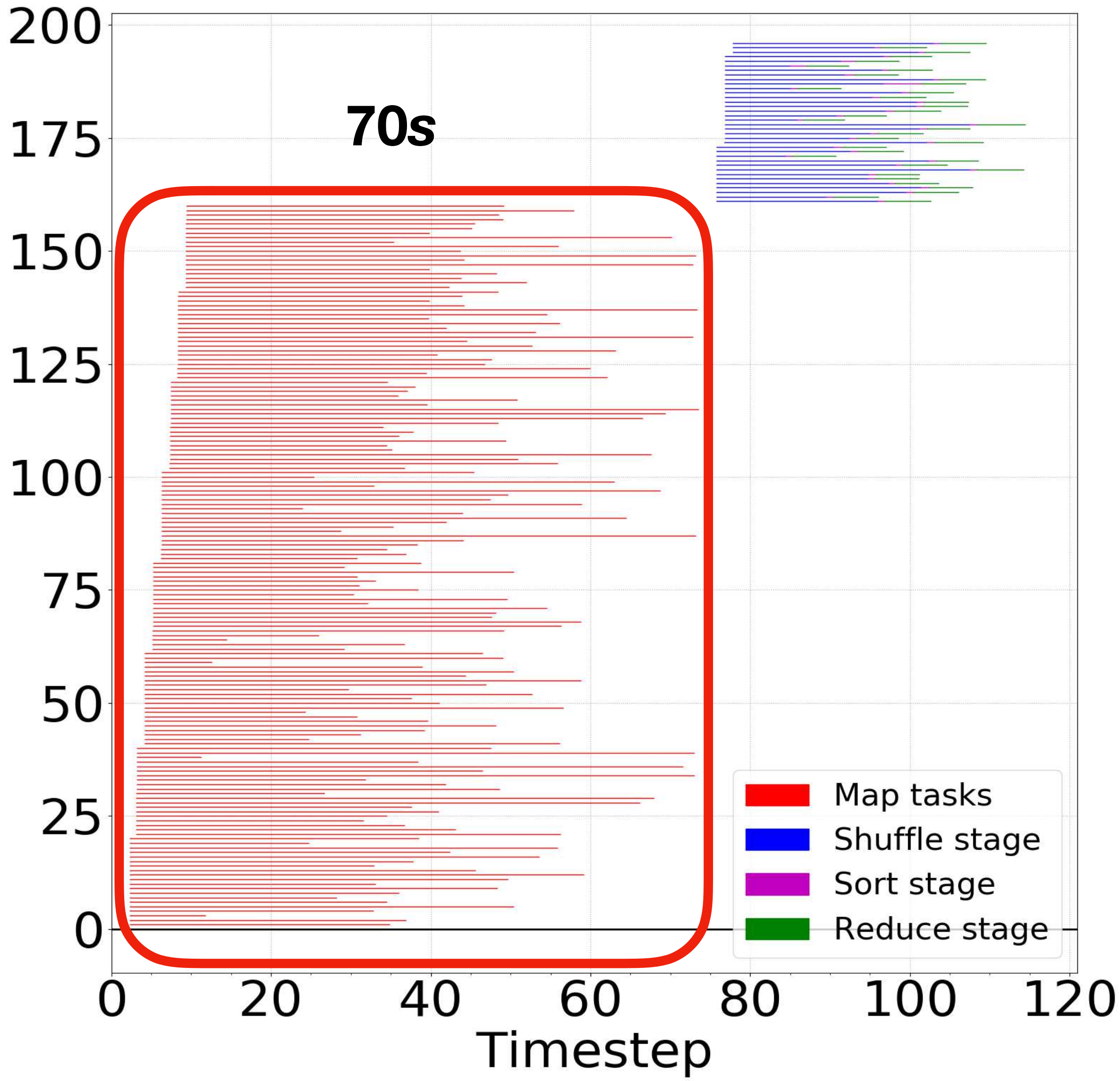
EC

Zoom-in on tasks runtimes distribution

Sorting 40 GB

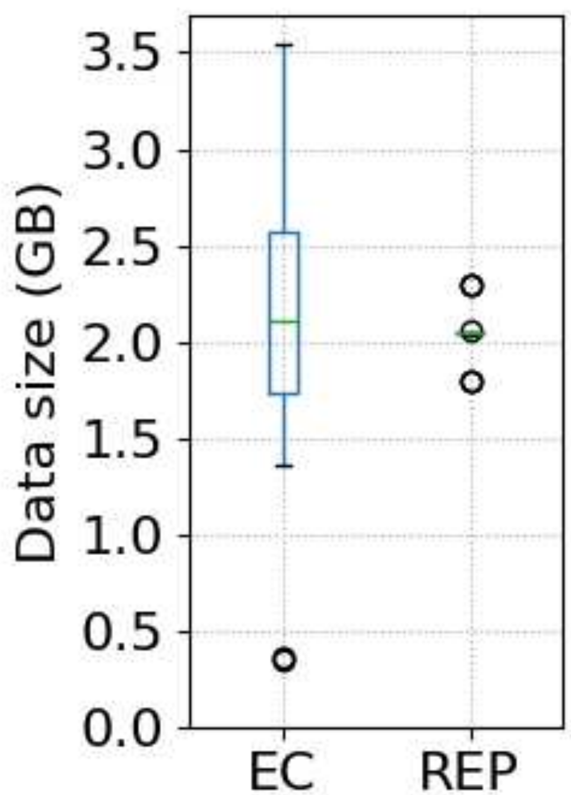


REP



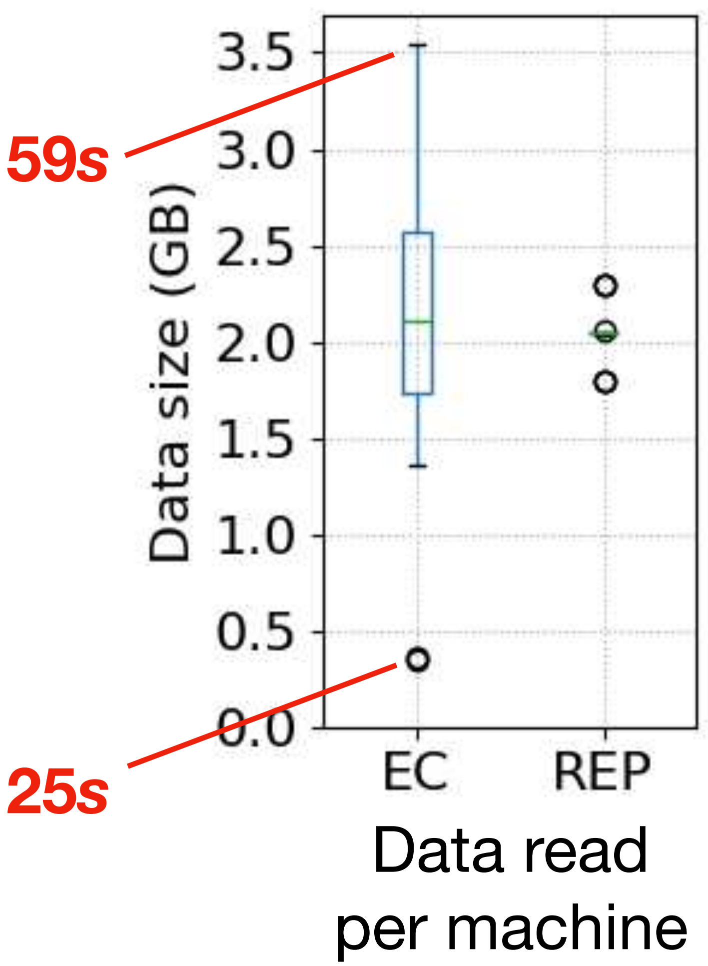
EC

