

Dataflow Streaming

The Evolution of Real-Time Data Processing

Apache Beam Summit

Thursday, 5 Sep 2024 12 PM

Tom Stepp, Google

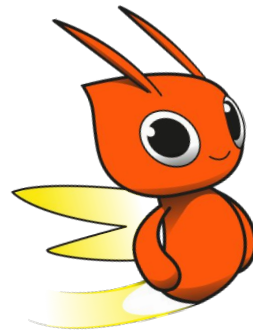


Table of Contents

Streaming Modes 01

Autoscaling Hints 02

Inflight Updates 03

Active Load Balancing 04

01

Streaming Modes

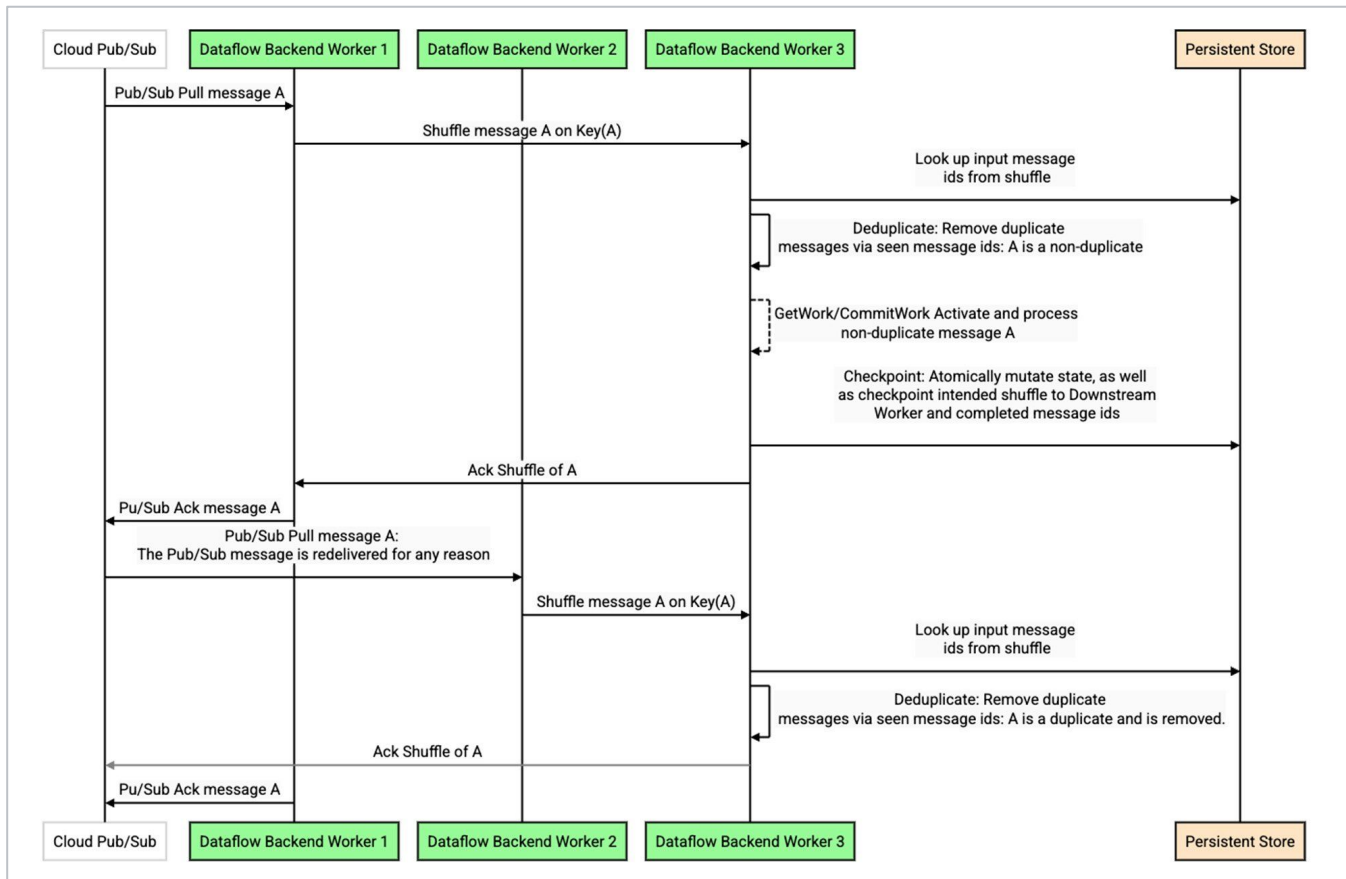
Intro

- Historically, Dataflow Streaming has offered exactly-once processing.
- We provide at-least-once mode as an alternative for lower latency and cost.
- How does each mode work and how do you choose?

