

Tuning your data flow

Date published: 2019-06-26

Date modified: 2025-10-08



Legal Notice

© Cloudera Inc. 2025. All rights reserved.

The documentation is and contains Cloudera proprietary information protected by copyright and other intellectual property rights. No license under copyright or any other intellectual property right is granted herein.

Unless otherwise noted, scripts and sample code are licensed under the Apache License, Version 2.0.

Copyright information for Cloudera software may be found within the documentation accompanying each component in a particular release.

Cloudera software includes software from various open source or other third party projects, and may be released under the Apache Software License 2.0 (“ASLv2”), the Affero General Public License version 3 (AGPLv3), or other license terms. Other software included may be released under the terms of alternative open source licenses. Please review the license and notice files accompanying the software for additional licensing information.

Please visit the Cloudera software product page for more information on Cloudera software. For more information on Cloudera support services, please visit either the Support or Sales page. Feel free to contact us directly to discuss your specific needs.

Cloudera reserves the right to change any products at any time, and without notice. Cloudera assumes no responsibility nor liability arising from the use of products, except as expressly agreed to in writing by Cloudera.

Cloudera, Cloudera Altus, HUE, Impala, Cloudera Impala, and other Cloudera marks are registered or unregistered trademarks in the United States and other countries. All other trademarks are the property of their respective owners.

Disclaimer: EXCEPT AS EXPRESSLY PROVIDED IN A WRITTEN AGREEMENT WITH CLOUDERA, CLOUDERA DOES NOT MAKE NOR GIVE ANY REPRESENTATION, WARRANTY, NOR COVENANT OF ANY KIND, WHETHER EXPRESS OR IMPLIED, IN CONNECTION WITH CLOUDERA TECHNOLOGY OR RELATED SUPPORT PROVIDED IN CONNECTION THEREWITH. CLOUDERA DOES NOT WARRANT THAT CLOUDERA PRODUCTS NOR SOFTWARE WILL OPERATE UNINTERRUPTED NOR THAT IT WILL BE FREE FROM DEFECTS NOR ERRORS, THAT IT WILL PROTECT YOUR DATA FROM LOSS, CORRUPTION NOR UNAVAILABILITY, NOR THAT IT WILL MEET ALL OF CUSTOMER’S BUSINESS REQUIREMENTS. WITHOUT LIMITING THE FOREGOING, AND TO THE MAXIMUM EXTENT PERMITTED BY APPLICABLE LAW, CLOUDERA EXPRESSLY DISCLAIMS ANY AND ALL IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO IMPLIED WARRANTIES OF MERCHANTABILITY, QUALITY, NON-INFRINGEMENT, TITLE, AND FITNESS FOR A PARTICULAR PURPOSE AND ANY REPRESENTATION, WARRANTY, OR COVENANT BASED ON COURSE OF DEALING OR USAGE IN TRADE.

Contents

- Tuning your data flow..... 4**
- Timer and event driven thread pools.....4**
 - Viewing the total number of threads in a cluster..... 5
 - Viewing the total number of active threads.....5
 - Viewing the number of cores.....6
 - Configuring thread pool size.....6
- Concurrent tasks.....7**
- Run duration..... 8**
- Recommendations..... 9**

Tuning your data flow

Several concepts and parameters affect the performance of NiFi data flows. Understanding the options and recommendations for tuning your NiFi environment, configuring parameters, and allocating resources enables you to optimize the performance for your use case and type of data flow.

Fine tuning a NiFi workflow depends on a lot of parameters, and there is no single answer for every use case and type of data flow. A data flow can process small or large FlowFiles, can have a small or large number of events to process per second and can rely on various processors with their own specific characteristics.

In addition, there is no resource isolation between the data flows running in the same NiFi environment. So when running multiple data flows in the same NiFi environment, you must perform fine tuning tasks globally to ensure there is no side effect between data flows.

**Tip:**

To avoid complex situations, you can deploy NiFi data flows on dedicated NiFi clusters.

You can process significant data without configuration changes in a default NiFi environment. You may need to change some parameters when your use cases become larger or more complex.

