Metrics & Monitoring

Konstantin Knauf, Solutions Architect

Metrics & Monitoring

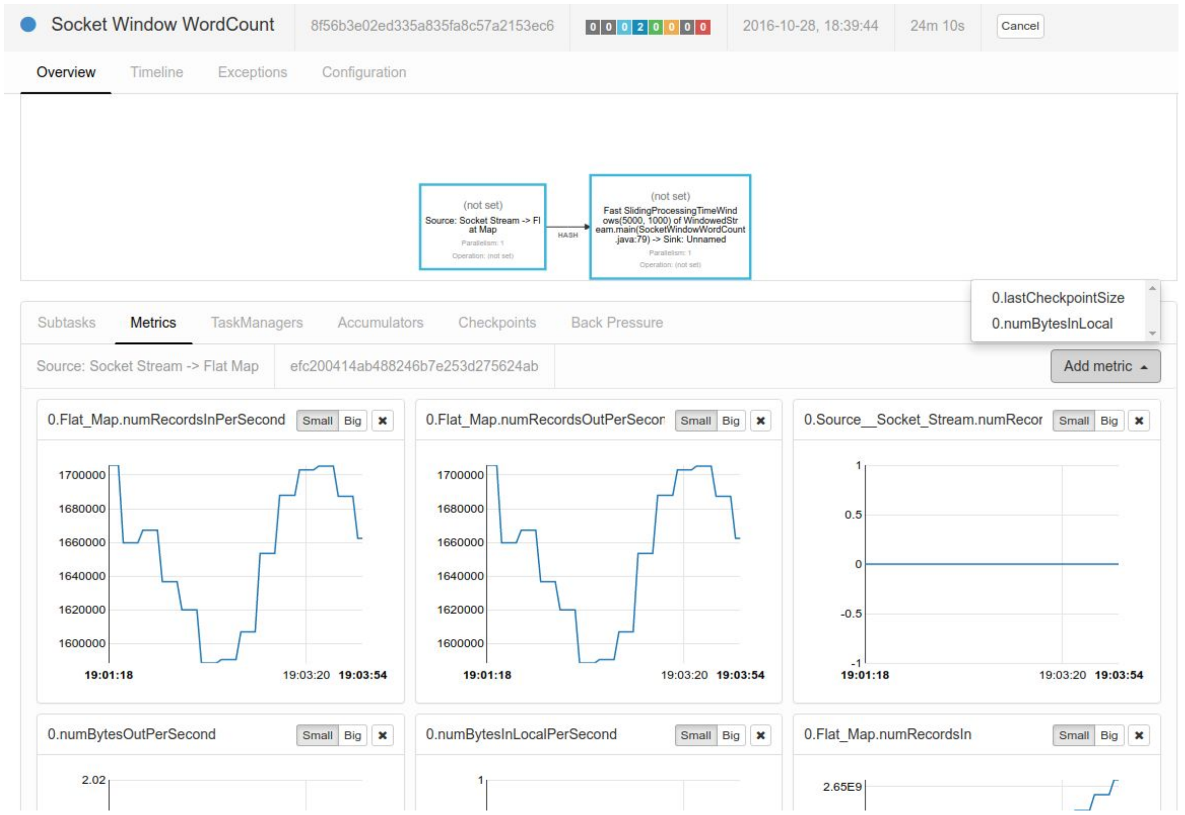
Agenda

- Flink's Metrics System "How"
  - `Metrics`
  - `MetricsReporter`
- Key Metrics for Continuous Monitoring *"What"*
  - Health
  - Throughput & Progress
  - Latency
- Key Metrics for Troubleshooting "What else"

Flink’s Metrics System

Metrics

- <identifier, measurement>
- Types
  - Counter
  - Meter (rate)
  - Histogram
  - Gauge (arbitrary value)



Socket Window WordCount
8f56b3e02ed335a835fa8c57a2153ec6
2016-10-28, 18:39:44
24m 10s
Cancel
Overview
Timeline
Exceptions
Configuration
Source: Socket Stream -> Flat Map
HASH
Fast SlidingProcessingTimeWindows(5000, 1000) of WindowedStream.main(SocketWindowWordCount.java:79) -> Sink: Unnamed
Subtasks
Metrics
TaskManagers
Accumulators
Checkpoints
Back Pressure
0.lastCheckpointSize
0.numBytesInLocal
Add metric
Source: Socket Stream -> Flat Map
efc200414ab488246b7e253d275624ab
0.Flat_Map.numRecordsInPerSecond
0.Flat_Map.numRecordsOutPerSecon
0.Source__Socket_Stream.numRecor
0.numBytesOutPerSecond
0.numBytesInLocalPerSecond
0.Flat_Map.numRecordsIn
Small
Big
19:01:18
19:03:20
19:03:54

Example

```
public static class MyMap extends RichMapFunction<String, String> {
  private Counter count;

  @Override
  public void open(Configuration config) {
    count = getRuntimeContext()
      .getMetricGroup()
      .counter("numRecordsIn");
  }

  @Override
  public String map(String input) {
    count.inc();
    // return something
  }
}
```

