WHITEPAPER

PEPPERDATA OVERVIEW AND DIFFERENTIATORS

INTRODUCTION

Prospective customers will often pose the question, “How is Pepperdata different from tools like Ganglia, Cloudera Manager, and Ambari?” The purpose of this whitepaper is to address this question comprehensively. In this document, we speak in terms of jobs and tasks but the same also applies in a YARN framework. Pepperdata fully supports both Hadoop 1 and Hadoop 2 (YARN).

PEPPERDATA OVERVIEW

Pepperdata is an enterprise software product that installs in about 20 minutes on your existing Hadoop cluster without any modifications to scheduler, workflow, or job submission processes. It is compatible with all of the Hadoop distributions (Cloudera, Hortonworks, MapR, IBM BigInsights, Pivotal PHD), and it runs on the Apache version as well. Once installed, Pepperdata provides you with three immediate benefits:

VISIBILITY

Captures an unprecedented level of detail on cluster resource usage

Pepperdata collects 200+ metrics in real-time for the four resources CPU, RAM, disk I/O, and network for any given job or task, by user or group or queue. This allows operators to quickly identify what job is causing a problem and which user submitted it. And it allows users to see what and how their jobs are doing on the cluster while they are running. Because users and operators are finally able to see what the jobs are actually doing on the cluster, the jobs can be improved.

CONTROL

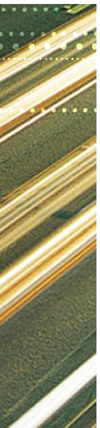
Enables you to implement service-level policies that guarantee on-time completion of high-priority jobs

Pepperdata senses contention among the four resources in real-time and will slow down low-priority jobs just enough to ensure the high-priority SLAs are always maintained. This SLA enforcement is ideal for multi-tenant environments, such as a cluster that is a central service that various business units utilize or for a cluster that has a lot of mixed workloads running. With Pepperdata there is no longer any need to isolate workloads onto separate clusters to protect high-priority production jobs. And with our HBase protection, you no longer have to worry about HBase and MapReduce jobs interfering with one another when running on the same cluster.

CAPACITY

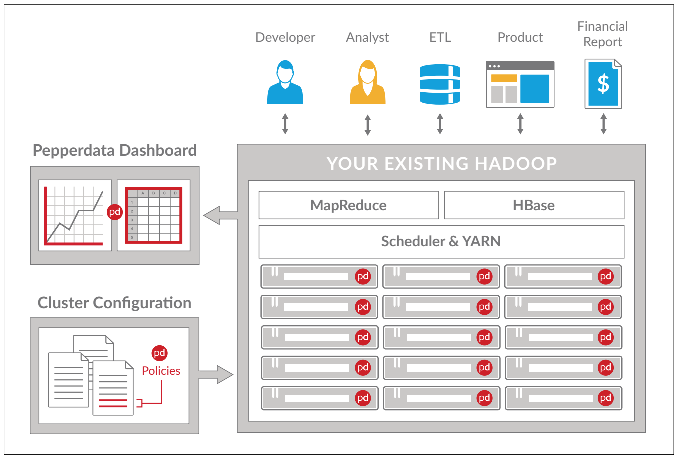
Increases cluster throughput by 30-50%

Pepperdata knows the actual true hardware resource capacity of your cluster and allows more tasks to run on nodes that have free resources at any given moment. In many instances jobs will run much faster because Pepperdata will dynamically allow them to use more of the true resource on the cluster when it is available. By installing Pepperdata on your existing cluster, it will feel like you just added more servers because there will be an immediate lift in terms of the number of jobs you can run concurrently, or the same jobs run faster. And Pepperdata software costs a lot less than purchasing more servers, allowing you to take full advantage of your investments in your infrastructure.



pepperdata

PEPPERDATA REAL-TIME ARCHITECTURE



Pepperdata's architecture

PEPPERDATA DIFFERENTIATORS

To answer the question of how Pepperdata is different from Ganglia, Cloudera Manager, or Ambari, it is useful to consider the question in the context of the three value propositions that Pepperdata provides.