Data read skew as a root cause for performance degradation under EC

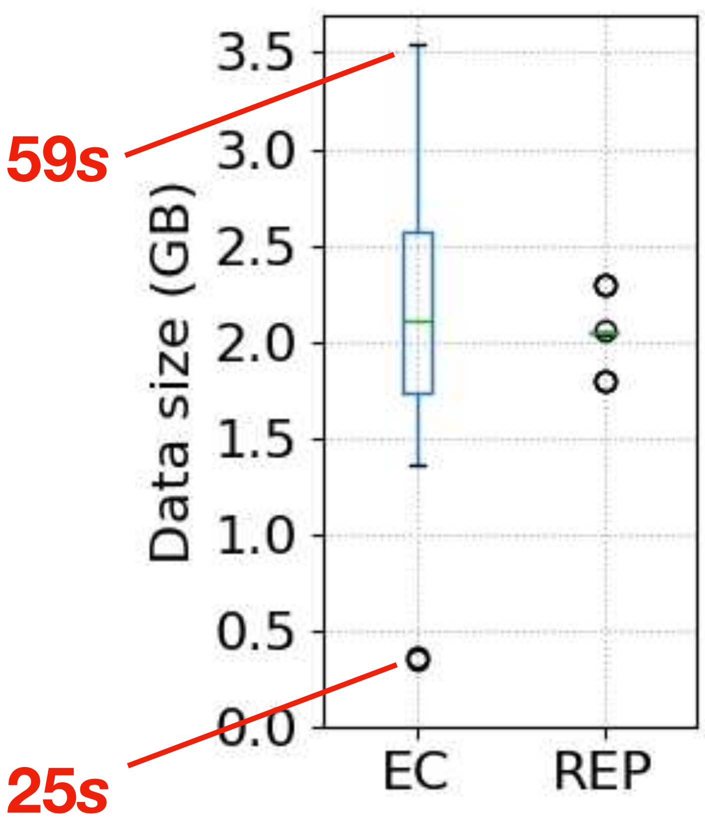


Data read
per machine

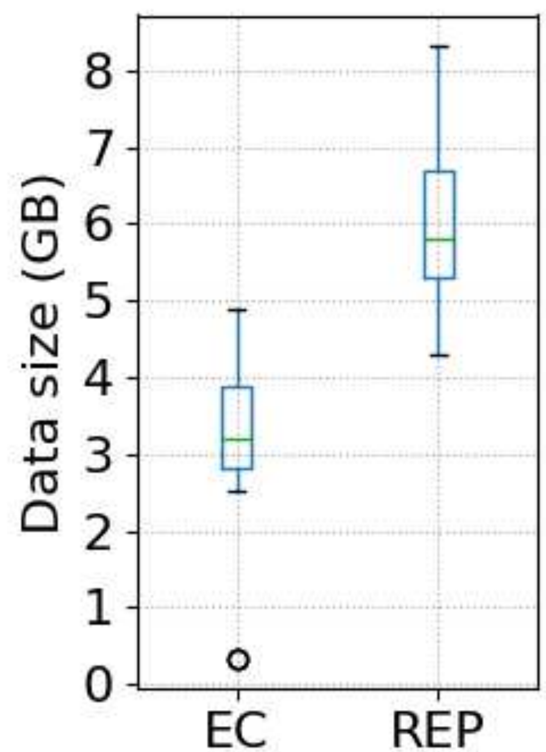
Data read skew as a root cause for performance degradation under EC



