Flink 监控与性能优化

极客时间

扫码试看/订阅

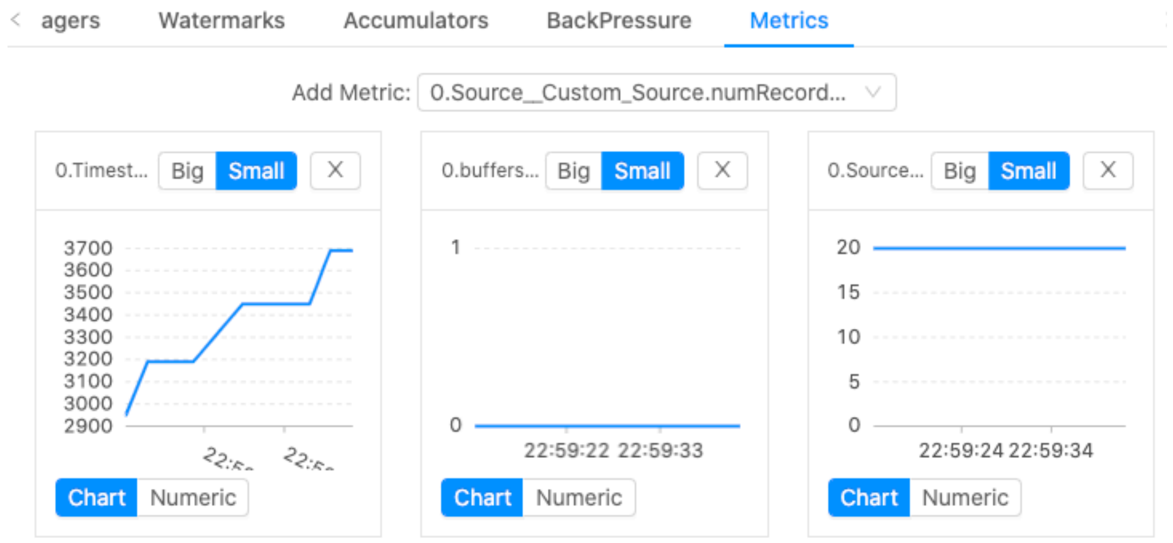
《Flink核心技术与实战》视频课程

目录

Metric 指标分类与采集

Metric 指标监控

- <identifier, measurement>
- Metric 类型：
  - Counter
    - 计数器
  - Gauge
    - 最简单的Metric，反映一个值
  - Meter
    - 单位时间内发生事件的次数
  - Histogram
    - 统计数据分布，Mean，Max，Min，StdDec 等
- Exposed via MetricReporters
- Also a REST API
- Visualized in the WebUI