Comparison with CONTROL and CAPACITY value propositions

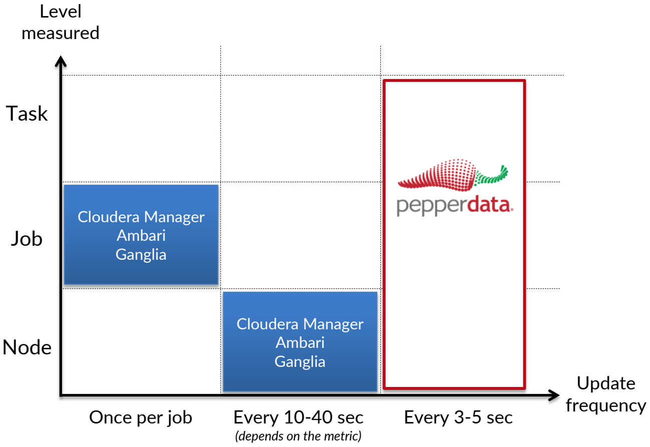
Ganglia, Cloudera Manager, and Ambari do nothing in the areas of SLA enforcement or increasing Hadoop's throughput, so those two value propositions are clear differentiators for Pepperdata.

Comparison with VISIBILITY value proposition

So that leaves the value proposition of visibility. That's where most people need some further explanation to get the difference. People often ask, "Don't those tools provide Hadoop cluster monitoring too?"

To help explain the difference, we've prepared the following diagram which should shed some light. The Y-axis represents ever more increasing level of granularity in what is measured as you move up the axis. Node is the least amount of granularity, followed by job, and then the task. The X-axis represents when something is measured, in ever increasing frequency as you move right along the axis.

All three tools essentially provide the same level of visibility, with Ambari actually relying on Ganglia for its collection of metrics. In the following pages we now go into detail on each of the three tools shown in the chart above and make comparisons with the functionality provided by that of the Pepperdata Dashboard.



Pepperdata comparison chart with Ganglia, Cloudera Manager, and Ambari

Ganglia

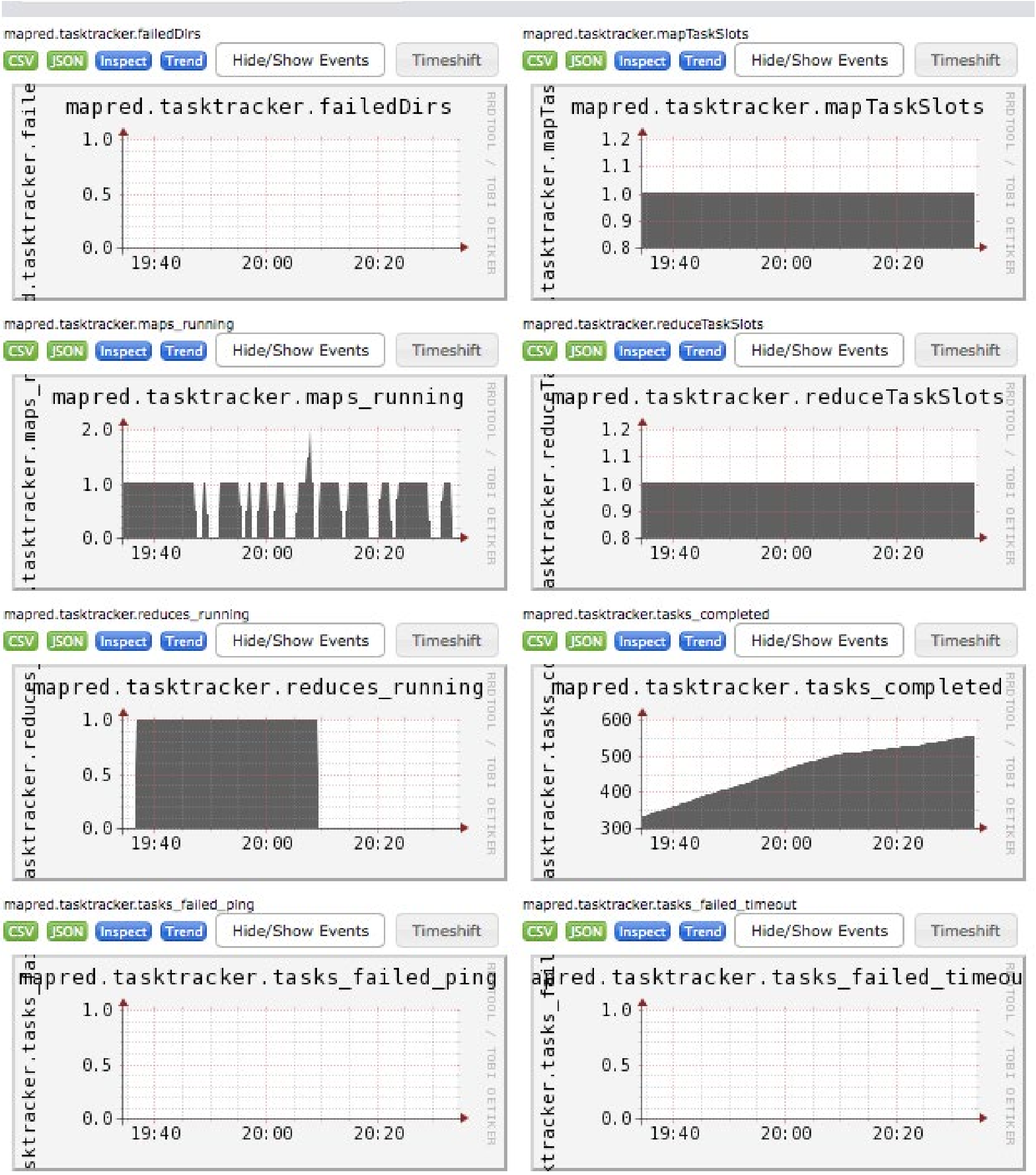
Ganglia is an open-source distributed system monitoring tool for high-performance computing systems such as clusters. (This section will also explain most of Ambari's monitoring functionality, since it uses Ganglia for most of its metrics.) There is a Hadoop plugin for Ganglia to provide some Hadoop-specific metrics. It allows the user to remotely view live or historical statistics (such as CPU load averages or network utilization) for all machines that are being monitored.