Data read skew as a root cause for performance degradation under EC

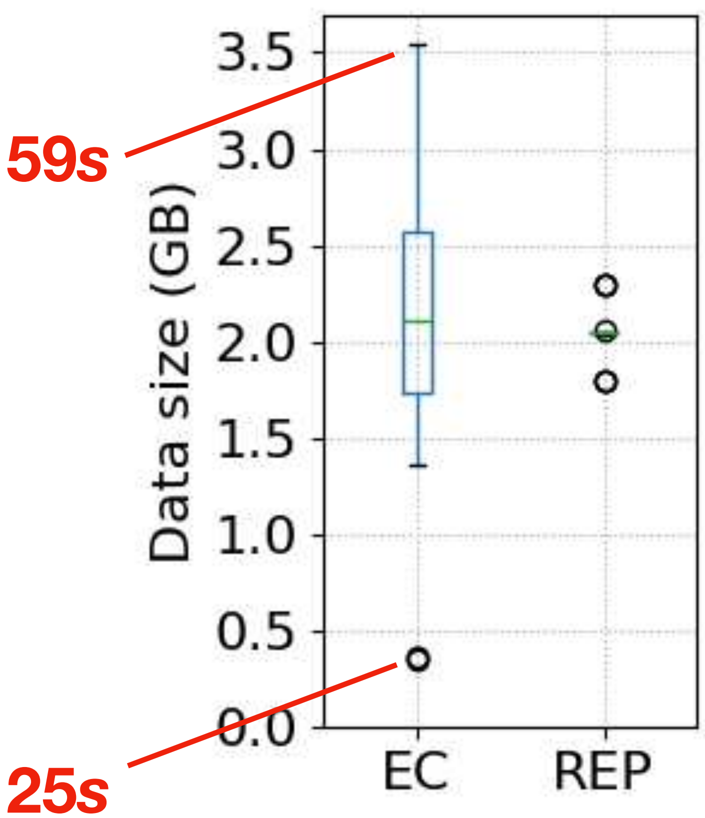


Data read
per machine

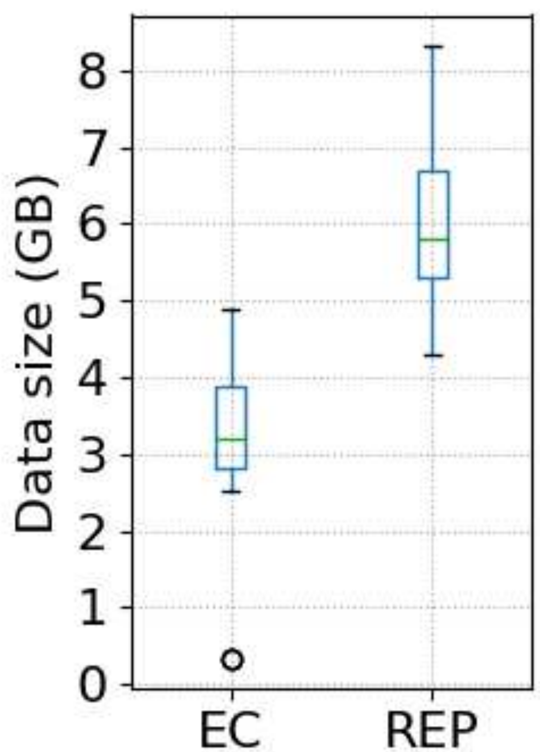


Block distribution in
HDFS

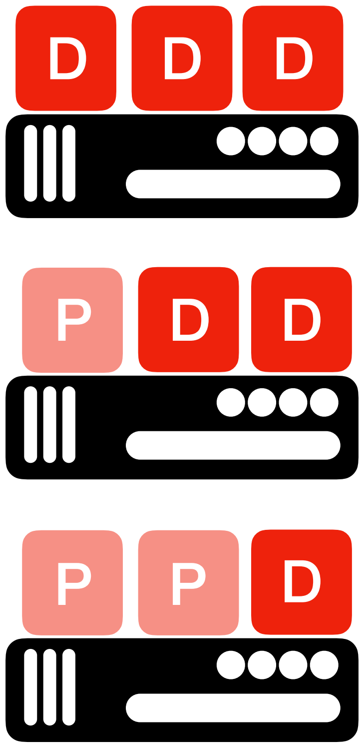
Data read skew as a root cause for performance degradation under EC