The first bottleneck in NiFi is usually the I/O operations on disks, and it is highly recommended that you have one or many dedicated disks for each one of the NiFi repositories. For example, processing millions of events per second on a 6-node NiFi cluster is relatively easy with the right hardware configuration and without fine tuning.

See Cloudera's *Sizing recommendations* and the *Processing one billion events per second* blog post for more information.

Related Information

[Sizing recommendations](#)

[Processing one billion events per second with Apache NiFi](#)

Timer and event driven thread pools

Learn about timer and event driven thread pools and how to configure them to optimize data flow performance.

There are two thread pools in NiFi:

- Event Driven Thread Pool
- Timer Driven Thread Pool

Event driven thread pool sizing strategy

The Event Driven strategy is an experimental feature and should not be used in production. You can change the associated value to 0.

Timer driven thread pool sizing strategy

The Timer Driven Thread Pool size is set to 10 by default. This number is the pool size per node in a NiFi cluster. If you have a 3-node cluster, you would have 30 threads available across the cluster.

**Tip:**

It is a good practice to start by setting the Timer Driven Thread Pool size number equal to three times the number of cores available on your NiFi nodes. For example, if you have nodes with eight cores, you would set this value to 24.

This thread pool represents the threads that NiFi components can use to perform the processing they are supposed to do. Threads are distributed across all the components and allow the data flows to run with shared resources in the following logic. Based on the “run scheduling” configuration of the processor, the processor’s tasks get scheduled and each task asks for a thread to perform its work. The thread request goes in a queue, and as soon as a thread is available, the processor will get access to the thread, run its work and release the thread in the pool.

The components in NiFi are designed to run simple processing very efficiently, so processors usually do not keep a thread for a long time. There are exceptions with specific processors, or when processing a huge file.

Viewing the total number of threads in a cluster

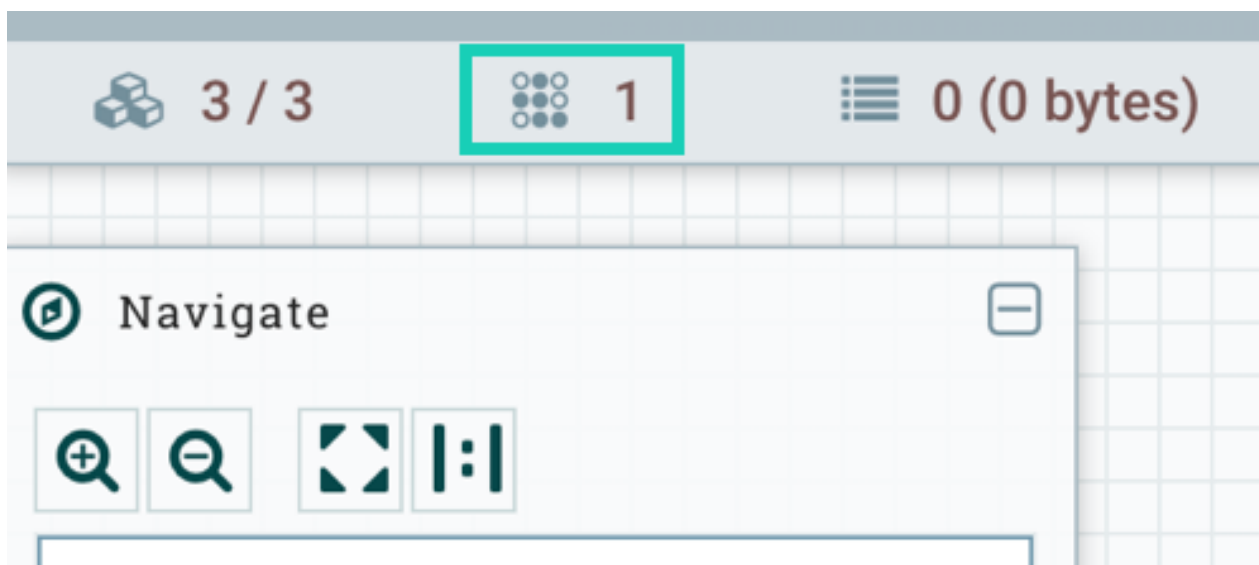
Learn how to view the total number of threads utilized by a cluster.

Procedure

The total number of threads used across the cluster is displayed by the thread icon in the Status Bar.

