Big Data Systems

Djellel Difallah

Spring 2023

Lecture 9 (cont.) – Apache Hadoop V2 and V3

Changes in Job Scheduling, HDFS, and other features

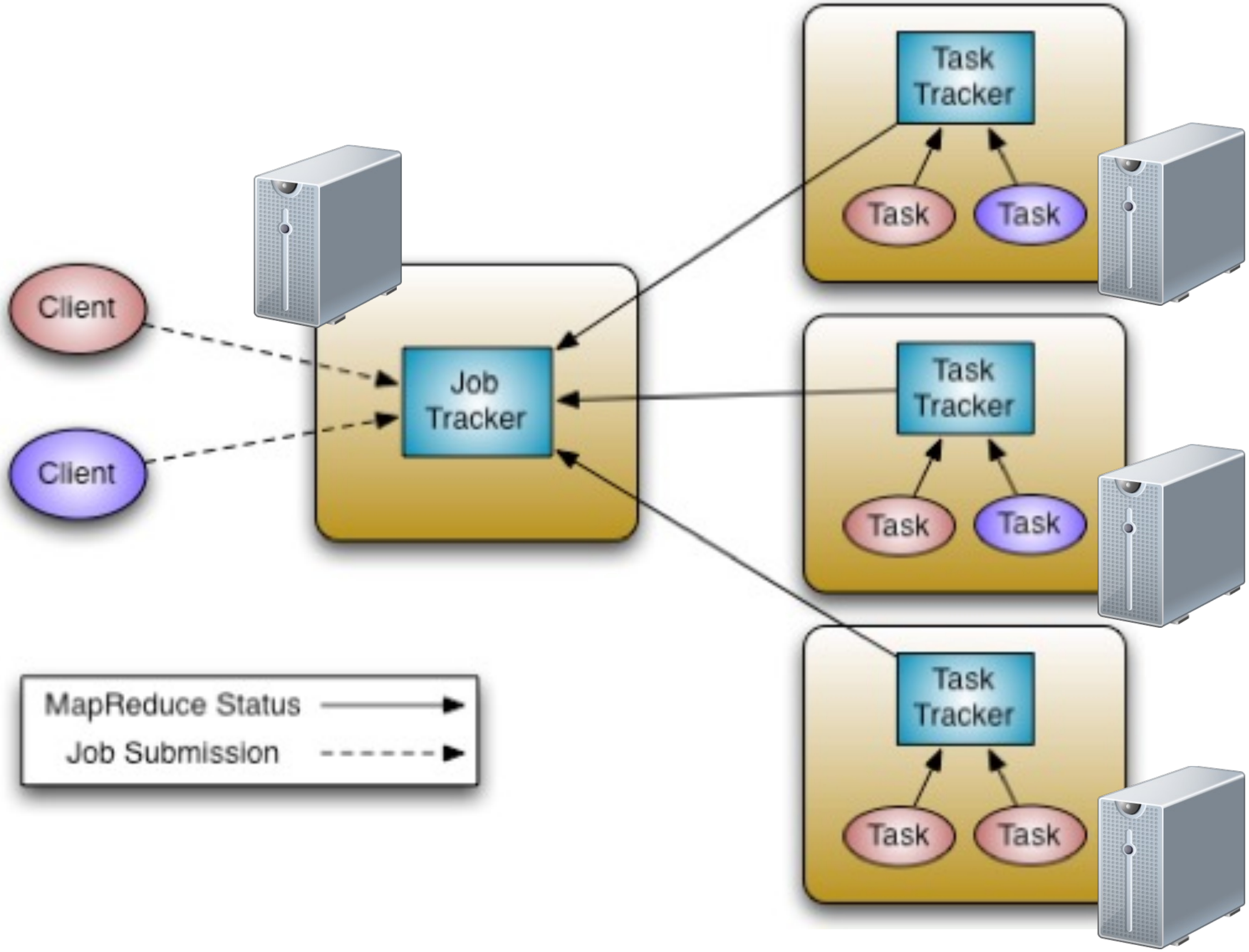
Outline

- Hadoop V2
  - Change 1: YARN (Yet Another Resource Negotiator)
  - Change 2: High Availability HDFS
  - Scheduling
    - FIFO
    - Capacity Scheduling
    - Fair Scheduling
    - Delay Scheduling
- Hadoop V3
  - YARN
    - Timeline service
  - HDFS
    - Erasure Coding

