Qubole®

Workload-Aware Auto-Scaling

A new paradigm for Big Data Workloads

Executive Summary

Auto-Scaling has become a common concept with the advent of the Public Cloud. It was one of the first techniques that allowed applications to exploit the elasticity of the Cloud. However - as the Cloud gained popularity and more complex applications moved to the Cloud – first generation Auto-Scaling technologies fell behind in serving the requirements of such applications.

In this document we describe **Workload-Aware Auto-Scaling.** This is an alternative architectural approach to Auto-Scaling that is better suited for new classes of applications like **Hadoop, Spark and Presto** that have now become commonplace in the Cloud. We show that traditional auto-scaling technologies are ill-suited for Big Data applications and that Workload-Aware Auto-Scaling technologies such as that offered by **Qubole** are vastly superior. These technologies result in significant benefits to Reliability, Cost and Responsiveness for Big Data Applications.

Auto-Scaling – a Short History

AWS introduced Auto-Scaling Groups in 2009[1]. In its introduction, the blog notes:

Auto-Scaling lets you define scaling policies driven by metrics collected by Amazon CloudWatch. Your Amazon EC2 instances will scale automatically based on actual system load and performance but you won't be spending money to keep idle instances running.

Auto-Scaling defined in this manner was largely targeted for stateless applications – like web servers – where the state was stored on external databases & caches. Real-time metrics like CPU and Memory utilization were used by applications to dynamically add or remove nodes – as shown in the Figure below:

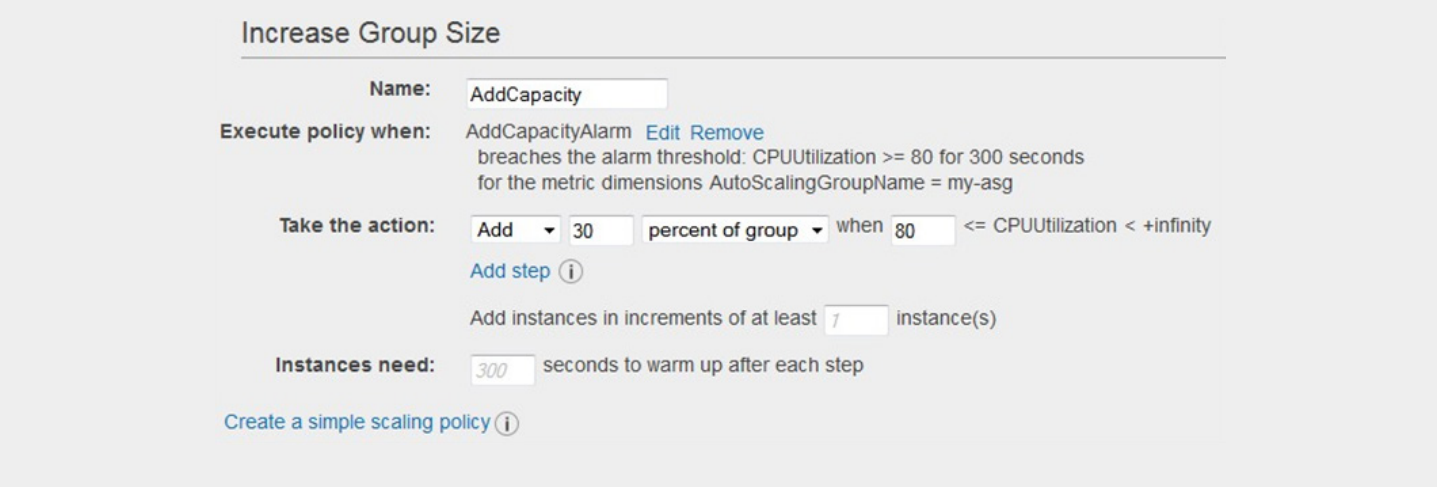


Figure 1: Auto-Scaling using CPU Utilization in AWS

Simple strategies like this work fairly well for web applications. Some salient characteristics of web applications are relevant to the way these auto-scaling systems were designed:

- They are Stateless
- Every application request (say a HTTP request) is usually very short lived
- Application workloads are driven by external clients and not known in advance
- Usually applications want to minimize response latency (as opposed to optimizing cost)
- All nodes are usually symmetrical from the point of view of CPU/memory usage
- An application (hosted on a single auto-scaling group) is homogeneous
- Application workload changes are often smooth (say increasing gradually during working hours and declining thereafter)

Enter Big-Data

As AWS was introducing Auto-Scaling groups in 2009 – Big Data was just coming into being – with Hadoop and later Spark and Presto becoming commonly used to wrangle with large data sets. The Public Cloud was a match made in heaven for Big Data. Large data sets were much more easily stored in Cloud Storage Systems like S3 – and large scale and bursty computation requirements of Big Data applications were ideally suited for the large and elastic pools of compute resources available in these Clouds.

