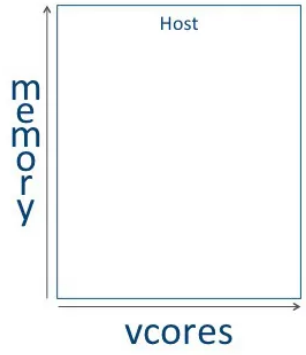
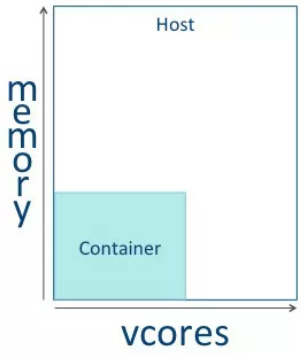
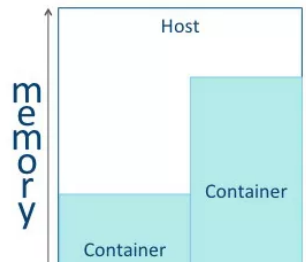
Tuning YARN (#concept_vbk_m43_fr)

This topic applies to YARN clusters only, and describes how to tune and optimize YARN for your cluster.

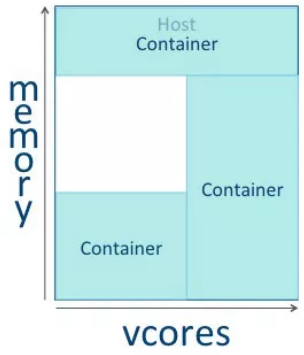
Note: Download the Cloudera YARN tuning spreadsheet (http://tiny.cloudera.com/yarn-tuning-guide?_ga=1.89859227.1261687805.1469631256) to help calculate YARN configurations. For a short video overview, see Tuning YARN Applications (https://www.youtube.com/watch?v=lykWFhrGvJ4&feature=youtu.be&list=PLe-h9HrA9qfC-5K7aSxvnq9ODJFmD5RyD) .

Overview (#concept_ulg_sbw_jv)

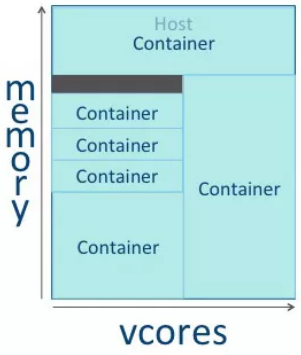
This overview provides an abstract description of a YARN cluster and the goals of YARN tuning.

A YARN cluster is composed of host machines. Hosts provide memory and CPU resources. A *vcore*, or virtual core, is a usage share of a host CPU.	
Tuning YARN consists primarily of optimally defining *containers* on your worker hosts. You can think of a container as a rectangular graph consisting of memory and vcores. Containers perform tasks.	
Some tasks use a great deal of memory, with minimal processing on a large volume of data.	

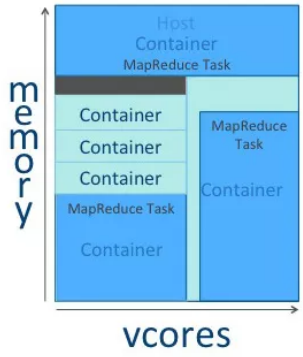
Other tasks require a great deal of processing power, but use less memory. For example, a Monte Carlo Simulation that evaluates many possible "what if?" scenarios uses a great deal of processing power on a relatively small dataset.



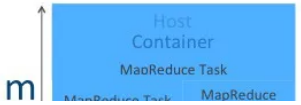
The YARN Resource Manager allocates memory and vcores to use all available resources in the most efficient way possible. Ideally, few or no resources are left idle.

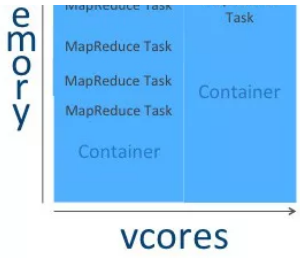


An *application* is a YARN client program consisting of one or more tasks. Typically, a task uses all of the available resources in the container. A task cannot consume more than its designated allocation, ensuring that it cannot use all of the host CPU cycles or exceed its memory allotment.



Tune your YARN hosts to optimize the use of vcores and memory by configuring your containers to use all available resources, beyond those required for overhead and other services.





There are three phases to YARN tuning. The phases correspond to the tabs in the YARN tuning spreadsheet (http://tiny.cloudera.com/yarn-tuning-guide).

1. Cluster configuration, where you configure your hosts.
2. YARN configuration, where you quantify memory and vcores.
3. MapReduce configuration, where you allocate minimum and maximum resources for specific map and reduce tasks.