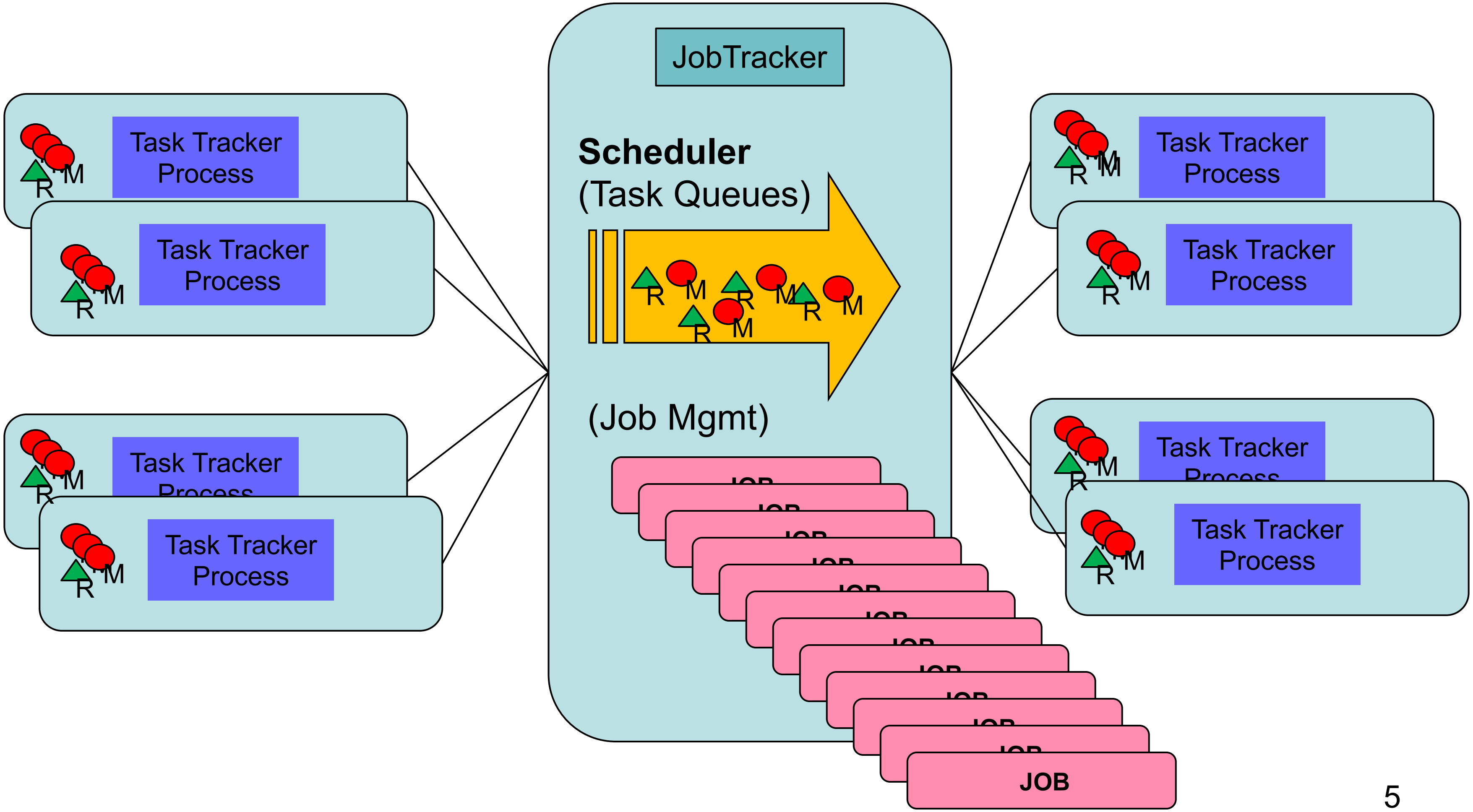
General Computation Cluster

HADOOP V2

Map Reduce Framework



Map Reduce Framework



Hadoop v1.0
Problems with the JobTracker (JT)