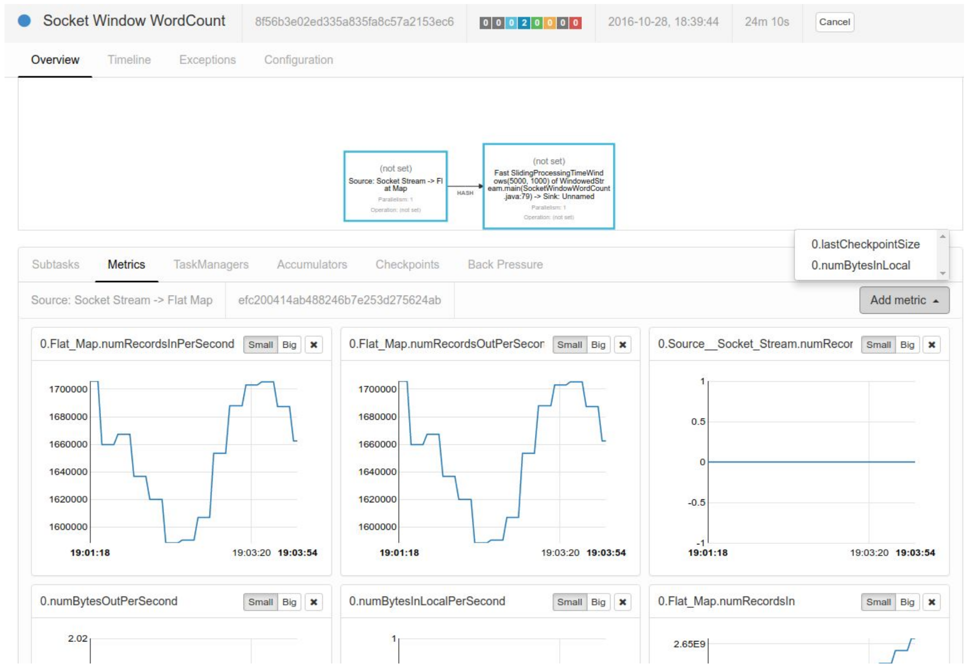
Metrics

Scopes

- metrics scope to different levels of a Flink deployment
- the keys to attach to metrics in a certain scope can be configured
  - metrics.scope.jm: <host>.jobmanager
  - metrics.scope.task:
    <host>.taskmanager.<tm_id>.<job_name>.<task_name>.<subtask_index>
- Checkout https://ci.apache.org/projects/flink/flink-docs-release-1.7/monitoring/metrics.html#scope for details

Accessing Metrics

- WebUI → TaskMetrics
- REST API
- **`MetricsReporters`**



/jobs/<id>/metrics

/jobs/<id>/checkpoints

/jobs/<id>/vertices/<id>/metrics?get=0.numRecordsOutPerSecond

/taskmanagers/<id>/metrics?get=<metric>

Accessing Metrics

Metrics Reporters

- Datadog
- Ganglia
- Graphite
- JMX

- Prometheus
- StatsD
- SLF4J
- InfluxDB

Or write your own...

Accessing Metrics

A Simple Log4jReporter

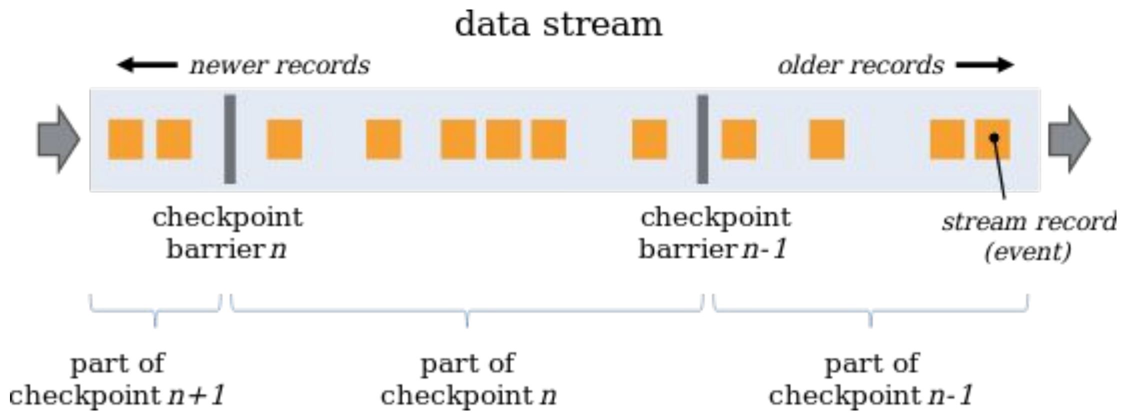
```
public static class Log4JReporter implements MetricReporter, Scheduled {
  private static final Logger LOG = LoggerFactory.getLogger(Log4jReporter.class);

  private final Map<Counter, String> counters = new ConcurrentHashMap<>();

  public void notifyOfAddedMetric(Metric metric, String metricName, MetricGroup group) {
    if (metric instanceof Counter) {
      counters.put((Counter) metric, group.getMetricIdentifier(metricName));
    }
  }

  public void notifyOfRemovedMetric(Metric metric, String metricName, MetricGroup group) {
    if (metric instanceof Counter) {
      counters.remove(metric);
    }
  }

  public void report() {
    for (Map.Entry<Counter, String> metric : counters.entrySet()) {
      LOG.info(metric.getValue() + ": " + metric.getKey());
    }
  }
}
```

Key Metrics for Continuous Monitoring

Key Metrics

General Health

- Is “RUNNING”?
  - `uptime`
  - `fullRestarts`
- Checkpointing Consistently?
  - `numberOfCompletedCheckpoints`
  - `numberOfFailedCheckpoints`
  - `lastCheckpointSize`



Key Metrics

Throughput & Progress

- Task & Operator Level Throughput
  - `numRecords(In|Out)PerSecond`
  - `numRecords(In|Out)`
- Progress & Event-Time Lag
  - `currentOutputWatermark`
- Keeping Up
  - `(Kafka) records-lag-max`
  - `(Kinesis) millisBehindLatest`