Data read per machine



Block distribution in HDFS



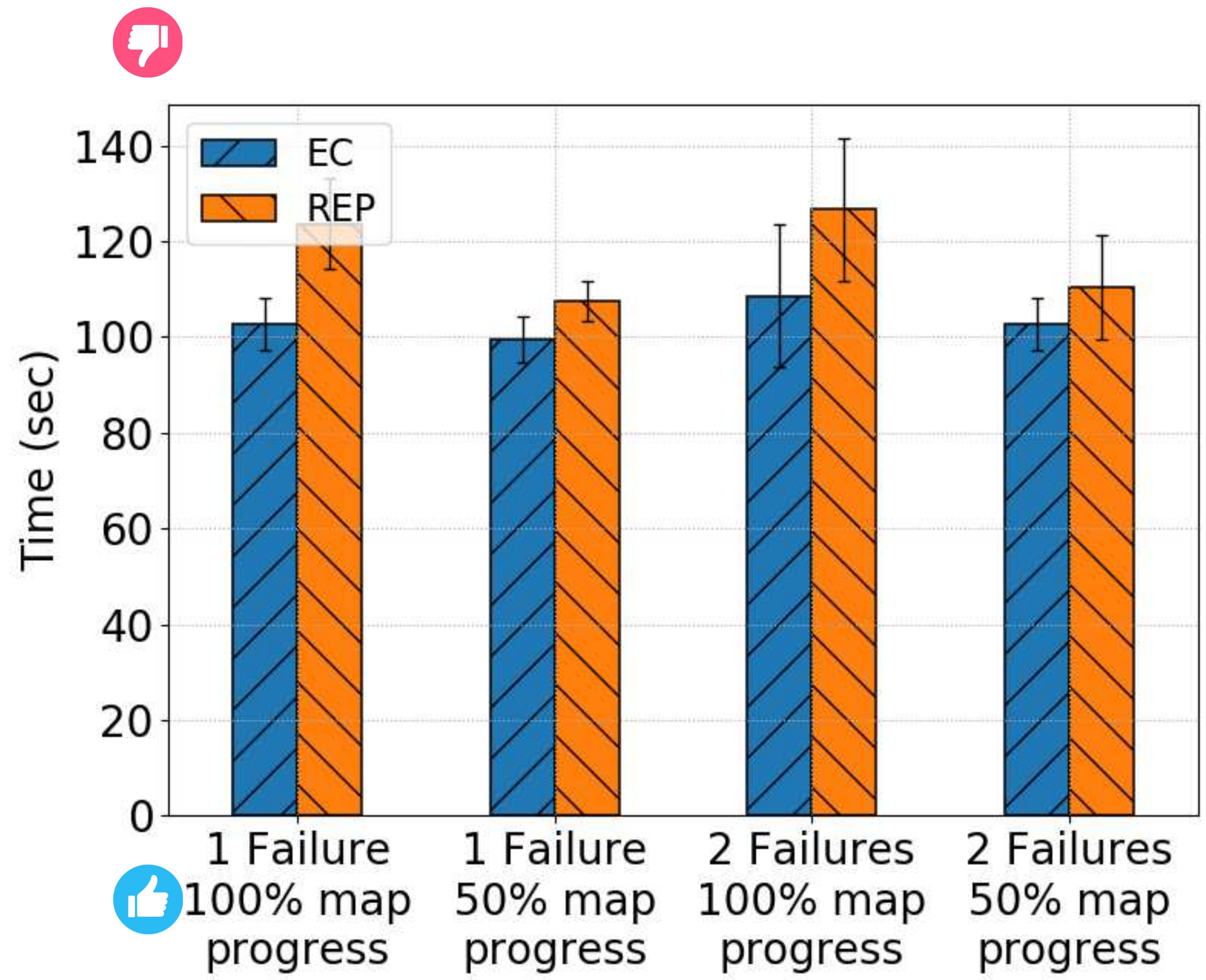
Data/Parity

Though they have different functionalities, original and parity chunks are treated the same when distributed across DNs. This results in a high variation in the data reads amongst the nodes.

The impact of failures

Non-overlapping Shuffle,
HDD,
10 Gbps

Job execution time of Sort application - 40 GB



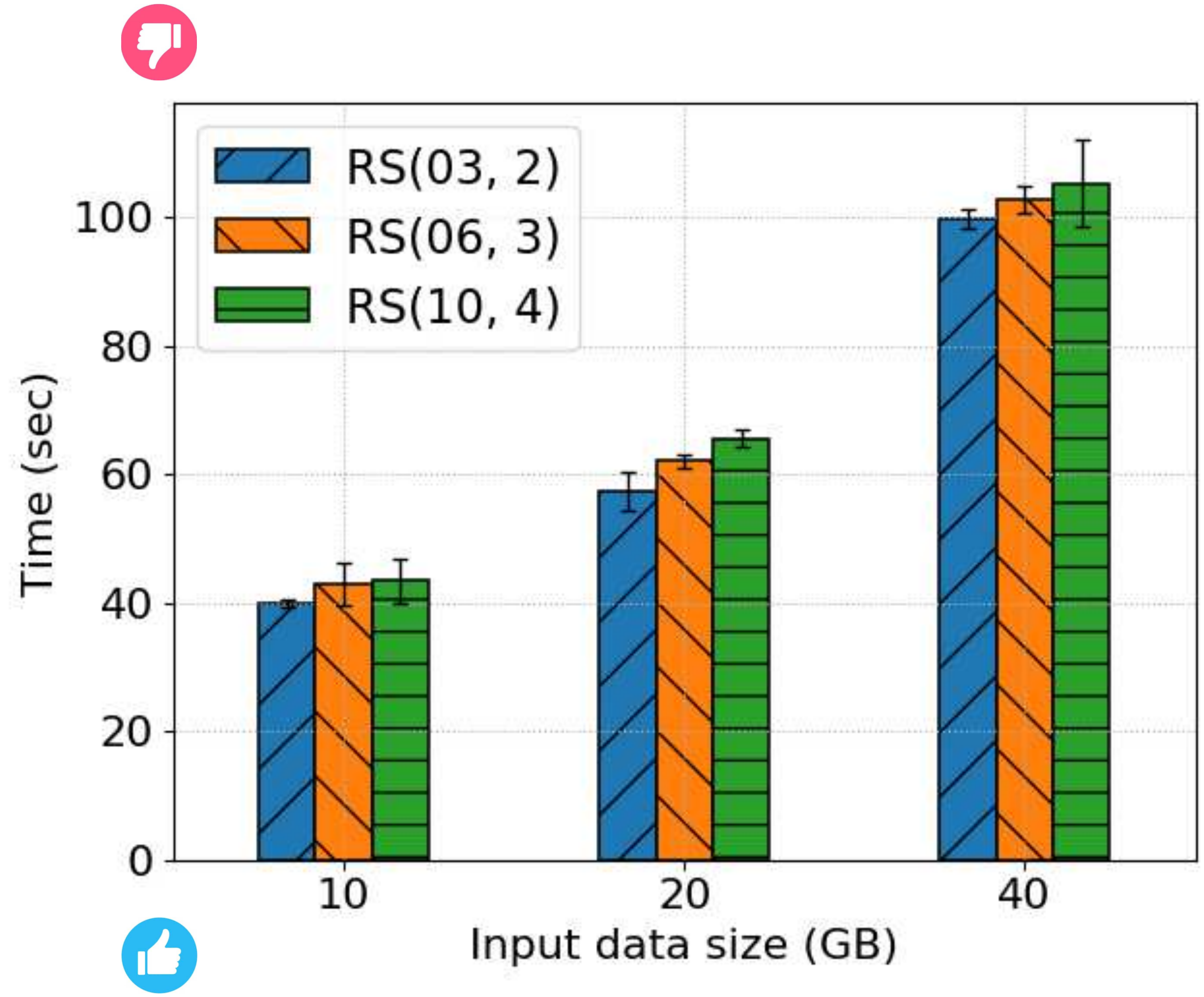
Degraded reads under EC with striped layout introduces negligible overhead (unlike contiguous layout[1]) and therefore the performance under EC is comparable to that under REP.

