
BEAM
SUMMIT

Troubleshooting Slow Running Beam Pipelines

By Mehak Gupta
Google Cloud, Canada

BEAM SUMMIT
NYC 2023

About Me

Hello!

I’m Mehak

Technical Solutions Specialist at Google Cloud

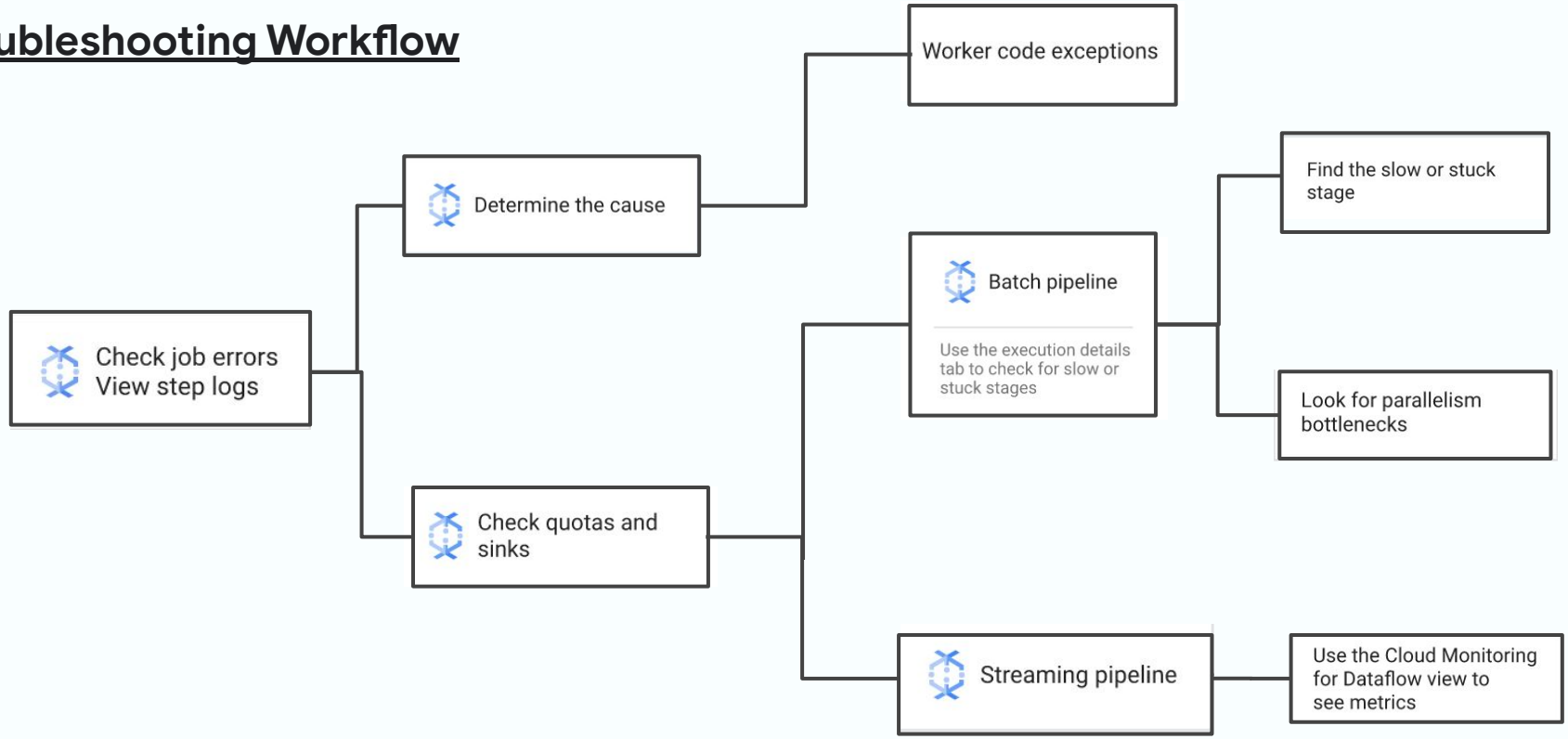
Goals

- Apache Beam pipeline troubleshooting techniques that would empower professionals to research and resolve Beam issues by themselves.
- Self service skills would reduce MTTR (Mean Time To Recover) significantly
- Share some tricks and samples of troubleshooting slow running beam pipelines using Dataflow as an example

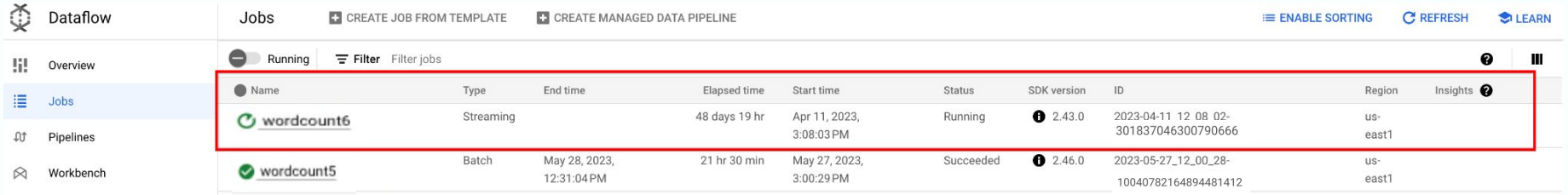
How to identify if the beam pipeline is slow/stuck

- Pipeline is running from a long time without reporting results
- Increased data watermark or system latency
- Pipeline is not consuming input

