MET: Workload aware elasticity for NoSQL

Francisco Cruz, Francisco Maia, Miguel Matos,
Rui Oliveira, João Paulo, José Pereira and Ricardo Vilaça

EuroSys'13
Prague, April 2013

Context

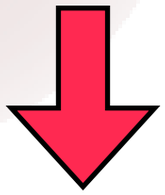
- Elasticity of a specific component– **NoSQL database:**
  - manage the bulk of data from modern web applications.
  - scalable and dependable systems.
  - data partitioned across several computing nodes.
  - high availability and high performance.

Context

- Elasticity of a specific component– **NoSQL database:**
  - manage the bulk of data from modern web applications.
  - scalable and dependable systems.
  - data partitioned across several computing nodes.
  - high availability and high performance.

Context

- Elasticity of a specific component– **NoSQL database:**
  - manage the bulk of data from modern web applications.
  - scalable and dependable systems.
  - data partitioned across several computing nodes.
  - high availability and high performance.



…but **not autonomously elastic**.

Context

› **An external system is required to control when and how to add or remove nodes.**

› Examples of such systems are Amazon's AutoScaling or Tiramola [SIGMOD '12].

Context

› **An external system is required to control when and how to add or remove nodes.**

› Examples of such systems are Amazon's AutoScaling or Tiramola [SIGMOD '12].

However...

› Just based on resource usage metrics.

Context

› **An external system is required to control when and how to add or remove nodes.**

› Examples of such systems are Amazon's AutoScaling or Tiramola [SIGMOD '12].

However...

› Just based on resource usage metrics.

› Oblivious to the data access patterns.

Context

› **An external system is required to control when and how to add or remove nodes.**

› Examples of such systems are Amazon's AutoScaling or Tiramola [SIGMOD '12].

However...

› Just based on resource usage metrics.

› Oblivious to the data access patterns.

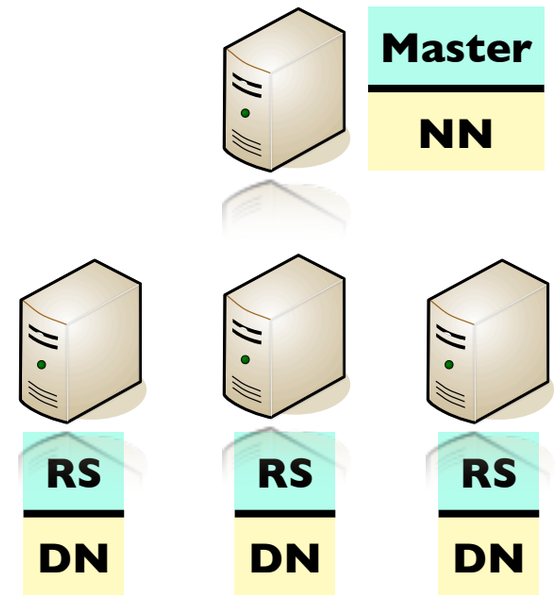
› Homogeneous system, all nodes are considered equal.

HBase

HBase

HBase

- Hierarchical architecture (Master and RegionServers–**RS**).

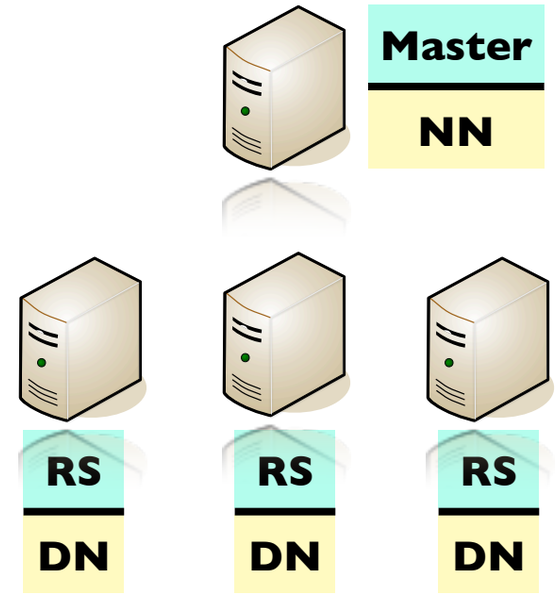


HBase

- Hierarchical architecture (Master and RegionServers–**RS**).
- Multi-dimensional map (HTable) with an unbounded number of attributes.

HTable Customer

ID	Name	Address	...
1			
2			
3			
4			
5			
6			

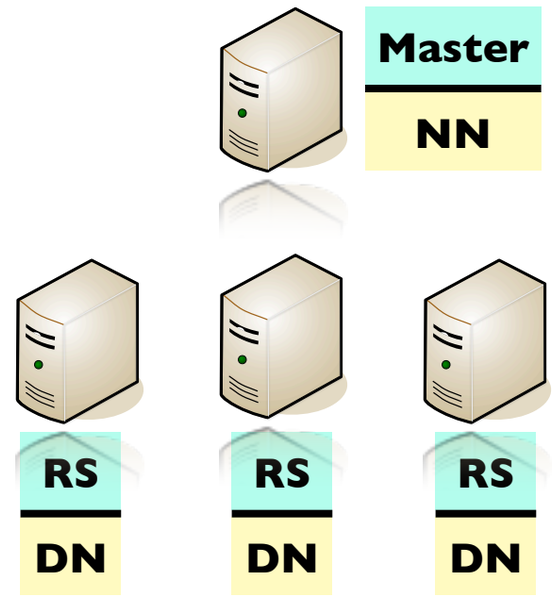


HBase

- Hierarchical architecture (Master and RegionServers–**RS**).
- Multi-dimensional map (HTable) with an unbounded number of attributes.
- Row range horizontally partitioned into **Regions.**

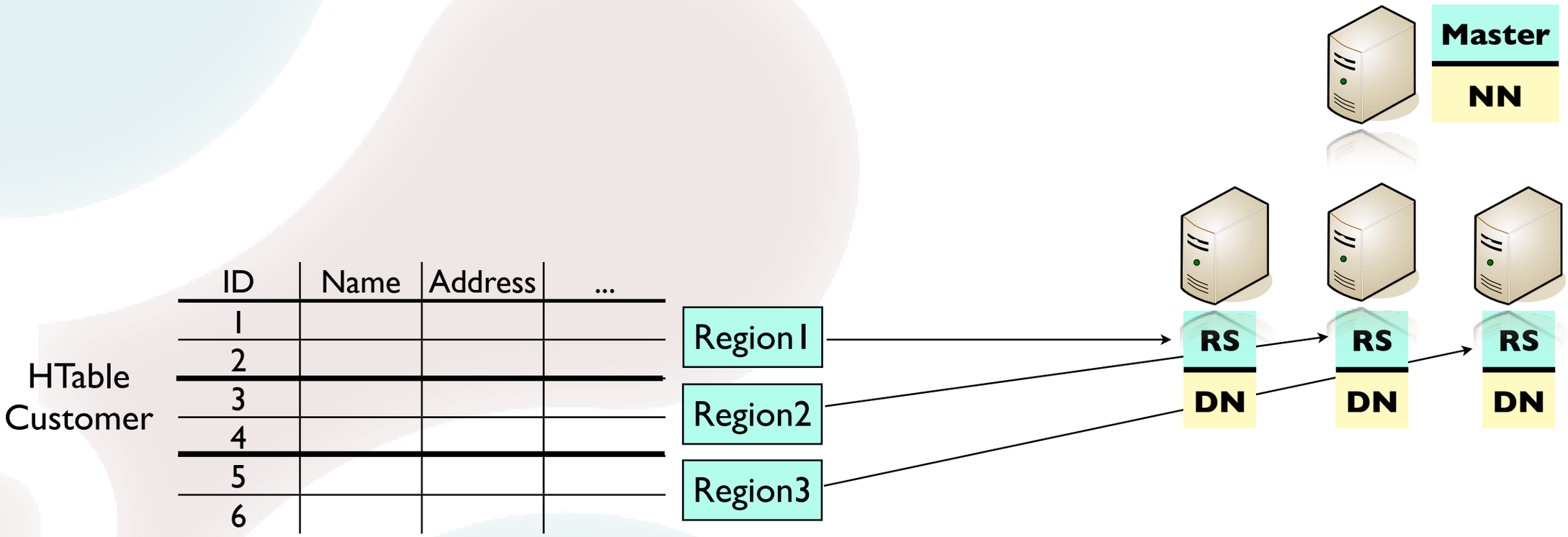
HTable Customer

ID	Name	Address	...	
1				Region1
2				
3				Region2
4				
5				Region3
6				



HBase

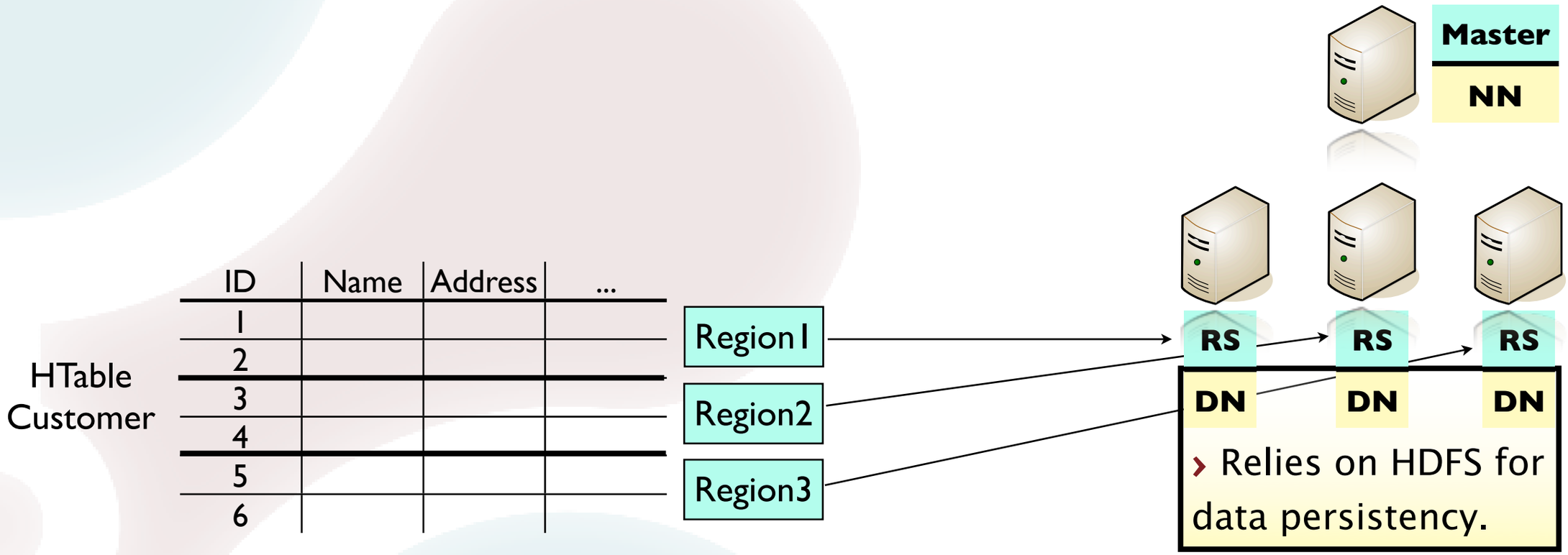
- Hierarchical architecture (Master and RegionServers–**RS**).
- Multi-dimensional map (HTable) with an unbounded number of attributes.
- Row range horizontally partitioned into **Regions.**
- Assignment of **Regions** to **RegionServers** uses a randomized data placement component.



HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

HBase

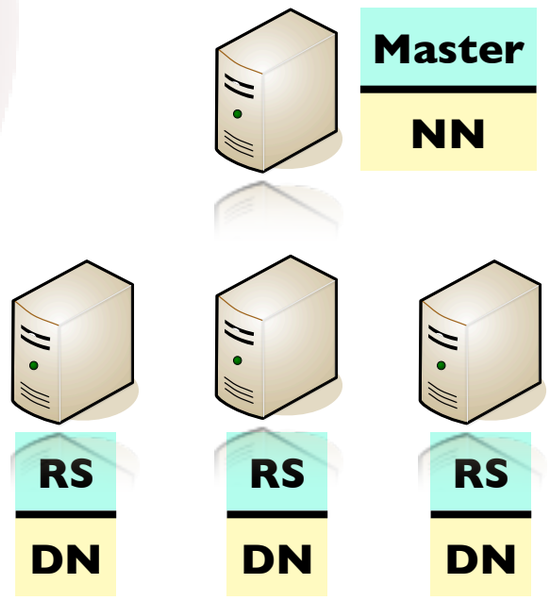
- Hierarchical architecture (Master and RegionServers–**RS**).
- Multi-dimensional map (HTable) with an unbounded number of attributes.
- Row range horizontally partitioned into **Regions.**
- Assignment of **Regions** to **RegionServers** uses a randomized data placement component.



HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

HBase

› **RegionServer**/**DataNode** have many configuration parameters – block size; handler count; behavior of the GC; write buffer size; etc.



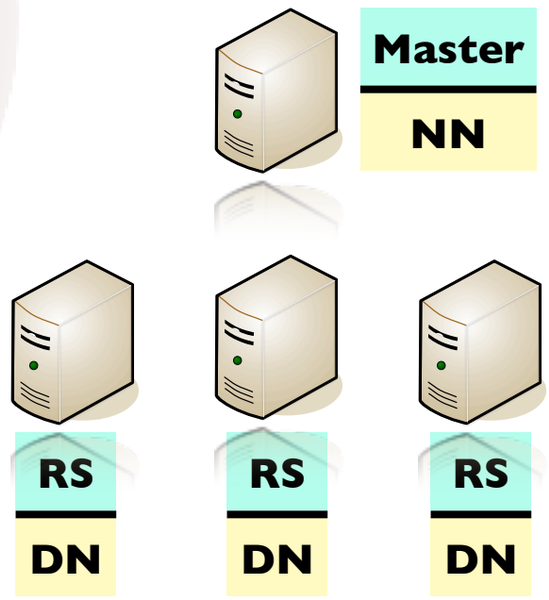
HBase

› **RegionServer**/**DataNode** have many configuration parameters – block size; handler count; behavior of the GC; write buffer size; etc.

› 2 parameters most significantly affect cluster's performance:

  › block cache size – favors read requests;

  › memstore size – favors write requests;



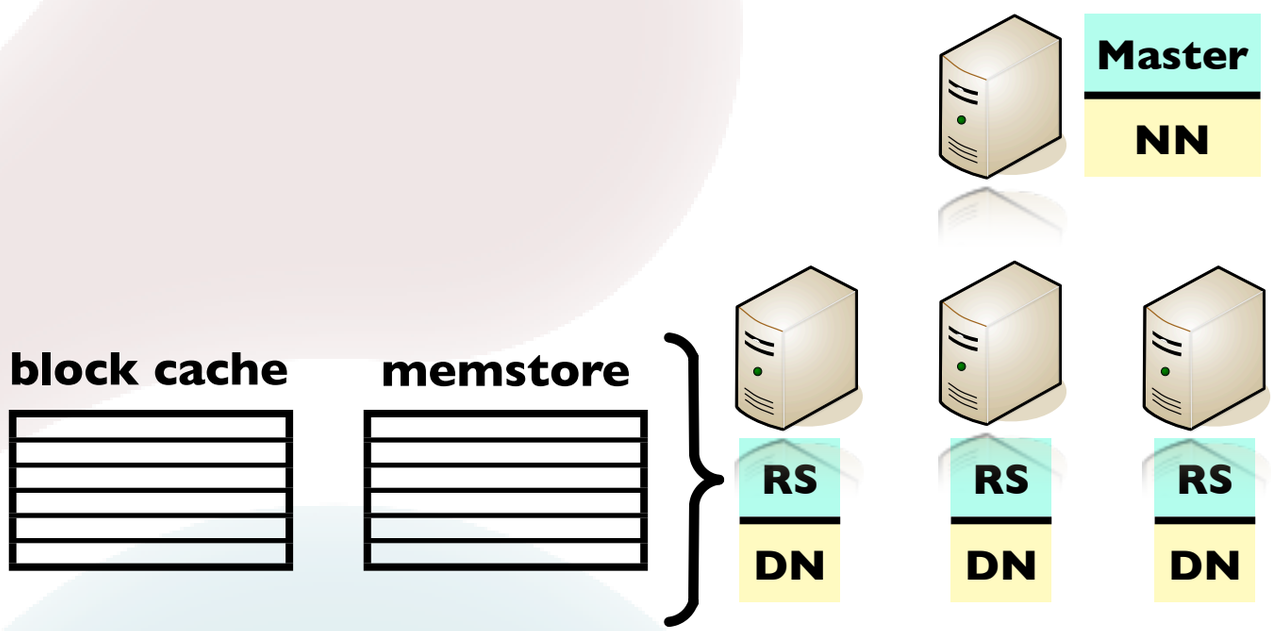
HBase

› **RegionServer**/**DataNode** have many configuration parameters – block size; handler count; behavior of the GC; write buffer size; etc.

› 2 parameters most significantly affect cluster's performance:

  › block cache size – favors read requests;

  › memstore size – favors write requests;



HBase

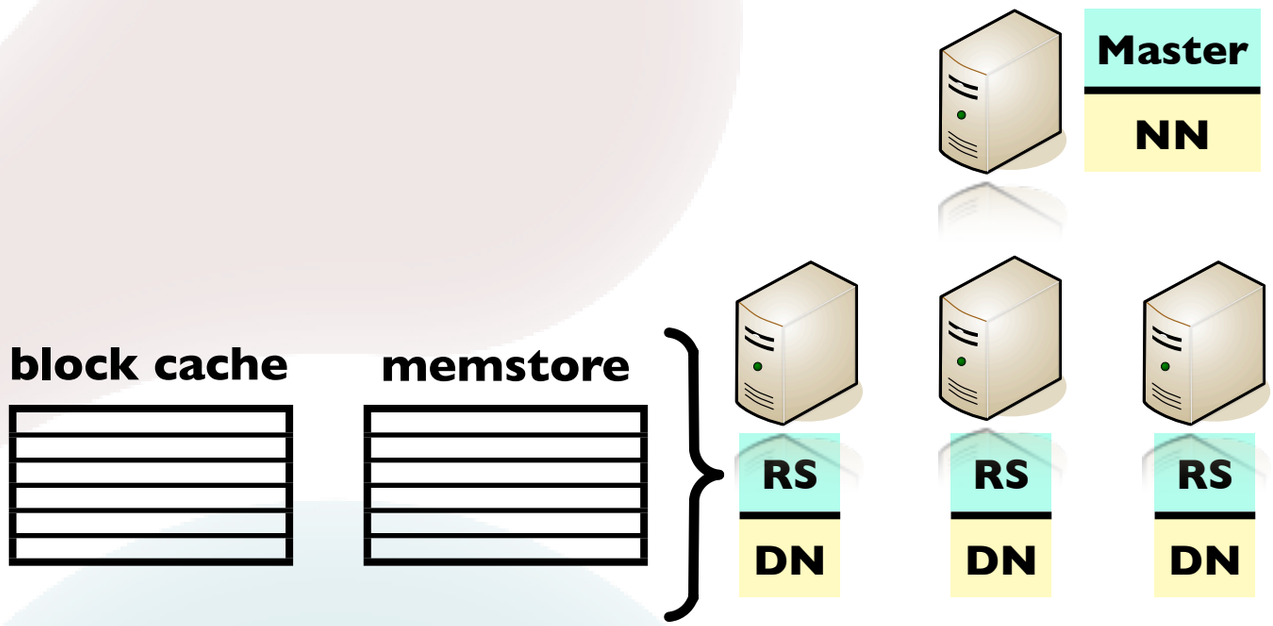
› **RegionServer**/**DataNode** have many configuration parameters – block size; handler count; behavior of the GC; write buffer size; etc.

› 2 parameters most significantly affect cluster's performance:

  › block cache size – favors read requests;

  › memstore size – favors write requests;

› Parameters reconfiguration requires shutting down the **RegionServer**.



HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Heterogeneity

Heterogeneity facts

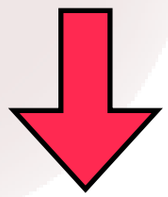
- Different applications -> different data access patterns.
- Access hotspots.

- In NoSQL databases data co-location is no longer a requirement due to:
  - no clear relationship between different entities;
  - data de-normalized;
  - computation done on the client side;
  - no support of atomic multi-item operations.

Heterogeneity facts

- Different applications -> different data access patterns.
- Access hotspots.

- In NoSQL databases data co-location is no longer a requirement due to:
  - no clear relationship between different entities;
  - data de-normalized;
  - computation done on the client side;
  - no support of atomic multi-item operations.



Allows for a different approach to data load balancing in NoSQL databases.

Heterogeneity hypothesis

› What if we cluster data partitions by their type of data access?

Heterogeneity hypothesis

› What if we cluster data partitions by their type of data access?

› What if we specifically configure each node taking into account the load they are expected to serve?

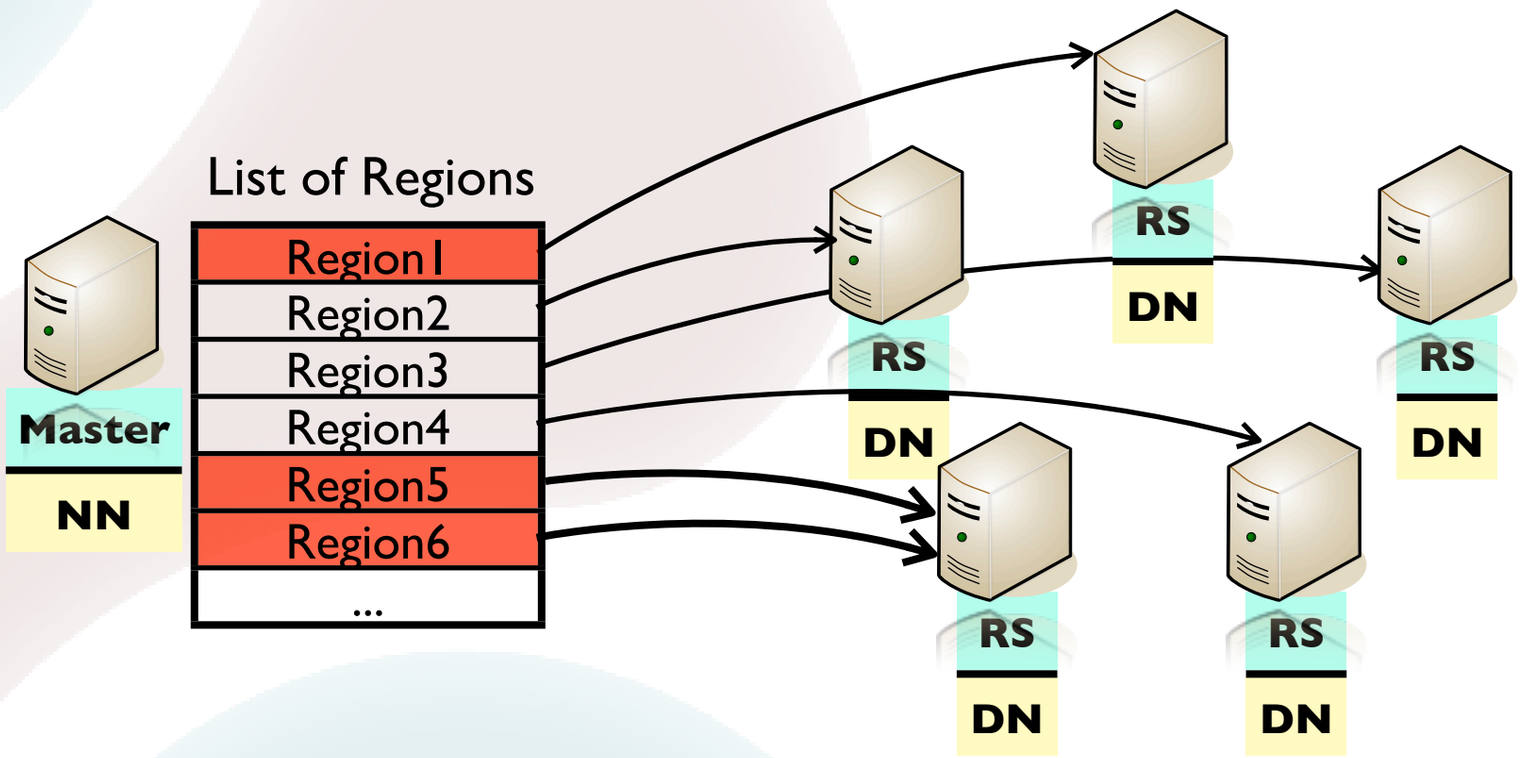
Heterogeneity - validation

- Placement and configuration strategies:
  - Random and Homogeneous;
  - Manual and Homogeneous;
  - Manual and Heterogeneous

Heterogeneity - validation

- Placement and configuration strategies:
  - Random and Homogeneous;
  - Manual and Homogeneous;
  - Manual and Heterogeneous

- Regular behaviour of HBase.
- Random data load balancer.
- Homogeneous node configuration.