Key Metrics

Latency

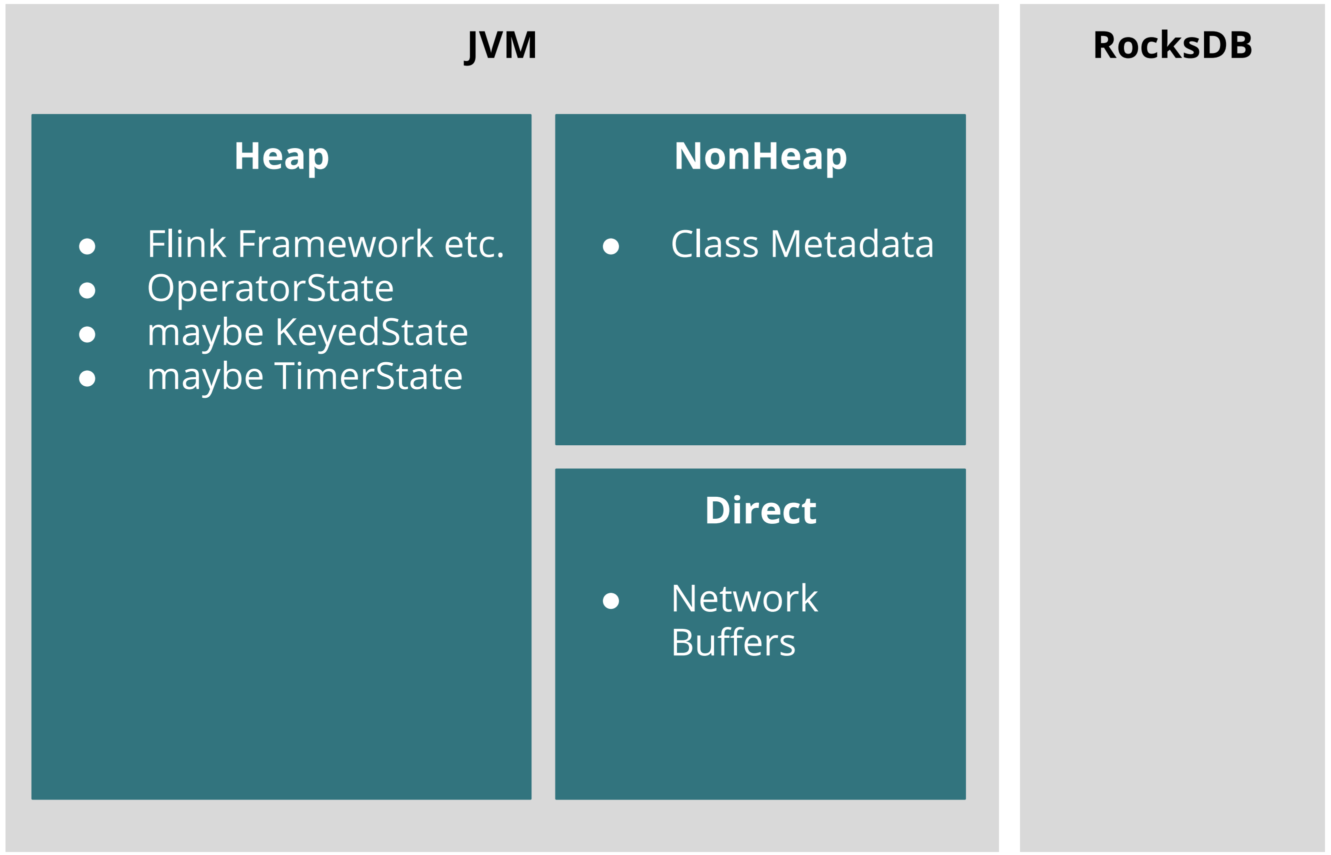
- Add timestamp to events at multiple stages, e.g.
    - event creation
    - ingestion
    - publishing
- custom metrics for reporting these metrics

Key Metrics for Troubleshooting

JVM Metrics

Memory

- `Status.JVM.Memory.`
    - `NonHeap.Committed`
    - `Heap.Used`
    - `Heap.Committed`
    - `Direct.MemoryUsed`
    - `Mapped.MemoryUsed`
    - `G1 Young Generation.Time`
    - `G1 Old Generation.Time`



JVM Metrics

CPU

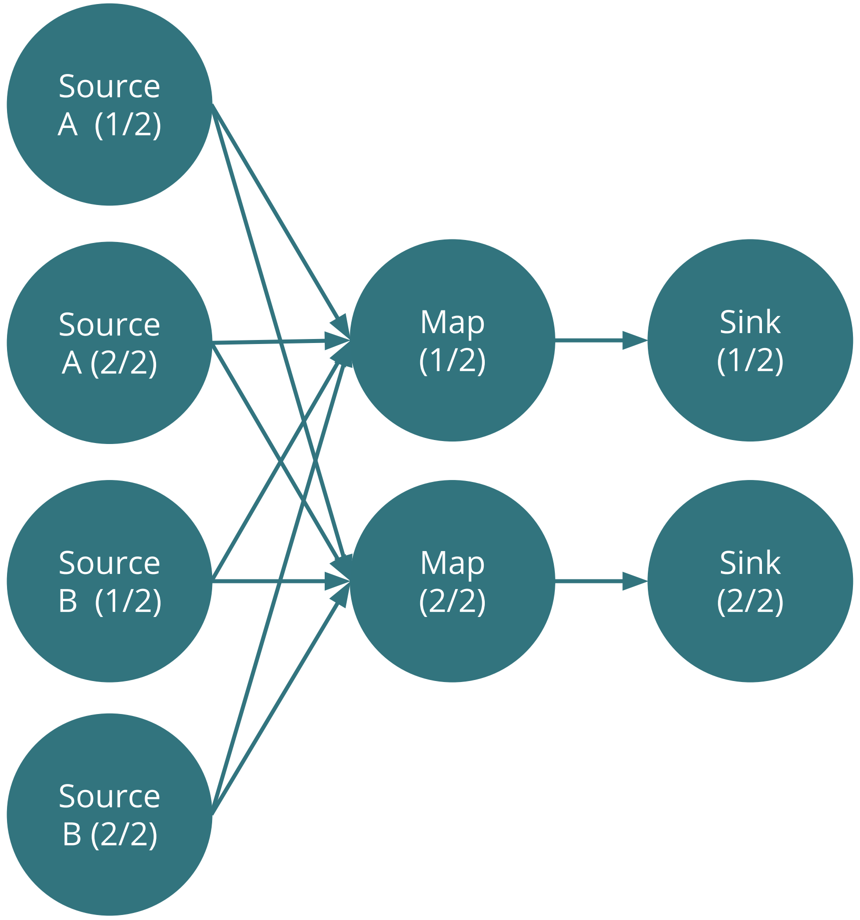
Note: 0.021 = 100% load for a Taskmanager container with 1 CPU on a 48 core machine.