Exactly Once

- Some applications require exactly once processing.
- Deduplication of events increases overall cost and latency of the system.
- Dataflow ensures that the message will be processed and not lost (at least once).
- State updates and outputs to a subsequent stage, are also reflected at most once.
- This guarantee enables performing exact aggregations, such as exact sums or counts.

Exactly Once: Deduplication



Streaming Modes

Exactly-once

- Ensures records are not dropped or duplicated as the data moves through the pipeline.

At-least-once

- Guarantees that records are processed at least once, with possible duplicate records.
- Significantly lowers the cost and latency of your job.

Which mode?

Exactly once

- Pipelines with aggregations, such as count, sum, or mean.
- Cases that rely on records being processed once and only once.

At-least-once

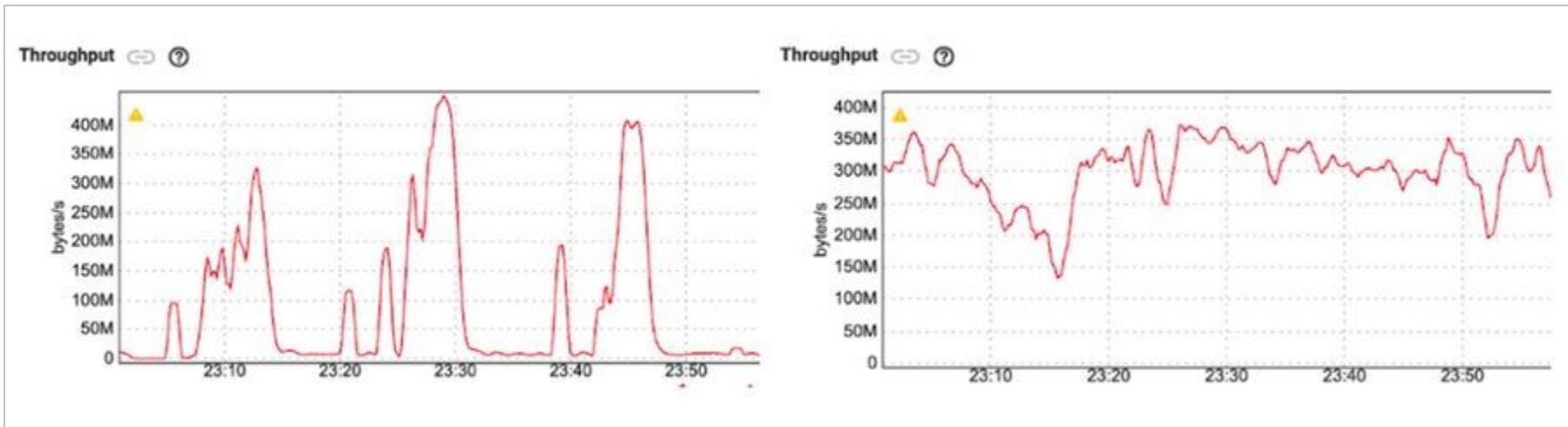
- Deduplication is performed downstream from Dataflow.
- Map-only without aggregations.
- Output sink can't guarantee exactly-once delivery.
- Input source from Pub/Sub which is significantly optimized when using at-least-once mode.

Additional Considerations

- At-least-once mode can significantly reduce the cost and latency of a pipeline.
- When using at-least-once mode the rate of duplicate records depends on the number of retries, the baseline rate is typically low (<1%).
- Align your I/O semantics with the streaming mode. For example, set BigQuery write mode to `STORAGE_API_AT_LEAST_ONCE`.
- Not all transforms are idempotent, such as a transform that appends uses current timestamp. In that case, a duplicate record can produce several distinct outputs.