1 Li et al., Degraded-First Scheduling for MapReduce in Erasure-Coded Storage Clusters, DSN, 2014

The impact of RS schemes

Non-overlapping Shuffle,
HDD,
10 Gbps

Job execution time of Sort application

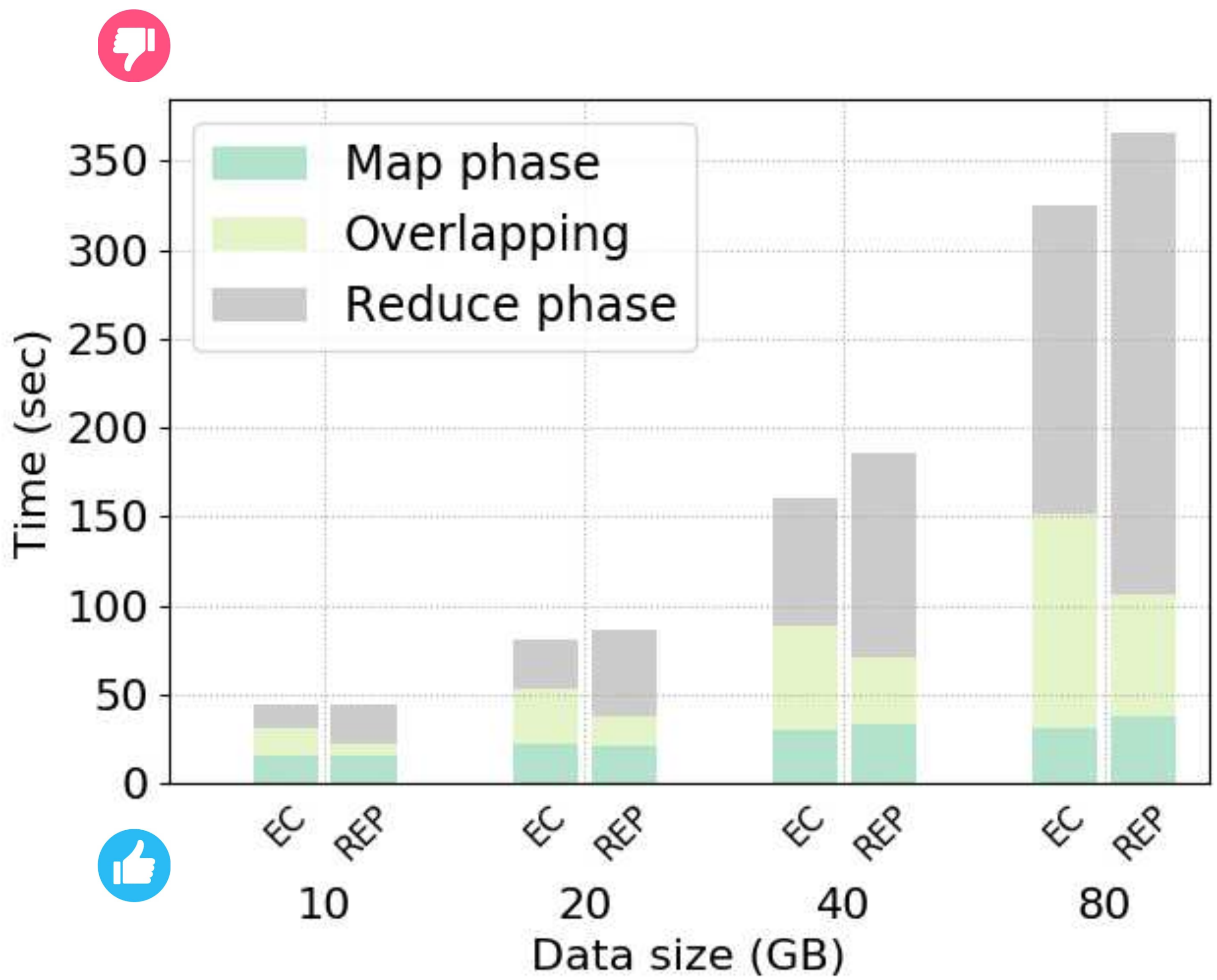


While increasing the stripe size can improve failure resiliency, it reduces local data accesses (map inputs) and increases the probability of data read imbalance.