Example

In this example, there is one thread across a 3-node cluster.



Viewing the total number of active threads

Learn how to view the total number of active threads in a cluster.

About this task

The total number of threads includes, but is not limited to, the threads currently used by the processors to process data. However you can see this number in the Nodes tab with the Active threads count.

Procedure

1. Log into the NiFi canvas.
2. From the Global Menu in the upper-right of your screen, select Cluster and click the Nodes tab.

Results

You can see the number of active threads in the Active threads count column.



Note: The total number of threads includes, but is not limited to, the threads currently used by the processors to process data.

Example

NiFi Cluster

NODES	SYSTEM	JVM	FLOWFILE STORAGE	CONTENT STORAGE	PROVENANCE STORAGE	VERSIONS
Displaying 3 of 3						
Filter	by address ▼					
Node Address	Active Thread Count	Queue / Size	Status	Started At	Last Heartbeat	
nifi-c-compute2.field.hor...	0	0 / 0 bytes	CONNECTED, COORDINAT...	03/20/2020 19:41:02 UTC	03/25/2020 11:04:26 UTC	ⓘ
nifi-c-compute0.field.hor...	0	0 / 0 bytes	CONNECTED	03/20/2020 19:41:05 UTC	03/25/2020 11:04:27 UTC	ⓘ
nifi-c-compute1.field.hor...	0	0 / 0 bytes	CONNECTED, PRIMARY	03/20/2020 19:41:31 UTC	03/25/2020 11:04:27 UTC	ⓘ

Viewing the number of cores

Learn how to view the number of cores available to your NiFi cluster, so that you can better determine thread pool sizing.

Procedure

1. Log into the NiFi canvas.
2. From the Global Menu in the upper-right of your screen, select Cluster and click the System tab.

Results

You can see the number of cores in the Cores column.

Example

NiFi Cluster

NODES	SYSTEM	JVM	FLOWFILE STORAGE	CONTENT STORAGE	PROVENANCE STORAGE	VERSIONS
Displaying 3 of 3						
Filter	by address ▼					
Node Address	Cores	Core Load Average	Total Threads	Daemon Threads		
nifi-c-compute2.field.hortonw...	8	0.16	197	104		
nifi-c-compute0.field.hortonw...	8	0.57	155	104		
nifi-c-compute1.field.hortonw...	8	0.2	97	52		

Configuring thread pool size

Learn how to configure thread pool size to optimize your Flow Management cluster performance.

Procedure

1. Log into the NiFi canvas.
2. From the Global Menu in the upper-right of your screen, select Controller Settings.
3. In the General tab, edit the Maximum Timer Driven Thread Count and Maximum Event Driven Thread Count fields according to your sizing strategy.

Example**NiFi Settings**

GENERAL	REPORTING TASK CONTROLLER SERVICES	REPORTING TASKS	REGISTRY CLIENTS
Maximum Timer Driven Thread Count ?			
10			
Maximum Event Driven Thread Count ?			
5			
APPLY			

Concurrent tasks

Learn how and when to configure concurrent tasks.

When configuring a processor, the Scheduling tab provides a configuration option named Concurrent Tasks.

Configure Processor

 Invalid

SETTINGS

SCHEDULING

PROPERTIES

COMMENTS

Scheduling Strategy 
Timer driven 

Concurrent Tasks 
1

Execution 
All nodes 

Run Duration 
0ms 25ms 50ms 100ms 250ms 500ms 1s 2s

Lower latency Higher throughput

Run Schedule 
0 sec

CANCEL

APPLY

This controls the maximum number of threads, from the Thread Pools configured above, that the processor is allowed to use at any one time. Said a different way, this controls how many FlowFiles should be processed by this processor at the same time. Increasing this value typically allows the processor to handle more data in the same amount of time.

Concurrent Tasks increases how many FlowFiles are processed by a single processor by using system resources that then are not usable by other Processors. This provides a relative weighting of processors — it controls how much of the system's resources should be allocated to this processor instead of other processors.

This field is available for most processors. There are, however, some types of processors that can only be scheduled with a single concurrent task. It is also worth noting that increasing this value for a processor may increase thread contention and, as a result, setting this to a large value can harm productivity.