The auto-scaling policies described above are easy to comprehend and it is not surprising that the same architecture got co-opted for running Big Data workloads. We see a similar approach in commercial Cloud Hadoop offerings (AWS EMR and Hortonworks) for scaling a Hadoop Cluster in 2017 as we saw for scaling web applications in 2009.

A. AWS EMR[2]

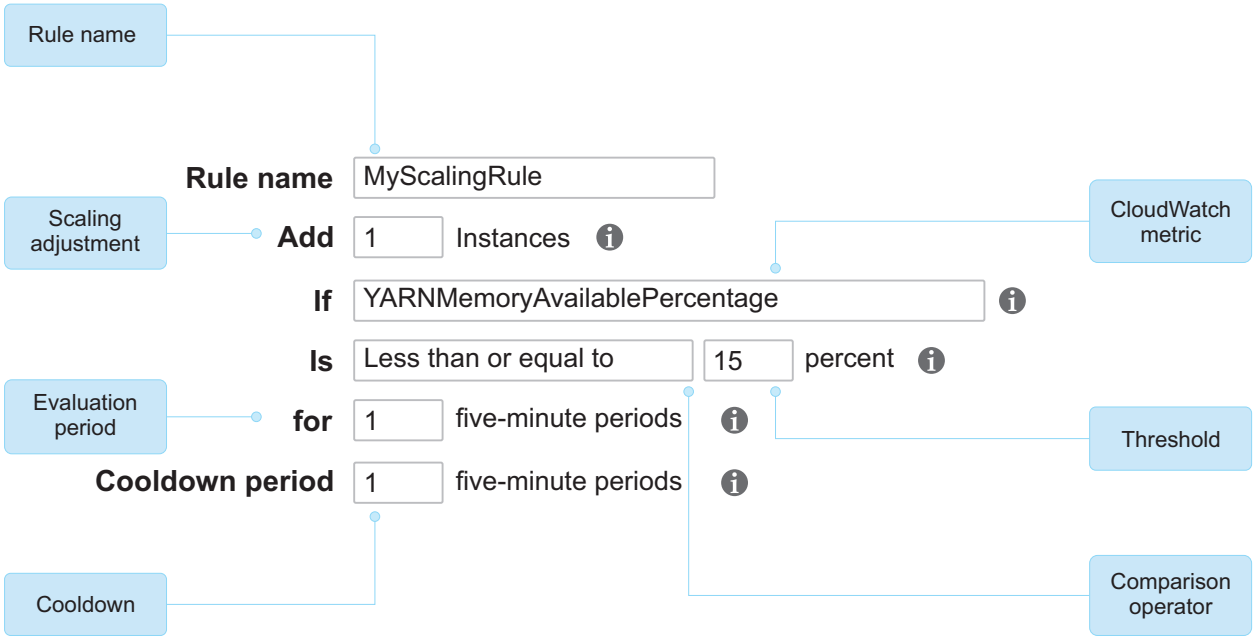


Figure 2: Auto-Scaling using memory utilization in AWS EMR

Hortonworks HDP-AWS[3]

Scaling Event	Scaling Policy
If **CPU usage** is more than 70% for 5 minutes (averaged across all cluster nodes).	Add 3 compute nodes
If **HDFS usage** is less than 10% for 15 minutes.	Remove 5 worker nodes
If **Memory usage** is more than 50% for 10 minutes (averaged across all cluster nodes).	Add 3 compute nodes

Figure 3: Configuring Auto-Scaling in HortonWorks Data Cloud

Hadoop is not a Web Server!

Big Data workloads are a complete contrast to standard Applications. A single cluster (the rough equivalent of an auto-scaling group) is submitted multiple simultaneous discrete applications. Each of these applications can comprise up to hundreds of thousands of tasks. Some of the differing characteristics of these applications are as follows:

Stateful Servers:

Most big data applications store state on each node while they are running.
Removing nodes without accounting for this state can cause workloads and even the entire cluster to fail. The various kinds of state that can be stored in each node include:

- Data belonging to HDFS
- Data belonging to a distributed cache (like RDDs in Spark)
- Intermediate data produced by tasks that are needed by subsequent tasks in the application (for example: Map Outputs in Map-Reduce parlance)

Non-Uniform Server Load:

Nodes in a big data cluster often have widely varying load factors.
Some nodes have more memory intensive tasks and some have more CPU intensive ones. Running tasks can be of widely varying time durations. The amount of data stored on each node can also vary widely depending on what applications it has been part of.