Troubleshooting Workflow



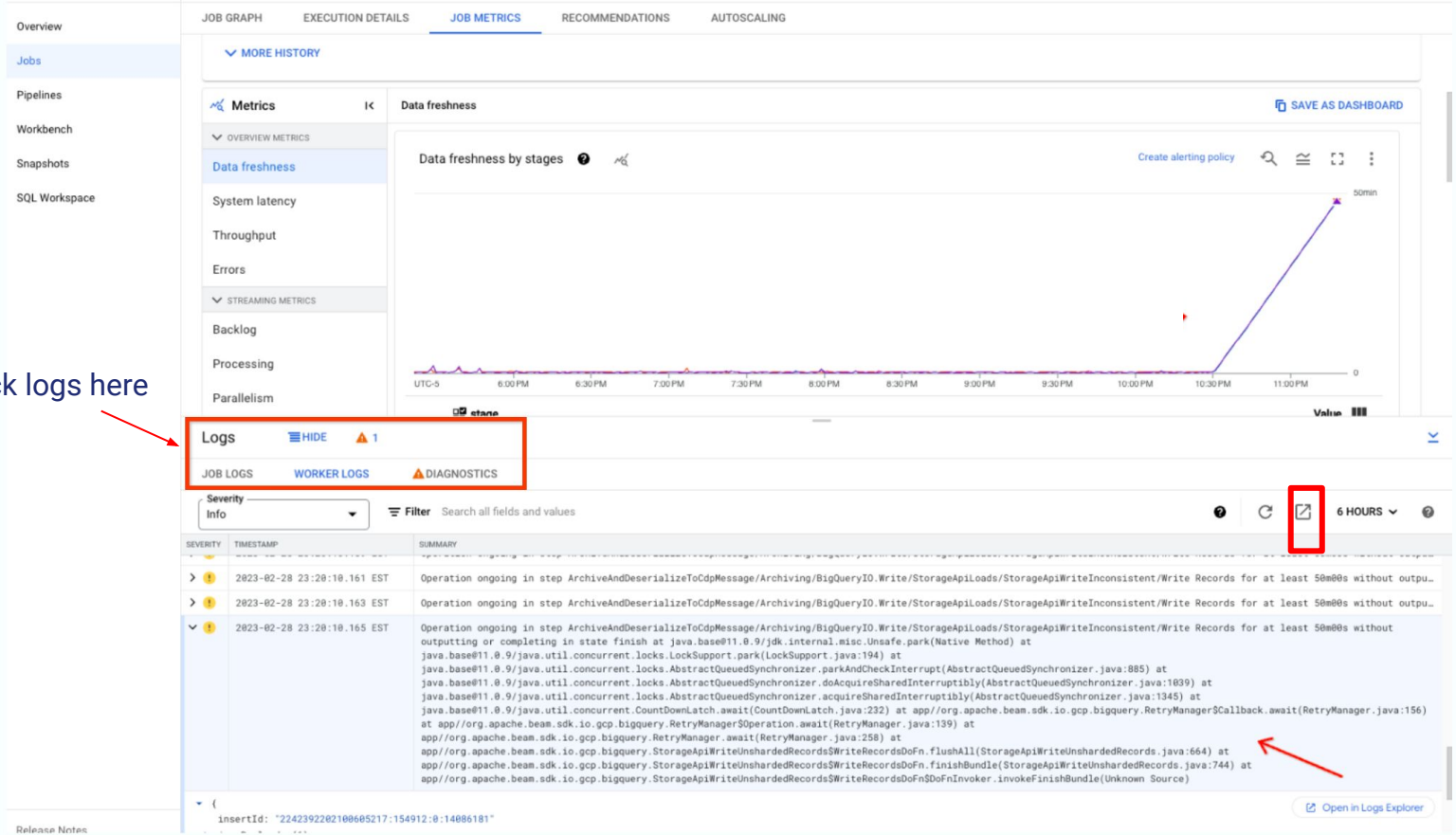
Troubleshoot slow/stuck dataflow jobs



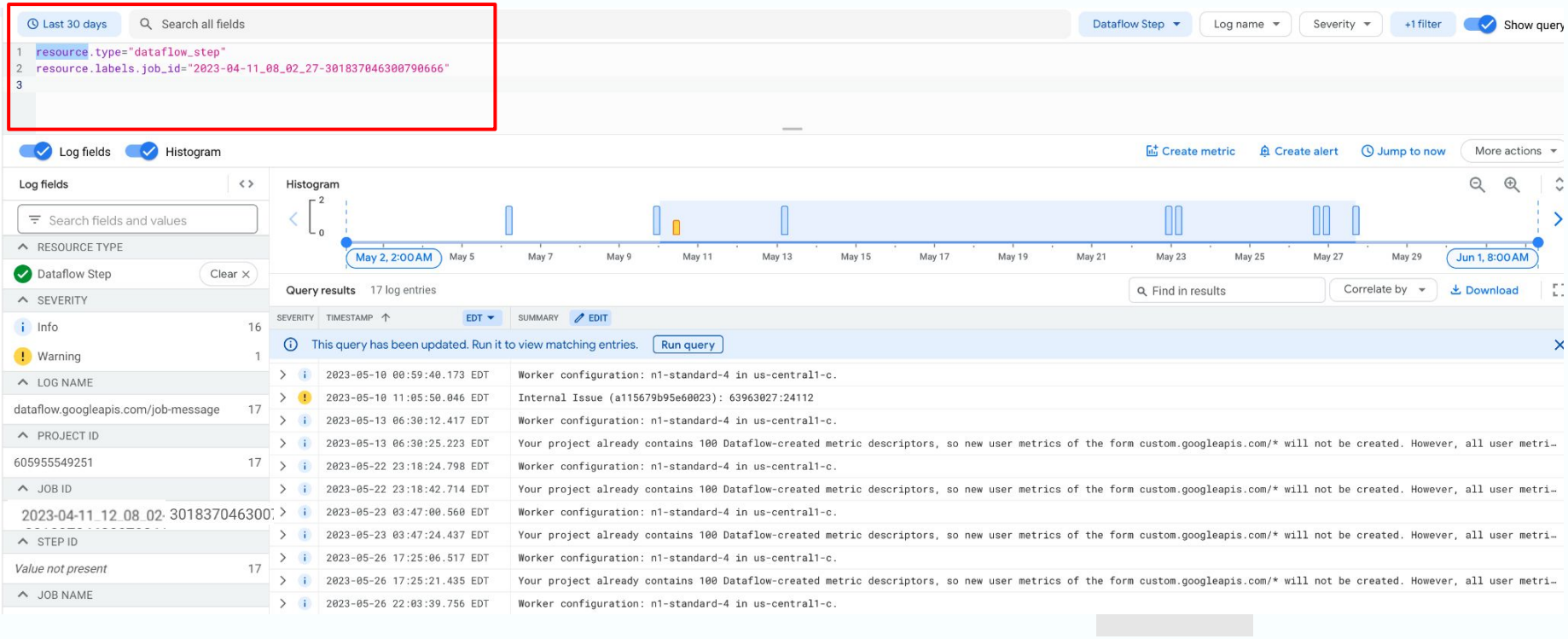
Troubleshooting using Logs Explorer View