Ganglia's current architecture has some issues that have proven to be challenges when deployed at Internet-level scale (thousands of nodes). For example, within a single cluster the quadratic message load incurred by a multicast-based listen/announce protocol will not scale properly to thousands of nodes. There are limitations on all metrics having to fit within a single IP datagram, the lack of a hierarchical namespace, large I/O overhead incurred by the use of RRDtool, and lack of access control mechanisms on the monitoring namespace.

Out of the box, Ganglia provides node-level metrics every 10-40 seconds, depending on the metric, and sends them to the collector every 90-180 seconds. That's it. There's no visibility by job or by task, so all you'll be able to see is that CPU is running hot, or disks on the node are thrashing, but you won't know what's causing it. Ganglia also has a Hadoop plugin, but it just gives you access to the existing internal Hadoop metrics. Some of those metrics are interesting (for example, HDFS reads/writes over time), but you don't have a good way to correlate them with jobs.

The metrics provided by YARN, MapReduce, and HBase are cumulative numbers, such as the numbers of jobs running, how many tasks are blocked, failed, killed, and running, and similar numbers. There is no visibility into which processes are causing the resource utilization you are seeing within your cluster, making these interesting numbers for capacity planning, but not very good for troubleshooting, chargebacks, or any usage which needs deeper introspection.

On this page is an image from Ganglia's latest release (3.6.1) with the Hadoop plugin. There is some time-series information, but it's all summary-level information and doesn't give any detail about jobs and their tasks and nothing about the user, group, or queue so there is no way to tell who is doing what on the cluster.


mapred.tasktracker.failedDirs
CSV
JSON
Inspect
Trend
Hide/Show Events
Timeshift
mapred.tasktracker.failedDirs
1.0
0.5
0.0
19:40
20:00
20:20
mapred.tasktracker.mapTaskSlots
mapred.tasktracker.mapTaskSlots
1.2
1.1
1.0
0.9
0.8
mapred.tasktracker.maps_running
mapred.tasktracker.maps_running
2.0
1.0
0.0
mapred.tasktracker.reduceTaskSlots
mapred.tasktracker.reduceTaskSlots
mapred.tasktracker.reduces_running
mapred.tasktracker.reduces_running
mapred.tasktracker.tasks_completed
mapred.tasktracker.tasks_completed
600
500
400
300
mapred.tasktracker.tasks_failed_ping
mapred.tasktracker.tasks_failed_ping
mapred.tasktracker.tasks_failed_timeout
apred.tasktracker.tasks_failed_timeou
RRDTOOL / TOBI OETIKER