- Metrics
  - `Status.JVM.CPU.Load`
  - `Status.JVM.CPU.Time`
- Leave some slack for catch-up scenarios (& RocksDB)

Troubleshooting Latency

Latency Tracking

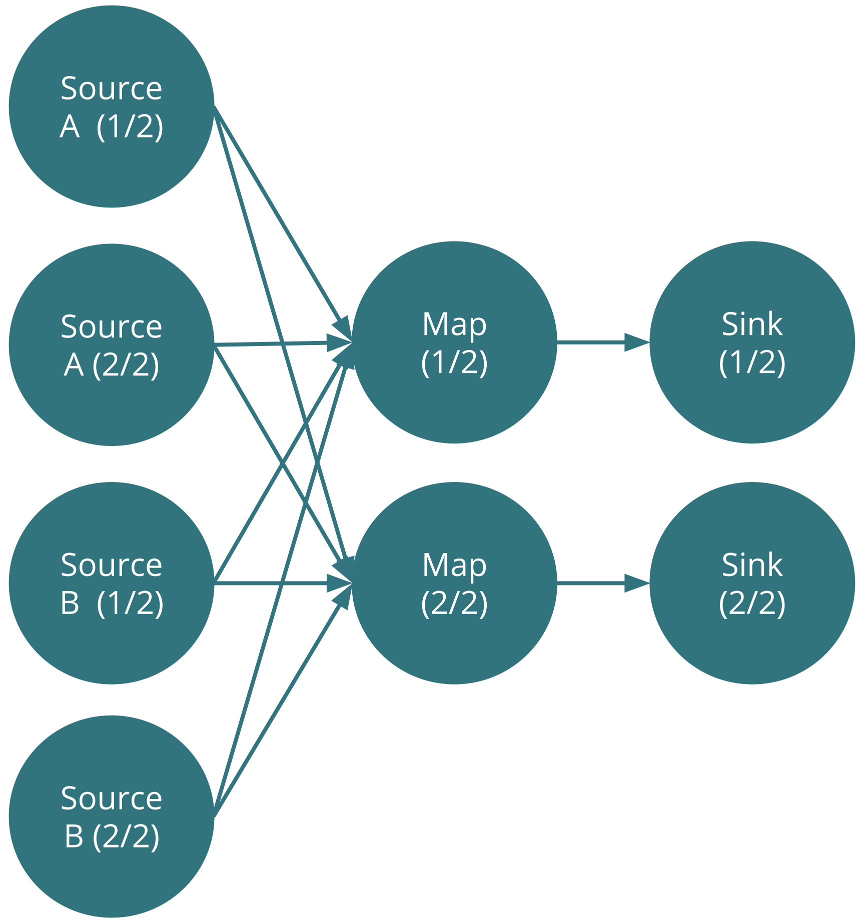
- For each operator-subtask a latency histogram is exposed
- Enabled via `metrics.latency.interval`
- scoped to job
- `latency.source_id.<source_id>.operator_id.<operator_id>.operator_subtask_index.<subtask_index>.`



Troubleshooting Latency

Latency Tracking

- For each operator-subtask a latency histogram is exposed
- Enabled via `metrics.latency.interval`

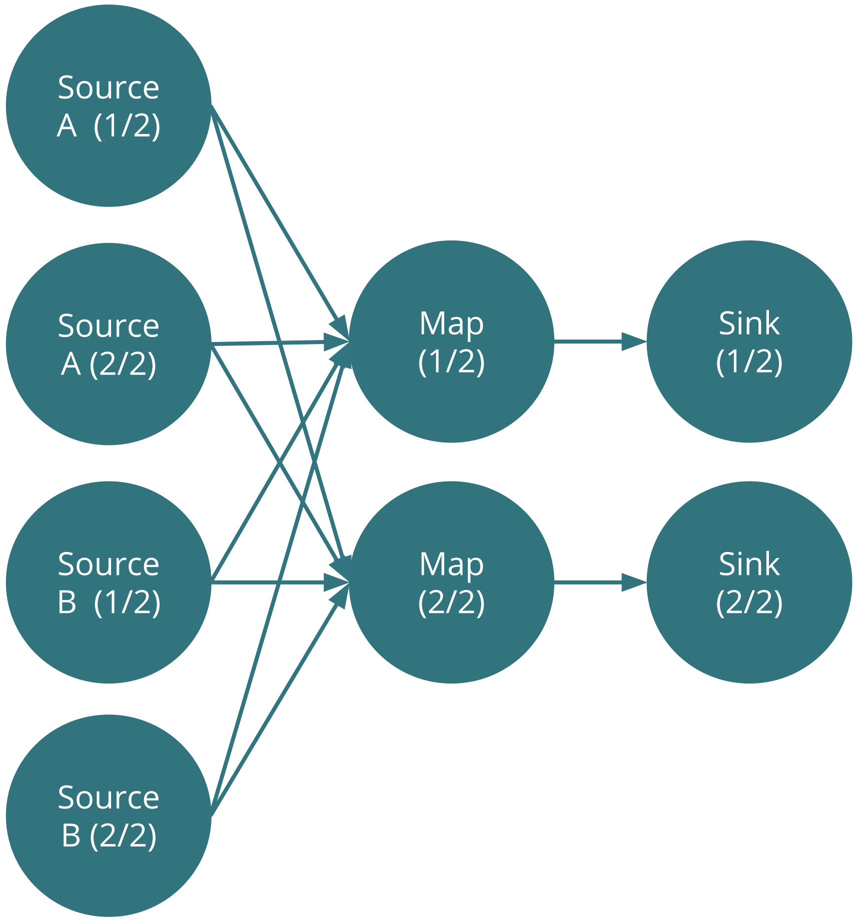


Troubleshooting Backpressure

Latency Tracking

```
metrics.latency.granularity: single
```

- Per Subtask
  - Latency histogram for both sources
- Overall
  - 4 (P*#Operators)