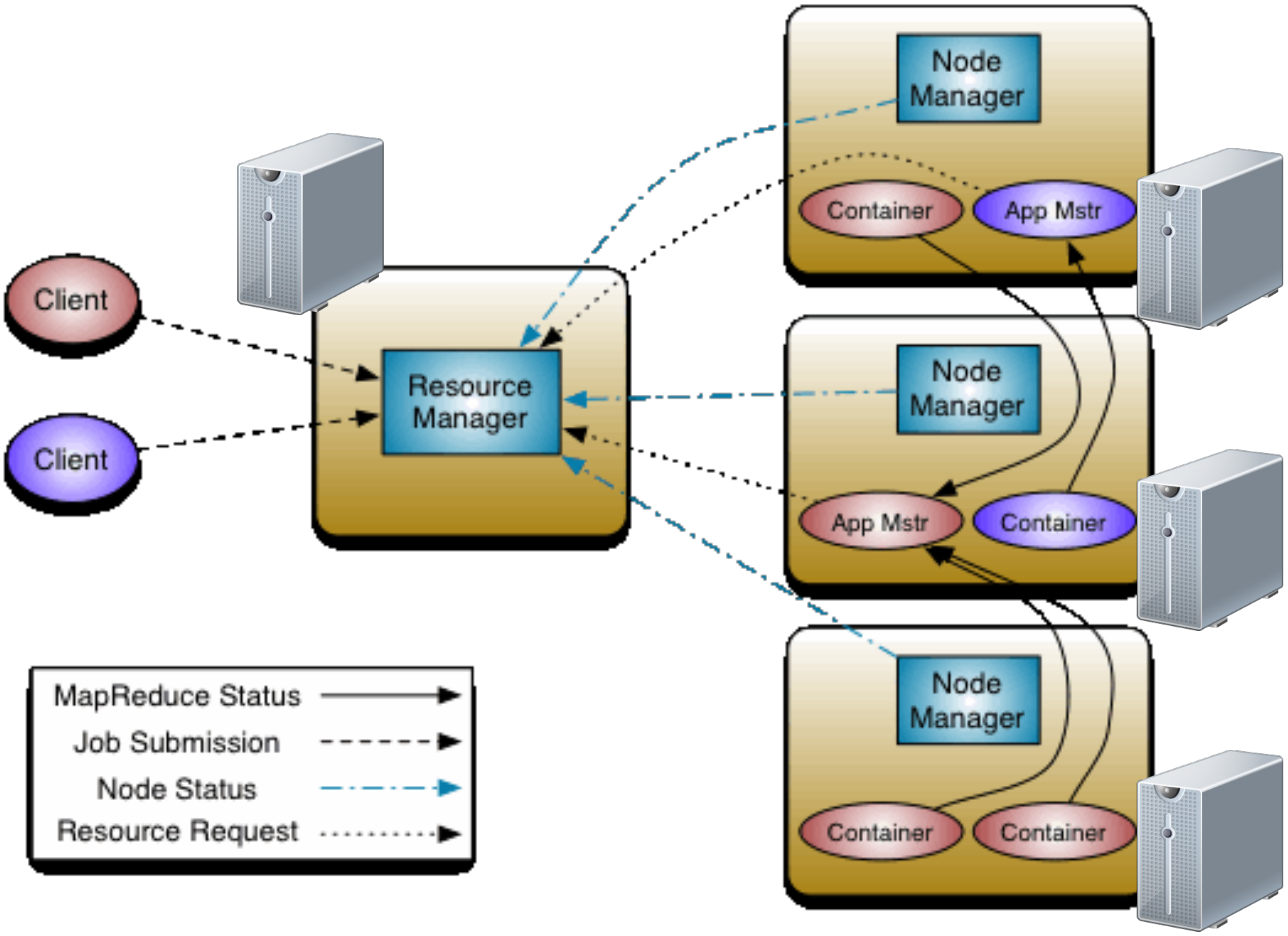
- Scalability
  - Limited horizontal scaling
  - **~ 4000/5000 nodes and 40000 (M/R) tasks running currently**
- Fault Tolerance
  - If the JT dies, *all* jobs must restart
- Maintenance
  - Stop jobs to upgrade JT
- Rigid programing model
  - Hadoop v1.0 support only MapReduce

YARN
Generalized Cluster Management

- The main change in Hadoop V2 :
  - 💡 **separate the resources management and job tracking**
- YARN (Yet Another Resources Negotiator)
  - Responsible of “resources” in the cluster.
  - Resource = any combination of CPU/Memory/Other x Time required by a program
- Job tracking for individual jobs is detached to another node in the network.
  - Any node can become a job tracker for a given Job
- Tasks are now “containers”

Vavilapalli, Vinod Kumar, et al. **"Apache Hadoop YARN: Yet another resource negotiator."** *Proceedings of the 4th annual Symposium on Cloud Computing*. ACM, 2013.