The impact of slow networks

Overlapping Shuffle,
HDD,
1 Gbps

Job execution time of Sort application

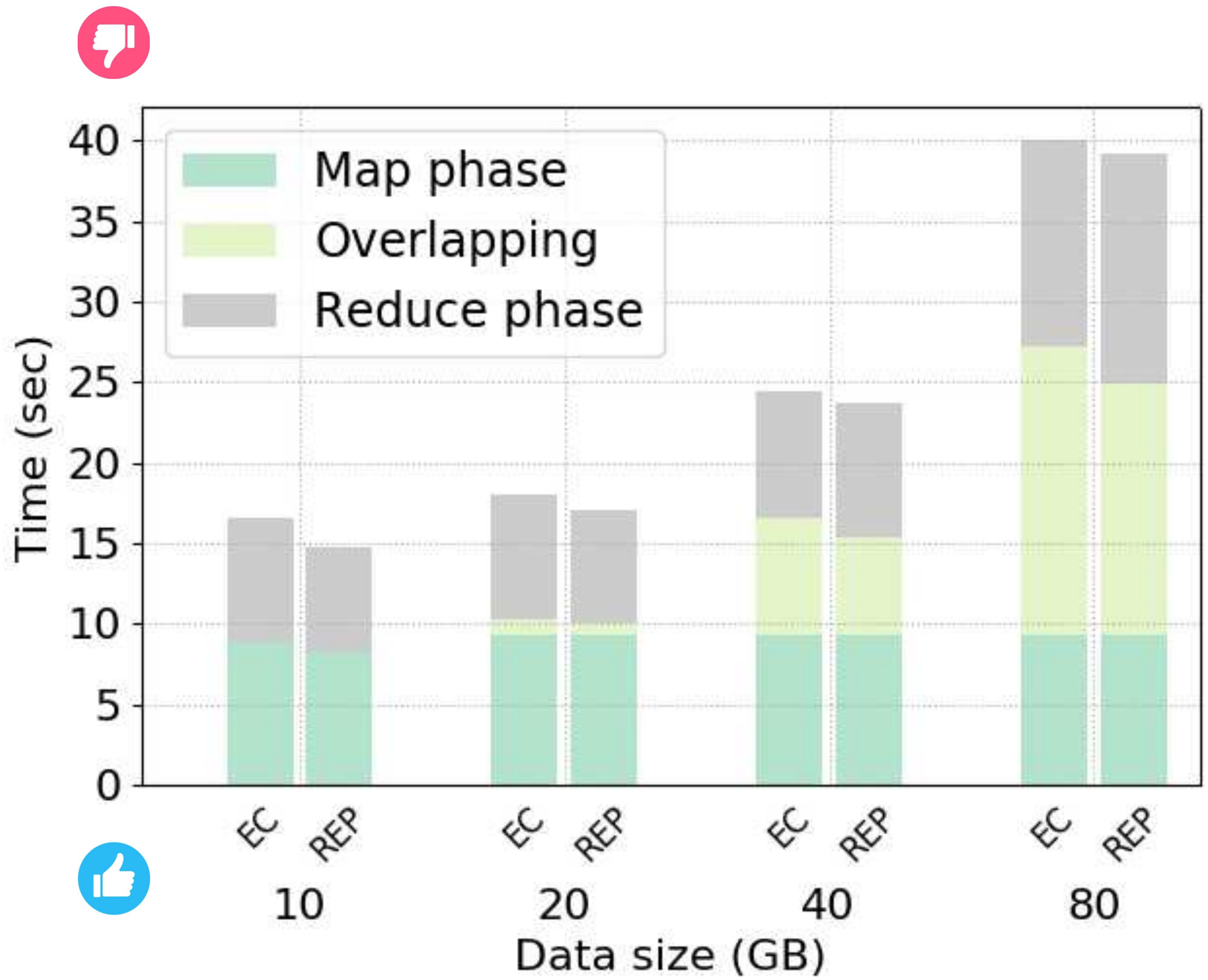


Reading the input data under EC is slightly affected when the network bandwidth is reduced. However, EC brings considerable advantage when the output dat size is big.

The impact of memory

Overlapping Shuffle,
MEM,
10 Gbps

Job execution time of Sort application



Using high-speed storage devices eliminate the stragglers caused by disk contention, therefore, EC brings the same performance as replication.

Is it time to revisit EC in Data-intensive clusters?

- EC is a potential data availability technique for large-scale data processing.
- EC performs the same or even better than replication.
- Rooms for improvement:
  - When running MapReduce applications, unbalanced distribution leads to stragglers map task which prolong the job execution time.
  - EC overhead is NOT negligible when storing data in memory[1,2].

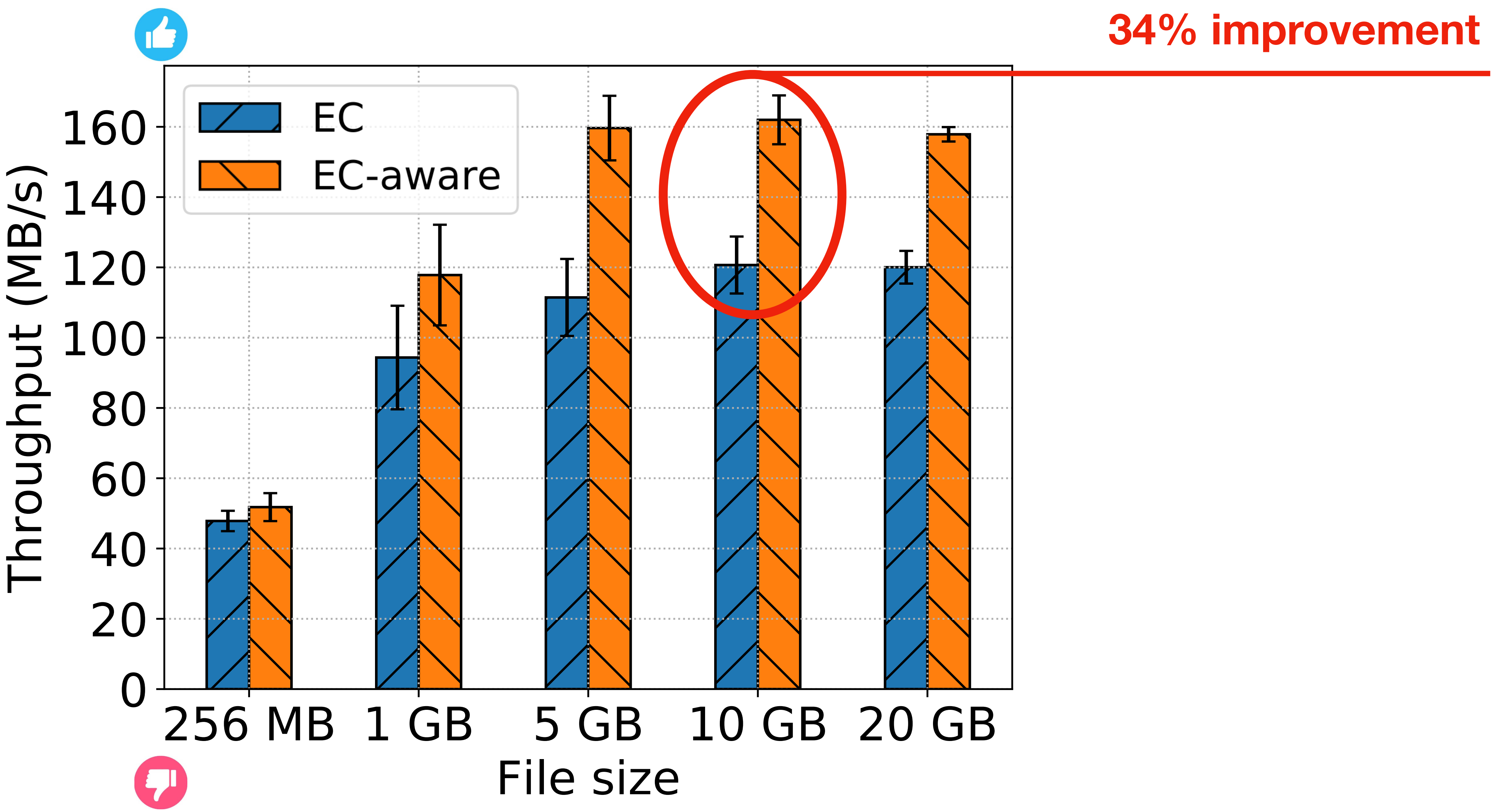
MASCOTS'19

[1] Michael et al., EC-Store: Bridging the Gap between Storage and Latency in Distributed Erasure Coded Systems, ICDCS'18.
[2] Rashmi et al., EC-Cache: Load-Balanced, Low-Latency Cluster Caching with Online Erasure Coding, OSDI'16.