There are many configurable properties for YARN and MapReduce. You can see the exhaustive list at Cloudera Manager Configuration Properties (cm_metrics.html#xd_583c10bfdbd326ba--7f25092b-13fba2465e5--7e52). The YARN tuning spreadsheet lists the essential subset of these properties that are most likely to improve performance for common MapReduce applications.

Cluster Configuration (#concept_om2_cs3_fr)

In the Cluster Configuration tab, you define the worker host configuration and cluster size for your YARN implementation.

Step 1: Worker Host Configuration (#step_1_worker_configuration)

Step 1 is to define the configuration for a single worker host computer in your cluster.

STEP 1: Worker Host Configuration

Enter your likely machine configuration in the input boxes below. If you are uncertain what machines you plan on buying, put in some minimum values that will suit what you expect to buy. Last updated early 2016.

Host Components	Quantity	Description
RAM	256	Gigabytes
CPU	48	8 CPUs: 6 cores, 3.5 GHz, 15MB cache
HDD (Hard Disk Drive)	36	12x3TB SATA III Hard Drives in JBOD Configuration
Ethernet	2	1 Gigabit Ethernet

As with any system, the more memory and CPU resources available, the faster the cluster can process large amounts of data. A machine with 8 CPUs, each with 6 cores, provides 48 vcores per host.

3 TB hard drives in a 2-unit server installation with 12 available slots in JBOD (Just a Bunch Of Disks) configuration is a reasonable balance of performance and pricing at the time the spreadsheet was created. The cost of storage decreases over time, so you might consider 4 TB disks. Larger disks are expensive and not required for all use cases.

Two 1-Gigabit Ethernet ports provide sufficient throughput at the time the spreadsheet was published, but 10-Gigabit Ethernet ports are an option where price is of less concern than speed.

Step 2: Worker Host Planning (#step_2_worker_host_planning)

Step 2 is to allocate resources on each worker machine.

STEP 2: Worker Host Planning

Now that you have your base Host configuration from Step 1, use the table below to allocate resources, mainly CPU and memory, to the various software components that run on the host.

Service	Category	CPU (cores)	Memory (MB)	CM Static Service %	Notes
Operating System	Overhead	1	8192	N/A	Most operating systems use 4-8GB minimum.
Cloudera Manager agent	Overhead	1	1024	N/A	Allocate 1GB for Cloudera Manager agents, which track resource usage on a host.
Other services	Overhead	0	0	N/A	Enter the required cores or memory for services not listed above.
HDFS DataNode	CDH	1	1024	4	Allocate 1GB for the HDFS DataNode.
Impala daemon	CDH	0	0	0	(Optional Service) Suggestion: Allocate at least 16GB memory when using Impala.
Hbase RegionServer	CDH	0	0	0	(Optional Service) Suggestion: Allocate no more than 12-16GB memory when using HBase Region Servers.
Solr Server	CDH	0	0	0	(Optional Service) Suggestion: Minimum 1GB for Solr server. More will be necessary depending on index sizes.
YARN NodeManager	CDH	1	1024	N/A	Allocate 1GB for the YARN NodeManager.
Available Resources		44	250880		
Physical Cores to Vcores Multiplier		4			Set this ratio based on the expected number of concurrent threads per core. Use 1 for CPU intensive tasks up to 4 for standard I/O bound tasks.

YARN Available Memory	250880	This value will be used in STEP 4 for YARN Configuration

Start with at least 8 GB for your operating system, and 1 GB for Cloudera Manager. If services outside of CDH require additional resources, add those numbers under Other Services.

The HDFS DataNode uses a minimum of 1 core and about 1 GB of memory. The same requirements apply to the YARN NodeManager.

The spreadsheet lists three optional services. For Impala, allocate at least 16 GB for the daemon. HBase RegionServer requires 12-16 GB of memory. Solr Server requires a minimum of 1 GB of memory.

Any remaining resources are available for YARN applications (Spark and MapReduce). In this example, 44 CPU cores are available. Set the multiplier for vcores you want on each physical core to calculate the total available vcores.

Step 3: Cluster Size (#step_3_cluster_size)

Having defined the specifications for each host in your cluster, enter the number of worker hosts needed to support your business case. To see the benefits of parallel computing, set the number of hosts to a minimum of 10.

STEP 3: Cluster Size

Enter the number of nodes you have (or expect to have) in the cluster

	Quantity
Number of Worker Hosts in the cluster	10

YARN Configuration (#concept_ij3_ns3_fr)

On the YARN Configuration tab, you verify your available resources and set minimum and maximum limits for each container.

Steps 4 and 5: Verify Settings (#steps_4_5_verify_resources)

Step 4 pulls forward the memory and vcore numbers from step 2. Step 5 shows the total memory and vcores for the cluster.