Troubleshoot slow/stuck dataflow jobs

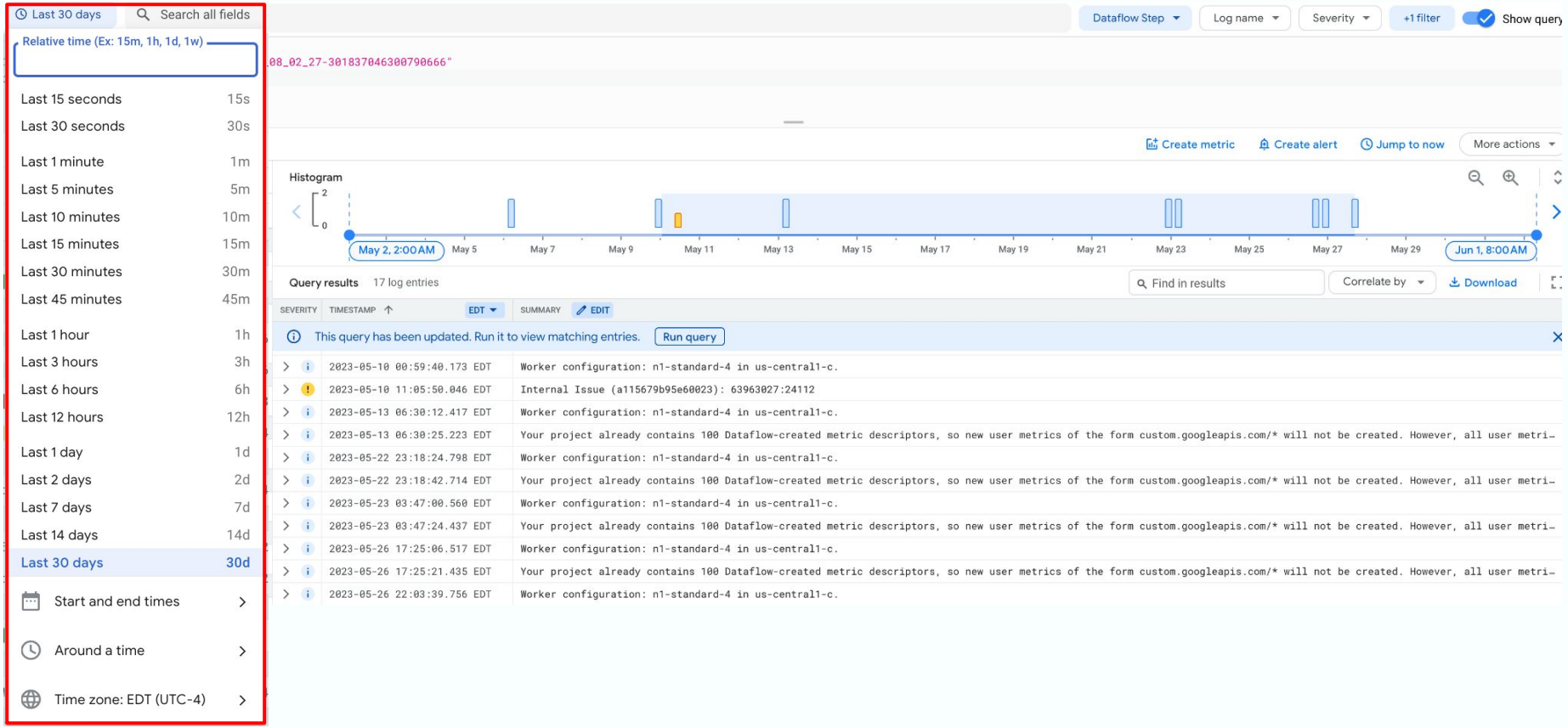
Check logs here



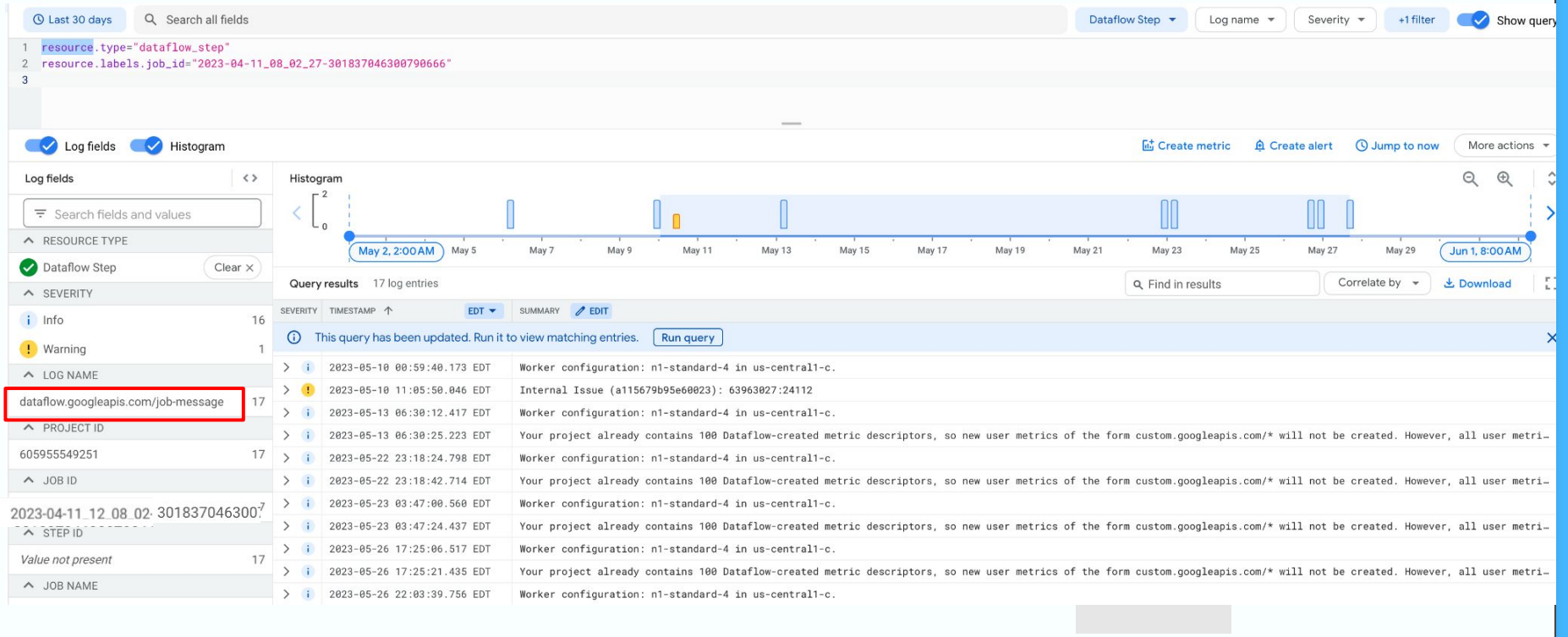
Troubleshoot slow/stuck dataflow jobs



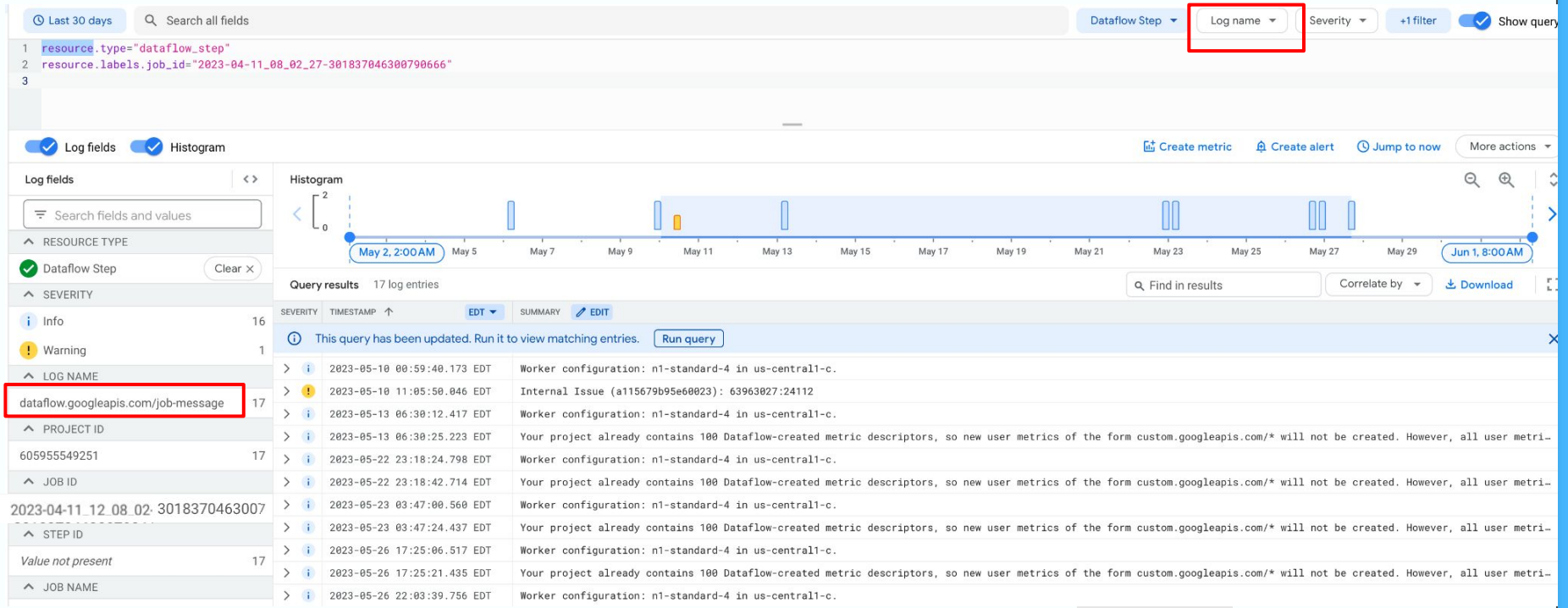
Troubleshoot slow/stuck dataflow jobs