Performance Comparison

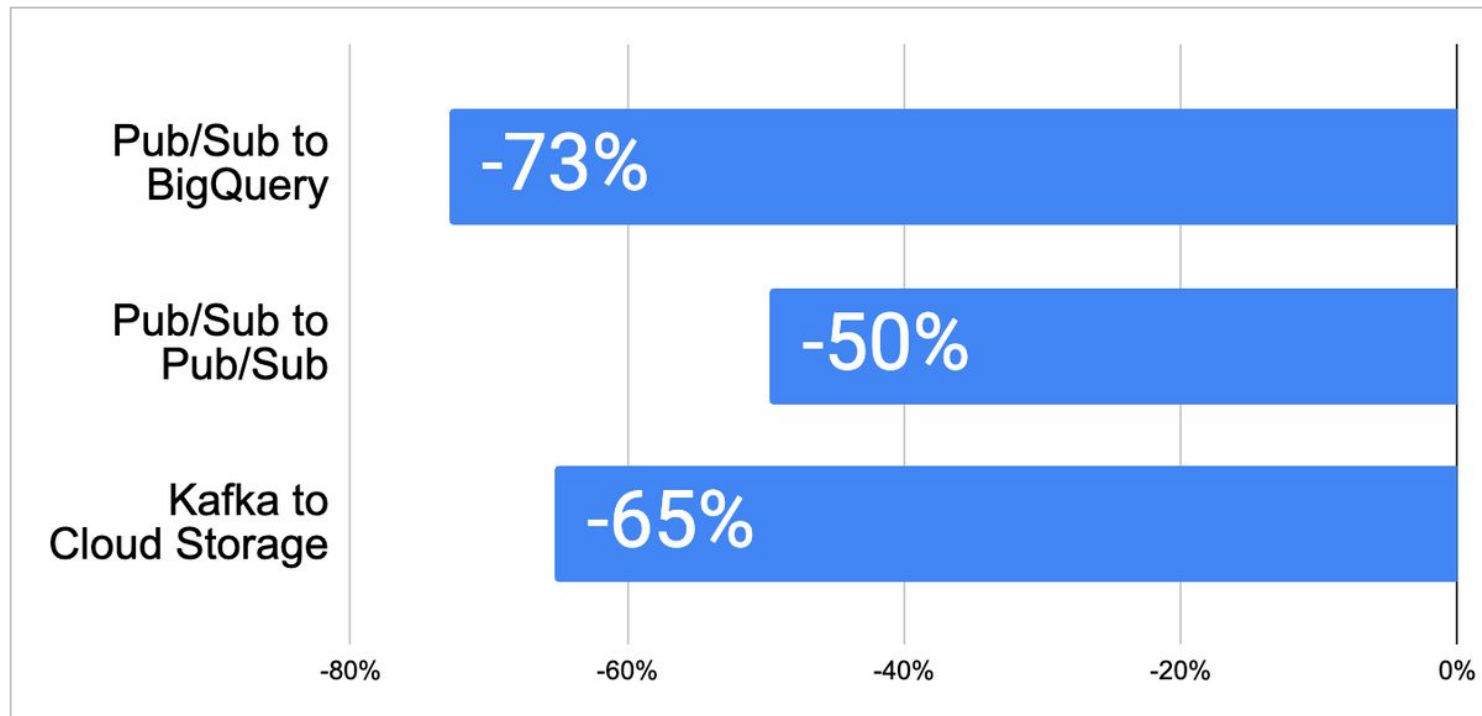
Pubsub-to-Pubsub pipeline with stragglers



Exactly Once

At Least Once

Cost Comparison



Customer Success











“By incorporating at-least-once mode in our platform that is built on Dataflow and Pub/Sub, we have seen a portion of our Dataflow jobs cut costs by 50%.

Since this is used by several consumers, 7 downstream systems are now cheaper overall with this simple change. Because of the way this system works, there has been 0 effects of duplicates!”

Specify Streaming Modes

- Jobs:
 - `--dataflowServiceOptions=streaming_mode_at_least_once` (Java)
 - `--dataflow_service_options=streaming_mode_at_least_once` (Python, Go)
- Templates:
 - `--additional-experiments=streaming_mode_at_least_once`
- Custom Templates metadata file:
 - `"streaming": true,`
 - `"supportsAtLeastOnce": true,`
 - `"supportsExactlyOnce": true,`
 - `"defaultStreamingMode": "AT_LEAST_ONCE",`

Viewing Streaming Mode

Job ID	2024-08-18_17_14_59-1269300216318115384
Job type	Streaming
Job status	 Running
SDK version	Apache Beam SDK for Java 2.59.0-SNAPSHOT
Job region 	us-west1
Worker location 	us-west1
Current workers 	1
Latest worker status	Autoscaling: Reduced the number of workers to 1 based on low average worker CPU utilization, and the pipeline having sufficiently low backlog and keeping up with input rate.
Straggler status 	No active straggler
Start time	August 18, 2024 at 5:15:00 PM GMT-7
Elapsed time	19 hr 18 min
Encryption type	Google-managed
Dataflow Prime 	Disabled
Runner v2 	Disabled
Streaming Engine 	Enabled
Vertical Autoscaling 	Disabled
Streaming Mode 	At least once