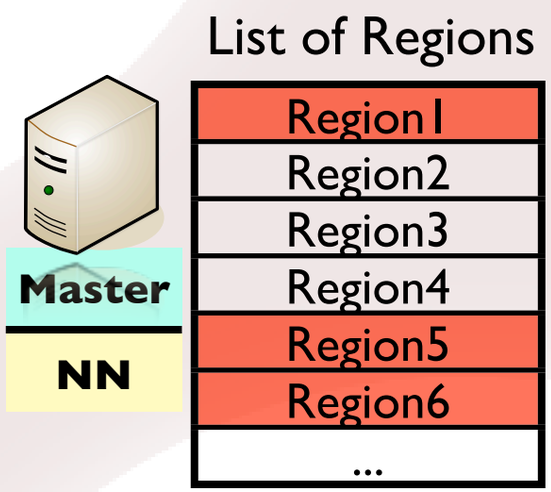


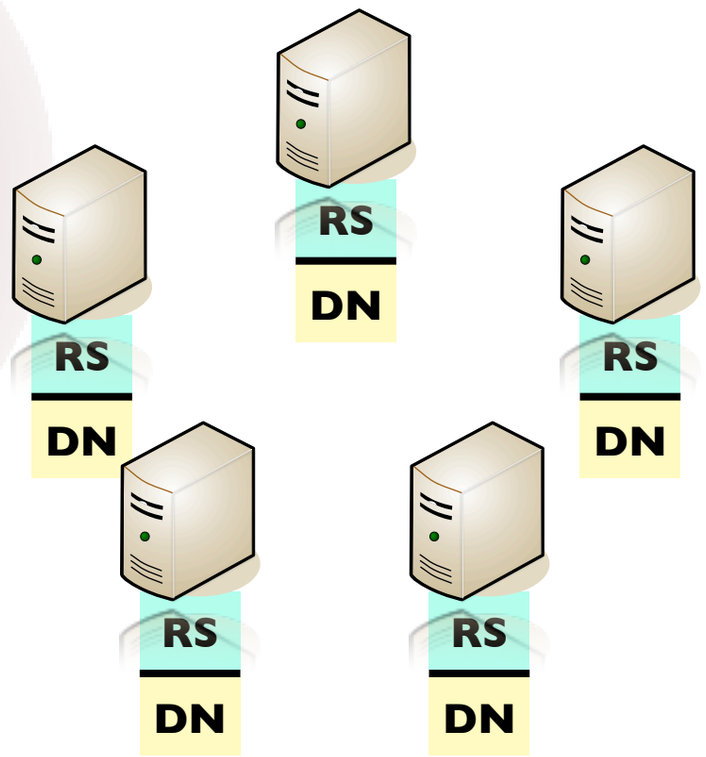
HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Heterogeneity - validation

- Placement and configuration strategies:
  - Random and Homogeneous;
  - Manual and Homogeneous;
  - Manual and Heterogeneous

- Manual data load balancer.
- Homogeneous node configuration.



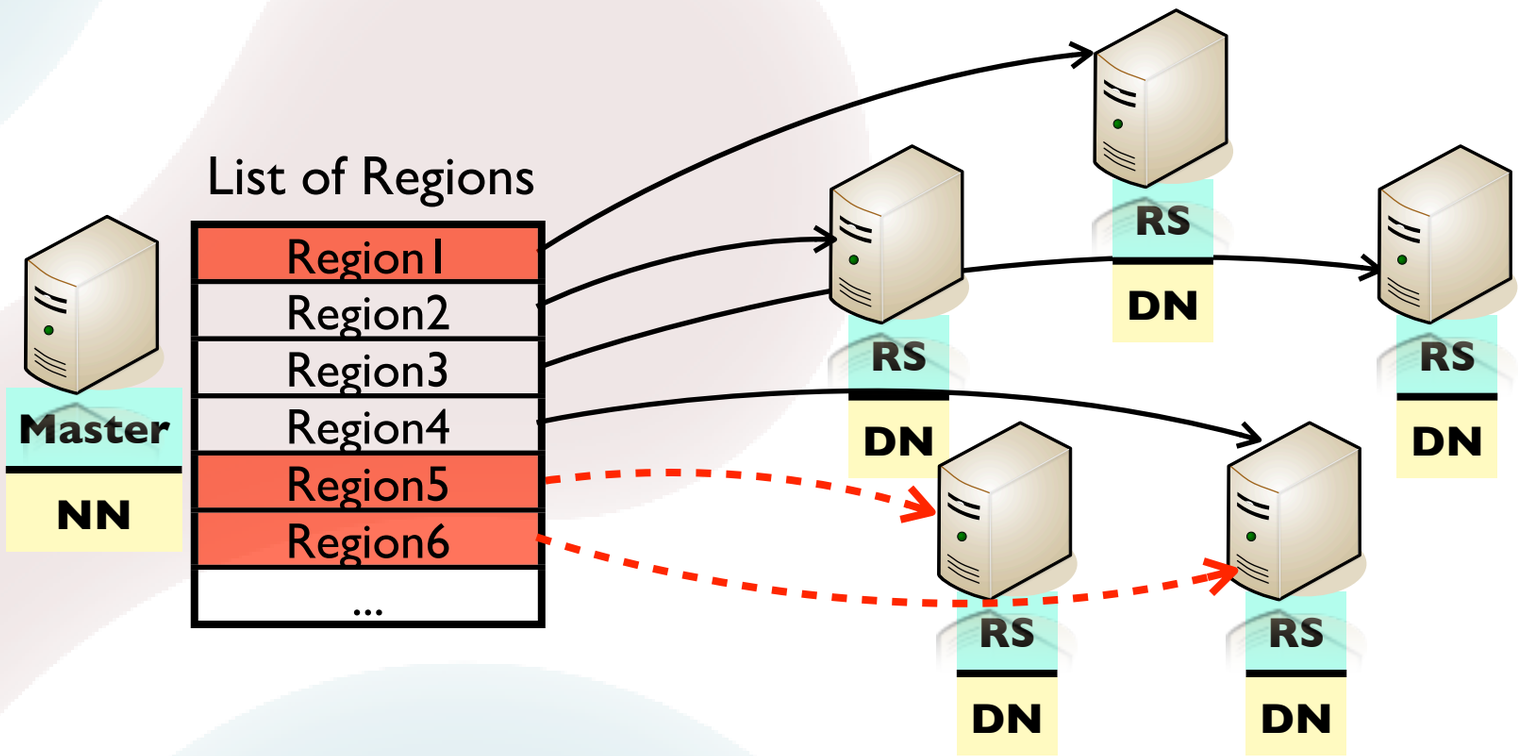


HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Heterogeneity - validation

- Placement and configuration strategies:
  - Random and Homogeneous;
  - Manual and Homogeneous;
  - Manual and Heterogeneous

- Manual data load balancer.
- Homogeneous node configuration.

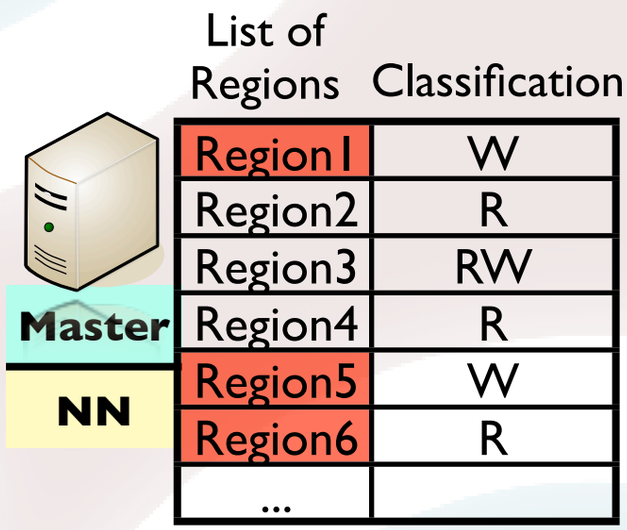


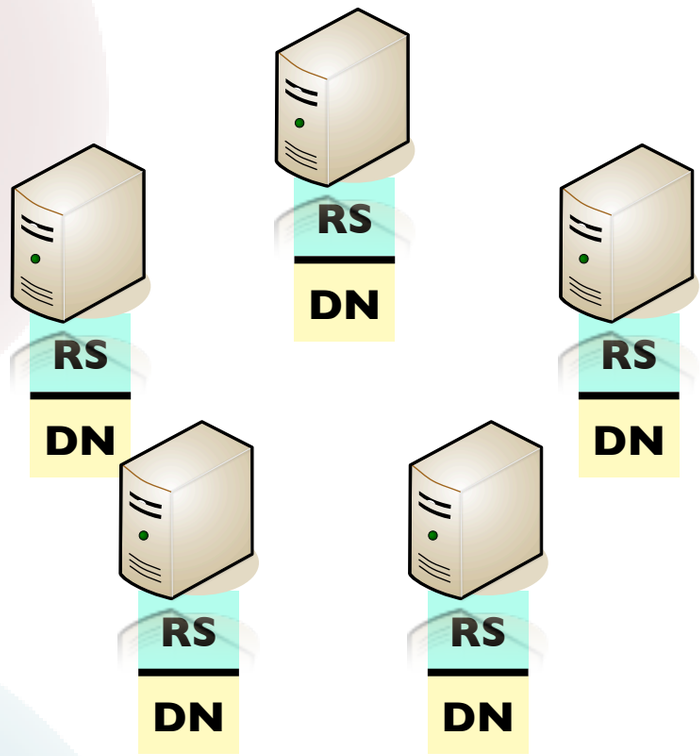
HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Heterogeneity - validation

- Placement and configuration strategies:
  - Random and Homogeneous;
  - Manual and Homogeneous;
  - Manual and Heterogeneous

- Classify partitions per access pattern (Read, Write, Read/Write and Scan).
- Manual data load balancer.
- Heterogeneous node configuration.