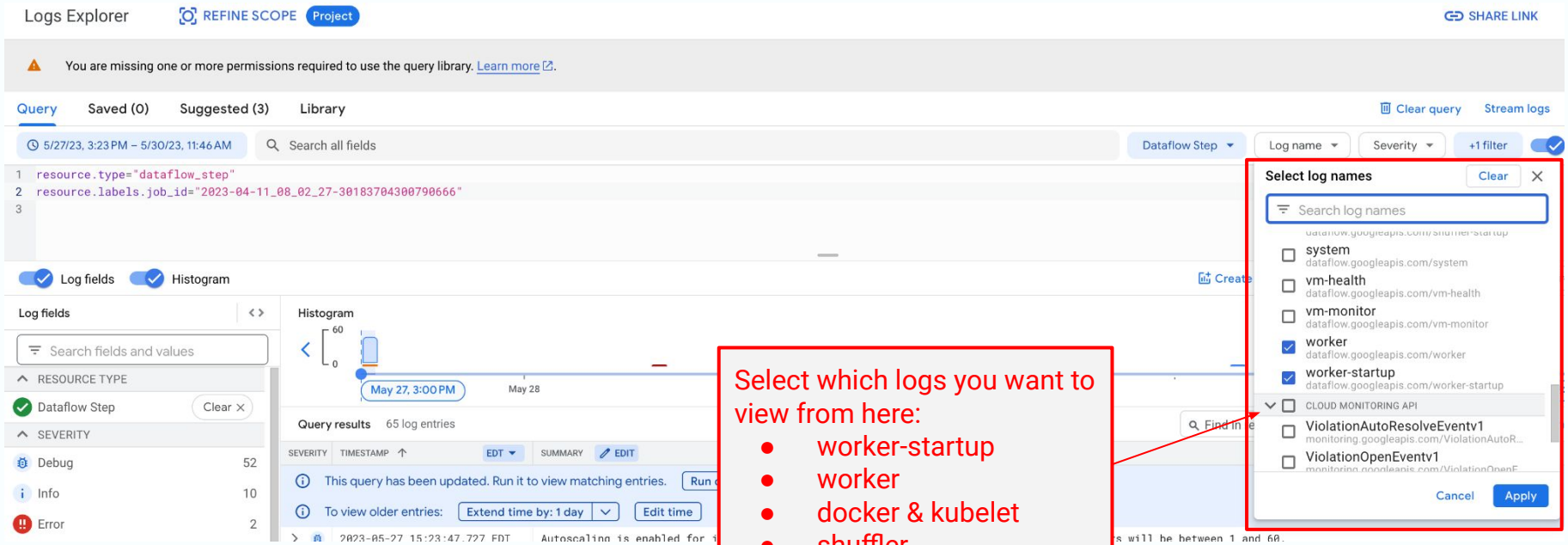
Troubleshoot slow/stuck dataflow jobs



Troubleshoot slow/stuck dataflow jobs



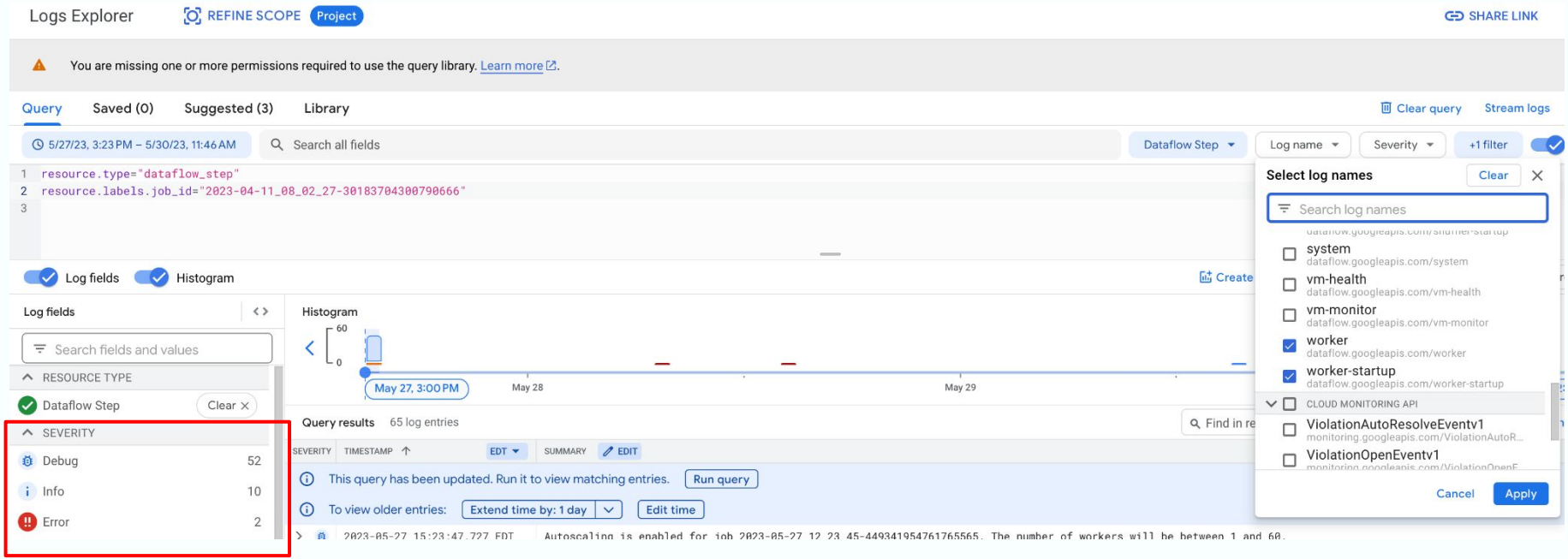
Troubleshoot slow/stuck dataflow jobs



Select which logs you want to view from here:

- worker-startup
- worker
- docker & kubelet
- shuffler

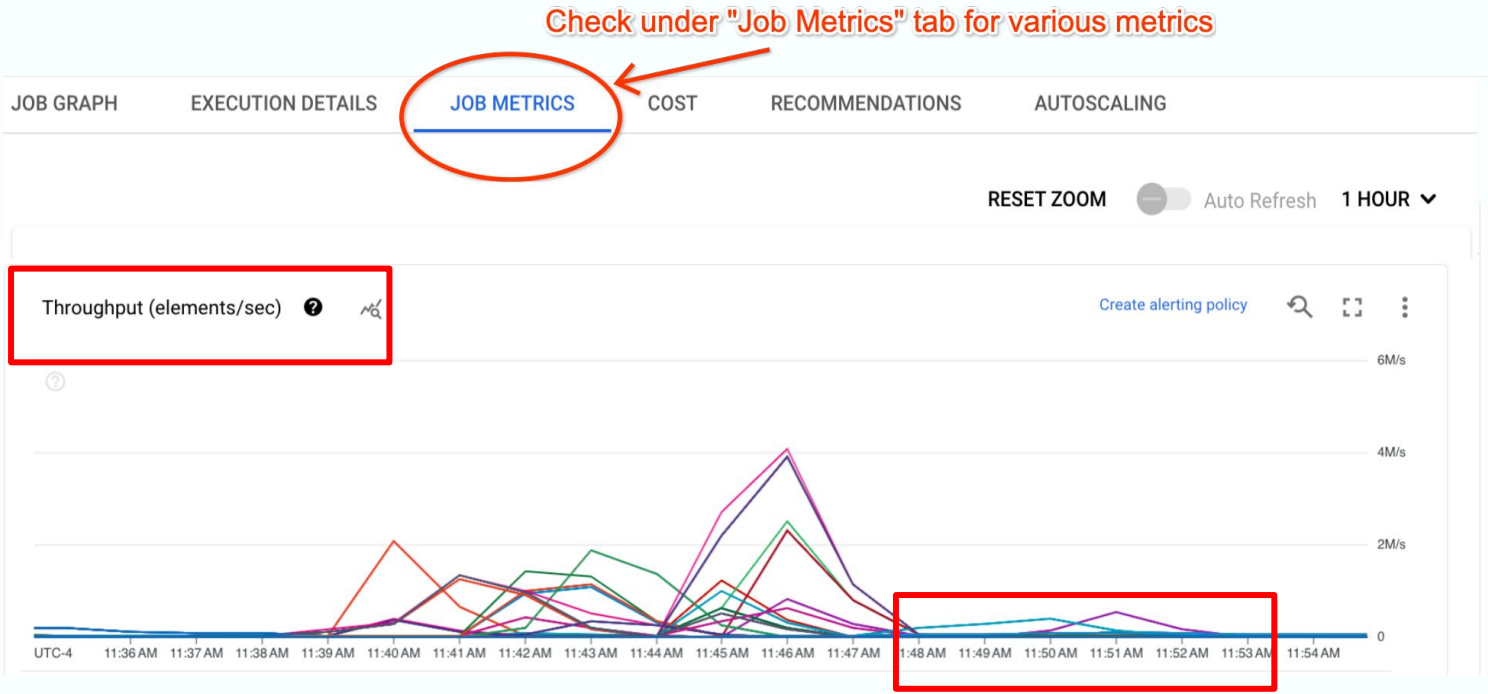
Troubleshoot slow/stuck dataflow jobs



Troubleshooting using Job Metrics Tab

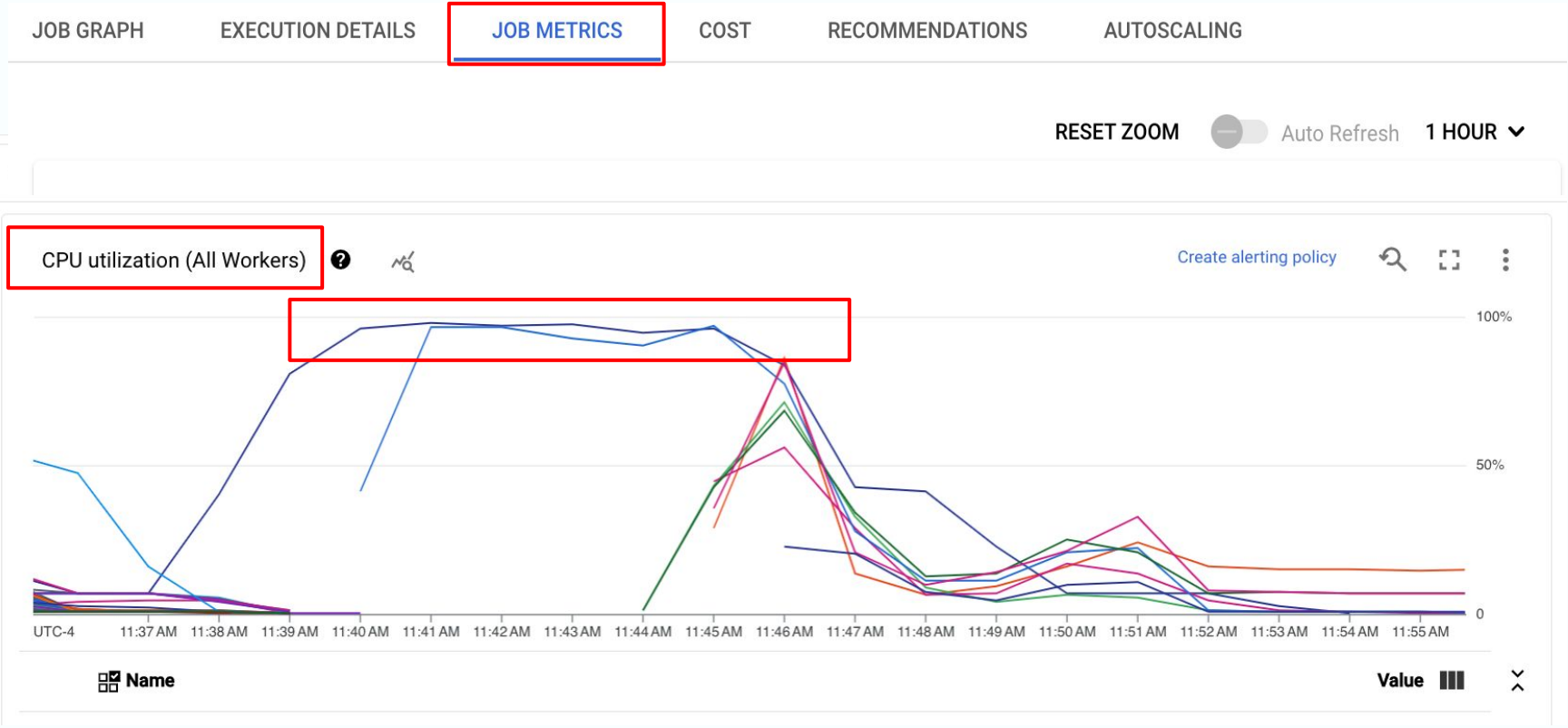
Troubleshoot slow/stuck dataflow jobs

Throughput dropping to zero



Troubleshoot slow/stuck dataflow jobs

High CPU Utilization

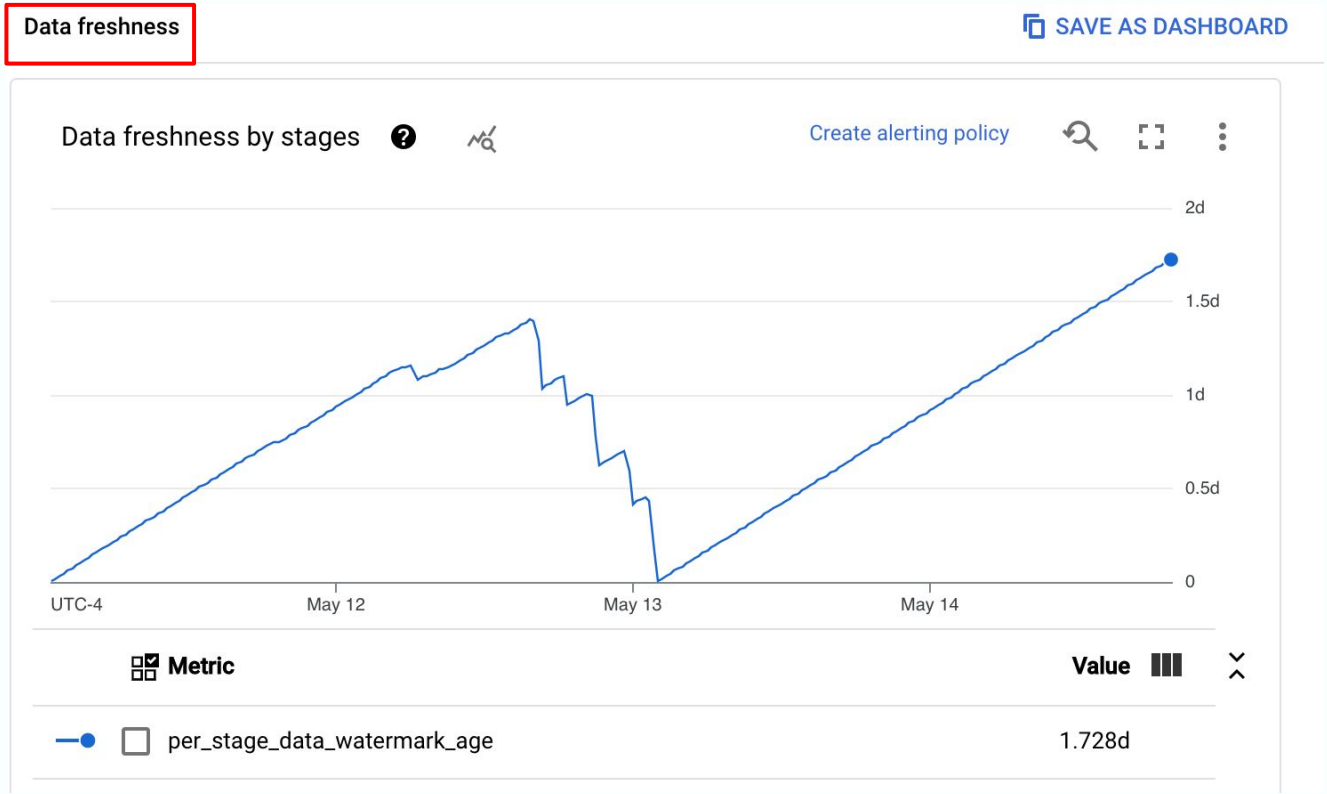


Troubleshoot slow/stuck dataflow jobs

High CPU Utilization

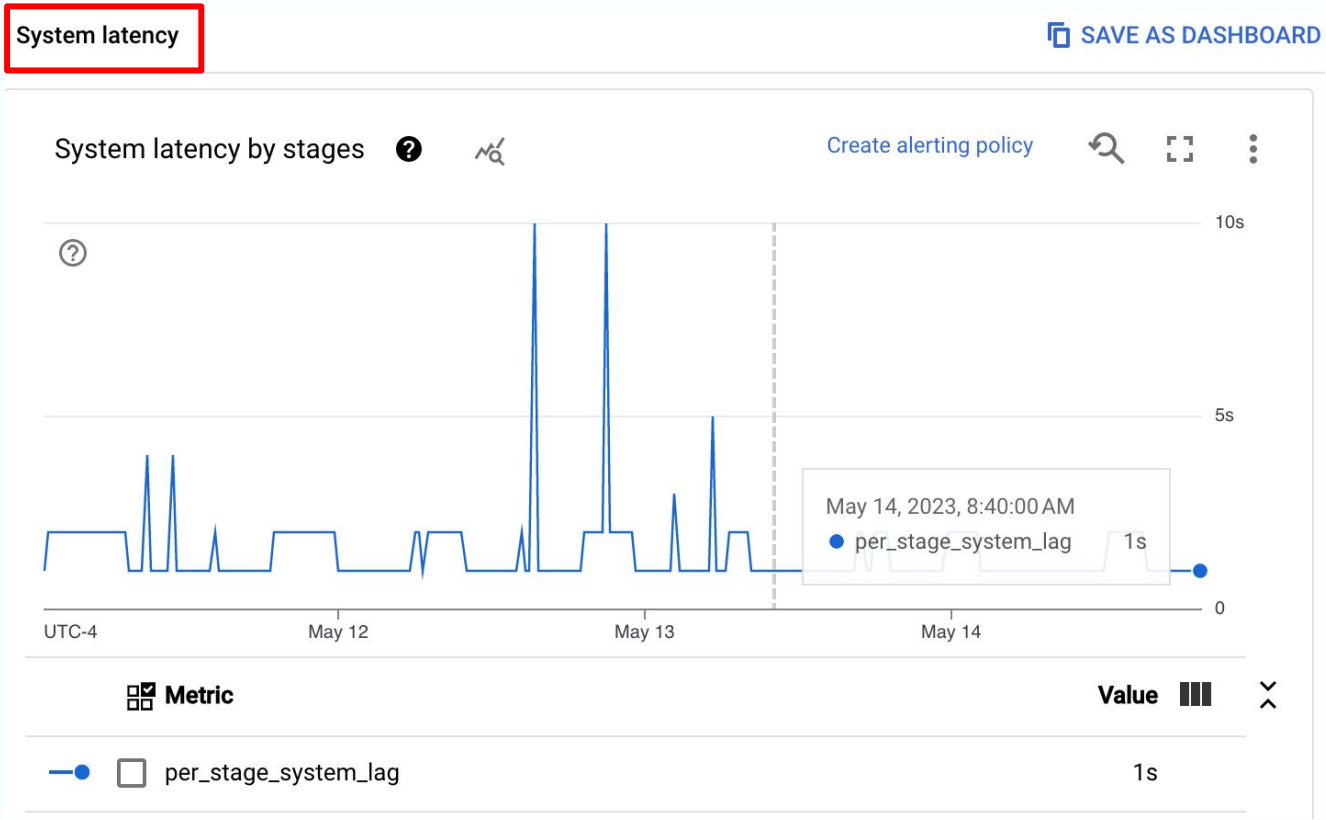
Troubleshoot slow/stuck dataflow jobs

Data Freshness



Troubleshoot slow/stuck dataflow jobs