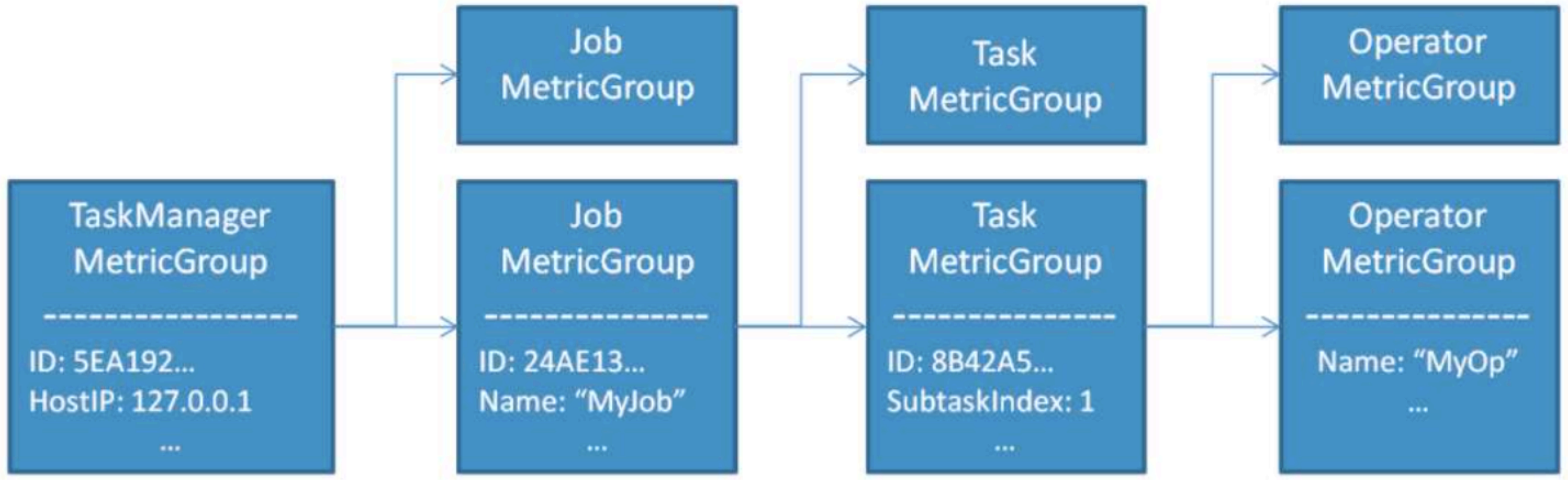


Metric 类型分类

- Counter
  - 计数器
- Gauge
  - 最简单的 Metric，反映一个值
- Meter
  - 单位时间内发生事件的次数
- Histogram
  - 统计数据分布，Mean，Max，Min，StdDec 等

MetricGroup

- Metric 在 Flink 内部有多层结构，以 Group 的方式组织
- Metric 唯一标识：Metric Group + Metric Name



eg: localhost.taskmanager.1234.MyJob.MyOperator.0.MyMetric

自定义 Counter

```
public class MyMapper extends RichMapFunction<String, String> {
  private transient Counter counter;

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

  @Override
  public String map(String value) throws Exception {
    this.counter.inc();
    return value;
  }
}
```

自定义 Counter

```java
public class MyMapper extends RichMapFunction<String, String> {
  private transient Counter counter;

  @Override
  public void open(Configuration config) {
    this.counter = getRuntimeContext()
      .getMetricGroup()
      .counter("myCustomCounter", new CustomCounter());
  }

  @Override
  public String map(String value) throws Exception {
    this.counter.inc();
    return value;
  }
}
```

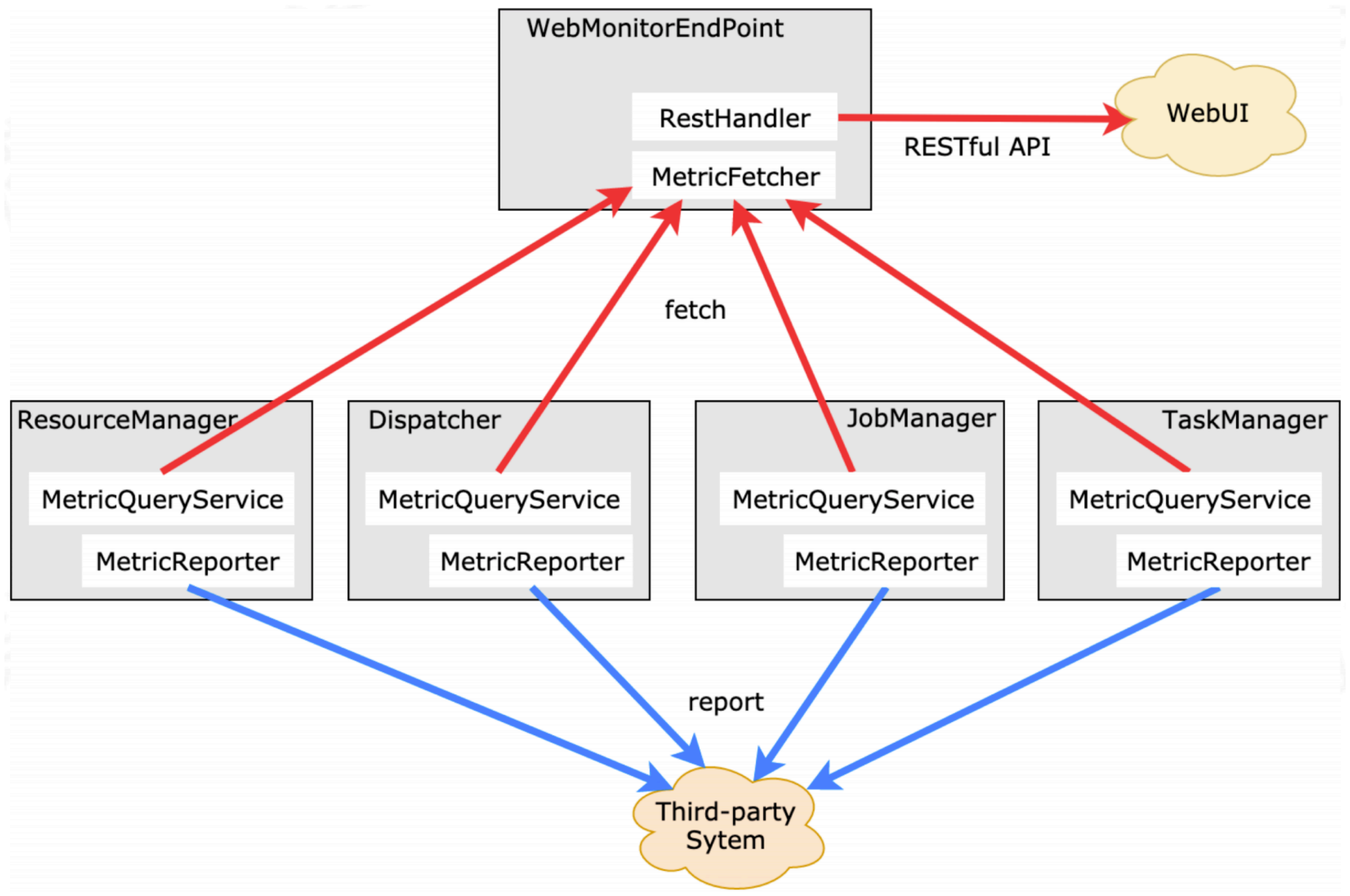
自定义 Gauge

```java
public class MyMapper extends RichMapFunction<String, String> {
  private transient int valueToExpose = 0;

  @Override
  public void open(Configuration config) {
    getRuntimeContext()
      .getMetricGroup()
      .gauge("MyGauge", new Gauge<Integer>() {
        @Override
        public Integer getValue() {
          return valueToExpose;
        }
      });
  }

  @Override
  public String map(String value) throws Exception {
    valueToExpose++;
    return value;
  }
}
```