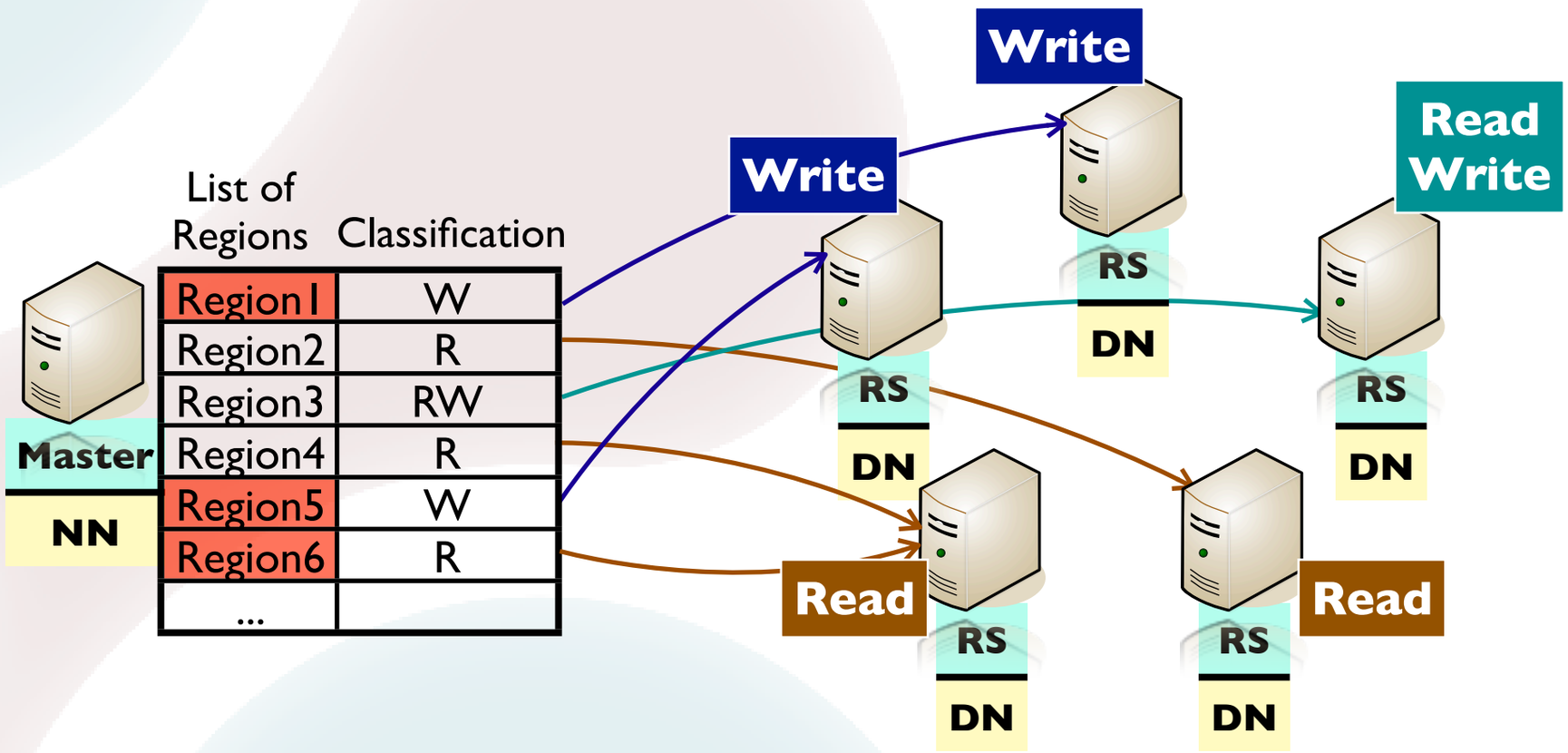


HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Heterogeneity - validation

- Placement and configuration strategies:
  - Random and Homogeneous;
  - Manual and Homogeneous;
  - Manual and Heterogeneous

- Classify partitions per access pattern (Read, Write, Read/Write and Scan).
- Manual data load balancer.
- Heterogeneous node configuration.



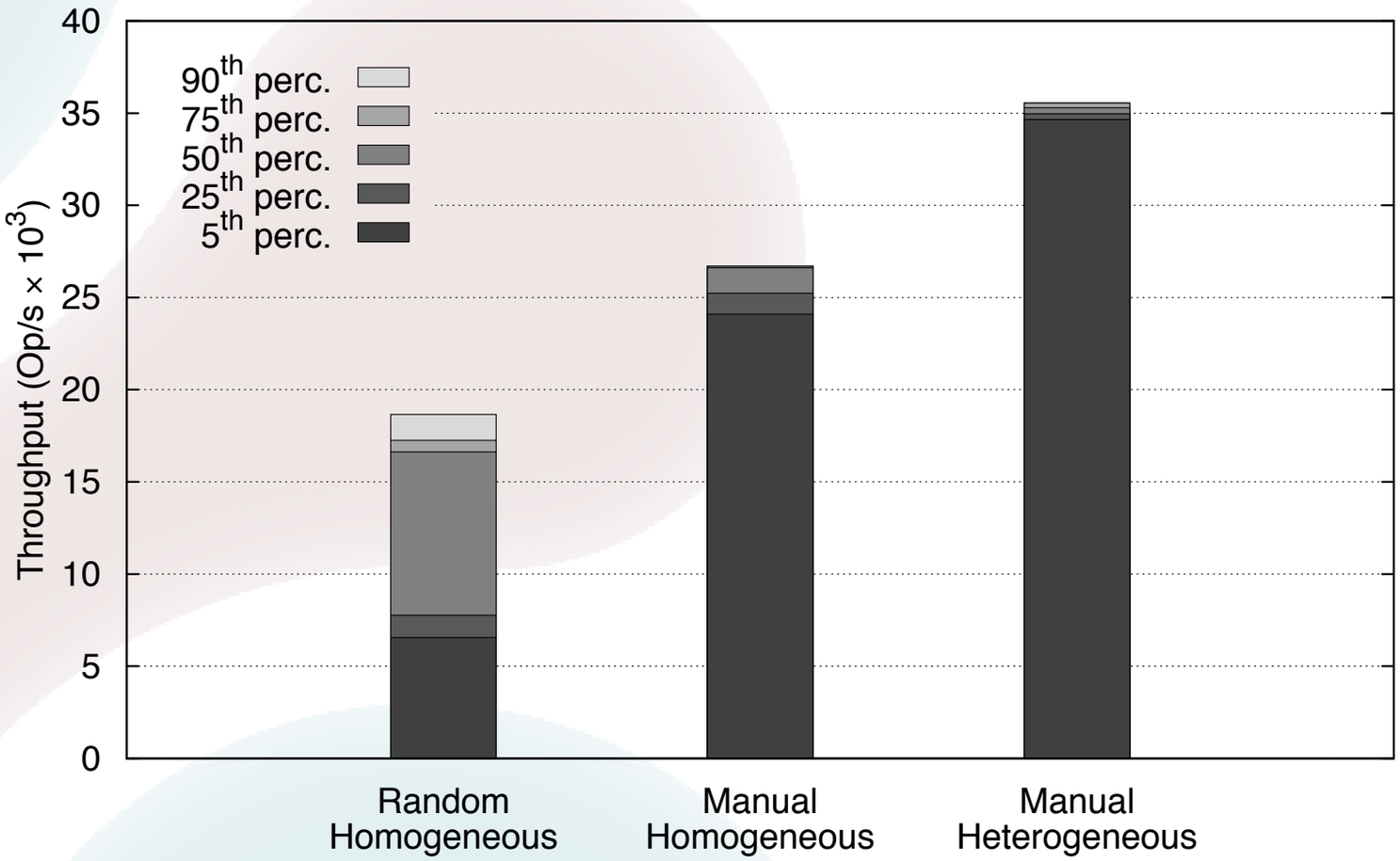
HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Heterogeneity - validation

› HBase in a multi-tenant environment using 6 YCSB different workloads generators and 5 **RegionServers/DataNodes**.
› Total read/write ratio - **65/35**.
› YCSB's hotspot distribution (50% of requests accessing 40% of the key space).

Heterogeneity - validation

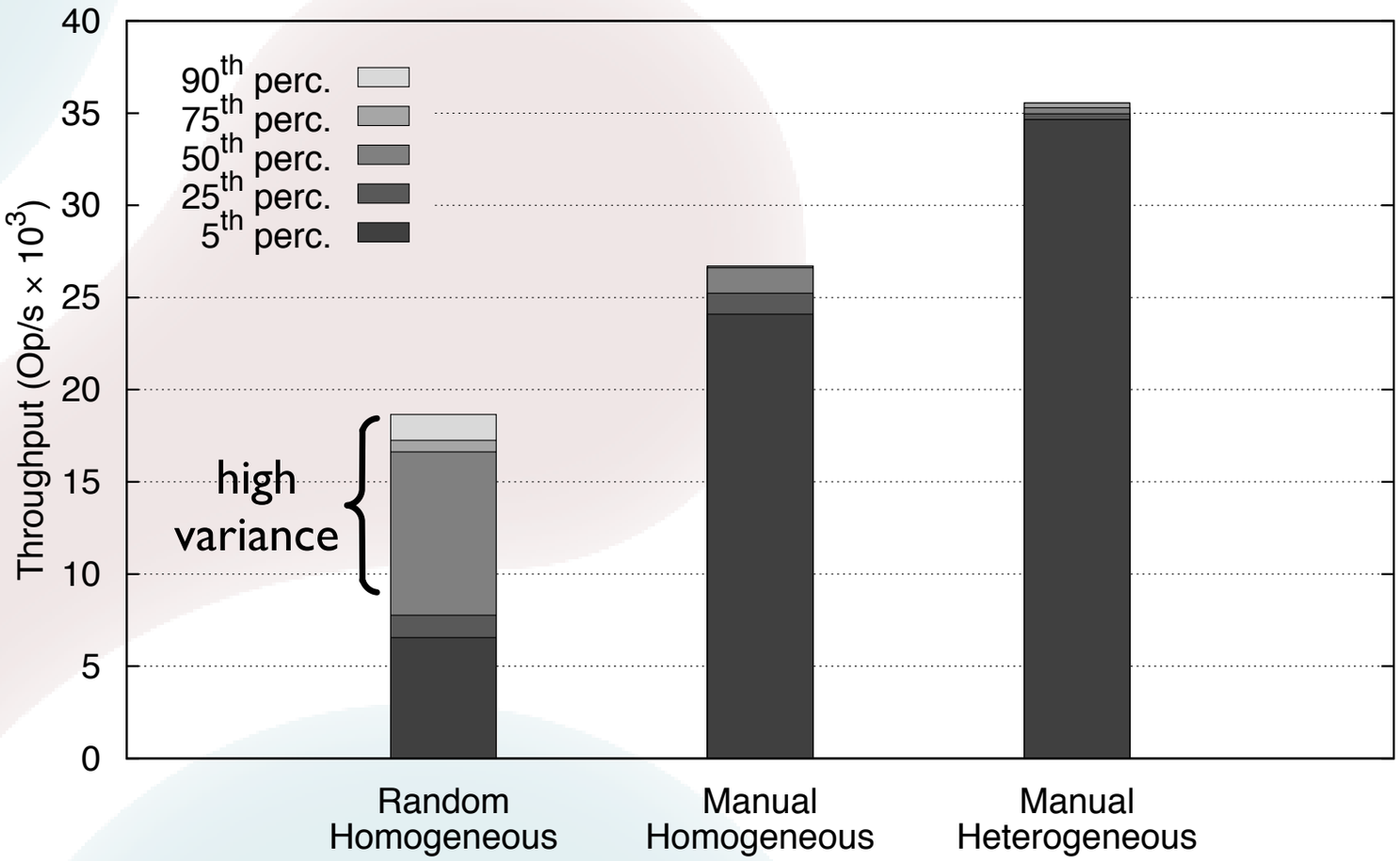
› HBase in a multi-tenant environment using 6 YCSB different workloads generators and 5 **RegionServers/DataNodes**.
› Total read/write ratio - **65/35**.
› YCSB's hotspot distribution (50% of requests accessing 40% of the key space).



HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Heterogeneity - validation

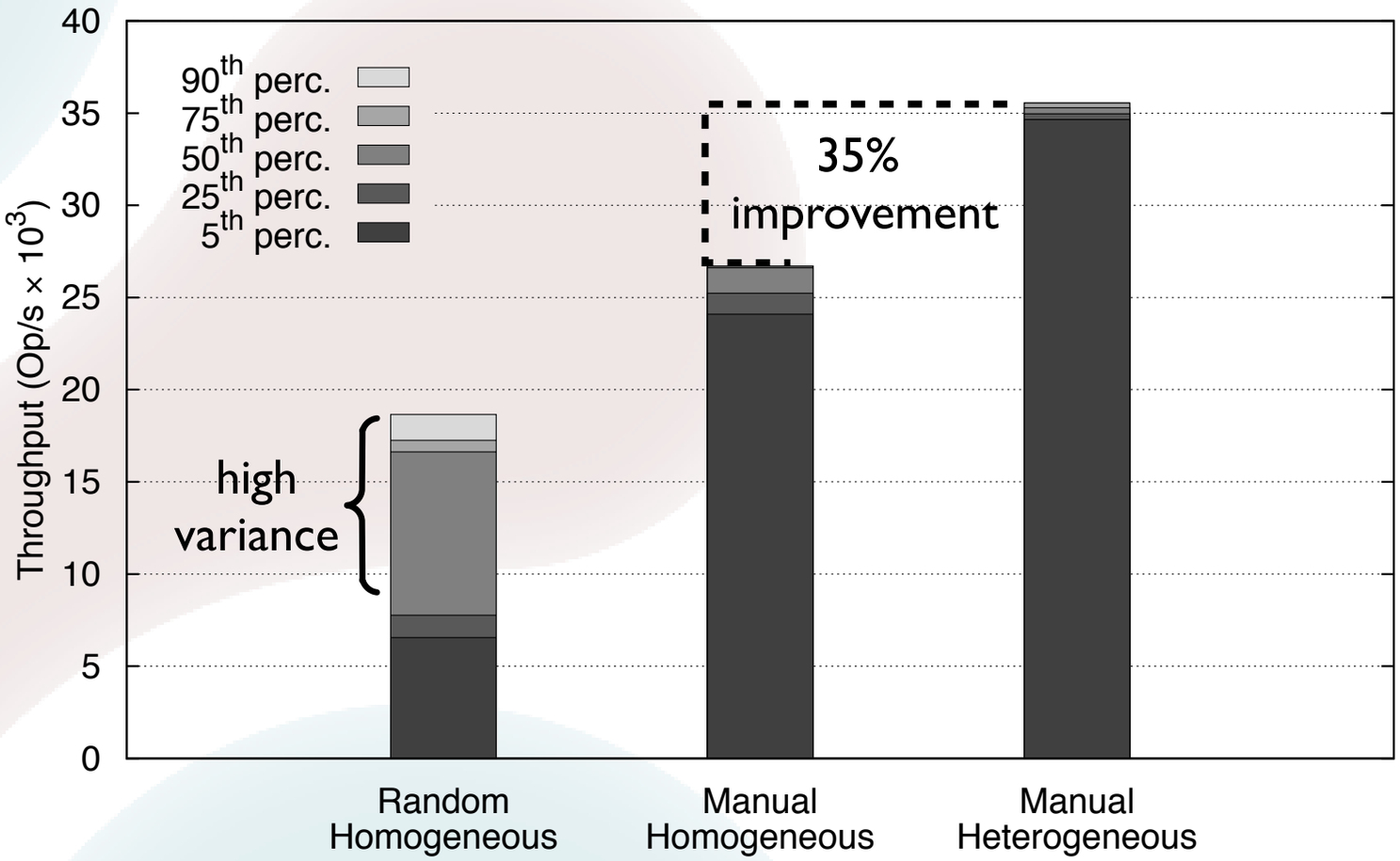
› HBase in a multi-tenant environment using 6 YCSB different workloads generators and 5 **RegionServers/DataNodes**.
› Total read/write ratio - **65/35**.
› YCSB's hotspot distribution (50% of requests accessing 40% of the key space).



HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Heterogeneity - validation

› HBase in a multi-tenant environment using 6 YCSB different workloads generators and 5 **RegionServers/DataNodes**.
› Total read/write ratio - **65/35**.
› YCSB's hotspot distribution (50% of requests accessing 40% of the key space).



HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

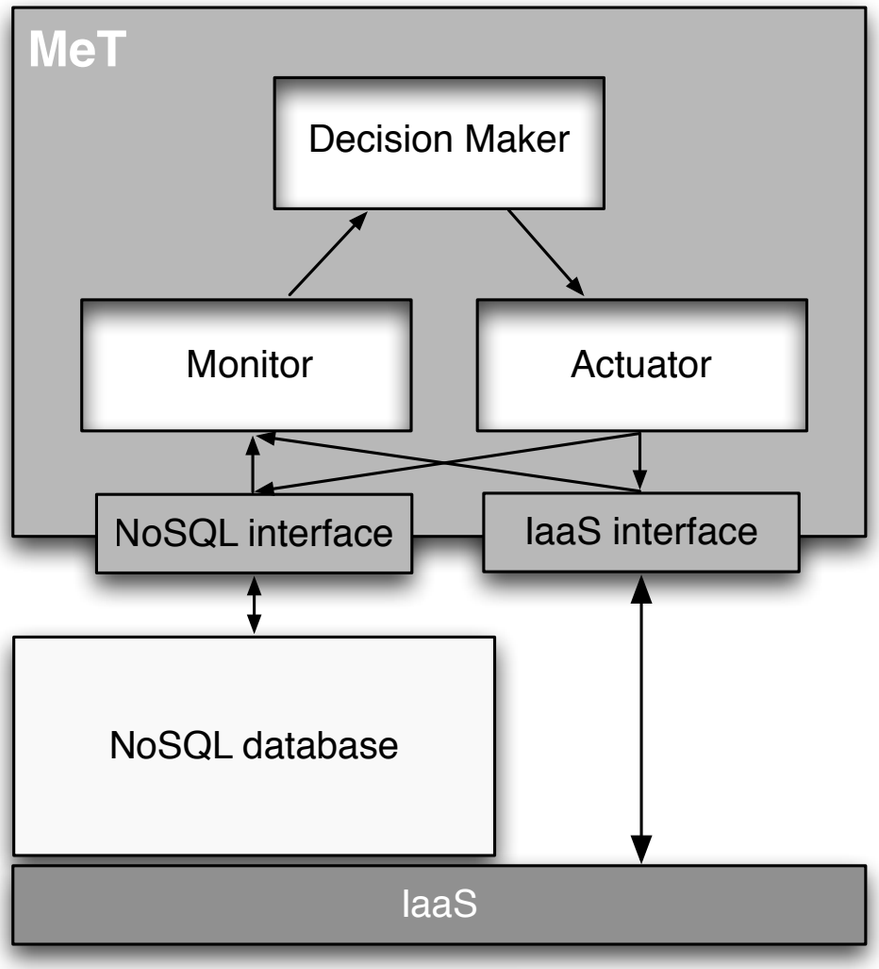
MeT

MeT

- Automatic management of NoSQL database clusters;
- Reconfigurations according to data access patterns;
- Use of heterogeneous configurations;
- Autonomous elasticity.

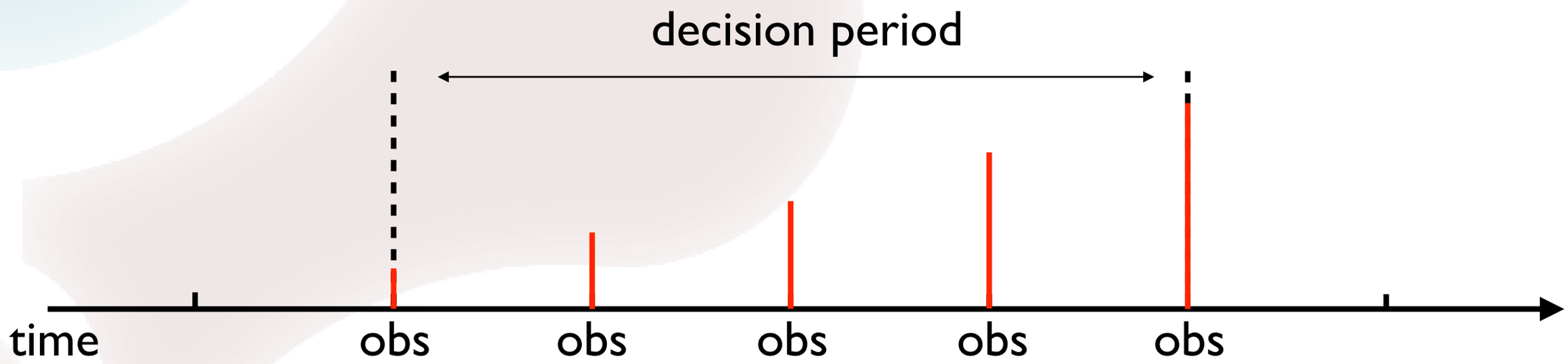
MeT

- Automatic management of NoSQL database clusters;
- Reconfigurations according to data access patterns;
- Use of heterogeneous configurations;
- Autonomous elasticity.



MeT - monitor

› Frequently gathers information about the current state of the cluster at two different levels:
  › Resource usage metrics;
  › NoSQL specific metrics.
› Exponential smoothing of observations.



MET - decision maker

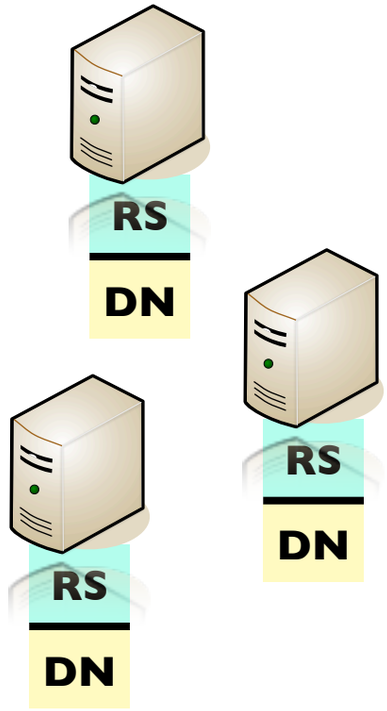
1. Decision Algorithm:
   - Based on resource usage metrics decides whether to add or remove nodes.

MeT - decision maker

1. Decision Algorithm:
   - Based on resource usage metrics decides whether to add or remove nodes.
2. Distribution Algorithm:

List of Regions

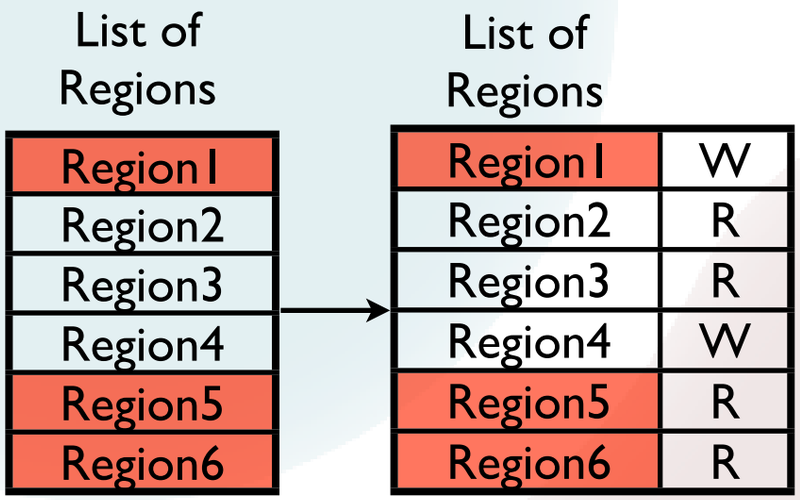
Region1
Region2
Region3
Region4
Region5
Region6



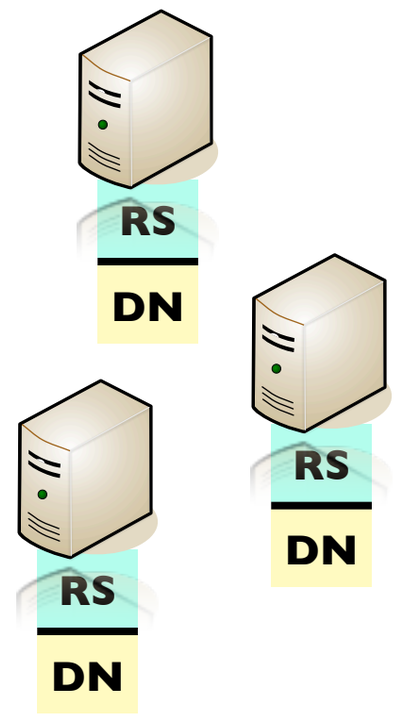
HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

MeT - decision maker

1. Decision Algorithm:
   - Based on resource usage metrics decides whether to add or remove nodes.
2. Distribution Algorithm:



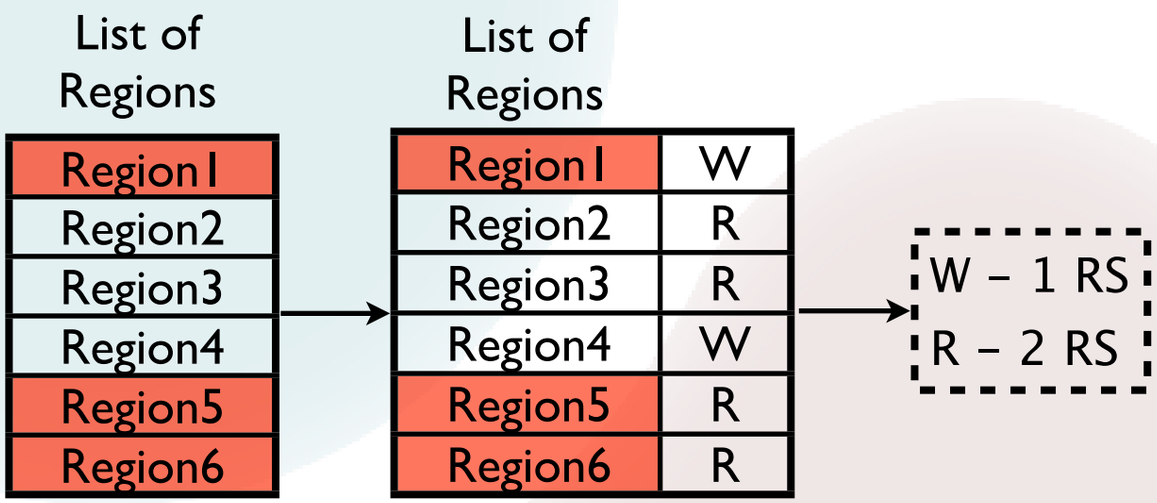
(i) classification

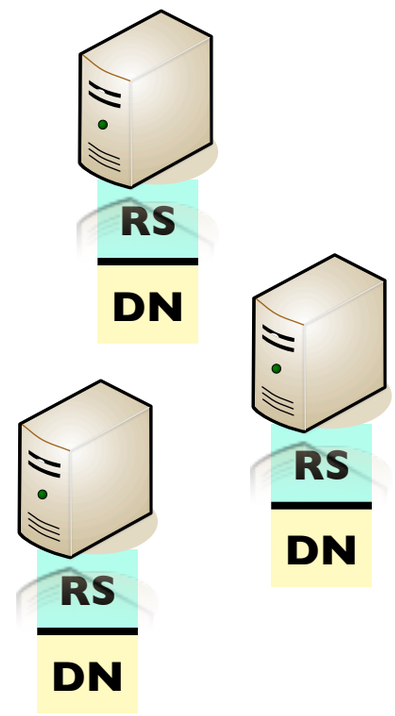


HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

MeT - decision maker

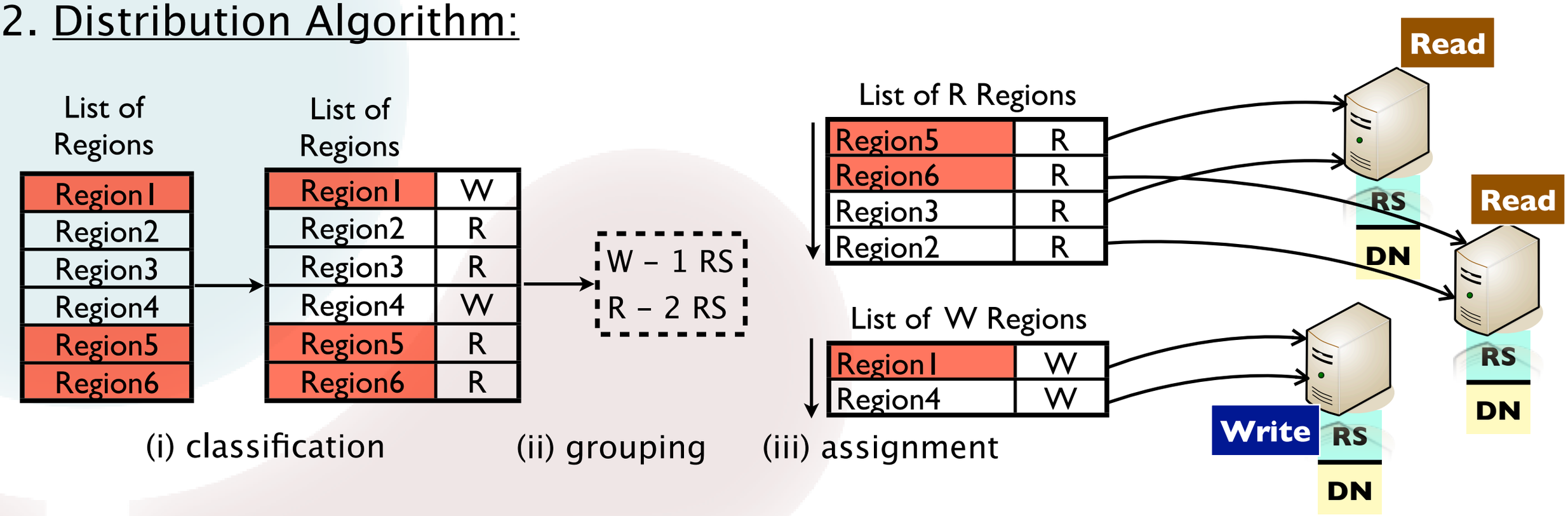
1. Decision Algorithm:
   - Based on resource usage metrics decides whether to add or remove nodes.
2. Distribution Algorithm:





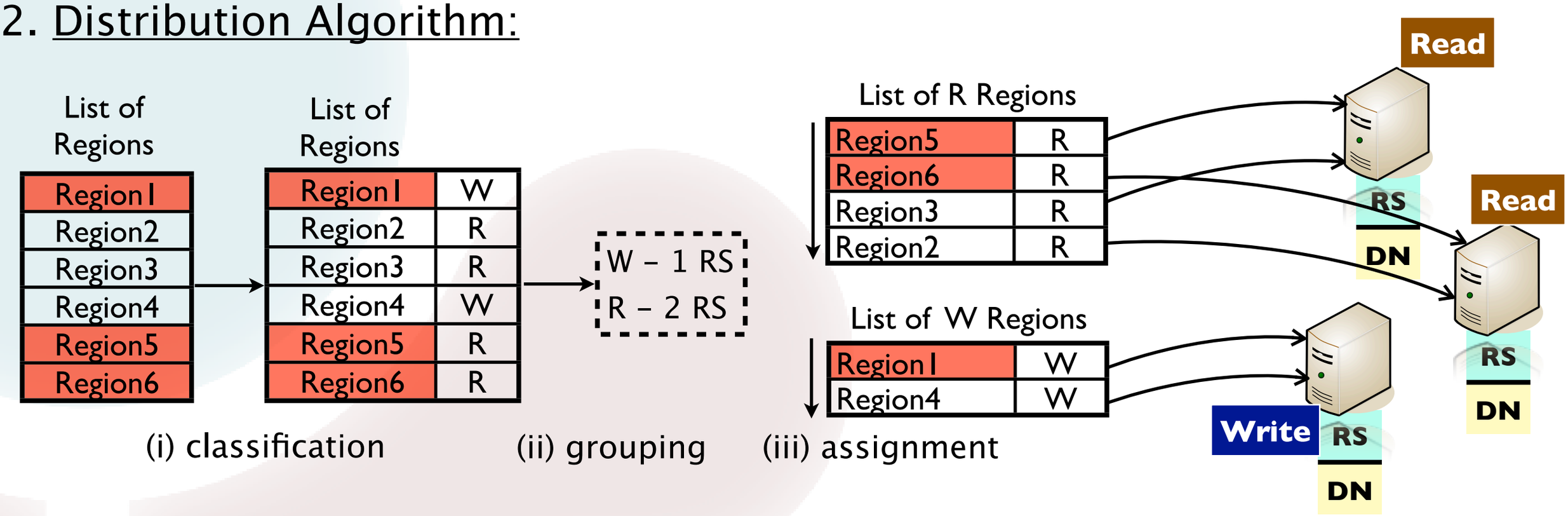
MET - decision maker

1. Decision Algorithm:
   - Based on resource usage metrics decides whether to add or remove nodes.
2. Distribution Algorithm:



MET - decision maker

1. Decision Algorithm:
   - Based on resource usage metrics decides whether to add or remove nodes.
2. Distribution Algorithm:

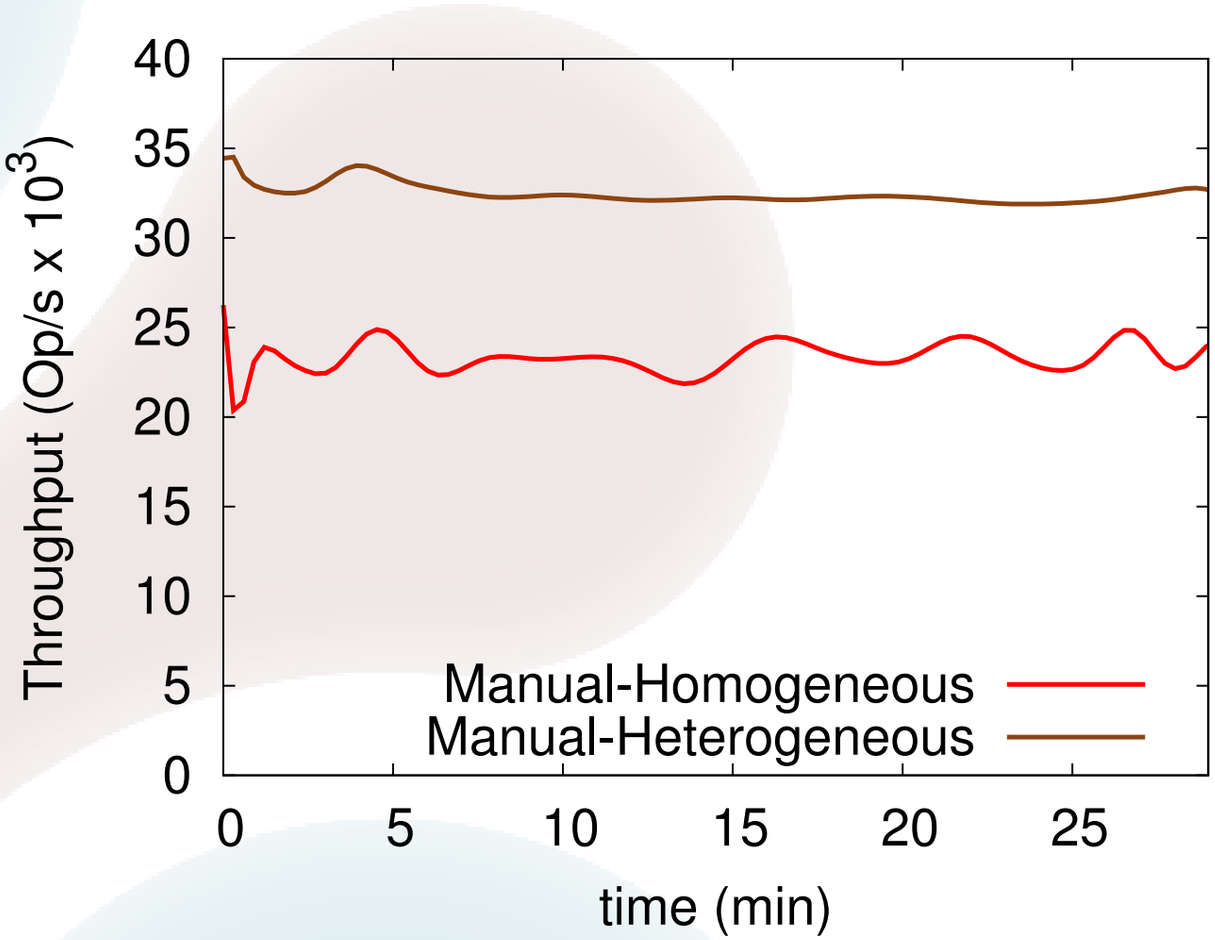


3. Output Computation:
   - Determine the best way to reach the targeted distribution.

Evaluation

Evaluation - convergence

› Same 6 YCSB workloads as previous tests. Same setting with 5 **RegionServers**/**DataNodes**.