EC-aware data placement

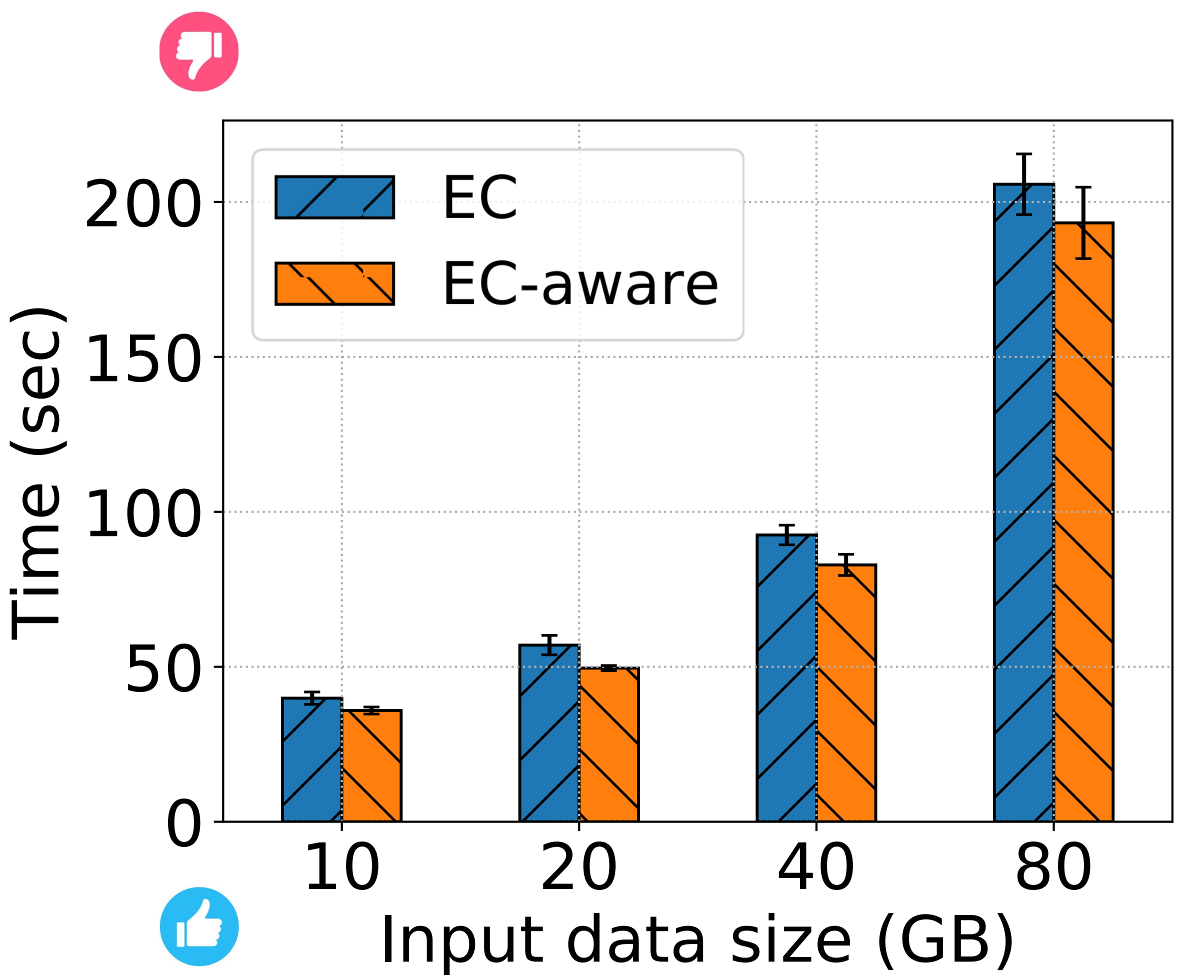
- Data distribution has a direct impact on the job performance.
- The goal: balance the read between different datanodes.
- Distribute the (data and parity) chunks considering their semantics:
  - Takes the semantic of the chunks during the the placement.
  - Is implemented in HDFS 3.