Dataflow Job Info tab

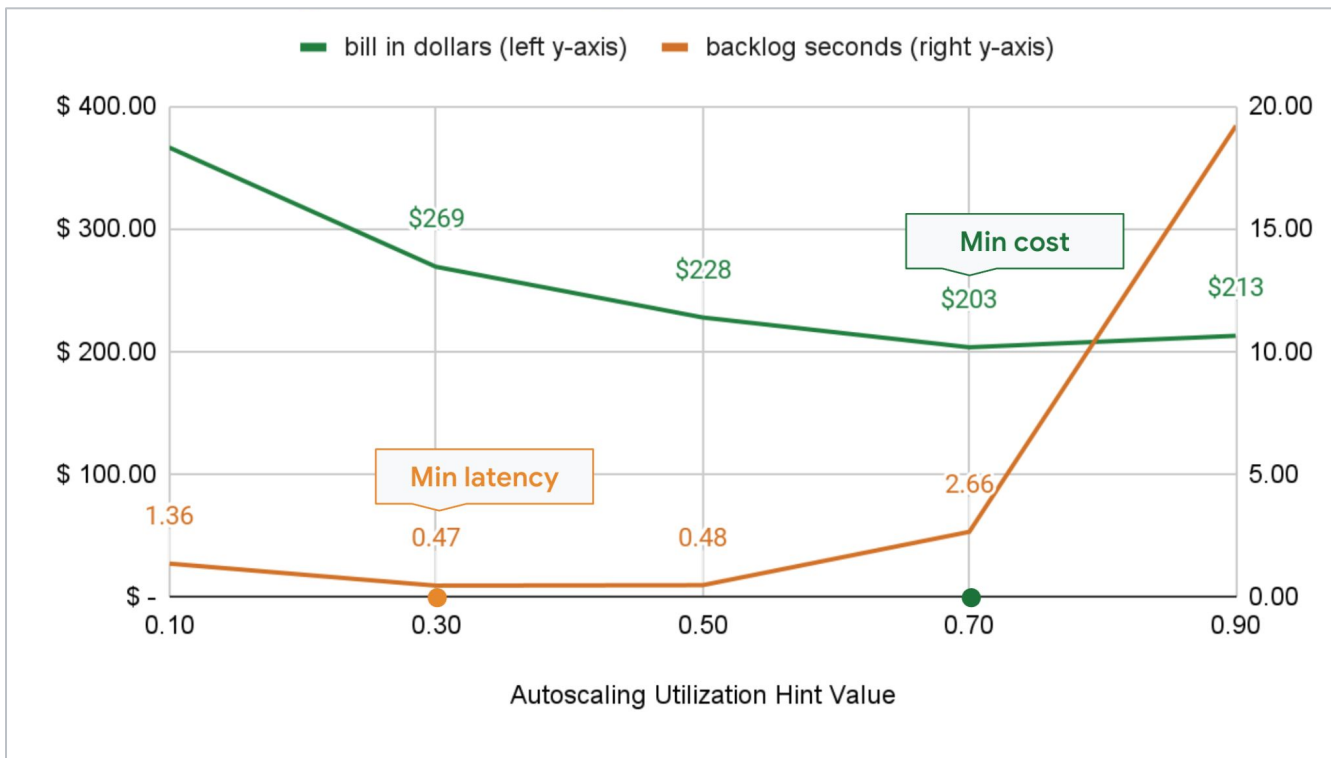
02

Autoscaling Hints

Background

- Customers often have different autoscaling preferences for their streaming pipelines.
- Some prefer more aggressive upscaling to achieve lower latency at peak traffic.
- Others may want to provision resources less aggressively to keep costs lower.
- Autoscaling Hint gives users the ability to calibrate the autoscaler behavior accordingly.

Trade-offs: Latency vs Cost



Pub/Sub to BigQuery Job

When to use it?

Consider reducing the autoscaling utilization hint to achieve lower latency when the pipeline:

Scales up too slowly: The autoscale lags behind traffic spikes and backlog seconds start to grow.

Scales down too much: Current worker CPU utilization is low and the backlog grows.