获取 Metric



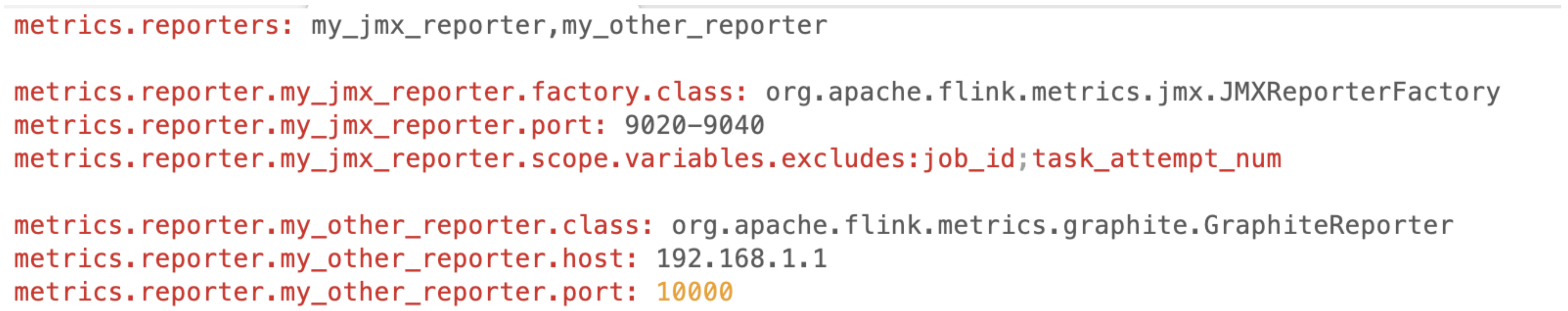
WebMonitorEndPoint
RestHandler
MetricFetcher
RESTful API
WebUI
fetch
ResourceManager
Dispatcher
JobManager
TaskManager
MetricQueryService
MetricReporter
report
Third-party Sytem

Metric Reporter

- Exposes metrics to the outside world
  - Ganglia
  - Graphite
  - JMX
  - StatsD
  - InfluxDB
  - Prometheus
  - or roll your own ...