STEP 4: YARN Configuration on Cluster

These are the first set of configuration values for your cluster. You can set these values in YARN->Configuration in Cloudera Manager.

YARN Configuration Property	Value	
yarn.nodemanager.resource.cpu-vcores	176	Copied from STEP 2 "Available Resources"
yarn.nodemanager.resource.memory-mb	250880	Copied from STEP 2 "Available Resources"

STEP 5: Verify YARN Settings on Cluster

Go to the Resource Manager Web UI (usually http://<ResourceManagerIP>:8088/ and verify the "Memory Total" and "Vcores Total" matches the values above. If your machine has no bad nodes, then the numbers should match exactly.

Resource Manager Property to Check	Value	Note
Expected Value for "Vcores Total"	1760	Calculated from STEP 2 "YARN Available Vcores" and STEP 3
Expected Value for "Memory Total" (in GB)	2450	Calculated from STEP 2 "YARN Available Memory" and STEP 3

Step 6: Verify Container Settings on Cluster (#step_6_verify_container_settings)

In step 6, you can change the four values that impact the size of your containers.

The minimum number of vcores should be 1. When additional vcores are required, adding 1 at a time should result in the most efficient allocation. Set the maximum number of vcore reservations for a container to ensure that no single task consumes all available resources.

Set the minimum and maximum reservations for memory. The increment should be the smallest amount that can impact performance. Here, the minimum is approximately 1 GB, the maximum is approximately 8 GB, and the increment is 512 MB.

STEP 6: Verify Container Settings on Cluster

In order to have YARN jobs run cleanly, you need to configure the container properties.

YARN Container Configuration Property (Vcores)	Value	Description
yarn.scheduler.minimum-allocation-vcores	1	Minimum vcore reservation for a container
yarn.scheduler.maximum-allocation-vcores	32	Maximum vcore reservation for a container
yarn.scheduler.increment-allocation-vcores	1	Vcore allocations must be a multiple of this value

YARN Container Configuration Property (Memory)	Value	
yarn.scheduler.minimum-allocation-mb	1024	Minimum memory reservation for a container
yarn.scheduler.maximum-allocation-mb	8192	Maximum memory reservation for a container
yarn.scheduler.increment-allocation-mb	512	Memory allocations must be a multiple of this value

Step 6A: Cluster Container Capacity (#step_6a_cluster_container_capacity)

Step 6A lets you validate the minimum and maximum number of containers in your cluster, based on the numbers you entered.

Step 6A: Cluster Container Capacity

This section will tell you the capacity of your cluster (in terms of containers).

Cluster Container Estimates	Value
Largest number of containers, based on memory configuration	2450
Smallest number of containers, based on memory configuration	306
Largest number of containers, based on vcore configuration	1760
Smallest number of containers, based on vcore configuration	55

Step 6B: Container Sanity Checking (#step_6b_container_sanity_checking)

Step 6B lets you see at a glance whether you have over-allocated resources.

STEP 6B: Container Sanity Checking

This section will do some basic checking of your container parameters in STEP 6 against the hosts.

Sanity Check	Check Status	Description
Vcore Max >= Vcore Min	GOOD	yarn.scheduler.maximum-allocation-vcores must be greater than or equal to yarn.scheduler.minimum-allocation-vcores
Memory Max >= Memory Min	GOOD	yarn.scheduler.maximum-allocation-mb must be greater than or equal to yarn.scheduler.minimum-allocation-mb
VCoreMin <= HostsVCores	GOOD	yarn.scheduler.minimum-allocation-vcores must be less than or equal to the yarn.nodemanager.resource.cpu-vcores

MapReduce Configuration (#concept_rk2_4ty_fr)

On the MapReduce Configuration tab, you can plan for increased task-specific memory capacity.

Step 7: MapReduce Configuration (#step_7_mapreduce_configuration)

You can increase the memory allocation for the ApplicationMaster, map tasks, and reduce tasks. The minimum vcore allocation for any task is always 1. The Spill/Sort memory allocation of 256 should be sufficient, and should be (rarely) increased if you determine that frequent spills to disk are hurting job performance.

STEP 7: MapReduce Configuration

Property	Property Type	Component	Value	Description
yarn.app.mapreduce.am.resource.cpu-vcores	Config	Application Master	1	AM container vcore reservation
yarn.app.mapreduce.am.resource.mb	Config	Application Master	1024	AM container memory reservation
mapreduce.map.cpu.vcores	Config	Map Task	1	Map task vcore reservation
mapreduce.map.memory.mb	Config	Map Task	1024	Map task memory reservation
mapreduce.reduce.cpu.vcores	Config	Reduce Task	1	Reduce task vcore reservation
mapreduce.reduce.memory.mb	Config	Reduce Task	1024	Reduce task memory reservation
mapreduce.task.io.sort.mb	Config	Spill/Sort (Map Task)	256	Spill/Sort memory reservation

Step 7A: MapReduce Sanity Checking (#step7a_mapreduce_sanity_checking)

Step 7A lets you verify at a glance that all of your minimum and maximum resource allocations are within the parameters you set.