Long Running Requests:

Individual tasks comprising a Big Data Application can run for hours.
Some tasks (like Reducers) run for long time gathering data from previous stages – or they can be long running simply because they are processing too much data (say due to Skews).

Workload Awareness:

Profile of Big Data applications running in a cluster are known up-front.
Unlike web applications where the requests are generated from external clients – in a Big Data cluster – task units are generated by an application that is submitted to a coordinator daemon[4] in the cluster. As such the characteristics of the application - the number of tasks it will generate (or even control over the same), the data it will read and the computation it will perform on it – are all known to the coordinator.

In many cases – applications are repeated. For example the same reporting query may run frequently, or the same ETL job may run periodically in the night. This further helps a smart coordinator anticipate in advance the nature of the workload submitted to it.

Utilization Sensitive:

Big-Data workloads are usually very cost sensitive.
A big subset is the ETL applications that care more about Reliability and Cost (which translates into a desire for high cluster utilization). Another big subset is ad-hoc query and analysis that are latency sensitive (but are also somewhat cost sensitive).

Workload Burstiness:

A Big Data cluster can go from idle to running 100k tasks in an instant.
This is contrast to usual web traffic where traffic usually goes up relatively smoothly even in the worst of days.

These differences can be summarized thus:

Auto-Scaling Type / Application Characteristic	Standard Application	Big Data Application
Stateful Servers	✗	✓
Uniform Load on Servers	✓	✗
Long Running Requests	✗	✓
Latency Sensitive	High	Variable
Utilization Sensitive	Low	High
Workload Burstiness	Moderate	Extremely High
Workload Awareness	✗	✓

Table 1: Standard versus Big-Data Application

All these completely upturn the assumptions that underlie old world auto-scaling technologies and make it a very poor fit. Consider these immediate observations:

- **Removing nodes while downscaling is hard:** both because of long running tasks as well as accumulated state.
- **Downscaling algorithms need to pick nodes carefully:** Nodes are no longer uniformly loaded – neither do they have equal amounts of application state.
- **Same auto-scaling policies cannot be applied to all workloads:** Some workloads want low latency, some high utilization. Some may have SLA constraints and some may have budget caps.
- **Usage of pre-emptible nodes(like AWS EC2 Spot Instances) is hard:** primarily because nodes are stateful. Pre-empted nodes can even lead to cluster failure. At the same time – usage of pre-emptible resources becomes extremely important to reduce costs – particularly for ETL workloads.
- **Cluster scaling has to take application characteristics into account:** as the most trivial example - one cannot repeatedly upscale by a small step function to satisfy a 100k task application. That may take a very long time.

Workload Aware Auto-Scaling

When we started building auto-scaling technologies at Qubole, we evaluated and rejected[4] existing approaches to auto-scaling as being insufficient for building a truly Cloud-Native Big Data solution. Instead we built Auto-Scaling into Hadoop and Spark where it has access to the details of the Big Data applications and the detailed state of the cluster nodes.

Being Workload Aware has made a dramatic difference to our ability to orchestrate Hadoop and Spark in the Cloud. The different ways in which we have used this awareness include:

Upscaling:

Qubole managed clusters look at a variety of criteria - beyond resource utilization - to come up with upscaling decisions. Some examples:

- **Parallelism-Aware:** If applications have limited parallelism (say a Job can only use 10 cores) - then upscaling will not scale the cluster beyond that number (even though the cluster may exhibit high resource utilization)
- **SLA-Aware:** Qubole monitors jobs for estimated completion time and adds compute resources if they can help meet SLA. If a Job can be predicted to complete in its required SLA then no upscaling is triggered on its behalf (even though resource utilization may be high). A large job with thousands of tasks will trigger a much stronger upscaling response than a small job.
- **Workload Aware Scaling Limits[5]:** If an application is limited in the number of CPU resources it can use (say because of limits put in by the administrator) - then it will not trigger upscaling if it is already using the maximum resources allowed.
- **Recommissioning:** Any upscaling requirements are first attempted to be fulfilled using any nodes that are currently in the process of Graceful Downscaling.

Furthermore a composite cluster upscaling decision is taken depending on the requirements of each of the jobs running in the cluster.