JMXReporter 应用

通过在 conf/flink-conf.yaml 文件中配置



```
metrics.reporters: my_jmx_reporter,my_other_reporter

metrics.reporter.my_jmx_reporter.factory.class: org.apache.flink.metrics.jmx.JMXReporterFactory
metrics.reporter.my_jmx_reporter.port: 9020-9040
metrics.reporter.my_jmx_reporter.scope.variables.excludes:job_id;task_attempt_num

metrics.reporter.my_other_reporter.class: org.apache.flink.metrics.graphite.GraphiteReporter
metrics.reporter.my_other_reporter.host: 192.168.1.1
metrics.reporter.my_other_reporter.port: 10000
```

InfluxdbReporter 应用

```
metrics.reporter.influxdb.factory.class: org.apache.flink.metrics.influxdb.InfluxdbReporterFactory
metrics.reporter.influxdb.host: localhost
metrics.reporter.influxdb.port: 8086
metrics.reporter.influxdb.db: flink
metrics.reporter.influxdb.username: flink-metrics
metrics.reporter.influxdb.password: qwerty
metrics.reporter.influxdb.retentionPolicy: one_hour
metrics.reporter.influxdb.consistency: ANY
metrics.reporter.influxdb.connectTimeout: 60000
metrics.reporter.influxdb.writeTimeout: 60000
metrics.reporter.influxdb.interval: 60 SECONDS
```

Flink RestAPI 介绍与使用

Some available requests

/config

/overview

/jobmanager/metrics

/jobs

/jobs/<id>/metrics

/jobs/<id>/checkpoints

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

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

…

RestAPI 主要功能

- 系统监控指标
- 任务管理
- 集群管理
- 配置信息
- 资源上传（Jars）

Metric REST API integration

- 基于实体聚合指标:
  /jobmanager/metrics
  /taskmanagers/<taskmanagerid>/metrics
  /jobs/<jobid>/metrics
  /jobs/<jobid>/vertices/<vertexid>/subtasks/<subtaskindex>
- 基于类型聚合指标:
  /taskmanagers/metrics
  /jobs/metrics
  /jobs/<jobid>/vertices/<vertexid>/subtasks/metrics
- 基于指定子集聚合指标:
  /taskmanagers/metrics?taskmanagers=A,B,C
  /jobs/metrics?jobs=D,E,F
  /jobs/<jobid>/vertices/<vertexid>/subtasks/metrics?subtask=1,2,3

Checkpoint 监控与调优

Checkpoint 实现原理

Checkpoint 监控指标

Checkpoint Counts

Triggered: The total number of checkpoints that have been triggered since the job started.

In Progress: The current number of checkpoints that are in progress.

Completed: The total number of successfully completed checkpoints since the job started.

Failed: The total number of failed checkpoints since the job started.

Restored: The number of restore operations since the job started. This also tells you how many times the job has restarted since submission. Note that the initial submission with a savepoint also counts as a restore and the count is reset if the JobManager was lost during operation.

Latest Completed Checkpoint: The latest successfully completed checkpoints. Clicking on More details gives you detailed statistics down to the subtask level.

Latest Failed Checkpoint: The latest failed checkpoint. Clicking on More details gives you detailed statistics down to the subtask level.