Improved Monitoring



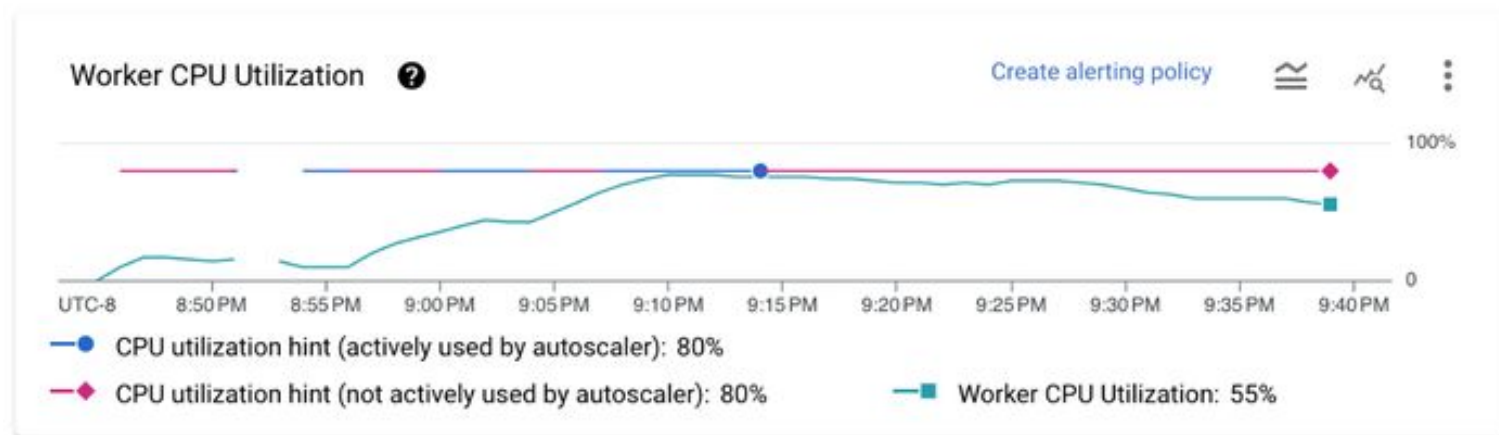
Autoscaling: shows current and target worker counts as time-series autoscaling data, along with min / max and target number of workers.

Improved Monitoring



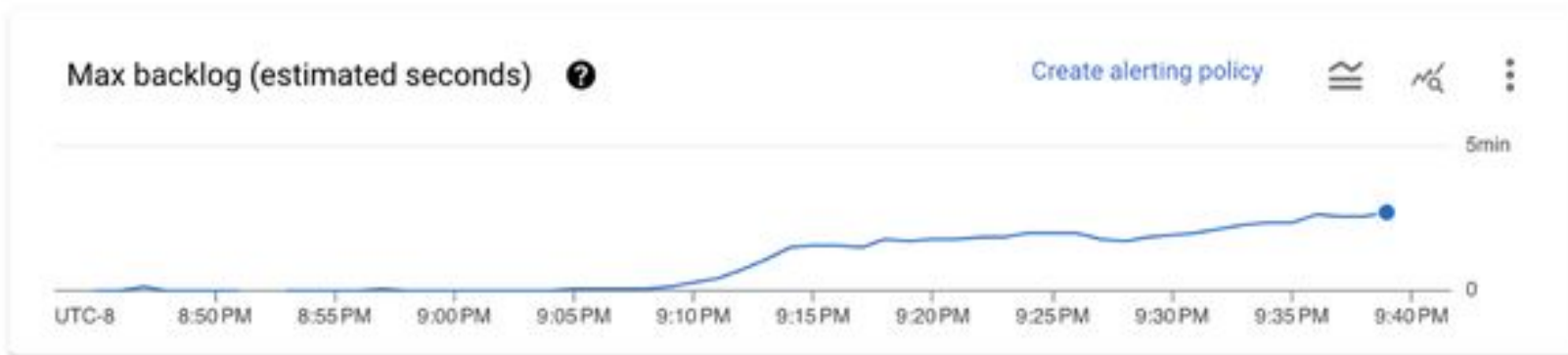
Autoscaling rationale: explains the factors driving autoscaling decisions for upscale, downscale, and no change.

Improved Monitoring



Worker CPU utilization: shows current user worker CPU utilization and customer hint value (when it is actively used in the autoscaling decision). This is an important factor in the autoscaling decisions.

Improved Monitoring



Max backlog: chart gives an indication of pipeline latency. This is another major factor in the autoscaling decisions.

How to use it?

Example:

- `--dataflowServiceOptions=worker_utilization_hint=0.3` (Java)
- `--dataflow_service_options=worker_utilization_hint=0.3` (Python, Go)

Note: Use hints in the range [0.1, 0.9] where lower values are more aggressive.

03

Inflight Updates

Inflight Updates

- Customers often want to update the min/max number of workers for their live streaming jobs, but can't afford the downtime.
- This feature allows users to adjust autoscaling at runtime, without pausing the data processing for long-running streaming jobs.

Why Update?

- **Save cost when latency spikes:** Latency spikes may cause excessive upscaling to handle the input load. Customers may want to apply a smaller worker limits to reduce the costs.
- **Handle expected load spikes:** When customers know about an event that may drastically increase their load, they may want to pre-scale up in advance.