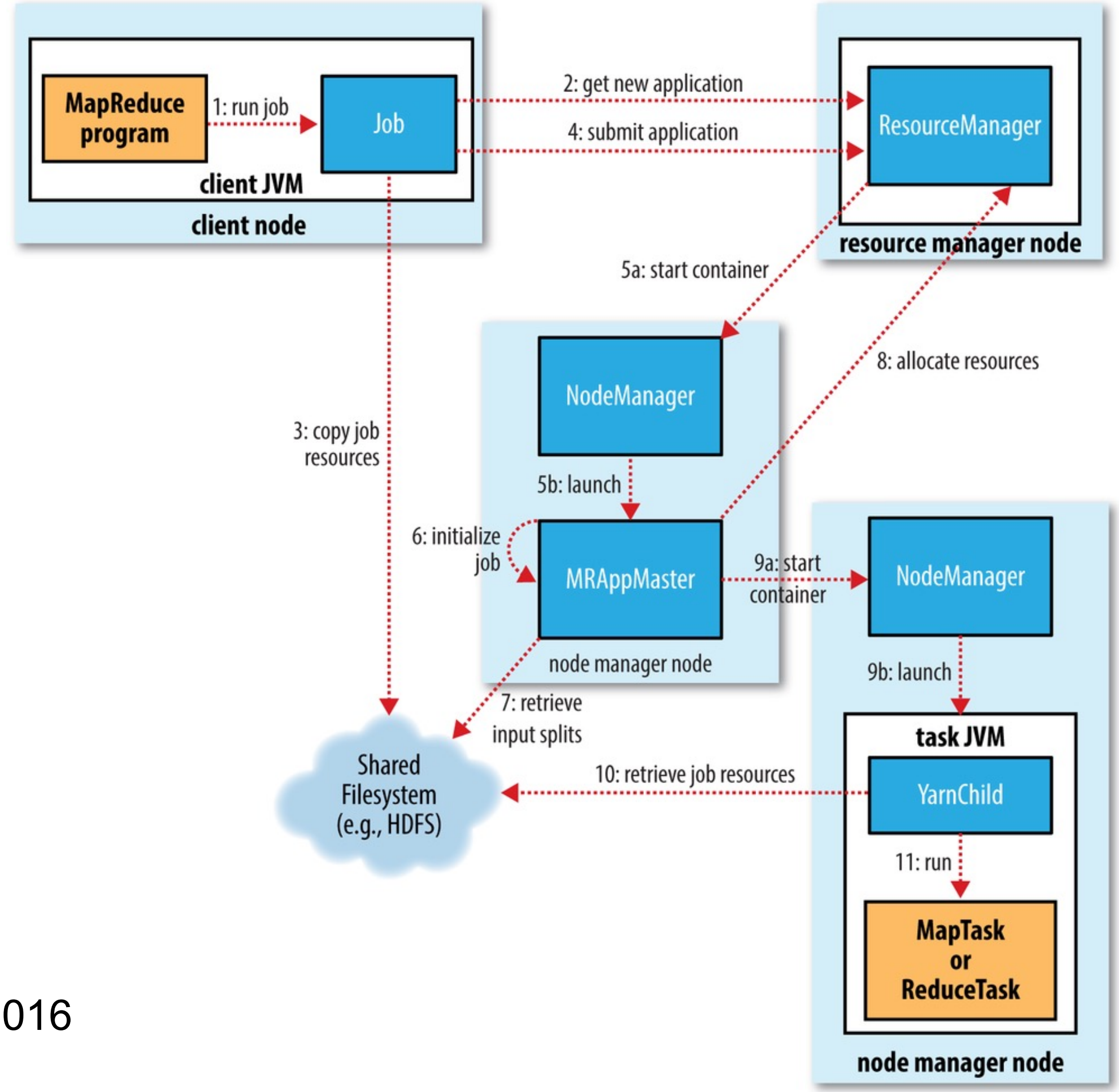
Hadoop YARN – Yet Another Resource Negotiator



YARN Components

- **Resource Manager**:
  - 1 Node per cluster
  - You can configure a RM failover node
  - Manages job scheduling and execution
  - Global resource allocation
- **Application Master**:
  - Replaces the Job Tracker
  - 1 per job, running on 1 node
  - Manages task scheduling and execution
- **Node Manager**
  - Similar to the TaskTracker
  - 1 process per node
  - Manages the lifecycle of task containers
  - Reports to RM on health and resource usage

Execution Flow
MapReduce on YARN



From Tom White 2016

Change 2: “HDFS 2”

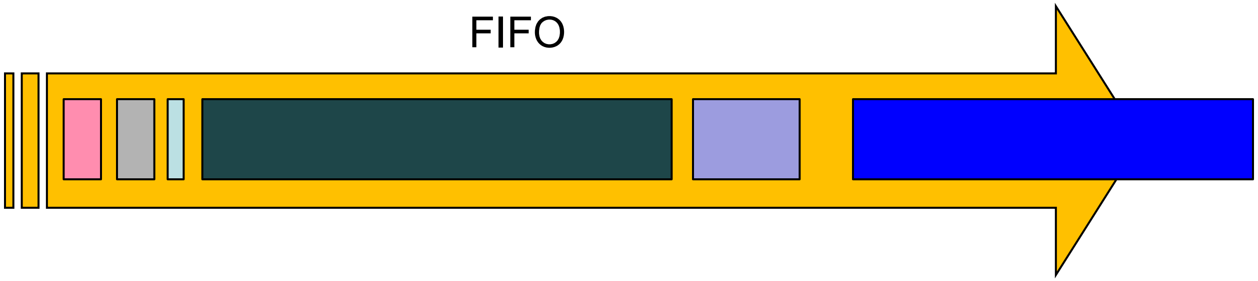
- The main change in Hadoop V2 :
  - 💡Improve the Name Node
- **High availability**
  - Add an additional *passive* Name Node
- **Federation**
  - Multiple name nodes responsible of sub namespaces

Resource Management Design Principles

- Provide fast response times to small jobs in a multitenant Hadoop cluster (many users!)
- Improve utilization and data locality
  - Proper load balance
  - Hotspot avoidance
  - Run task closer to data (same node ideally, or same rack)
- *This is done using Job Scheduling*
- Hadoop has a taste for simplicity.
  - YARN is a centralized resources negotiator with very basic scheduling and decision techniques