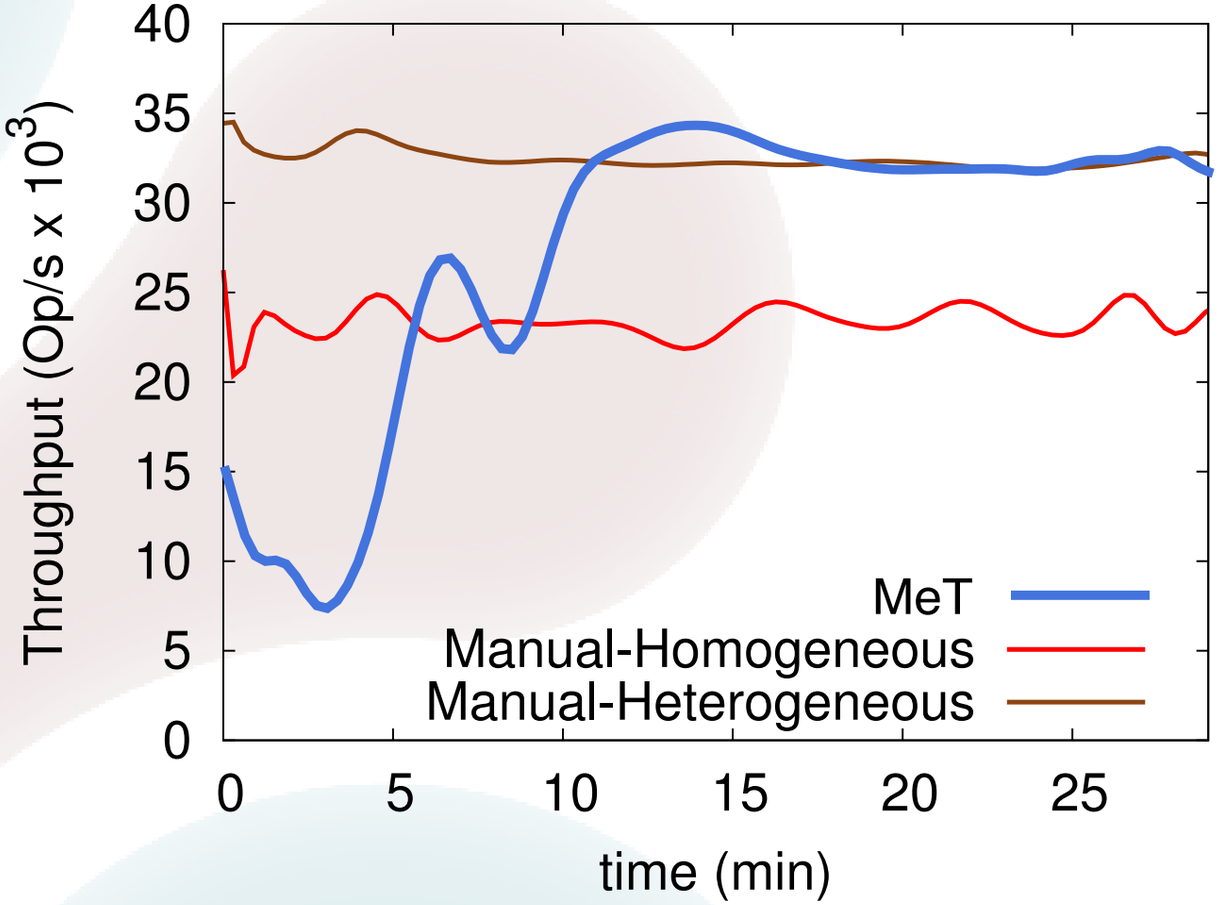


HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Evaluation - convergence

› Same 6 YCSB workloads as previous tests. Same setting with 5 **RegionServers**/**DataNodes**.

› Starting with the Random and Homogeneous strategy (out-of-the box HBase) we then used MeT to reconfigure the cluster.

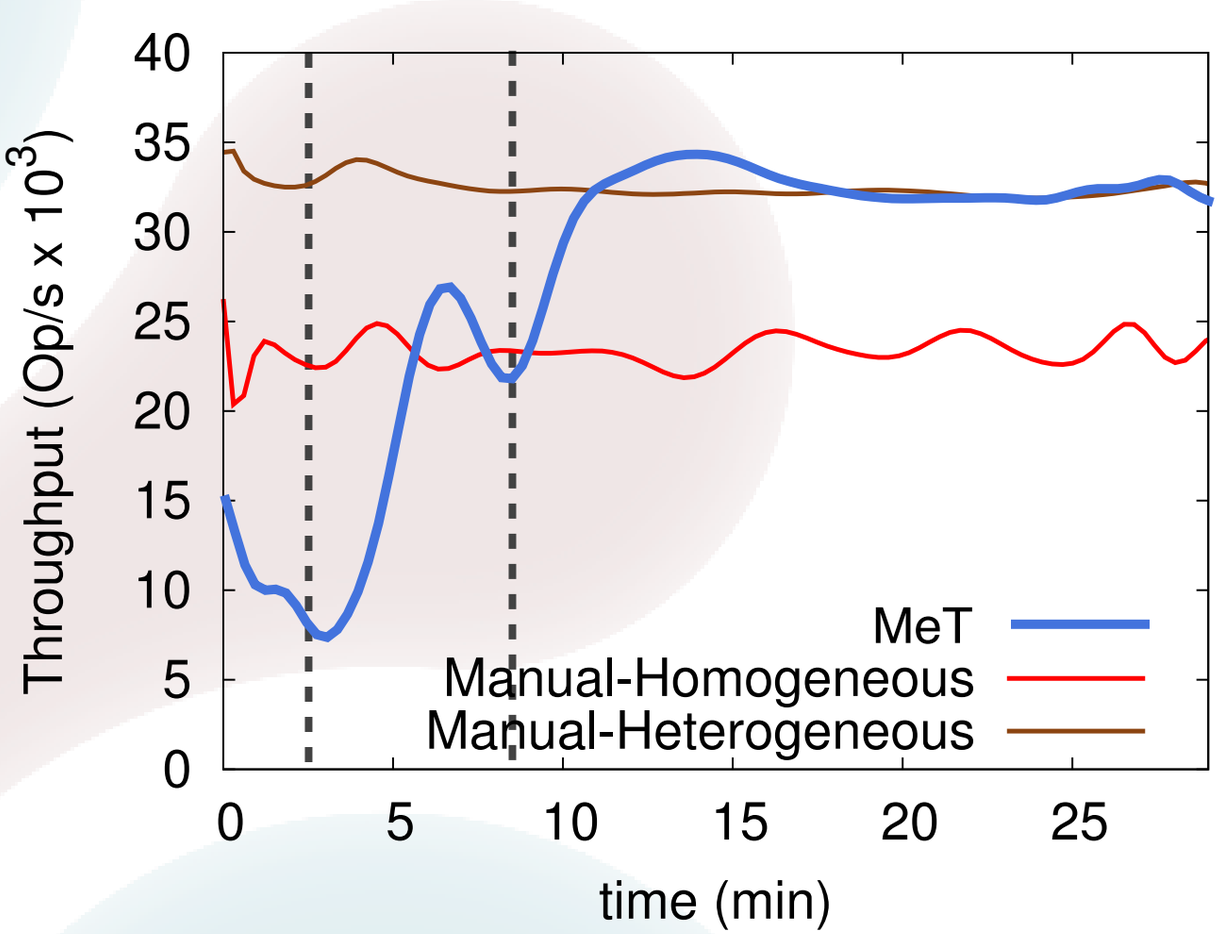


HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Evaluation - convergence

› Same 6 YCSB workloads as previous tests. Same setting with 5 **RegionServers**/**DataNodes**.

› Starting with the Random and Homogeneous strategy (out-of-the box HBase) we then used MeT to reconfigure the cluster.



HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Evaluation - versatility

- An optimized implementation of TPC-C for HBase.
  - 30 warehouses; 300 clients; 45 minutes long runs.
- Setting with 6 **RegionServers**/**DataNodes**.

Setting	Throughput (tpmC)
Manual and Homogeneous (base line)	25380
MeT	33720

Evaluation - versatility

- An optimized implementation of TPC-C for HBase.
  - 30 warehouses; 300 clients; 45 minutes long runs.
- Setting with 6 **RegionServers**/**DataNodes**.

Setting	Throughput (tpmC)
Manual and Homogeneous (base line)	25380
MeT	33720

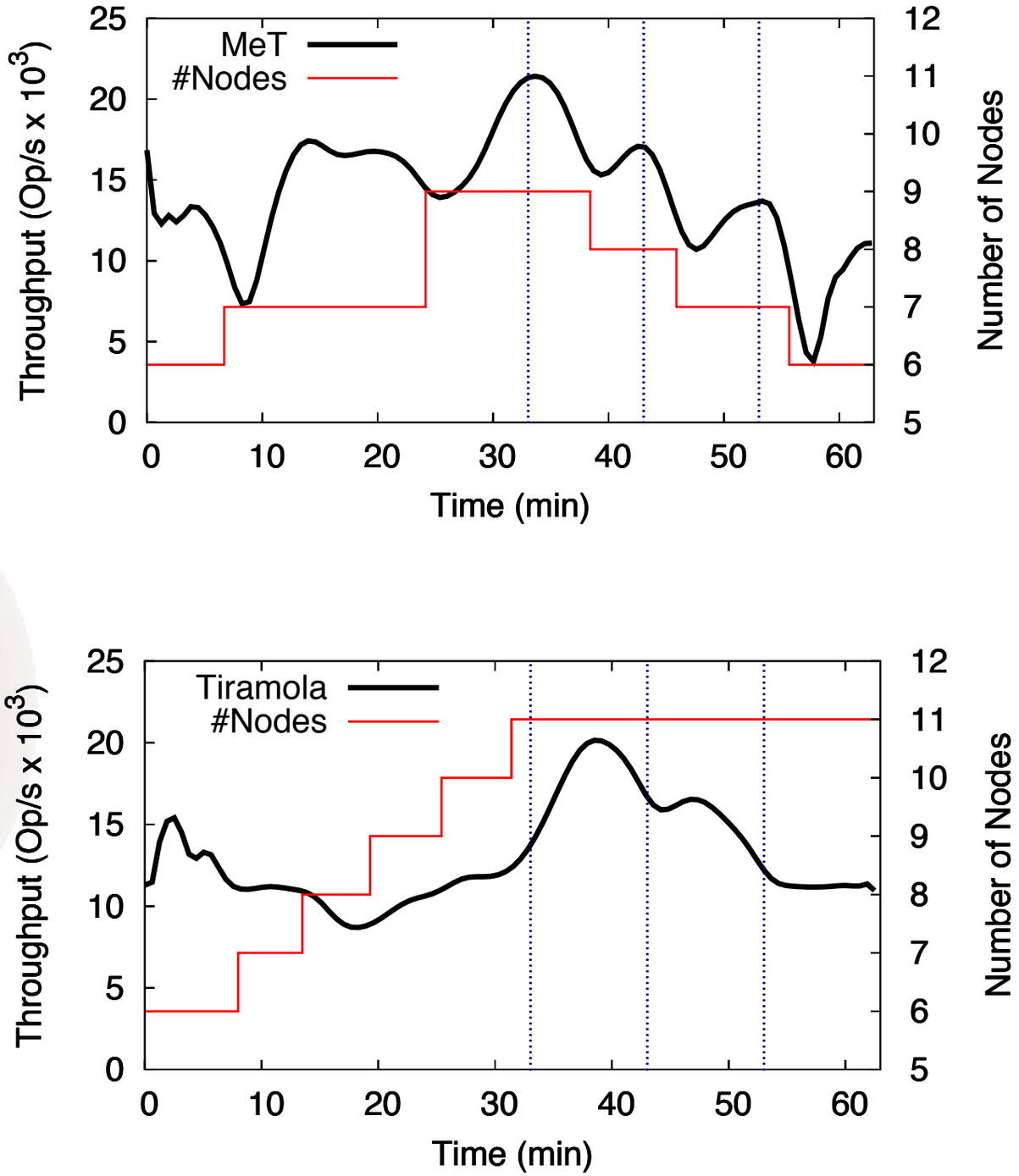
- 33% improvement.

Evaluation - elasticity

› HBase cluster on top of an OpenStack deployment (6RegionServers/DataNodes).