Perform Updates

An inflight update may include one or more of: min workers, max workers, autoscaling worker utilization hint.

```
gcloud dataflow jobs update-options \  
  --region=us-central1 \  
  --min-num-workers=3 \  
  --max-num-workers=25 \  
  --worker-utilization-hint=0.4 \  
2024-08-09_10_11_12-123456
```

04

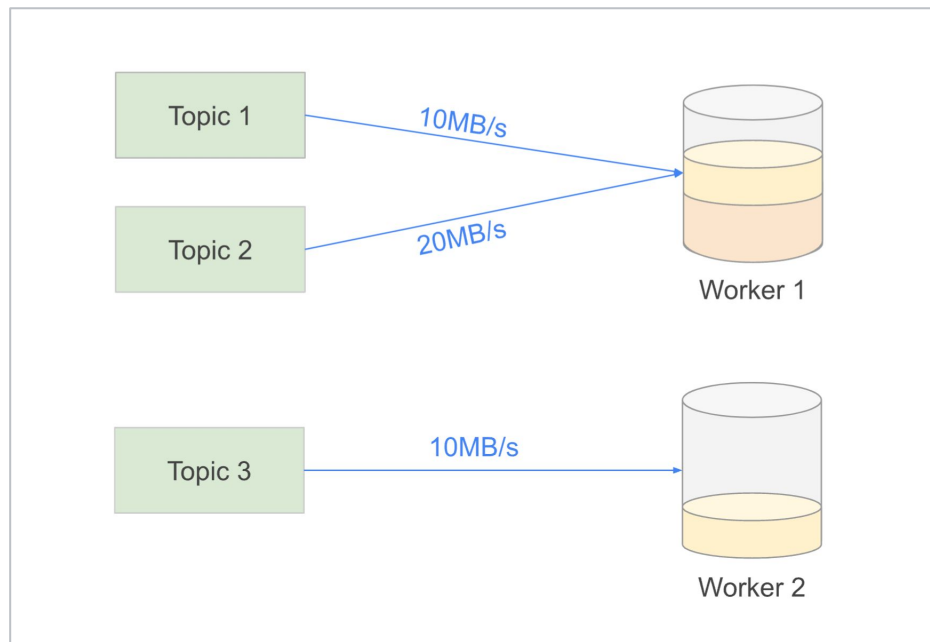
Active Load Balancing

Why Active Load Balancing?

- Scaling workloads costly, especially in streaming, where latency is heavily scrutinized.
- When there are hot keys, ranges, or workers, they become the bottleneck.
- Autoscaling reacts after there's a backlog and incurs overhead for adding workers.
- To help, we recently introduced load balancing in Dataflow to help with source reads.
- Better distributed workloads allows processing more data with less resources and lower latencies.

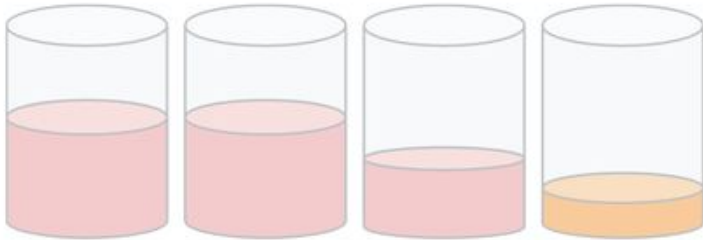
Active Load Balancing at Work

- When a pipeline starts up, Dataflow doesn't know in advance the amount of data coming in on any particular data source.
- Load may change throughout the life of the pipeline.

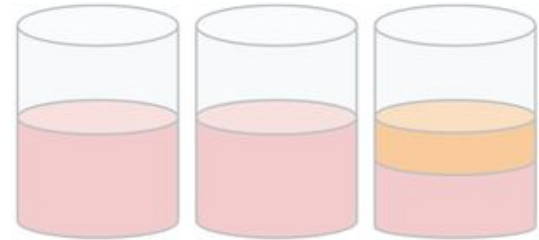


Active Load Balancing at Work

- Distributes load by moving work across workers to improve utilization and performance.
- Without Load Balancing a single worker could become the bottleneck for the entire pipeline.