System Latency

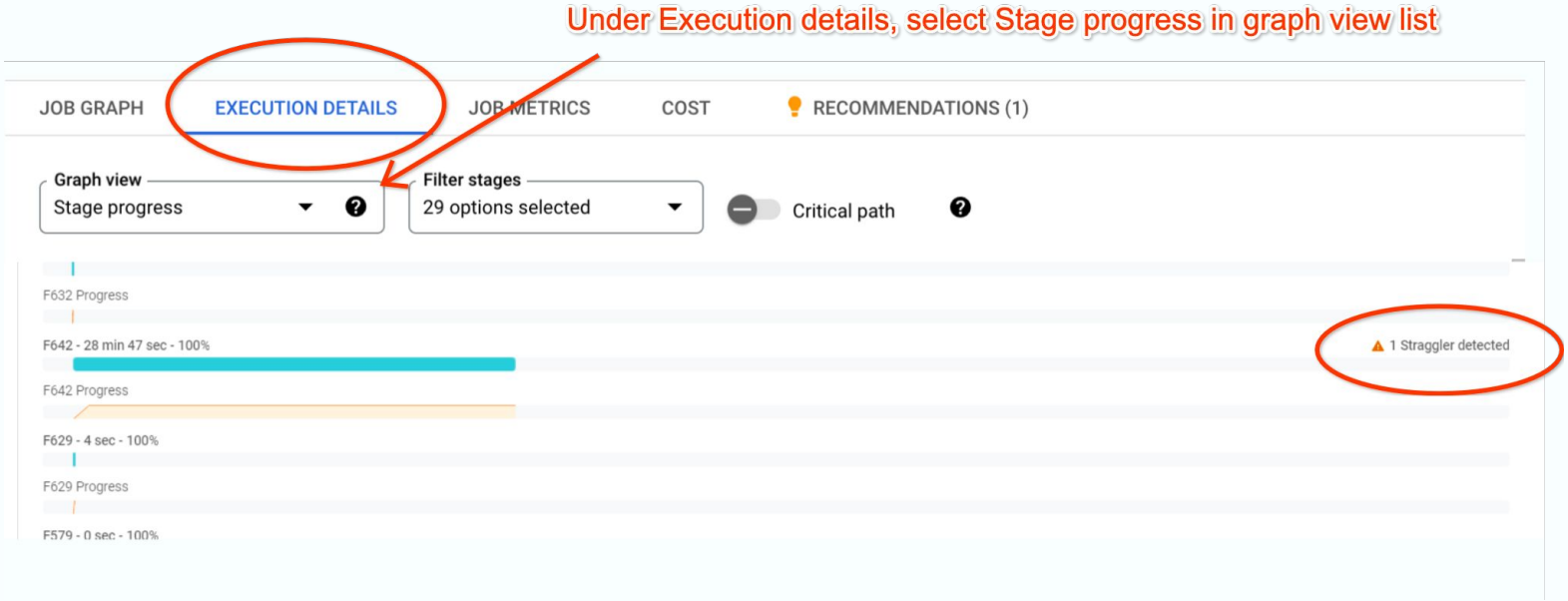


Stragglers in batch job

Troubleshoot slow/stuck dataflow jobs

When a batch job takes a long time to process data, it would be best to check on the [Straggler Workers]

How to check it?



Troubleshoot slow/stuck dataflow jobs

There can be various causes of stragglers:

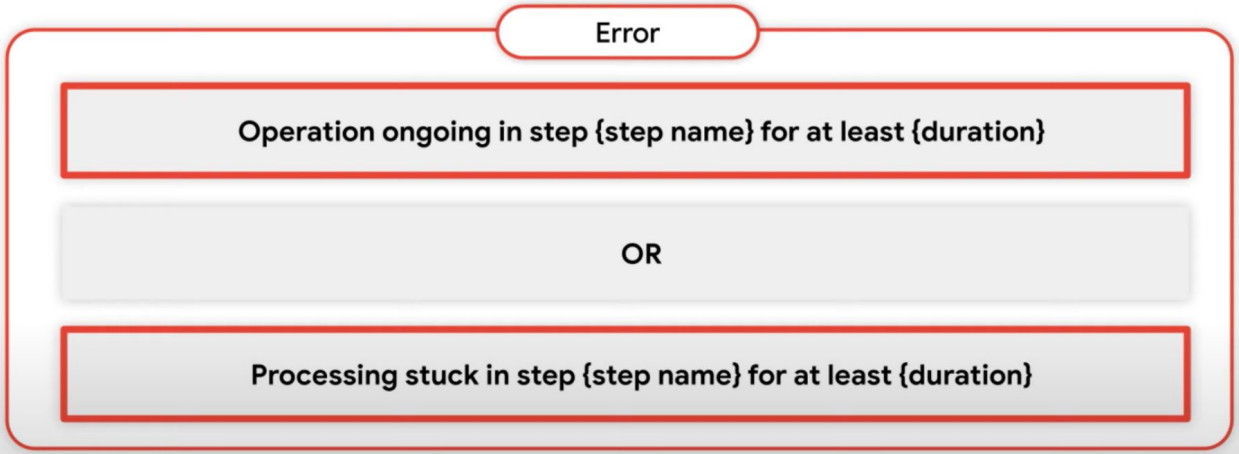
- **Hot Keys:** Hot keys can create stragglers because they limit ability of Dataflow to process elements in parallel.
  a. Re-key your data. Apply a ParDo transform to output new key-value pairs.