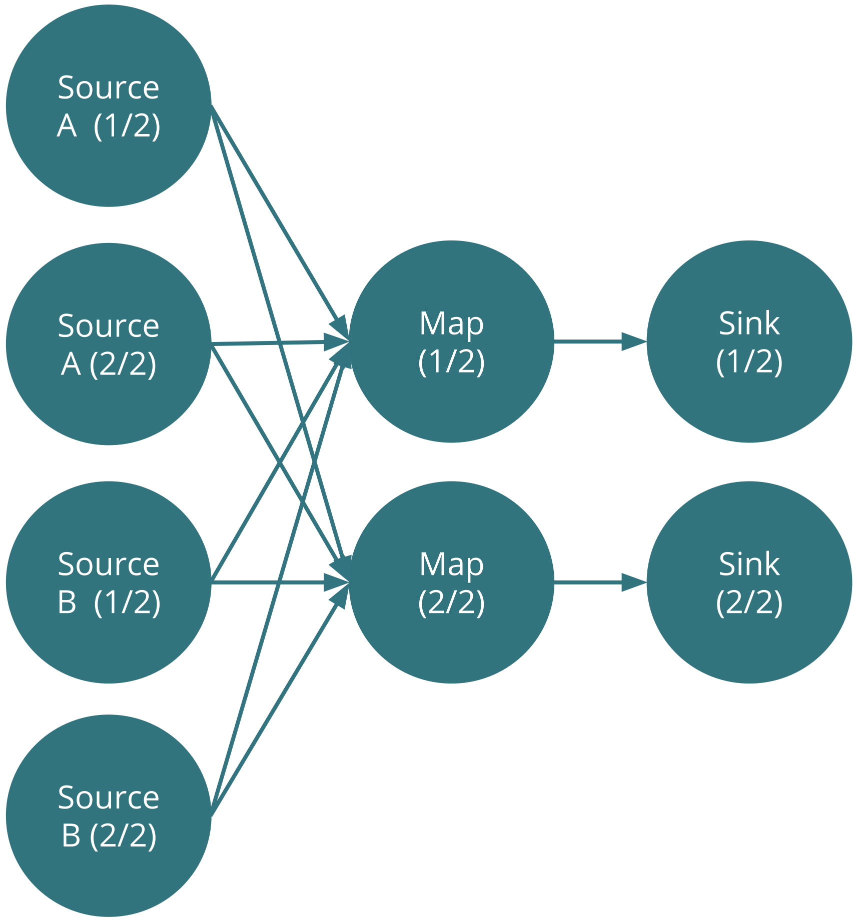


Troubleshooting Backpressure

Latency Tracking

```
metrics.latency.granularity: operator
```

- Per Subtask
  - Latency histogram for Source A
  - Latency histogram for Source B
- Overall
  - 8 histograms (P * #Sources * #Operators)