Tip:

As a best practice, you should not generally set this value to anything greater than 12.

Run duration

Learn about considerations for setting Run Duration.

For some processors, the right-hand side of the Scheduling tab contains a slider to choose a time period for Run Duration. This controls how long the processor should be scheduled to run each time when it is triggered. The left-hand side of the slider is marked Lower latency while the right-hand side is marked Higher throughput.

When a processor finishes running, it must update the repository to transfer the FlowFiles to the next connection. Updating the repository is expensive, so the more work that can be done at once before updating the repository, the more work the processor can handle (higher throughput).

Higher throughput means that the next processor cannot start processing those FlowFiles until the previous processor updates this repository. As a result, the time required to process the FlowFile from beginning to end (latency) is longer.

The slider provides a spectrum from which you can choose to favor lower latency or higher throughput. A higher Run Duration value also means that the processor keeps the thread longer, which may impact other processors.

Configure Processor

Invalid

SETTINGS SCHEDULING PROPERTIES COMMENTS

Scheduling Strategy ?
Timer driven

Concurrent Tasks ?
1

Execution ?
All nodes

Run Schedule ?
0 sec

Run Duration ?
0ms 25ms 50ms 100ms 250ms 500ms 1s 2s
Lower latency Higher throughput

CANCEL APPLY

Recommendations

Learn about the considerations involved in tuning your Flow Management cluster.

Fine tuning your Flow Management cluster depends on a range of parameters and requirements.

Thread pool size based on the number of cores

The first Flow Management cluster tuning recommendation is that you adjust the thread pool size based on the number of cores. Once you have done this, review the following recommendations for additional adjustments.

Increasing thread pool size

Do not increase the thread pool size unless you see that the active threads count is always equal to the maximum number of available threads. For example, if you have a 3-node cluster with eight cores per node, do not increase the thread pool size from 24 to a higher number, unless the active thread count displayed in the UI is often equal to 72. (3 nodes x 24 available threads per node = 72).

NiFi Cluster

NODES	SYSTEM	JVM	FLOWFILE STORAGE	CONTENT STORAGE	PROVENANCE STORAGE
-------	--------	-----	------------------	-----------------	--------------------

Displaying 3 of 3

Filter	by address ▼			
Node Address	Active Thread Count	Queue / Size	Status	
nifi-c-compute2.field.hor...	2	762 / 0 bytes	CONNECTED, COORDINAT...	
nifi-c-compute0.field.hor...	0	0 / 0 bytes	CONNECTED	
nifi-c-compute1.field.hor...	2	656 / 0 bytes	CONNECTED, PRIMARY	

You can use the NiFi Summary UI to identify the number of threads used per processor.

NiFi Summary

PROCESSORS	INPUT PORTS	OUTPUT PORTS	REMOTE PROCESS GROUPS	CONNECTIONS	PROCESS GROUPS
------------	-------------	--------------	-----------------------	-------------	----------------

Displaying 14 of 14

Filter	by name ▼				View: Single node Cluster				
Name	Type	Process Group...	Run Status ▼	In (Size) 5 min	Read Write 5 ...	Out (Size) 5 min	Tasks Time 5 ...		
Generate Tes...	ExecuteProce...	Sensor Simula...	Stopped	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	→	
UpdateAttrib...	UpdateAttrib...	NiFi Flow	Running (3)	5,102,914 (0 by...	0 bytes 0 bytes	0 (0 bytes)	5,102,914 00:1...	→	
GenerateFlo...	GenerateFlow...	NiFi Flow	Running (1)	0 (0 bytes)	0 bytes 0 bytes	5,107,903 (0 byt...	5,107,903 00:1...	→	
Kafka-Truck-...	PublishKafka...	Publish Enrich...	Running	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	→	
Kafka-Truck-...	PublishKafka...	Publish Enrich...	Running	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	→	
Kafka-Truck-...	PublishKafka...	Publish Enrich...	Running	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	→	
Kafka-Truck-...	PublishKafka...	Publish Enrich...	Running	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	→	
ReverseGeol...	LookupRecord	Route, Transfo...	Running	0 (0 bytes)	0 bytes 0 bytes	0 (0 bytes)	0 00:00:00.000	→	

Core load average of NiFi nodes

If the active threads count is equal to the maximum number of available threads, review the core load average on your NiFi nodes.