- **Re-shuffle your data** to avoid a single worker having extra load

Scenario 1: Long active user operation

Troubleshoot slow/stuck dataflow jobs

Processing Stuck/ Operation ongoing



Troubleshoot slow/stuck dataflow jobs

Processing Stuck/ Operation ongoing

From Logs Explorer

Query:

Results:

```
Operation ongoing in step Write to BQ/BatchLoads/SinglePartitionWriteTables/ParMultiDo(WriteTables) for at least 02h20m00s without outputting or completing in state finish
  at java.base@11.0.9/java.lang.Thread.sleep(Native Method)
  at app//com.google.api.client.util.Sleeper$1.sleep(Sleeper.java:42)
  at app//com.google.api.client.util.BackOffUtils.next(BackOffUtils.java:48)
  at app//org.apache.beam.sdk.io.gcp.bigquery.BigQueryHelpers$PendingJobManager.nextBackOff(BigQueryHelpers.java:162)
  at app//org.apache.beam.sdk.io.gcp.bigquery.BigQueryHelpers$PendingJobManager.waitForDone(BigQueryHelpers.java:148)
  at app//org.apache.beam.sdk.io.gcp.bigquery.WriteTables$WriteTablesDoFn.finishBundle(WriteTables.java:380)
  at app//org.apache.beam.sdk.io.gcp.bigquery.WriteTables$WriteTablesDoFn$DoFnInvoker.invokeFinishBundle(Unknown Source)
```

Troubleshoot slow/stuck dataflow jobs

Processing Stuck/ Operation ongoing

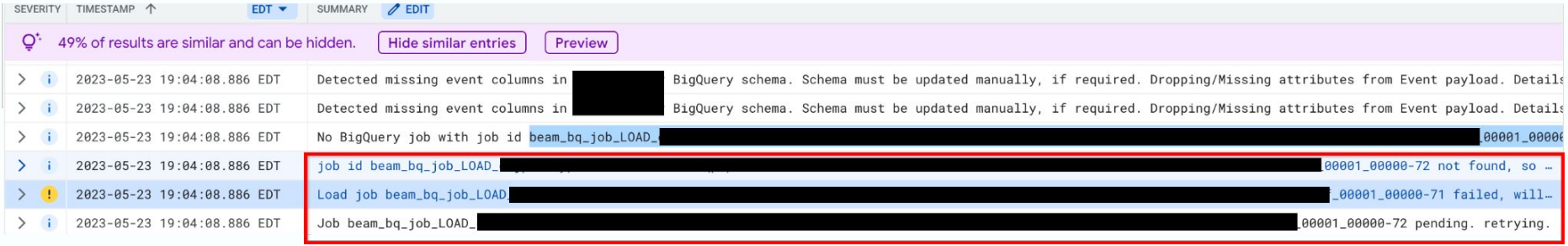
From Logs Explorer

```
Operation ongoing in step Write to BQ/BatchLoads/SinglePartitionWriteTables/ParMultiDo(WriteTables) for at least 02h20m00s without outputting or completing
in state finish
  at java.base@11.0.9/java.lang.Thread.sleep(Native Method)
  at app//com.google.api.client.util.Sleeper$1.sleep(Sleeper.java:42)
  at app//com.google.api.client.util.BackOffUtils.next(BackOffUtils.java:48)
  at app//org.apache.beam.sdk.io.gcp.bigquery.BigQueryHelpers$PendingJobManager.nextBackOff(BigQueryHelpers.java:162)
  at app//org.apache.beam.sdk.io.gcp.bigquery.BigQueryHelpers$PendingJobManager.waitForDone(BigQueryHelpers.java:148)
  at app//org.apache.beam.sdk.io.gcp.bigquery.WriteTables$WriteTablesDoFn.finishBundle(WriteTables.java:380)
  at app//org.apache.beam.sdk.io.gcp.bigquery.WriteTables$WriteTablesDoFn$DoFnInvoker.invokeFinishBundle(Unknown Source)
```

https://github.com/apache/beam/blob/master/sdks/java/io/google-cloud-platform/src/main/java/org/apache/beam/sdk/io/gcp/bigquery/BigQueryHelpers.java