Results: EC-aware placement improves the performance of data read

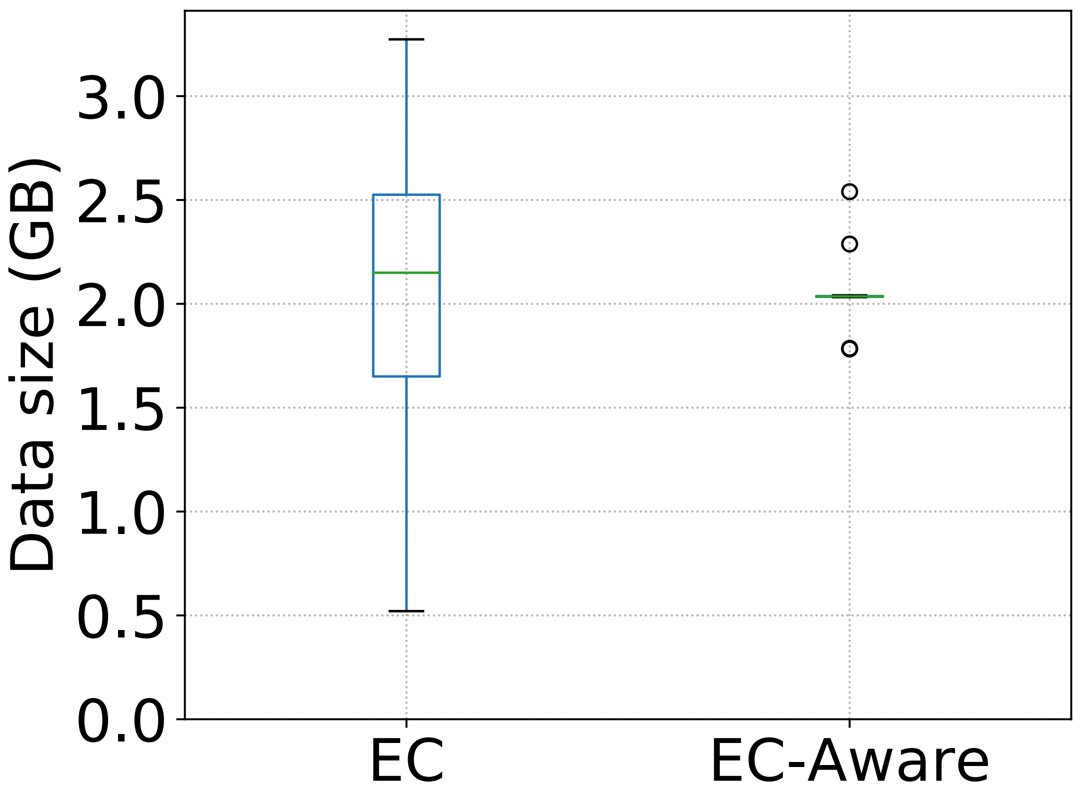


Read distinct files by 10 clients

Results: Making data placement EC-aware eliminates data read skew



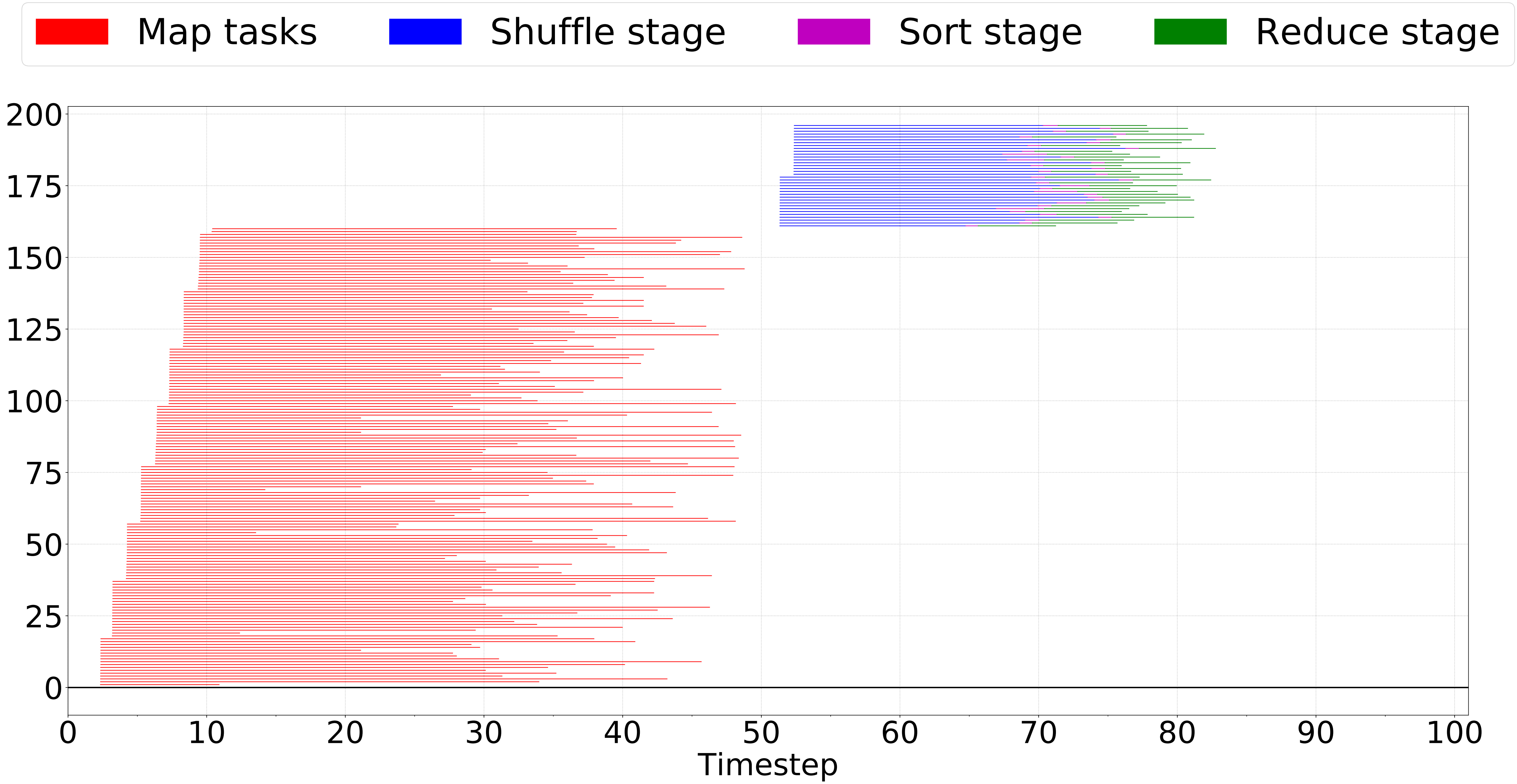
Job execution time



Data read per machine
(40 GB input size)

Sort application

Results: the need for dynamic task scheduling



Tasks timeline of Sort application with 40 GB input size

Conclusion & Perspectives

Conclusion

Service provisioning in distributed infrastructures

CCGrid’18

- VMIs retrieval in heterogeneous WAN: Nitro
  - Real system, exploits deduplication, provides optimal network-aware chunk scheduling, and is evaluated on top of Grid’5000.

ICCCN’19

- Network-aware container image placement: KCBP and KCBP-WC
  - Formal model, incorporates two K-center based network-aware placement algorithms, and is evaluated through simulation.

Big Data processing under erasure coding

MASCOTS’19

- Understand the performance of data-intensive applications under EC with <u>striped block layout</u>
  - Experimental study: HDFS and MapReduce, various software and hardware configurations.
- EC-aware chunk placement algorithm
  - Balances the data read, and is implemented and evaluated in HDFS.

Perspectives

- Dynamic placement of container images
  - Adapts with adding new images and sites.
  - Considers image access patterns.
- EC-aware and access-aware task scheduling
  - Balances the temporal I/O load between the datanodes.
  - Integrates EC-aware data retrieval at the scheduler level.
- Data processing in Edge **Poster in ICPP'19**
  - Map tasks under EC suffer from obvious performance degradation.
  - Network-aware task placement and chunk retrieval.