Screenshot of Ganglia TaskTracker metrics

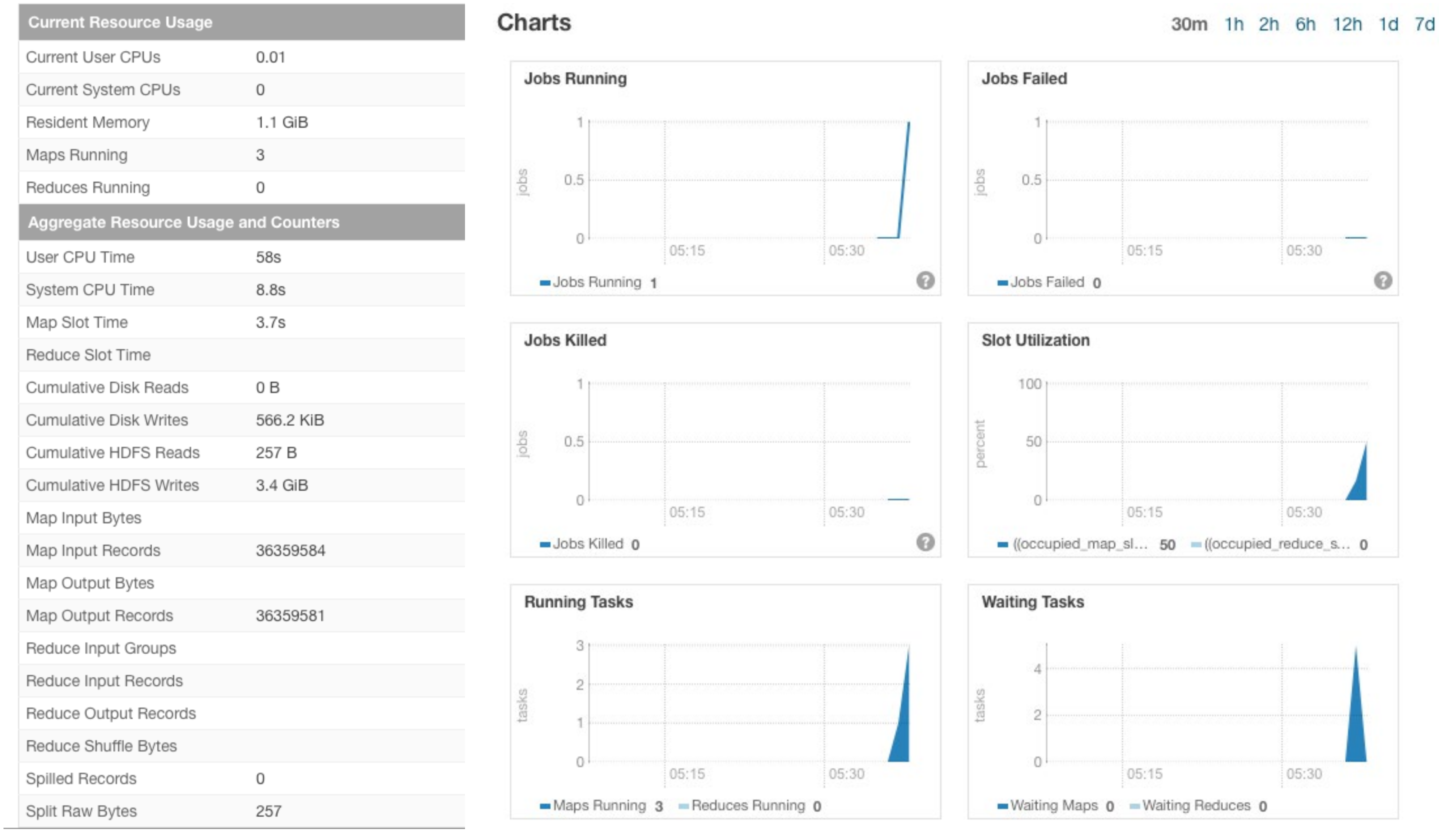
Cloudera Manager

Cloudera Manager allows Hadoop operators to deploy and centrally operate Cloudera's distribution of Hadoop. It automates the installation process, gives you a cluster-wide view of nodes and services running, and provides a single central console to enact configuration changes across your cluster. It also incorporates reporting and diagnostic tools, which is the area of functionality where it is necessary to explain how Pepperdata is different.