Downscaling:

- **Smart Victim Selection:** Tasks running on each node and the amount of state on each node are considered in addition to the node launch time to determine which nodes are safe and optimal to remove from the cluster when down-scaling.
- **Graceful Downscaling:** All state from a node is copied elsewhere before removing it from the cluster. This includes HDFS decommissioning and log archival – and in cases of forced downscaling – also involves *offloading* intermediate data to Cloud Storage.
- **Container Packing[6] :** Scheduling algorithms inside YARN are modified to pack tasks into a smaller set of nodes that allows more nodes to be available for downscaling.

Composite Health Checks:

We periodically check running nodes in a cluster against their membership and health status in HDFS (distributed storage system) and YARN (distributed computing system). Nodes that don't pass such composite health checks are automatically removed from the cluster.

Spot Loss Notification:

YARN based Hadoop and Spark clusters in Qubole are able to deal with Spot Loss Notifications provided by AWS. The cluster management software immediately shuts off Task scheduling on such nodes, stops further HDFS writes to such nodes, backs-up container logs on these nodes and tries its best to replicate any state left on such nodes to other surviving nodes in the system

Spot Rebalancing[7]:

We are not only able to downscale nodes that are free - but able to evaluate which nodes have the most accumulated state/tasks and may be the easiest to retire. In most cases we can even estimate the amount of time required to retire a node. This sophistication allows us to build features like Spot Rebalancing where a cluster with excess on-demand nodes can retire them when it finds that Spot Nodes have become available in the Spot market.

Cloud-Aware Workload Management

Just like Auto-Scaling technologies benefit enormously by being Workload-Aware – the dual is also true. Workload management technologies inside Big Data engines– like Hadoop, Spark and Presto – benefit enormously from being Cloud aware. A few examples are in order:

Spot Awareness[8]: HDFS and Job Schedulers in Qubole's Hadoop/Spark/Presto clusters are aware of which nodes are preemptible Spot nodes (and hence unreliable) and which nodes are regular ones. This knowledge allows us careful placement of data and tasks to allow applications to run reliably in the presence of Spot losses:

- HDFS Data Blocks are, by default, replicated to Spot and On-Demand nodes
- Important Tasks - like Application Master and Qubole Control Jobs (Shell Commands) are not placed on Spot Nodes (and this limitation is factored into Auto-Scaling logic)

Task Estimation: A key step in all Big Data technologies is dividing processing into small chunks that can be performed in parallel. The maximum computing resources available to an application can be used to dynamically compute such parallelism (this is now dynamic and configurable where it was previously static).

Heterogeneous clusters[9]: In heterogeneous clusters –any one of different types of nodes can be chosen for Upscaling. The knowledge of workload requirements at any instant can allow the cluster management software to choose the right instance for cluster upscaling or downscaling. Moreover the knowledge of different heterogeneous instance types can be used to automatically come up with optimal configurations for a specific job.

The table below summarizes the above technological differences between traditional and workload-aware auto-scaling technologies:

	Auto-Scaling Type	Traditional	Workload-Aware
Feature Group	**Features**		
	Load Monitoring	✓	✓
	Simple Health Check	✓	✓
Upscaling	*Parallelism Aware*	✗	✓
	SLA Aware	✗	✓
	Recommissioning	✗	✓
	Workload Specific Scaling Limits	✗	✓
Downscaling	*Smart Victim Selection*	✗	✓
	Graceful Downscaling	✗	✓
	Container Packing	✗	✓
	Composite Health Checks	✗	✓
Spot Nodes	*(Spot) Node Rebalancing*	✗	✓
	(Spot) Node Loss Notification Handling	✗	✓
	Spot Aware Scheduling	✗	✓
	Heterogeneous Clusters	✗	✓

Table 2: Traditional versus Workload-Aware Auto-Scaling

Conclusion

We have shown comprehensively how the nature of Big Data applications differs substantially from simple online applications like Web Servers. To truly take advantage of the Cloud – one has to integrate auto-scaling deep into the Big Data stack so that it is Workload-Aware. A true Cloud Native implementation also makes the Big Data stack aware of the Cloud resources and helps it adapt workload and data management in response to it.

The described technologies are already, or soon planned to be, part of the Qubole Big Data Platform offering.