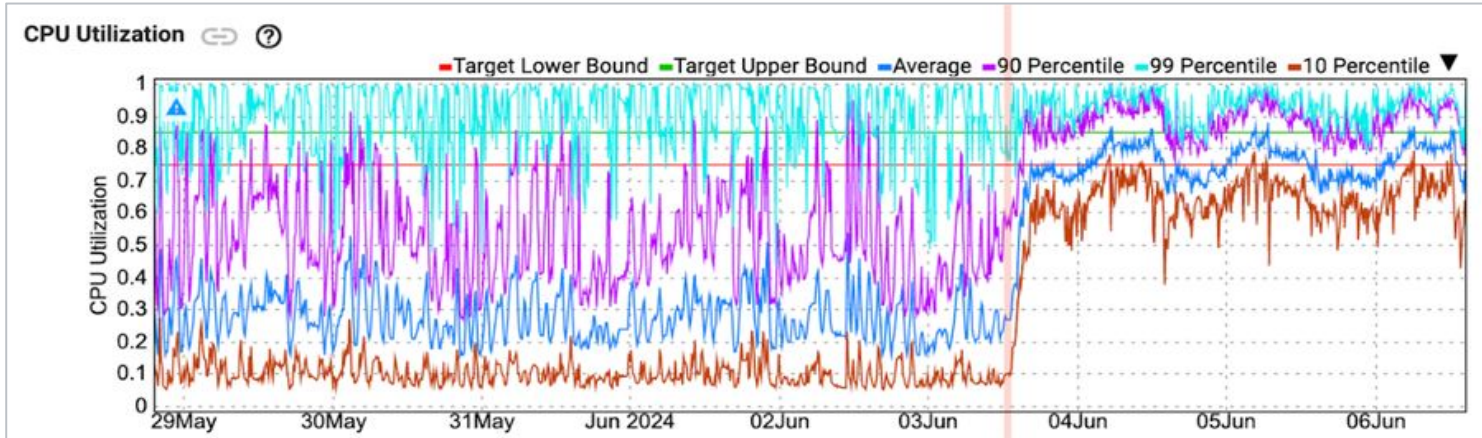
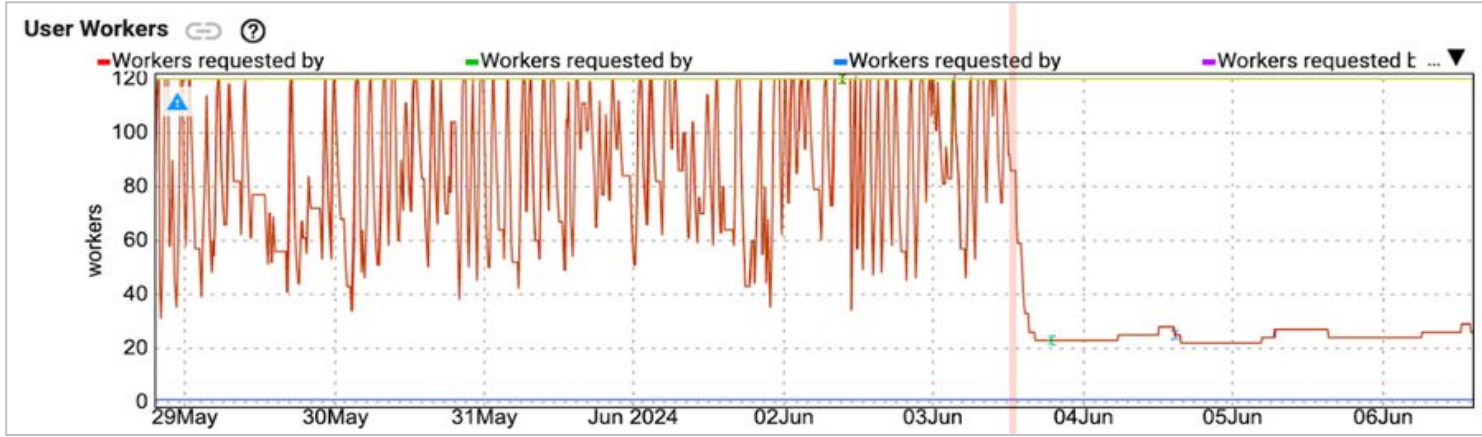