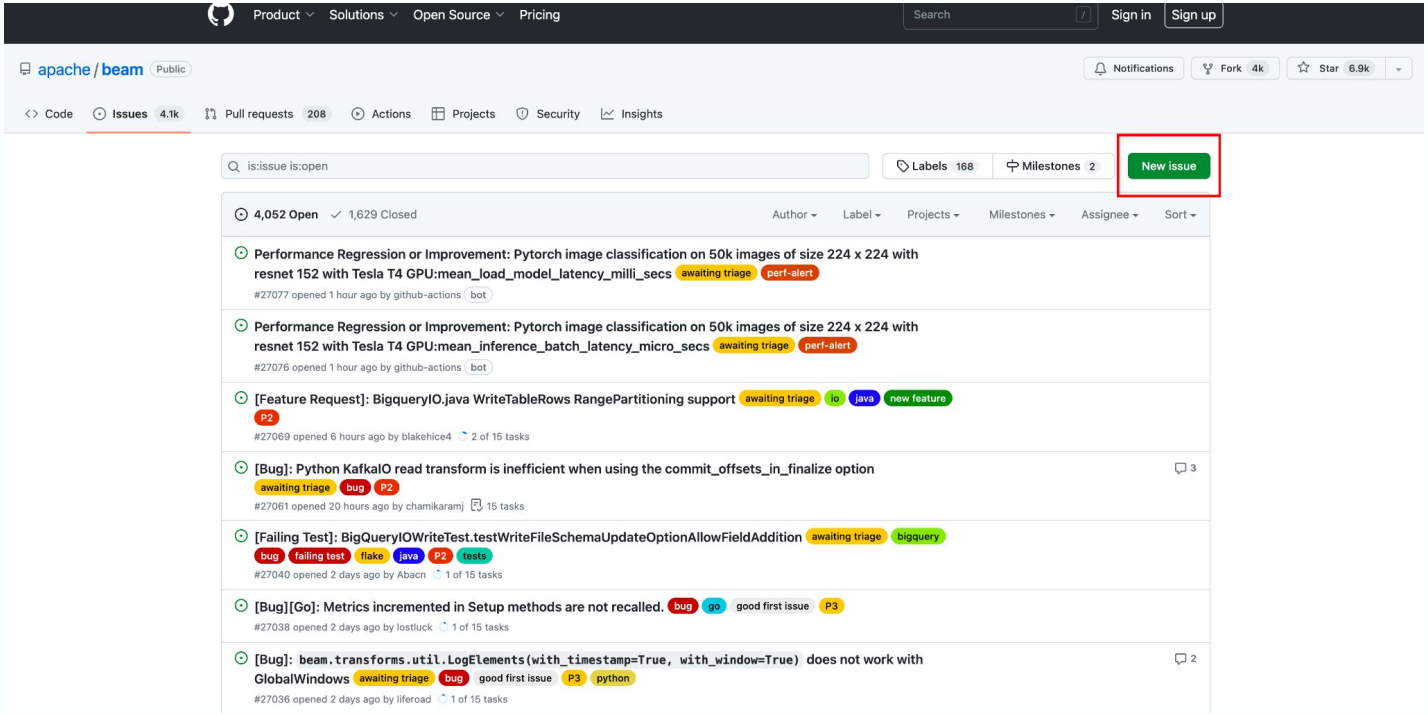
Troubleshoot slow/stuck dataflow jobs

Processing Stuck/ Operation ongoing



Troubleshoot slow/stuck dataflow jobs

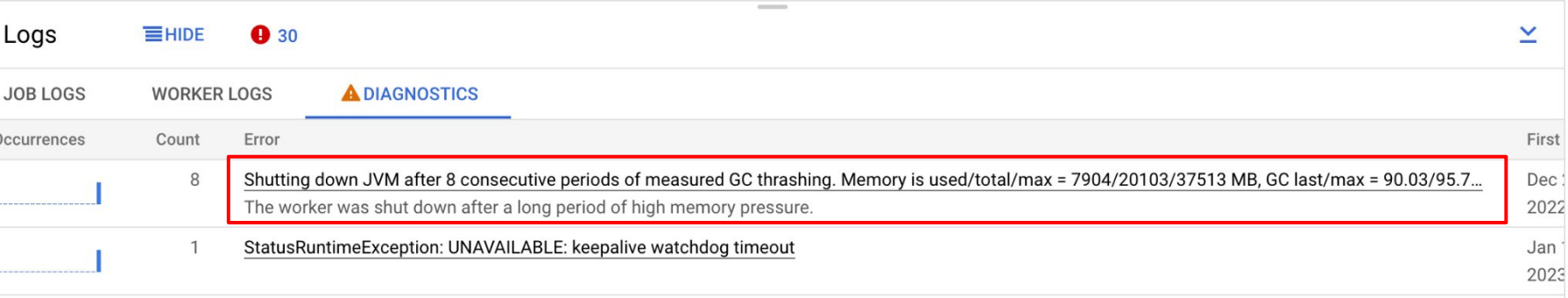
Apache Beam Issues/Feature Request



Scenario 2: GC Thrashing/OOM

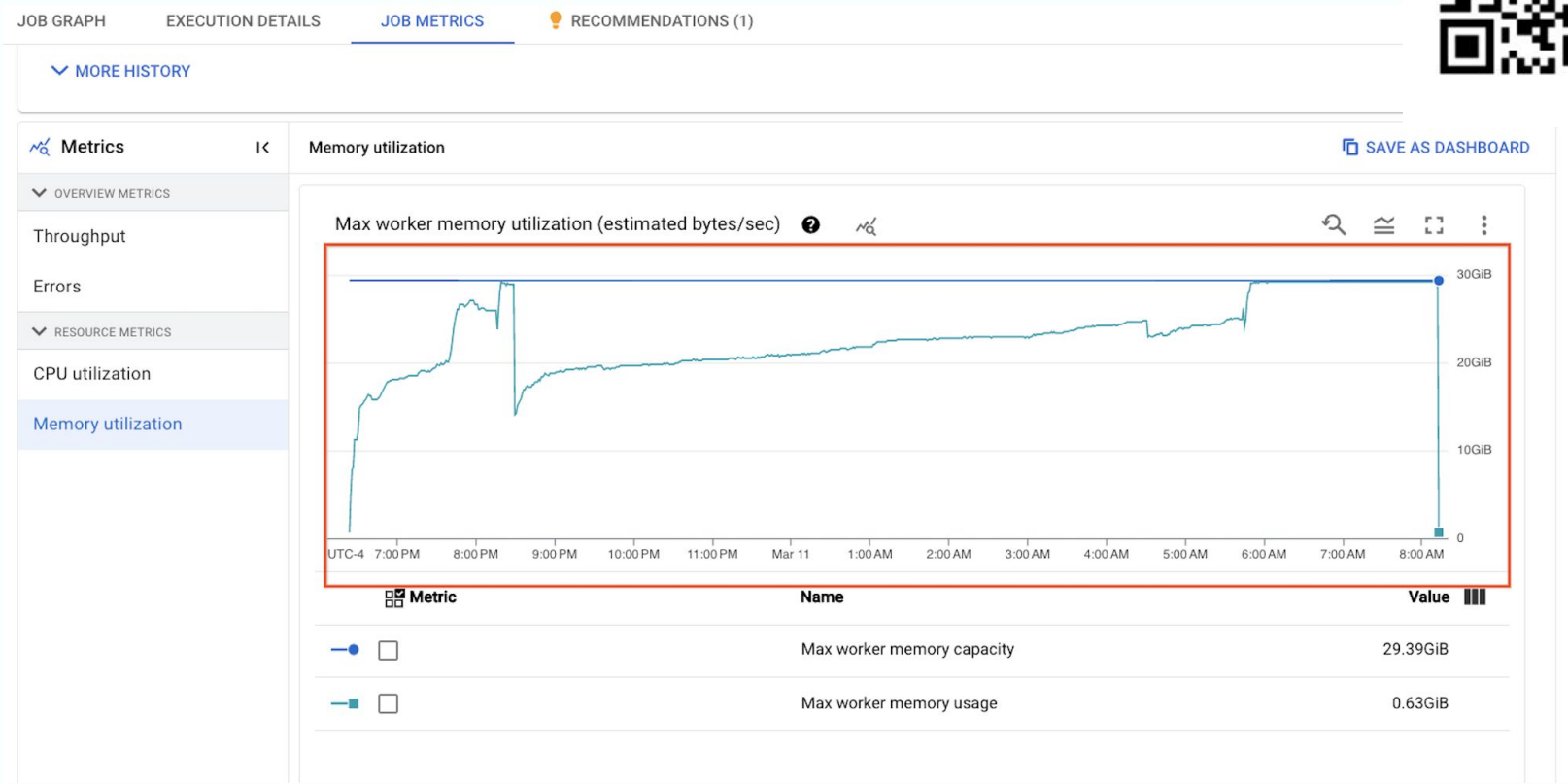
Troubleshoot slow/stuck dataflow jobs

GC Thrashing/OOM: Diagnostics Tab



Troubleshoot slow/stuck dataflow jobs

GC Thrashing/OOM



General Recommendations

- Use machine types with higher memory
  - **Link:** goo.gle/45USWe3
- Decrease the parallelism of processing by reducing the number of worker harness threads
  - **Link**: goo.gle/45RM6WT
- Do vertical autoscaling (Enable Dataflow Prime)
  - **Link:** goo.gle/3r3KZjv

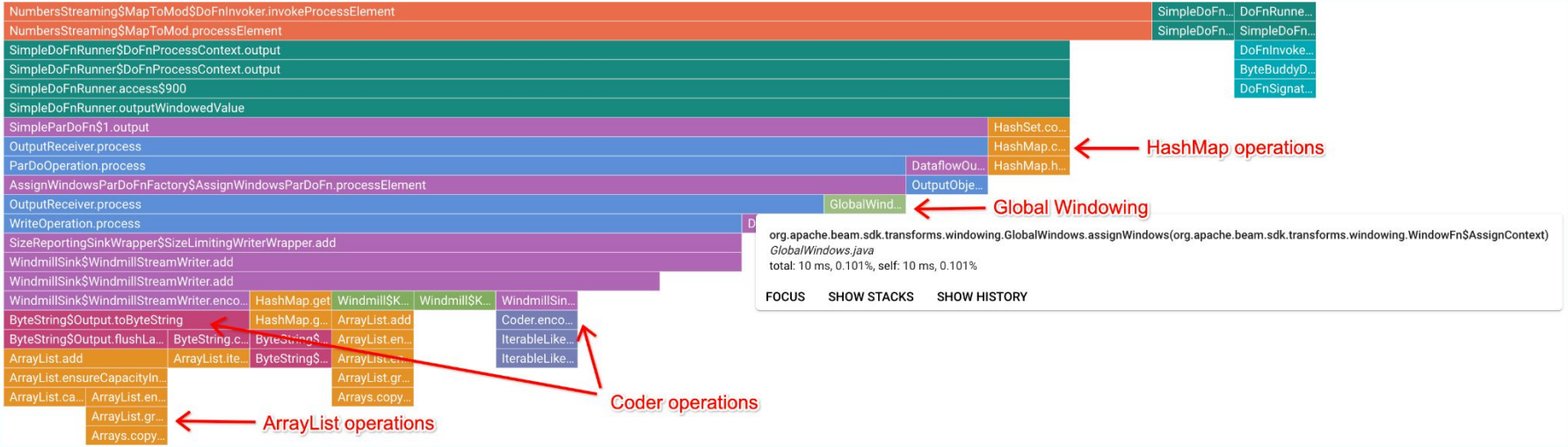
Performance Optimization using Dataflow profiling

Cloud Profiler is available for Dataflow pipelines written in Apache Beam SDK for Java and Python, version 2.33.0 or later.

It can be enabled at pipeline start time.

E.g. For Java SDK, to enable CPU profiling, start the pipeline with the following option

```
--dataflowServiceOptions=enable_google_cloud_profiler
```



QUESTIONS?

mhkgupta@google.com

linkedin.com/in/mhkgupta