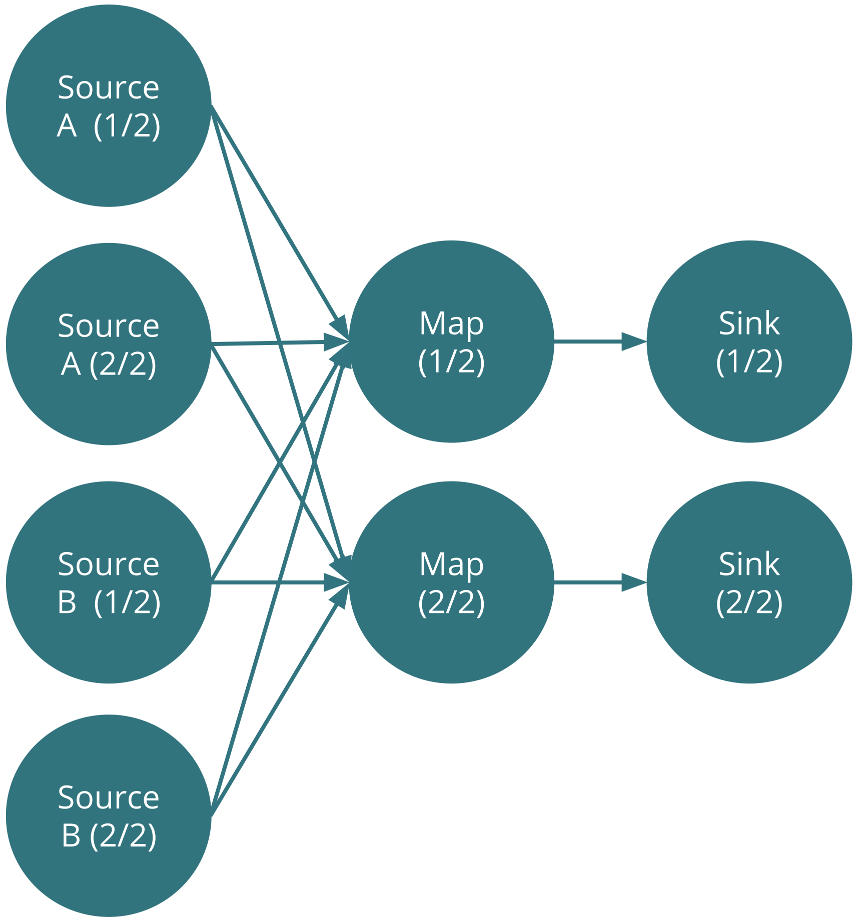


Troubleshooting Backpressure

Latency Tracking

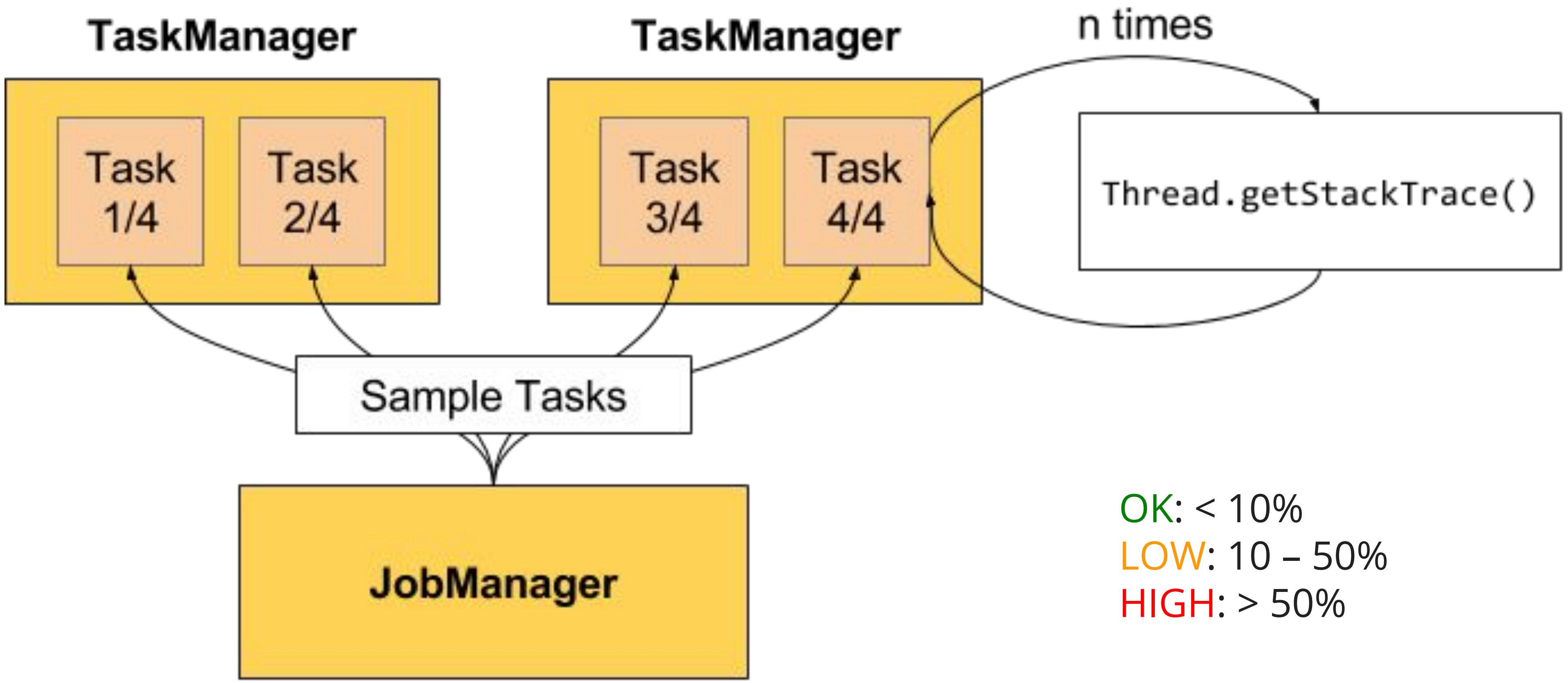
`metrics.latency.granularity: subtask`

- Per Subtask
  - Latency histogram for Source A (1/2)
  - Latency histogram for Source A (2/2)
  - Latency histogram for Source B (1/2)
  - Latency histogram for Source B (2/2)
- Overall
  - 16 histogram (**P^2** * #Sources * #Operators)