Hadoop YARN – FIFO

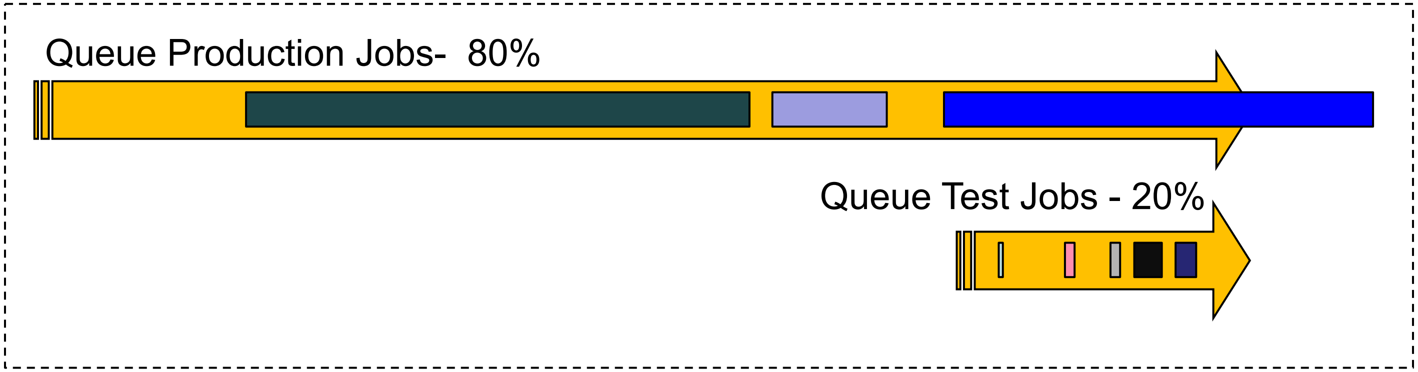
- Jobs that are admitted first take up all the resources until they finish
- Poor parallelism
- Small jobs starvation



Scheduler
FIFO

Hadoop YARN – Capacity Scheduler

- Organizes jobs into (hierarchical) queues
- Queue shares as %'s of cluster
  - Many configuration parameters are available e.g., can go overcapacity if not utilized
- FIFO scheduling within each queue
- Supports preemption
  - The system can stop a job or a task to maintain the promised capacity



Hadoop YARN – Fair Scheduler

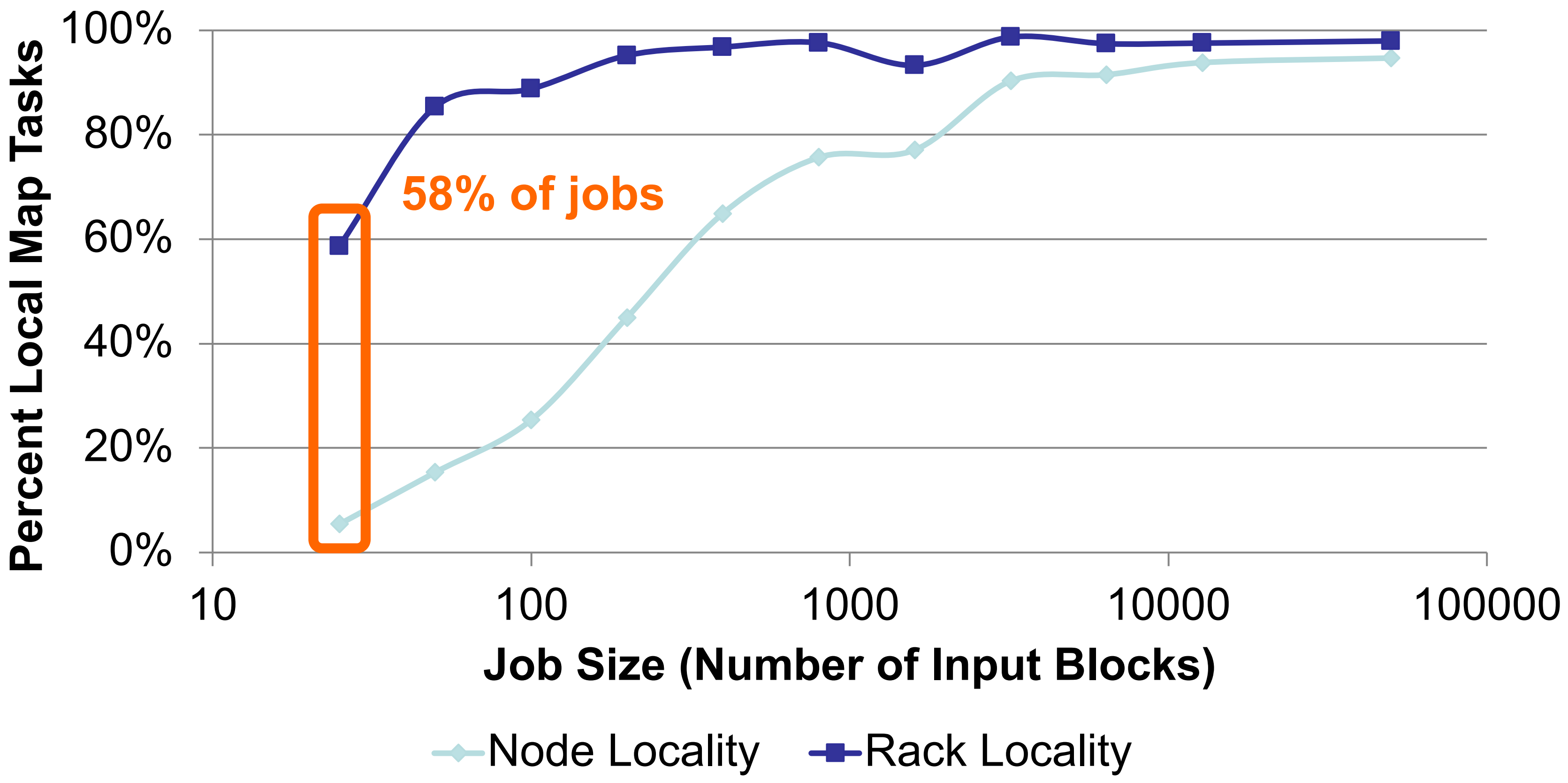
- Conceptually similar to Capacity Scheduler
- Group jobs into “**pools**” (queues)
- Assign each pool a guaranteed minimum share of resources.
  - On average every job in a pool will get an equal share overtime
- Divide extra capacity evenly between pools