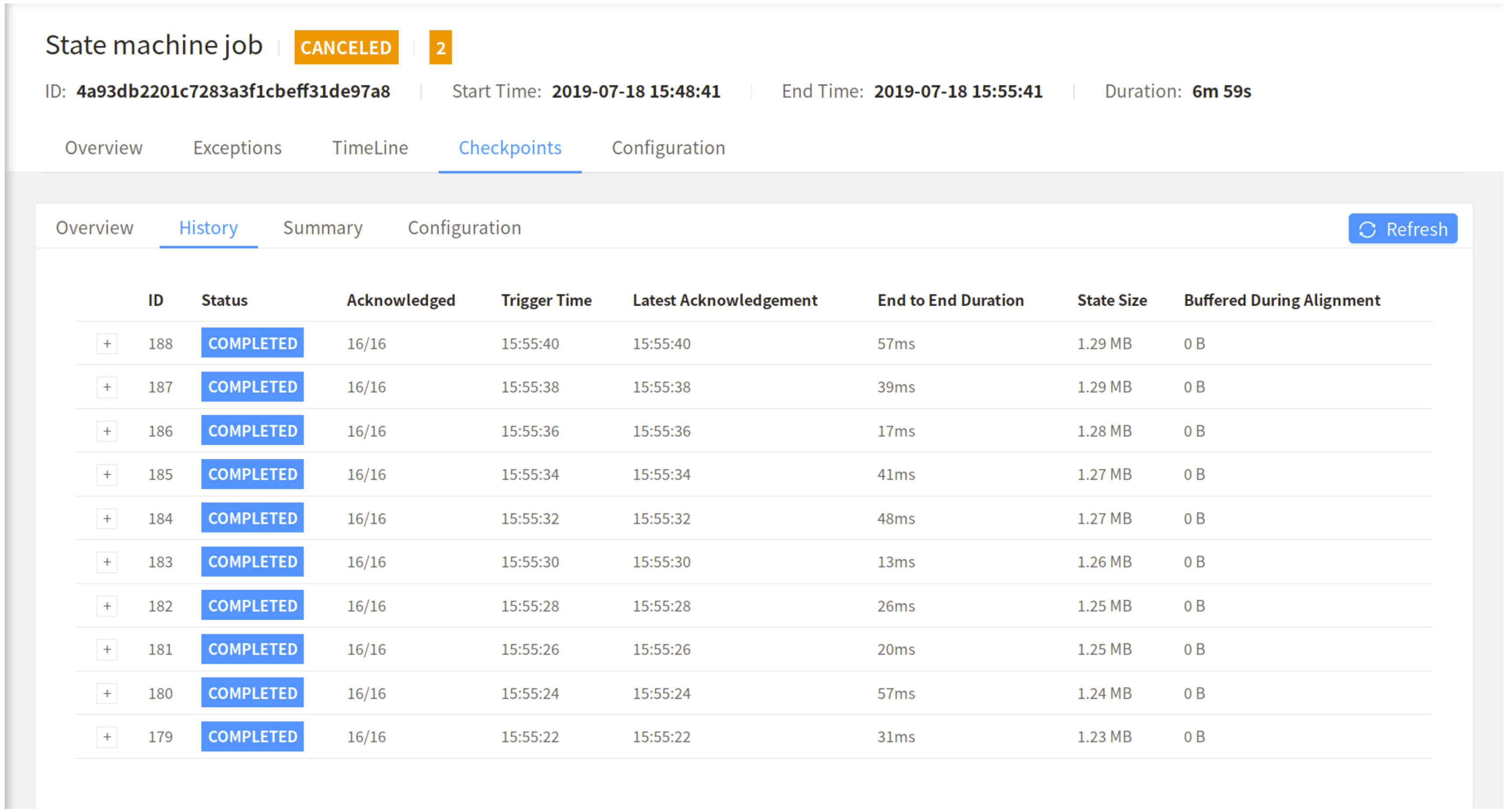
Latest Savepoint: The latest triggered savepoint with its external path. Clicking on More details gives you detailed statistics down to the subtask level.

Latest Restore: There are two types of restore operations.

Restore from Checkpoint: We restored from a regular periodic checkpoint.

Restore from Savepoint: We restored from a savepoint.