STEP 7A: MapReduce Sanity Checking

Sanity check MapReduce settings against container minimum/maximum properties.

Application Master Sanity Checks	Value	Description
yarn.app.mapreduce.am.resource.cpu-vcores >= container min	GOOD	Make sure ApplicationMaster vcore request fits within container limits
yarn.app.mapreduce.am.resource.cpu-vcores <= container max	GOOD	Ditto
yarn.app.mapreduce.am.resource.mb >= container min	GOOD	Make sure ApplicationMaster memory request fits within container limits
yarn.app.mapreduce.am.resource.mb <= container max	GOOD	Ditto

Map Task Sanity Checks	Value	Description
mapreduce.map.cpu.vcores >= container min	GOOD	Make sure Map Task vcore request fits within container limits

mapreduce.map.cpu.vcores <= container max	GOOD	Ditto
mapreduce.map.cpu.memory.mb >= container min	GOOD	Make sure Map Task memory request fits within container limits
mapreduce.map.cpu.memory.mb <= container max	GOOD	Ditto
Reduce Task Sanity Checks	Value	Description
mapreduce.reduce.cpu.vcores >= container min	GOOD	Make sure Reduce Task vcore request fits within container limits
mapreduce.reduce.cpu.vcores <= container max	GOOD	Ditto
mapreduce.reduce.cpu.memory.mb >= container min	GOOD	Make sure Reduce Task memory request fits within container limits
mapreduce.reduce.cpu.memory.mb <= container max	GOOD	Ditto

Configuring Your Cluster In Cloudera Manager (#configuring_in_cm)

When you are satisfied with the cluster configuration estimates, use the values in the spreadsheet to set the corresponding properties in Cloudera Manager. For more information, see Modifying Configuration Properties (cm_mc_mod_configs.html#cmug_topic_5_3)

Step	YARN/MapReduce Property	Cloudera Manager Equivalent

Cloudera Manager Property Correspondence

Step	YARN/MapReduce Property	Cloudera Manager Equivalent
4	`yarn.nodemanager.resource.cpu-vcores`	Container Virtual CPU Cores
4	`yarn.nodemanager.resource.memory-mb`	Container Memory
6	`yarn.scheduler.minimum-allocation-vcores`	Container Virtual CPU Cores Minimum
6	`yarn-scheduler.maximum-allocation-vcores`	Container Virtual CPU Cores Maximum
6	`yarn.scheduler.increment-allocation-vcores`	Container Virtual CPU Cores Increment
6	`yarn.scheduler.minimum-allocation-mb`	Container Memory Minimum
6	`yarn.scheduler.maximum-allocation-mb`	Container Memory Maximum
6	`yarn.scheduler.increment-allocation-mb`	Container Memory Increment
7	`yarn.app.mapreduce.am.resource.cpu-vcores`	ApplicationMaster Virtual CPU Cores
7	`yarn.app.mapreduce.am.resource.mb`	ApplicationMaster Memory
7	`mapreduce.map.cpu.vcores`	Map Task CPU Virtual Cores
7	`mapreduce.map.memory.mb`	Map Task Memory
7	`mapreduce.reduce.cpu.vcores`	Reduce Task CPU Virtual Cores
7	`mapreduce.reduce.memory.mb`	Reduce Task Memory
7	`mapreduce.task.io.sort.mb`	I/O Sort Memory

Categories: Clusters (../categories/hub_clusters.html) | Configuring (../categories/hub_configuring.html) | MapReduce (../categories/hub_mapreduce.html) | Memory (../categories/hub_memory.html) | Performance (../categories/hub_performance.html) | Planning (../categories/hub_planning.html) | Requirements (../categories/hub_requirements.html) | Sizing (../categories/hub_sizing.html) | Tuning (../categories/hub_tuning.html) | YARN (../categories/hub_yarn.html) | All Categories (../categories/hub.html)

- About Cloudera (http://www.cloudera.com/about-cloudera.html)
- Resources (http://www.cloudera.com/resources.html)
- Contact (http://www.cloudera.com/contact-us.html)
- Careers (http://www.cloudera.com/about-cloudera/careers.html)
- Press (/about-cloudera/press-center.html)
- Documentation (/documentation.html)

United States: +1 888 789 1488
Outside the US: +1 650 362 0488

- (https://www.linkedin.com/company/cloudera)
- (https://www.facebook.com/cloudera)

- (https://twitter.com/cloudera)
- (/contact-us.html)