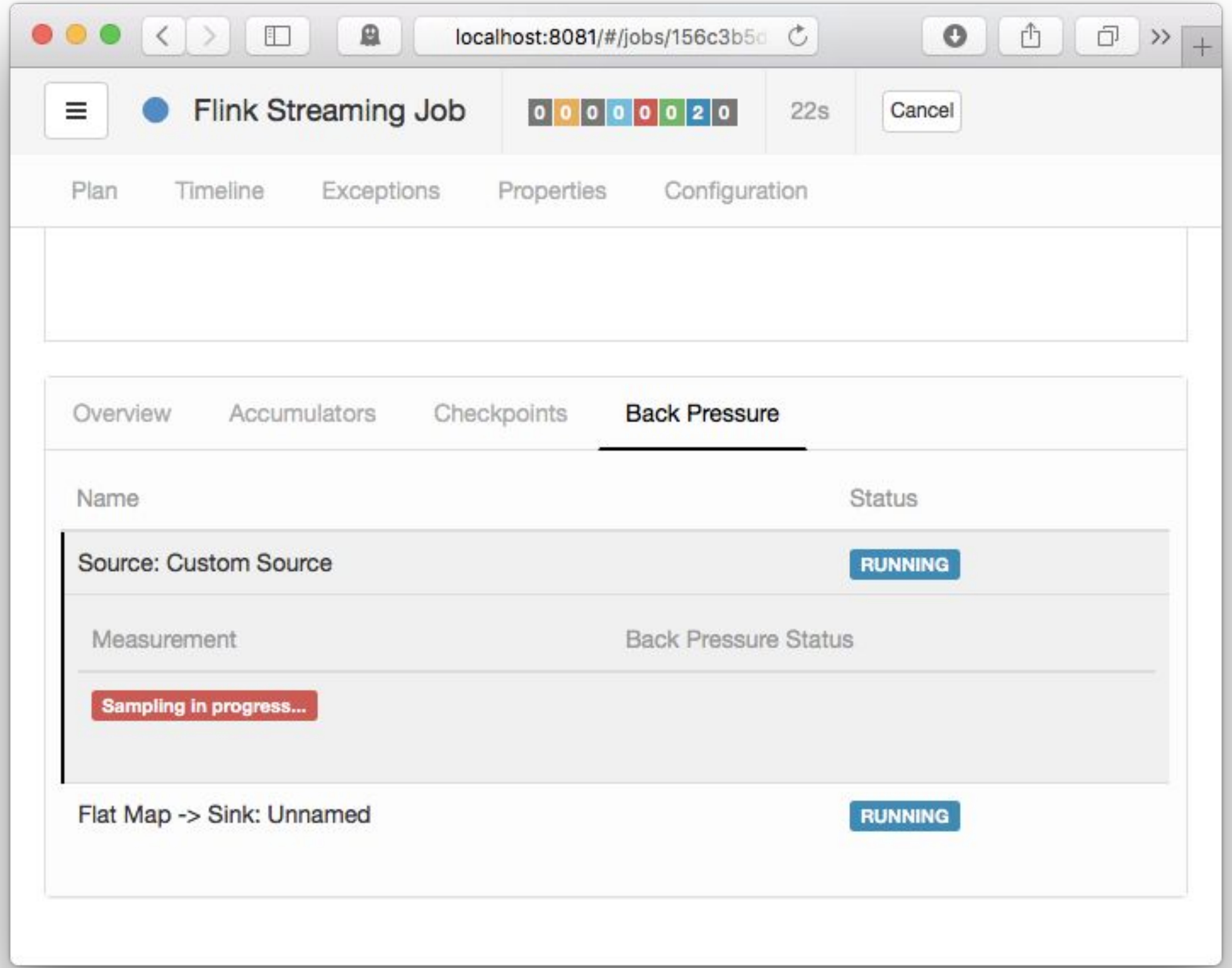


Troubleshooting Backpressure

Backpressure Monitor



OK: < 10%
LOW: 10 – 50%
HIGH: > 50%


localhost:8081/#/jobs/156c3b5c

Flink Streaming Job

0 0 0 0 0 0 2 0

22s

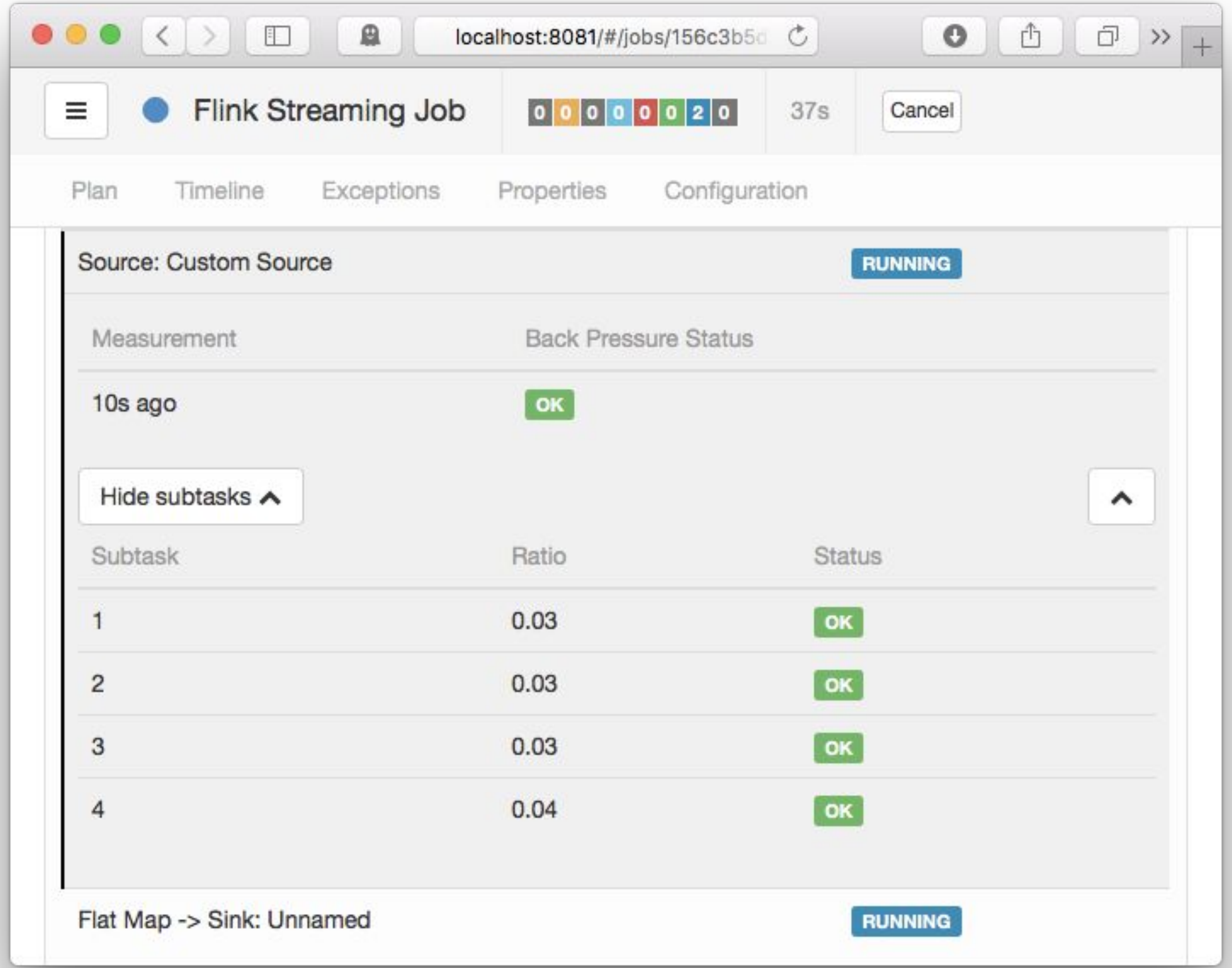
Cancel

Plan | Timeline | Exceptions | Properties | Configuration

Overview | Accumulators | Checkpoints | Back Pressure

Name	Status
Source: Custom Source	RUNNING
Flat Map -> Sink: Unnamed	RUNNING

Measurement	Back Pressure Status
Sampling in progress...	