- In practice:
  - Continuously maintain a sorted list of jobs by the number of running “Tasks” (small first)
  - When a resource is freed assign it to the head of the list

Data Locality and Scheduling

- Main challenge in the previous schedulers: **Data Locality**
  - For efficiency must run tasks near their input data
  - Strictly following any job queuing policy hurts locality: job picked by policy may not have data on free nodes

Data Locality and Scheduling
Data from Facebook Production Cluster



Zaharia, Matei, et al. **"Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling."** *Proceedings of the 5th European conference on Computer systems.* ACM, 2010.

Hadoop YARN – Delay Scheduling

- A technique that forces jobs to wait for a limited time if they cannot launch local tasks

- Empirically 1-5 seconds help reaching almost 100% map locality

- Delay scheduling works well under two conditions:
  - Sufficient fraction of tasks are *short* relative to jobs
  - There are *many locations* where a task can run efficiently
    - Blocks replicated across nodes, multiple tasks/node

Hadoop YARN – Delay Scheduling Pseudo code

*** Free container on node n**

```
Sort jobs according to queuing policy (FairShare, FIFO)
for j in jobs:
      if j has node-local task t on n:
               j.level := 0; j.wait := 0; return t
      else if j has rack-local task t on n and (j.level ≥ 1 or j.wait ≥ $T_1$):
               j.level := 1; j.wait := 0; return t
      else if j.level = 2 or (j.level = 1 and j.wait ≥ $T_2$)
                    or (j.level = 0 and j.wait ≥ $T_1 + T_2$):
               j.level := 2; j.wait := 0; return t
      else:
               j.wait += time since last scheduling decision
```

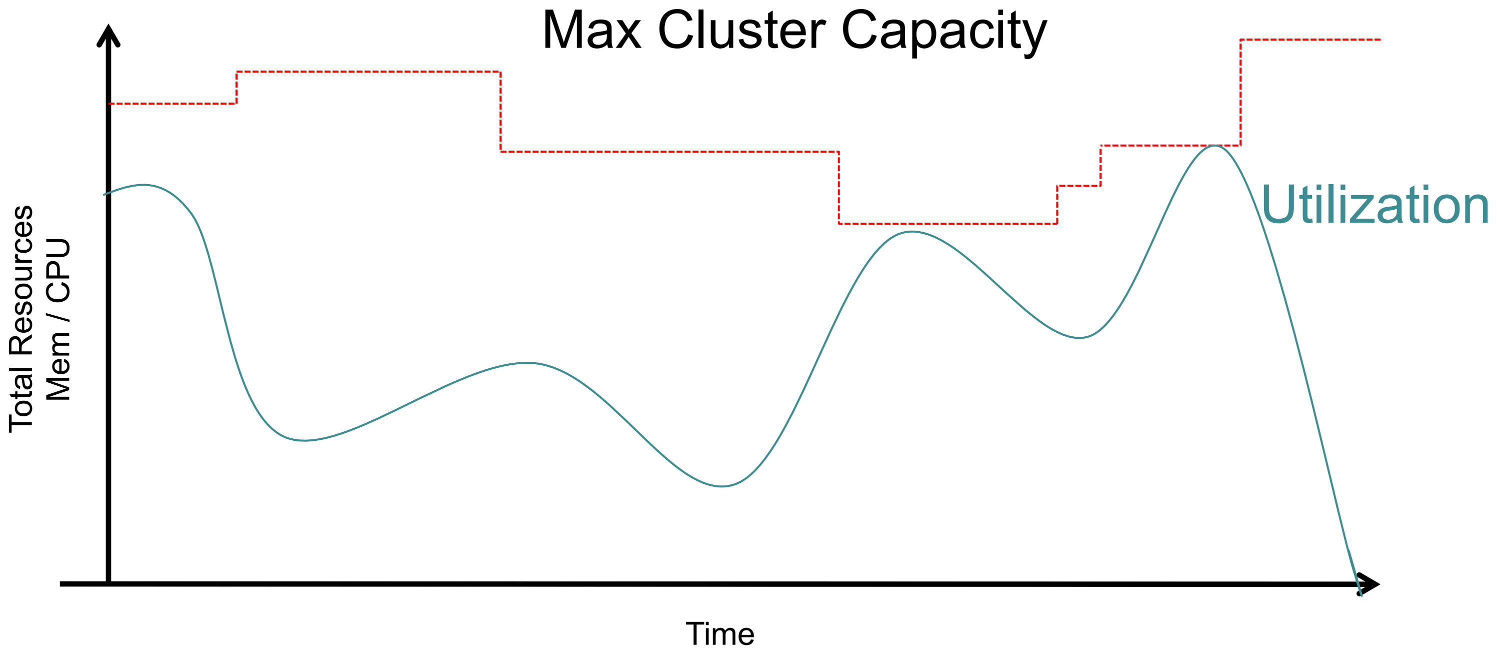
Zaharia, Matei, et al. **"Delay scheduling: a simple technique for achieving locality and fairness in cluster scheduling."** *Proceedings of the 5th European conference on Computer systems*. ACM, 2010.