History Tab

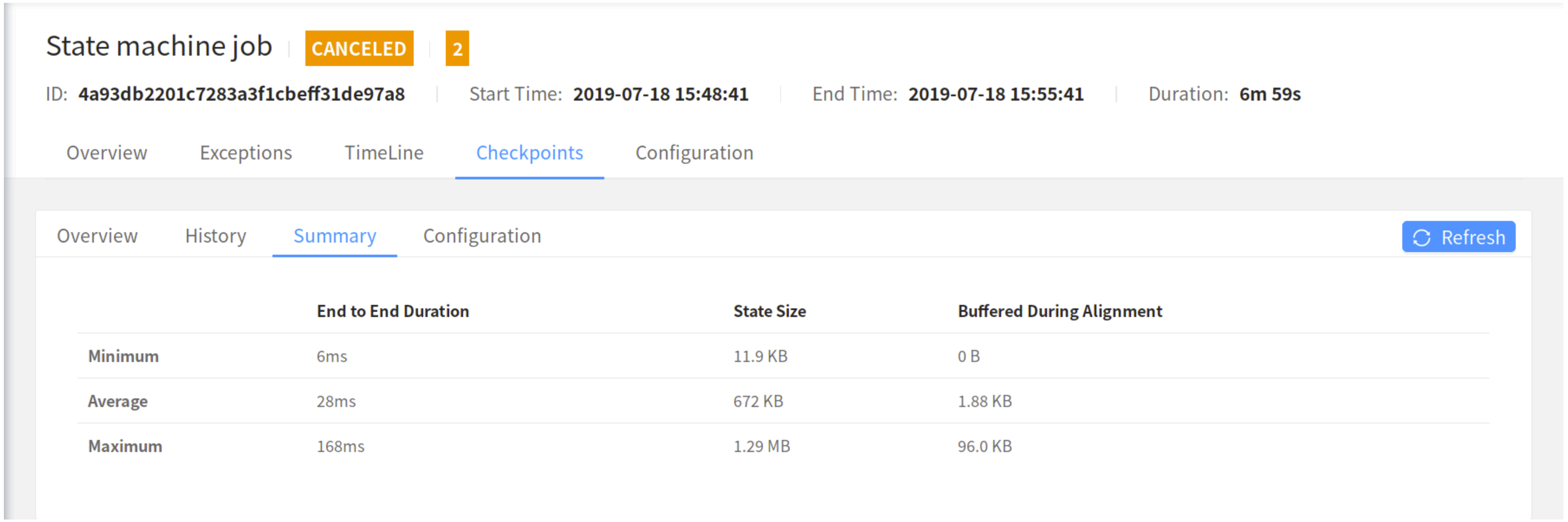


State machine job

ID: **4a93db2201c7283a3f1cbeff31de97a8** | Start Time: **2019-07-18 15:48:41** | End Time: **2019-07-18 15:55:41** | Duration: **6m 59s**

	ID	Status	Acknowledged	Trigger Time	Latest Acknowledgement	End to End Duration	State Size	Buffered During Alignment
+	188	COMPLETED	16/16	15:55:40	15:55:40	57ms	1.29 MB	0 B
+	187	COMPLETED	16/16	15:55:38	15:55:38	39ms	1.29 MB	0 B
+	186	COMPLETED	16/16	15:55:36	15:55:36	17ms	1.28 MB	0 B
+	185	COMPLETED	16/16	15:55:34	15:55:34	41ms	1.27 MB	0 B
+	184	COMPLETED	16/16	15:55:32	15:55:32	48ms	1.27 MB	0 B
+	183	COMPLETED	16/16	15:55:30	15:55:30	13ms	1.26 MB	0 B
+	182	COMPLETED	16/16	15:55:28	15:55:28	26ms	1.25 MB	0 B
+	181	COMPLETED	16/16	15:55:26	15:55:26	20ms	1.25 MB	0 B
+	180	COMPLETED	16/16	15:55:24	15:55:24	57ms	1.24 MB	0 B
+	179	COMPLETED	16/16	15:55:22	15:55:22	31ms	1.23 MB	0 B