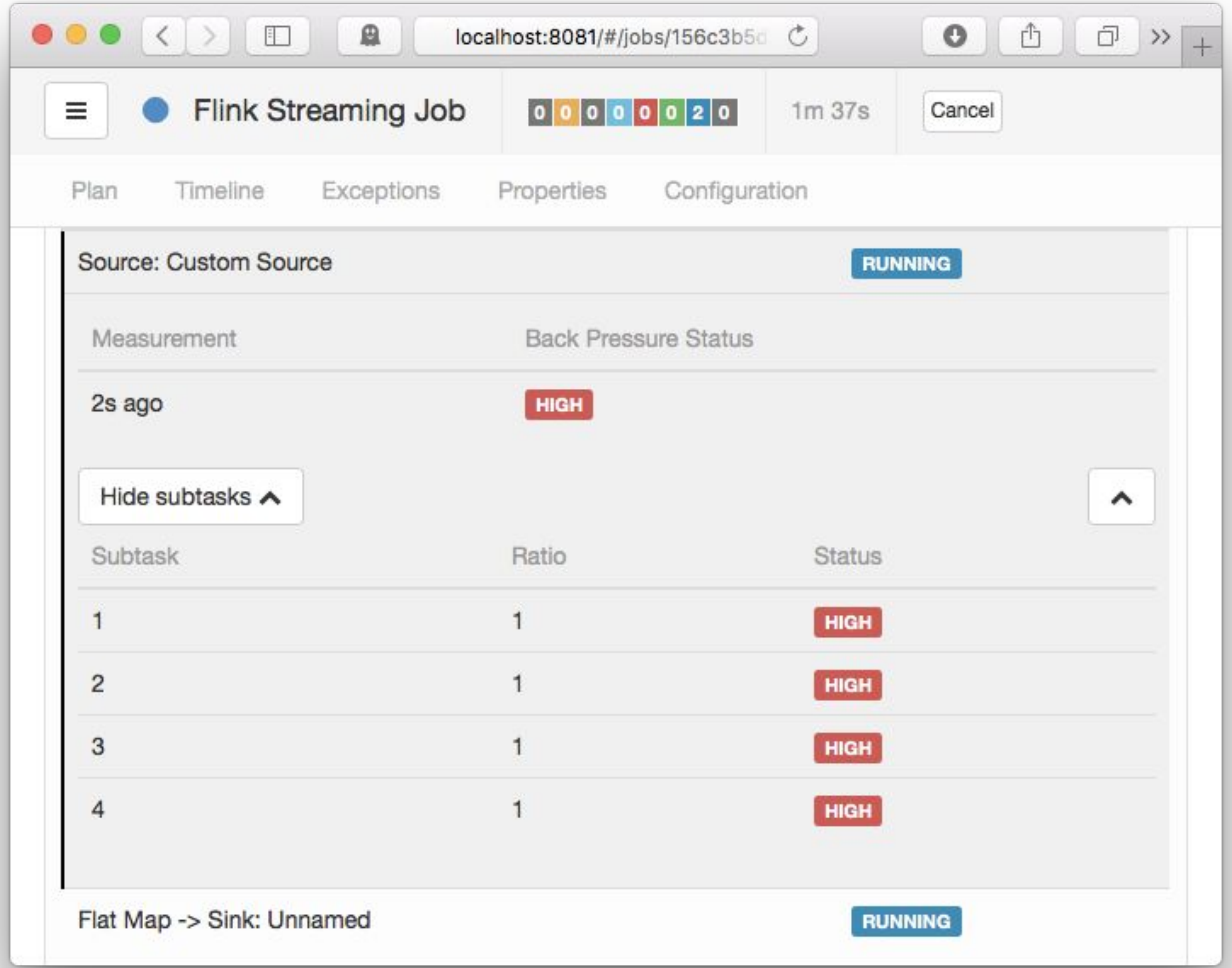
Source: Custom Source RUNNING

Measurement	Back Pressure Status
10s ago	OK

Hide subtasks

Subtask	Ratio	Status
1	0.03	OK
2	0.03	OK
3	0.03	OK
4	0.04	OK

Flat Map -> Sink: Unnamed RUNNING



Source: Custom Source RUNNING

Measurement	Back Pressure Status
2s ago	HIGH

Hide subtasks

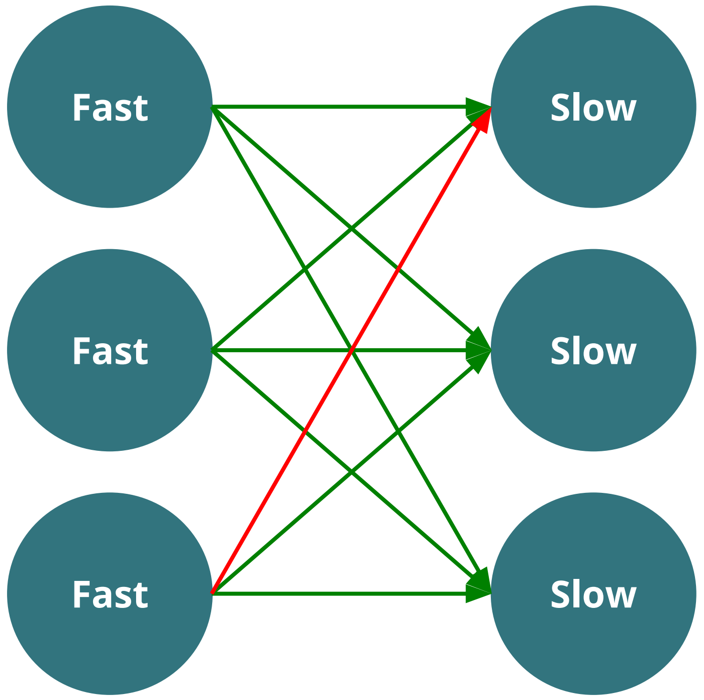
Subtask	Ratio	Status
1	1	HIGH
2	1	HIGH
3	1	HIGH
4	1	HIGH

Flat Map -> Sink: Unnamed RUNNING

Troubleshooting Backpressure

Asymmetric Backpressure

- situation where backpressure only occurs in one channel
- hard to detect, but can lead to checkpoint timeouts
- Metrics
  - `inputQueueLength`
  - `outputQueueLength`



ververica

konstantin@ververica.com www.ververica.com @VervericaData