Cloudera Manager collects time-series data for every minute and does a lot of pre-aggregation of the information for node-level metrics. It provides node and job-level metrics, but the job metrics are just collected from Hadoop counters so only provide you summary-level information for the job, just like Ganglia (with the Hadoop plugin). You have no way of know what Hadoop is doing while the job is running, similar to the limitations of Ganglia. These limitations are a result of the Hadoop metrics and node-level metrics being gathered in the same way, even if the presentation and aggregation of the data may vary. These tools will provide some visibility into other services that might not have the same Ganglia support in Hadoop (Pig, Hive, Impala), but will still have the same limitations.

Below is an image from Cloudera Manager's latest release 5.2. It has summary-level metrics along the left-hand column and some job time-series information to the right. This is all summary-level and doesn't tell you what job or what task and no information about who is running the job and no information about how a job is using the system resources like network or disk I/O.

Pepperdata actually complements Cloudera Manager nicely since Pepperdata provides a level of visibility that Cloudera Manager does not, and Cloudera Manager has a number of features (such as configuring the cluster) that Pepperdata does not have nor ever will. In the future there may be tighter integration with Cloudera Manager, but today Pepperdata already provides Customer Service Descriptor (CSD) support and Parcel support as an initial level of integration with Cloudera Manager.



Current Resource Usage	
Current User CPUs	0.01
Current System CPUs	0
Resident Memory	1.1 GiB
Maps Running	3
Reduces Running	0
Aggregate Resource Usage and Counters	
User CPU Time	58s
System CPU Time	8.8s
Map Slot Time	3.7s
Reduce Slot Time	
Cumulative Disk Reads	0 B
Cumulative Disk Writes	566.2 KiB
Cumulative HDFS Reads	257 B
Cumulative HDFS Writes	3.4 GiB
Map Input Bytes	
Map Input Records	36359584
Map Output Bytes	
Map Output Records	36359581
Reduce Input Groups	
Reduce Input Records	
Reduce Output Records	
Reduce Shuffle Bytes	
Spilled Records	0
Split Raw Bytes	257