Cloud Computing

HADOOP V3

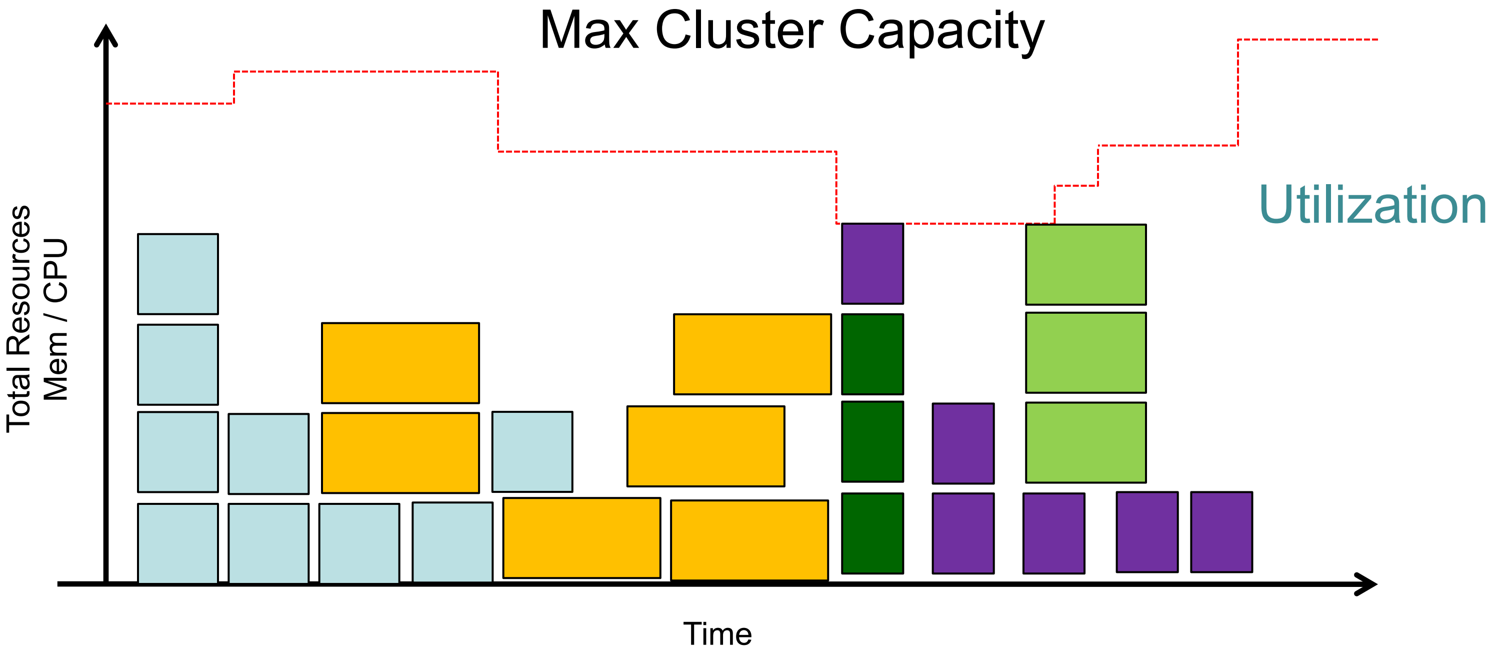
Cluster Resources

Resources: Cumulative #CPU and #Memory



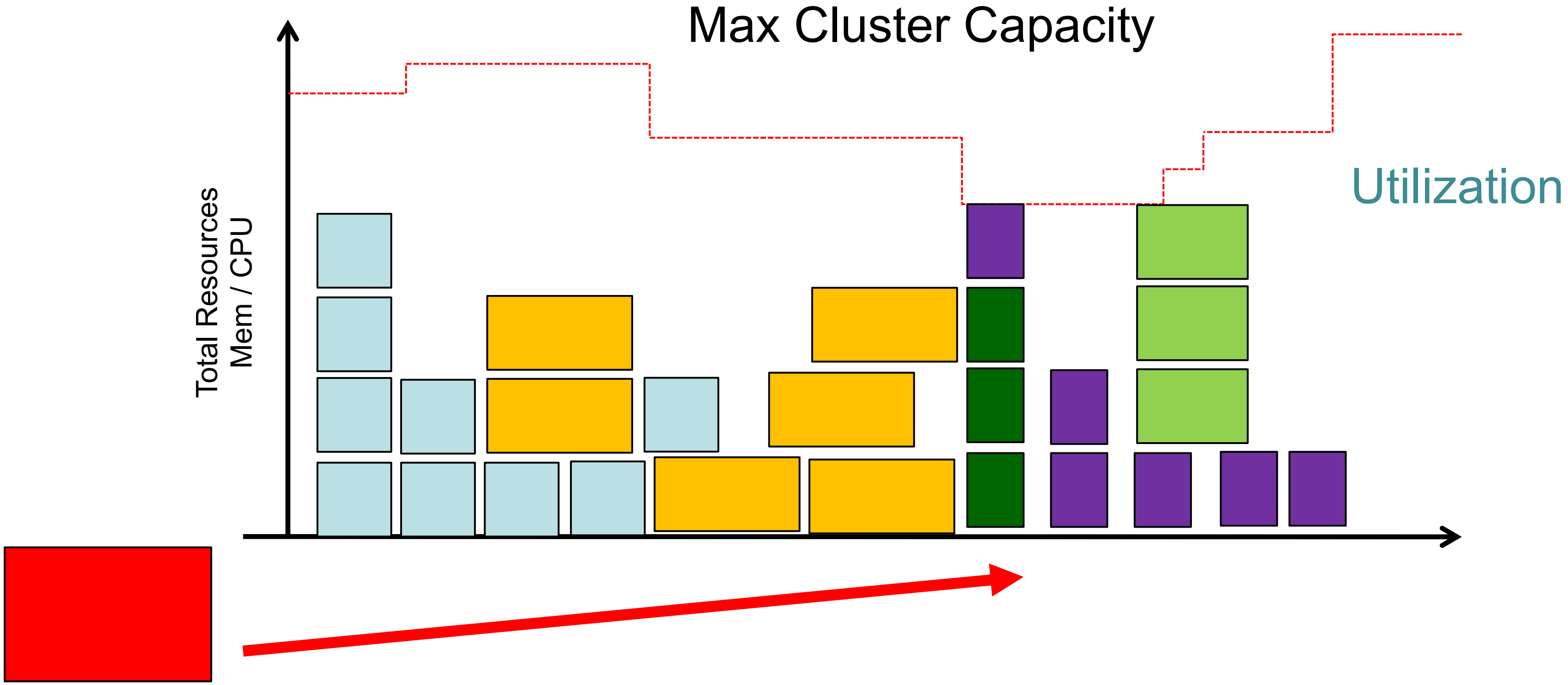
Cluster Resources

Resources: Cumulative #CPU and #Memory



Long Term Resource Planning

- Static capacity attribution managed by the system admin
  - Not feasible in a cloud setup (~millions of users)
- Many wasted opportunities for placing the jobs



Erasure Coding

- Replication Factor of 3 (default) leads to 200% storage overhead.
  - In a data processing this overhead mainly caters to data recovery is often unnecessary.

- In Hadoop v3 the overhead is reduced to 50% with support for Erasure Coding.
  - Blocks are not replicated using mirroring (exact copy)
  - Use of erasure coding to protect against hardware failure, e.g., recall parity blocks in RAID 5.

Misc. Features

	Hadoop V2	Hadoop V3
YARN Timeline Service	YARN timeline service introduced in Hadoop v2 has some scalability issues.	YARN Timeline service has been enhanced with ATS v2 which improves the scalability and reliability.
Intra DataNode Balancing	Nodes might have multiple disks added and replaced over time. HDFS Balancer in Hadoop v2 caused skew within a DataNode.	Intra DataNode Balancing has been introduced in Hadoop v3 to address the intra-DataNode skews which occur when disks are added or replaced.
Erasure Coding	Replication Factor of 3 for data recovery leading to 200% storage overhead. If a file has 6 data blocks then a total of 18 blocks will occupy the storage.	Storage overhead in Hadoop v3 is reduced to 50% with support for Erasure Coding. If a file has 6 data blocks, only 9 data blocks are required in total.
NameNode HA (High Availability)	Hadoop v2 can support an additional passive Namenode as standby.	Hadoop v3 supports 2 or more standby NameNodes