Evaluation - elasticity

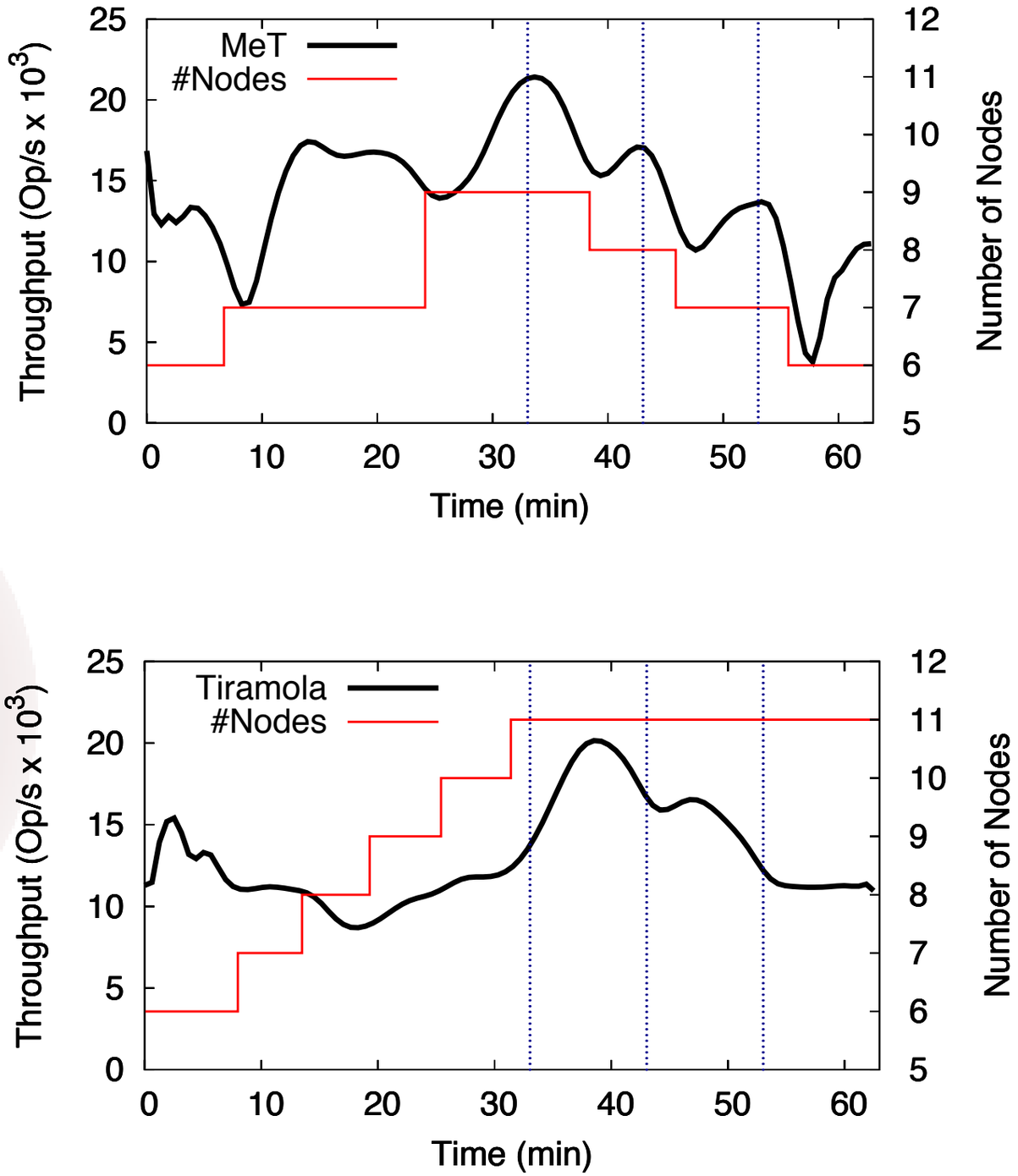
› HBase cluster on top of an OpenStack deployment (6RegionServers/DataNodes).
› MeT and Tiramola.



HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Evaluation - elasticity

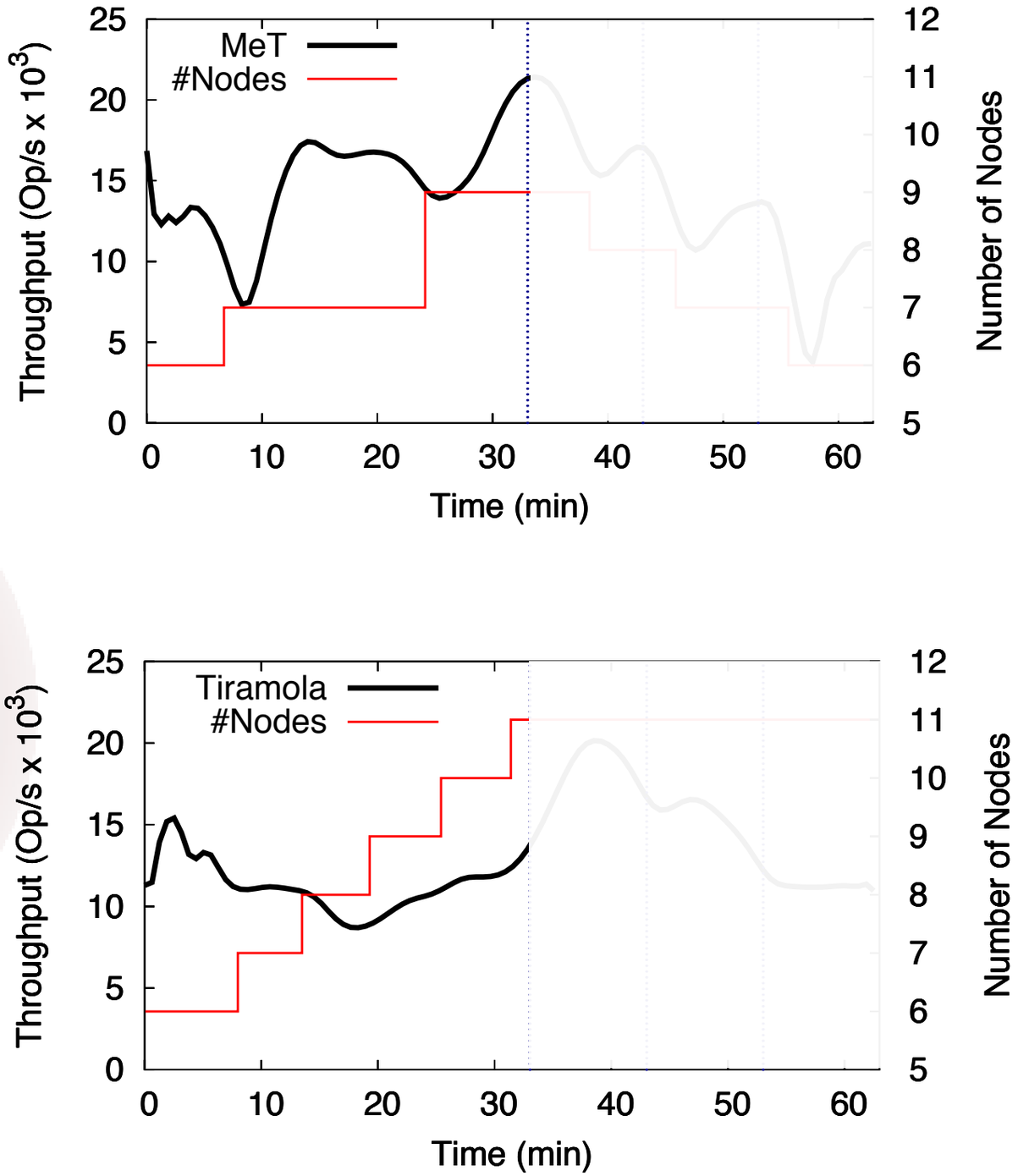
› HBase cluster on top of an OpenStack deployment (6RegionServers/DataNodes).
› MeT and Tiramola.
› Two phases:



HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Evaluation - elasticity

› HBase cluster on top of an OpenStack deployment (6RegionServers/DataNodes).
› MeT and Tiramola.
› Two phases:
    1.Starting with a set of YCSB workloads that overloaded the initial HBase cluster;



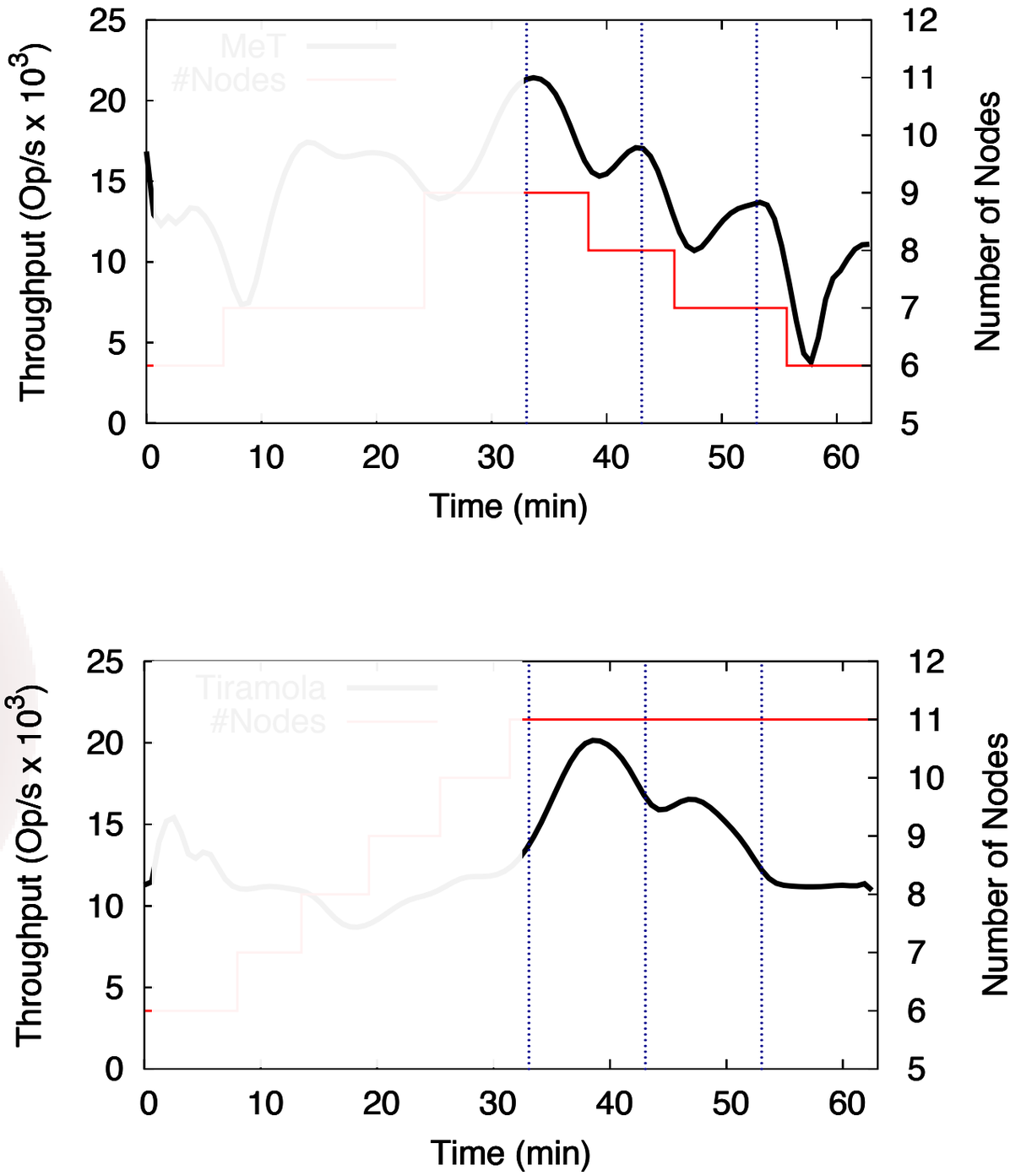
HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Evaluation - elasticity

› HBase cluster on top of an OpenStack deployment (6RegionServers/DataNodes).
› MeT and Tiramola.
› Two phases:

1. Starting with a set of YCSB workloads that overloaded the initial HBase cluster;

2. By the 33th minute we progressively switch-off some of the YCSB workloads until there is only one active.



HASLab
HIGH-ASSURANCE SOFTWARE LABORATORY

Conclusion

- **Heterogeneous** configuring NoSQL databases:
  - **throughput** can be improved by **35%**;
  - both in multi-tenant and single tenant scenarios.
- Data partitions allocated to:
  - **specifically configured nodes** considering their **access patterns;**
  - dynamic load balancer, based on NoSQL metrics.
- **MeT framework**:
  - combines all these contributions;
  - **automatically** provides **elasticity** to HBase;
  - open-source project at: **https://github.com/fmaia/met.**