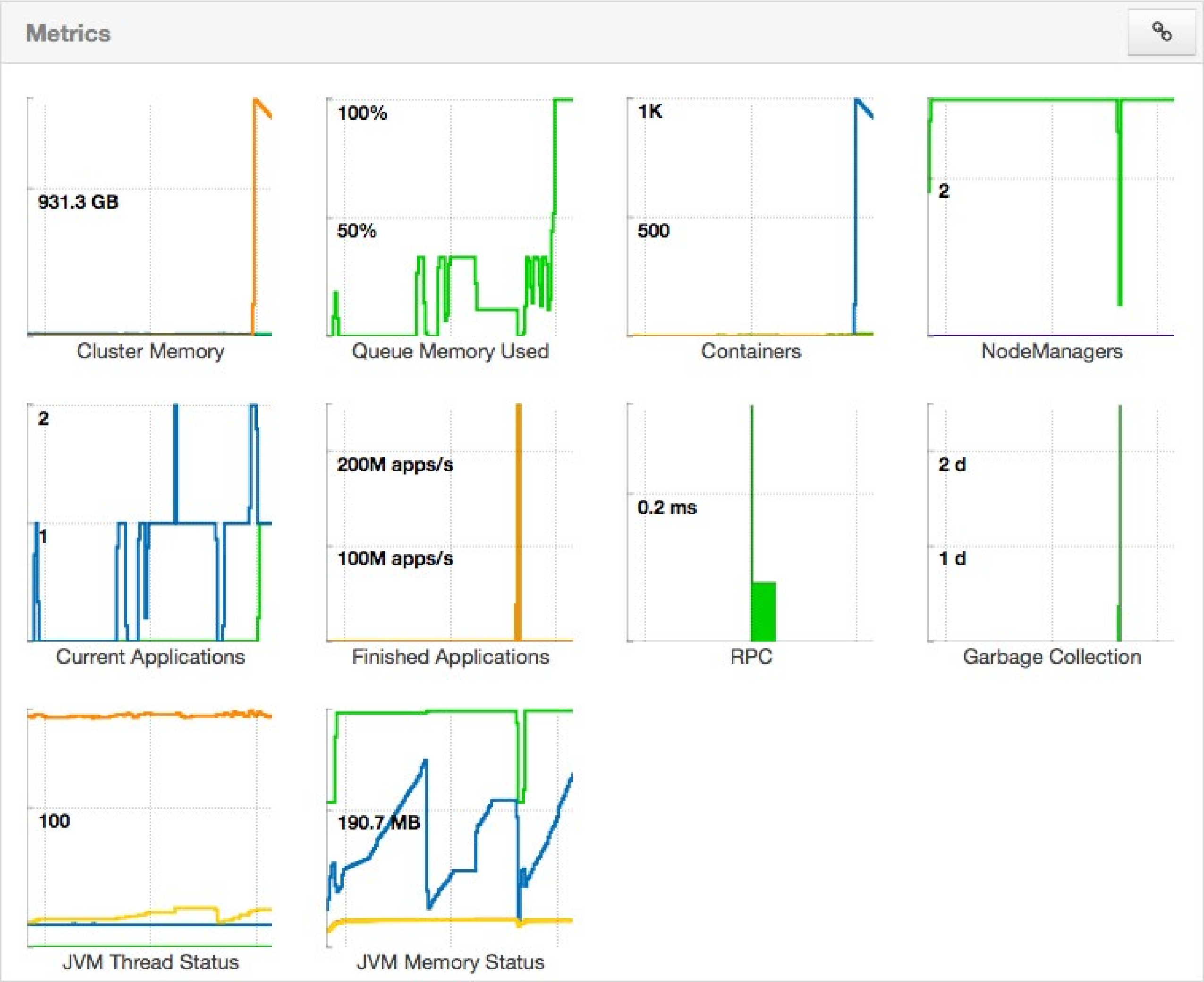
Screenshot from Cloudera Manager showing job information

Ambari

Ambari is an open source product that provides an operational framework for provisioning, managing and monitoring Hadoop clusters. Ambari includes a collection of operator tools and a set of APIs that mask the complexity of Hadoop, simplifying the operation of clusters. Ambari enables Hadoop operators to provision, integrate Hadoop with the existing enterprise architecture, and manage and monitor Hadoop.

As previously mentioned, the installation of Ambari actually includes Ganglia, so the metrics functionality of Ambari reflects that of Ganglia. Below is an image of Ambari showing summary-level metrics. This is the latest release of Ambari 1.6.1. If you click on any of the charts below, you actually are navigated directly into Ganglia, so the drill-through looks exactly the same as what we covered earlier in the document.

Since the metrics collection of Ambari as based on Ganglia, the node-level metrics are collected every 10-40 seconds depending on the metric, and sent to the collector every 90-180 seconds. Ambari is of course subject to all the shortcomings that Ganglia has (for example, https://issues.apache.org/jira/browse/AMBARI-5707).