NiFi Cluster

NODES	SYSTEM	JVM	FLOWFILE STORAGE	CONTENT STORAGE	PROVENANCE STORAGE	VERSIONS
-------	--------	-----	------------------	-----------------	--------------------	----------

Displaying 3 of 3

Filter	by address ▼			
Node Address	Cores	Core Load Average	Total Threads	Daemon Threads
nifi-c-compute2.field.hortonw...	8	0.16	197	104
nifi-c-compute0.field.hortonw...	8	0.57	155	104
nifi-c-compute1.field.hortonw...	8	0.2	97	52

If the core load average is below 80% of the number of cores (below 6.4 in the example), and if the active threads count is at its maximum, you can slightly increase the thread pool size. Start by increasing the thread pool size by $n+1$ times the number of cores, where N is the current value. You usually want to keep the load average around 80% the

number of cores to account for the loss of one node. You also want to have some resources available to process the additional amount of work on the remaining nodes.

Number of concurrent tasks

If:

- you have backpressure somewhere in your workflow
- your load average is low
- and the active threads count is not at the maximum

you can consider increasing the number of concurrent tasks where the processors are not processing enough data and are causing backpressure.

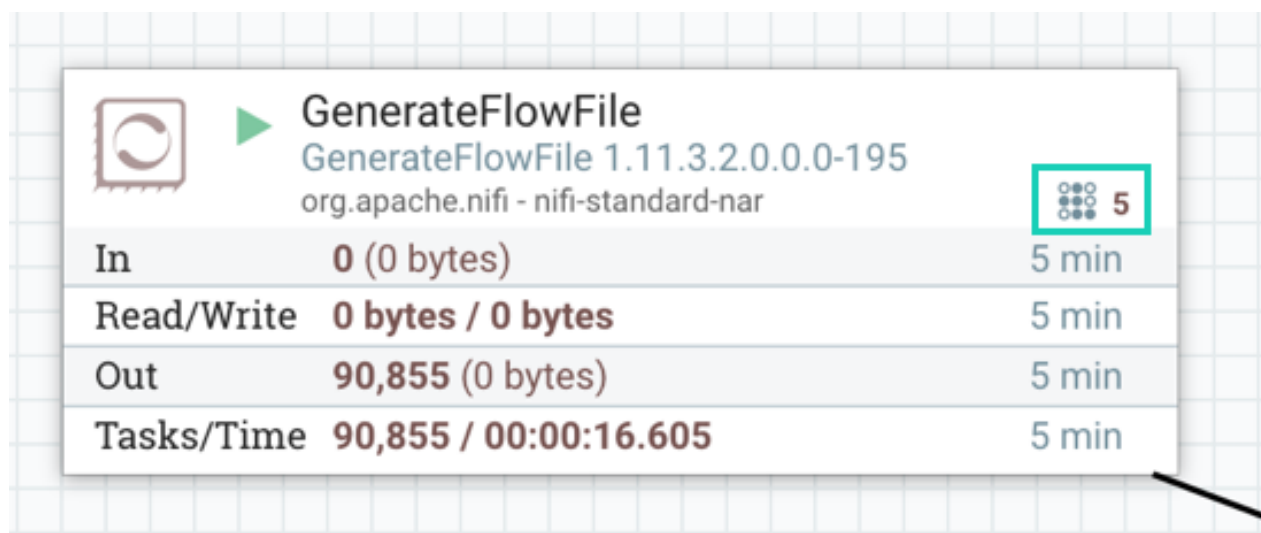
You should increase concurrent tasks iteratively. Begin by increasing the number of concurrent tasks by 1.

Check

- How things are evolving globally (the thread pool is shared across all the workflows running in the same NiFi environment)
- Load average
- Active thread count

Based on these considerations, decide if you need to increase this number again.

A processor displays active threads across the cluster:



If a processor has active threads and is not processing data as fast as expected while the load average on the server is low, the issue can be related to I/O operations on disks. In this case it is a good idea to check the I/O statistics on the disks used for the NiFi repositories.