Without Load Balancing

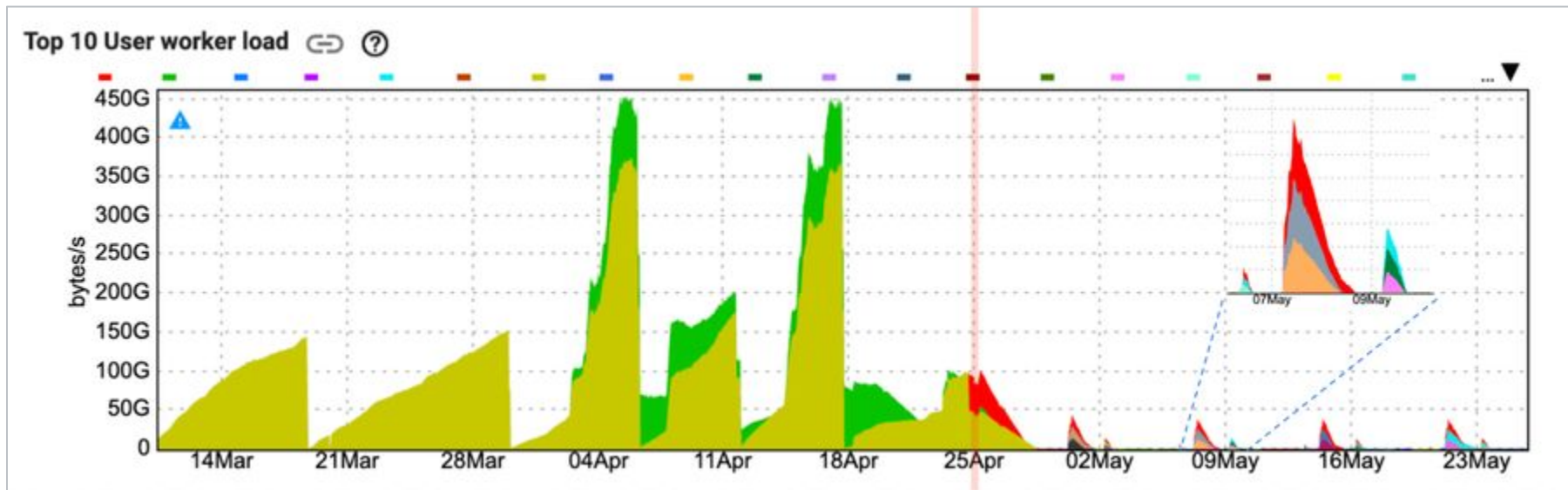


With Load Balancing

Customer Success: Case 1

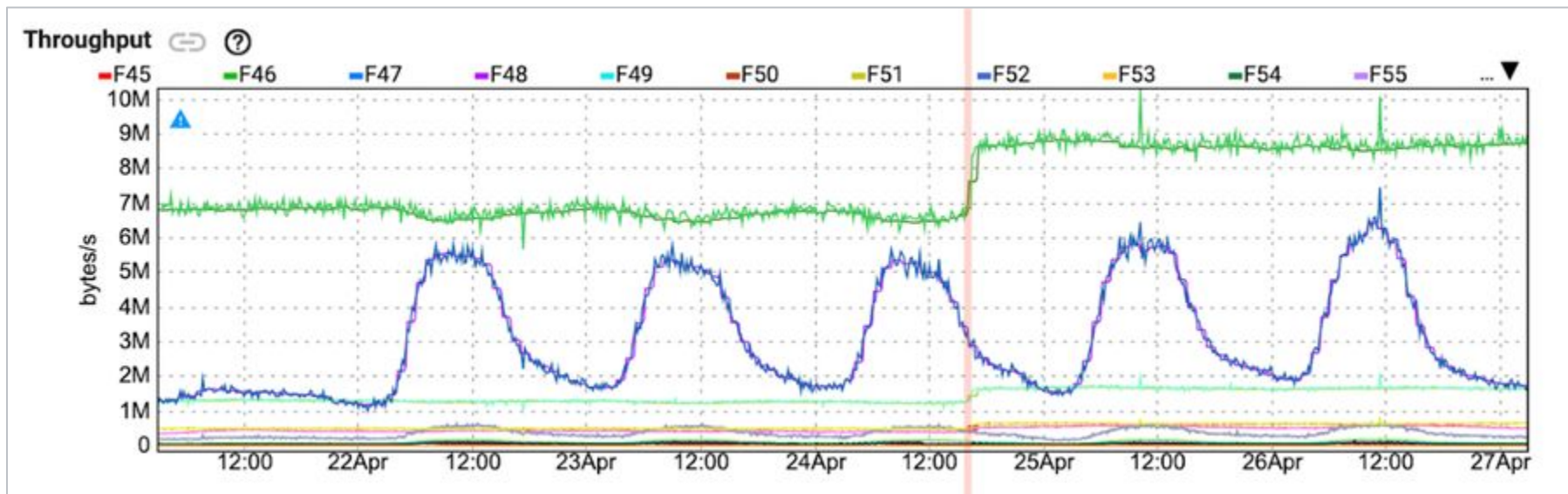


Customer Success: Case 2



Note: Each color represents a single worker and its assigned workload, not available externally.

Customer Success: Case 3



Thank you!