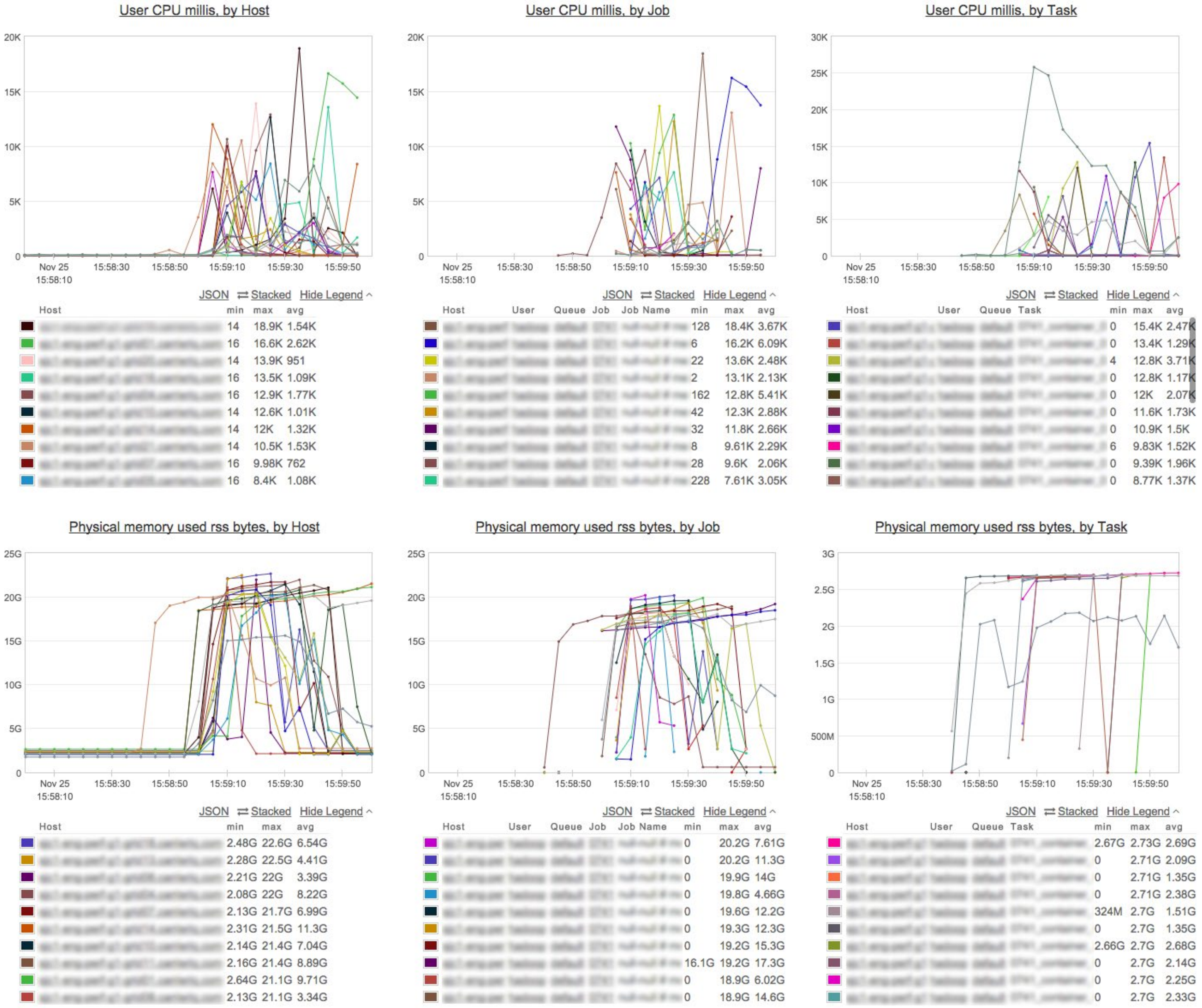
Ambari summary-level metrics

Pepperdata

All of these tools provide time-series data with much less granularity than Pepperdata, and they don't provide any time-series data for the jobs or tasks. Pepperdata provides both job- and task-level time-series information so you can see what's happening while the job is running.

The image below is an example with CPU and memory by host, by job, and even by the tasks that make up each job in real time. This allows you to see what your job is actually doing over time in the context of other jobs and tasks, allowing you to identify outliers very easily. Cloudera Manager and Ambari only tell you once the job/task has completed how much of a given resource was consumed, and then only the total for the entire run time of the job.

The legend in each chart has been exposed as well so you can see a listing of the jobs, tasks, hosts, queues, and users. Unlike the other tools discussed, Pepperdata collects 200+ metrics every 3-4 seconds about what the job is doing, while it is actually running, so you can pinpoint problems and compare its performance with other jobs and tasks running on the cluster.