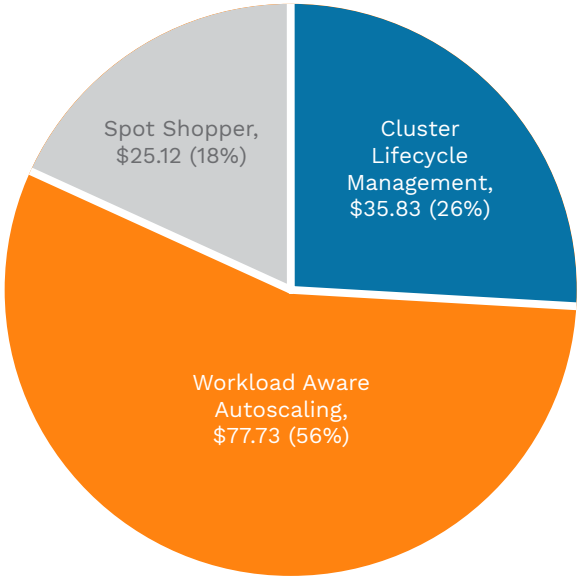
Appendix – Qubole TCO Savings in Practice

The Workload Aware Auto-Scaling White Paper has described why generic approaches to auto-scaling are inefficient and costly for big data use cases in the cloud. Qubole has pioneered workload aware auto-scaling for big data over the last several years and delivered the technology into a generally available production service in 2017. By working with over 200 big data customers of all sizes and in multiple industries, we have also been able to construct models that quantify the financial impact of utilizing workload-aware auto-scaling in real life environments. This appendix rounds out the technology paper with the financial insights.

First, the cost of ownership savings of using Qubole as a data platform in the cloud are 80% overall as measured in typical customer environments whether the comparison is to cloud or on-premise big data! Qubole customers have save $140M in costs since 2016 (without counting our largest customer who could distort the savings upwards). The costs savings measured primarily derive from 3 automation efficiencies Qubole brings to bear with automation agent technology. 100% of the savings are not due to workload aware auto-scaling (55% are), but 100% of the savings documented here across more than 200 customers are available to any business using the Qubole service. The 3 primary drivers of cost savings are:

1. Cluster Life Cycle Management (CLCM) – Amount saved by automatically terminating a cluster when it is inactive vs. allowing it to continue to run at a minimum capacity in the absence of CLCM.
2. Workload-Aware Auto-scaling – Amount saved by predictively adjusting the number of nodes to meet demand vs. allowing clusters to run at full capacity for the duration of the instance in the absence of the agent.
3. Spot Shopper savings – Amount saved by using SPOT instances at an assumed 80% discount over on-demand instance pricing thanks to the Qubole agent.

Savings by Qubole Automation Features ($140M in compute costs)



End notes

Page 2 - [1] New Features for Amazon EC2: Elastic Load Balancing, Auto Scaling, and Amazon CloudWatch - Jeff Barr, AWS Blog, May 18, 2009

Page 3 - [2] Using Automatic Scaling in Amazon EMR - AWS EMR documentation

Page 3 [3] EC2 Spot Notes - AWS EMR documentation

Page 7 [4] Auto-Scaling Hortonworks Data Cloud - HDP AWS documentation

Page 7 [5] Industry's First Auto-Scaling Hadoop Cluster - Joydeep Sen Sarma Qubole Blog, Jun 17, 2012

Page 8 [6] HDFS Decommissioning - Apache Hadoop 2.7.2 Documentation

Page 8 [7] Rebalancing Hadoop Clusters for Higher Spot Utilization - Hariharan Iyer, Qubole Blog, Jun 9, 2015

Page 9 [8] Riding the Spotted Elephant - Mayank Ahuja, Qubole Blog, No 12, 2015

Page 9 [9] Qubole announces Heterogeneous Clusters on AWS - Hariharan Iyer, Qubole Blog, Nov 30, 2016

About Qubole

Qubole is passionate about making data-driven insights easily accessible to anyone. Qubole customers currently process nearly an exabyte of data every month, making us the leading cloud-agnostic big-data-as-a-service provider. Customers have chosen Qubole because we created the industry's first autonomous data platform. This cloud-based data platform self-manages, self-optimizes and learns to improve automatically and as a result delivers unbeatable agility, flexibility, and TCO. Qubole customers focus on their data, not their data platform. Qubole investors include CRV, Lightspeed Venture Partners, Norwest Venture Partners and IVP. For more information visit **www.qubole.com**

For more information:

Contact:
sales@qubole.com

Try QDS for Free:
https://www.qubole.com/products/pricing/

469 El Camino Real, Suite 205
Santa Clara, CA 95050
(855) 423-6674 | info@qubole.com

WWW.QUBOLE.COM