User CPU millis, by Host

	Host	min	max	avg
■	[redacted]	14	18.9K	1.54K
■	[redacted]	16	16.6K	2.62K
■	[redacted]	14	13.9K	951
■	[redacted]	16	13.5K	1.09K
■	[redacted]	16	12.9K	1.77K
■	[redacted]	14	12.6K	1.01K
■	[redacted]	14	12K	1.32K
■	[redacted]	14	10.5K	1.53K
■	[redacted]	16	9.98K	762
■	[redacted]	16	8.4K	1.08K

User CPU millis, by Job

	Host	User	Queue	Job	Job Name	min	max	avg
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	128	18.4K	3.67K
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	6	16.2K	6.09K
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	22	13.6K	2.48K
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	2	13.1K	2.13K
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	162	12.8K	5.41K
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	42	12.3K	2.88K
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	32	11.8K	2.66K
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	8	9.61K	2.29K
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	28	9.6K	2.06K
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	228	7.61K	3.05K

User CPU millis, by Task

	Host	User	Queue	Task	min	max	avg
■	[redacted]	[redacted]	[redacted]	[redacted]	0	15.4K	2.47K
■	[redacted]	[redacted]	[redacted]	[redacted]	0	13.4K	1.29K
■	[redacted]	[redacted]	[redacted]	[redacted]	4	12.8K	3.71K
■	[redacted]	[redacted]	[redacted]	[redacted]	0	12.8K	1.17K
■	[redacted]	[redacted]	[redacted]	[redacted]	0	12K	2.07K
■	[redacted]	[redacted]	[redacted]	[redacted]	0	11.6K	1.73K
■	[redacted]	[redacted]	[redacted]	[redacted]	0	10.9K	1.5K
■	[redacted]	[redacted]	[redacted]	[redacted]	6	9.83K	1.52K
■	[redacted]	[redacted]	[redacted]	[redacted]	0	9.39K	1.96K
■	[redacted]	[redacted]	[redacted]	[redacted]	0	8.77K	1.37K

Physical memory used rss bytes, by Host

	Host	min	max	avg
■	[redacted]	2.48G	22.6G	6.54G
■	[redacted]	2.28G	22.5G	4.41G
■	[redacted]	2.21G	22G	3.39G
■	[redacted]	2.08G	22G	8.22G
■	[redacted]	2.13G	21.7G	6.99G
■	[redacted]	2.31G	21.5G	11.3G
■	[redacted]	2.14G	21.4G	7.04G
■	[redacted]	2.16G	21.4G	8.89G
■	[redacted]	2.64G	21.1G	9.71G
■	[redacted]	2.13G	21.1G	3.34G

Physical memory used rss bytes, by Job

	Host	User	Queue	Job	Job Name	min	max	avg
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	0	20.2G	7.61G
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	0	20.2G	11.3G
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	0	19.9G	14G
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	0	19.8G	4.66G
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	0	19.6G	12.2G
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	0	19.3G	12.3G
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	0	19.2G	15.3G
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	16.1G	19.2G	17.3G
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	0	18.9G	6.02G
■	[redacted]	[redacted]	[redacted]	[redacted]	[redacted]	0	18.9G	14.6G

Physical memory used rss bytes, by Task

	Host	User	Queue	Task	min	max	avg
■	[redacted]	[redacted]	[redacted]	[redacted]	2.67G	2.73G	2.69G
■	[redacted]	[redacted]	[redacted]	[redacted]	0	2.71G	2.09G
■	[redacted]	[redacted]	[redacted]	[redacted]	0	2.71G	1.35G
■	[redacted]	[redacted]	[redacted]	[redacted]	0	2.71G	2.38G
■	[redacted]	[redacted]	[redacted]	[redacted]	324M	2.7G	1.51G
■	[redacted]	[redacted]	[redacted]	[redacted]	0	2.7G	1.35G
■	[redacted]	[redacted]	[redacted]	[redacted]	2.66G	2.7G	2.68G
■	[redacted]	[redacted]	[redacted]	[redacted]	0	2.7G	2.14G
■	[redacted]	[redacted]	[redacted]	[redacted]	0	2.7G	2.25G
■	[redacted]	[redacted]	[redacted]	[redacted]	0	2.7G	2.33G

Screenshot showing Pepperdata providing host, job, and task-level breakdown of selected metrics