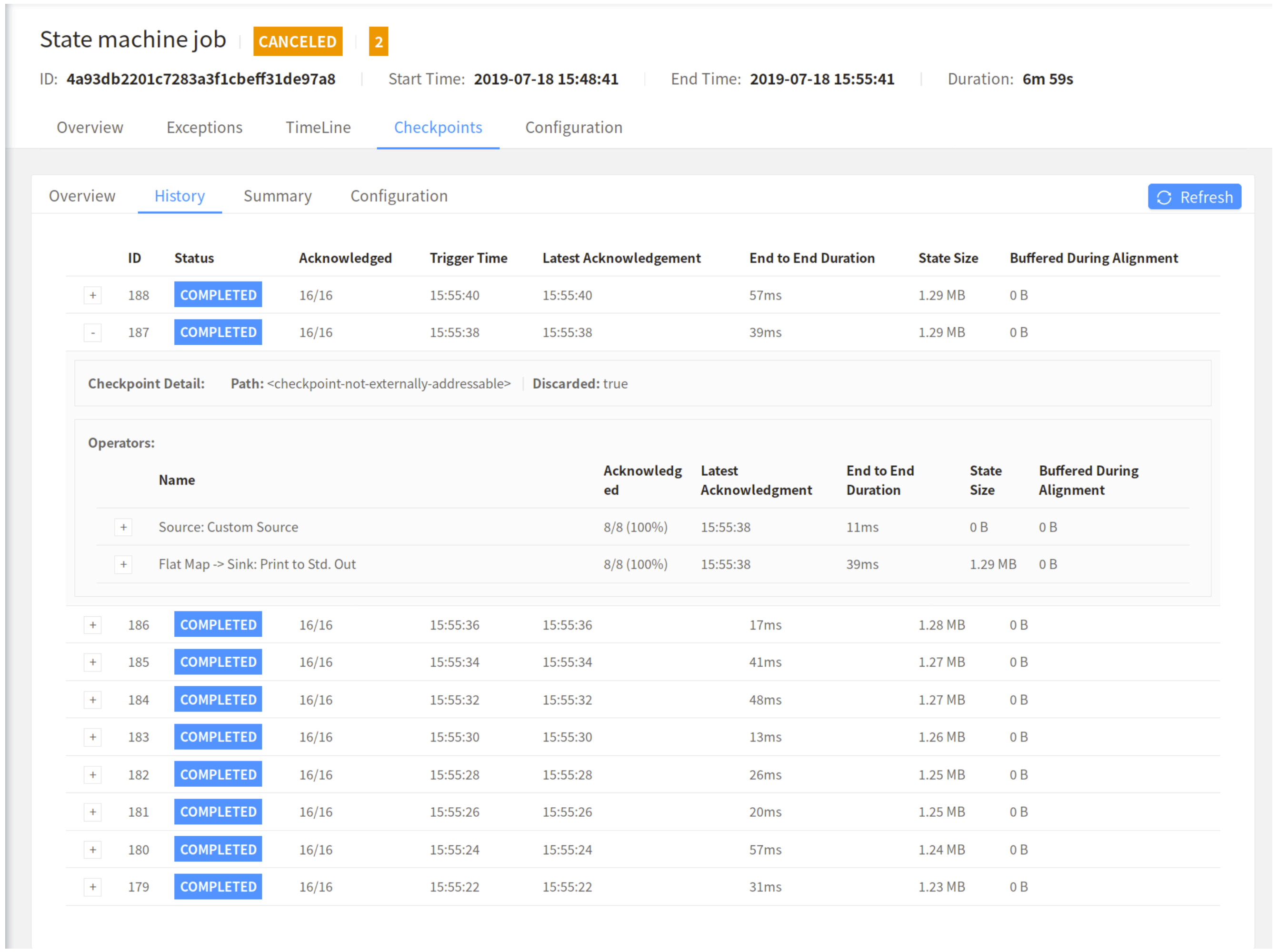
Summary Tab



ID: 4a93db2201c7283a3f1cbeff31de97a8 | Start Time: 2019-07-18 15:48:41 | End Time: 2019-07-18 15:55:41 | Duration: 6m 59s

	End to End Duration	State Size	Buffered During Alignment
Minimum	6ms	11.9 KB	0 B
Average	28ms	672 KB	1.88 KB
Maximum	168ms	1.29 MB	96.0 KB

Checkpoint Details



State machine job CANCELED 2

ID: 4a93db2201c7283a3f1cbeff31de97a8 | Start Time: 2019-07-18 15:48:41 | End Time: 2019-07-18 15:55:41 | Duration: 6m 59s

Overview | Exceptions | TimeLine | Checkpoints | Configuration

Overview | History | Summary | Configuration | Refresh

	ID	Status	Acknowledged	Trigger Time	Latest Acknowledgement	End to End Duration	State Size	Buffered During Alignment
+	188	COMPLETED	16/16	15:55:40	15:55:40	57ms	1.29 MB	0 B
-	187	COMPLETED	16/16	15:55:38	15:55:38	39ms	1.29 MB	0 B

Checkpoint Detail: **Path:** <checkpoint-not-externally-addressable> | **Discarded:** true

Operators:

	Name	Acknowledged	Latest Acknowledgment	End to End Duration	State Size	Buffered During Alignment
+	Source: Custom Source	8/8 (100%)	15:55:38	11ms	0 B	0 B
+	Flat Map -> Sink: Print to Std. Out	8/8 (100%)	15:55:38	39ms	1.29 MB	0 B

	ID	Status	Acknowledged	Trigger Time	Latest Acknowledgement	End to End Duration	State Size	Buffered During Alignment
+	186	COMPLETED	16/16	15:55:36	15:55:36	17ms	1.28 MB	0 B
+	185	COMPLETED	16/16	15:55:34	15:55:34	41ms	1.27 MB	0 B
+	184	COMPLETED	16/16	15:55:32	15:55:32	48ms	1.27 MB	0 B
+	183	COMPLETED	16/16	15:55:30	15:55:30	13ms	1.26 MB	0 B
+	182	COMPLETED	16/16	15:55:28	15:55:28	26ms	1.25 MB	0 B
+	181	COMPLETED	16/16	15:55:26	15:55:26	20ms	1.25 MB	0 B
+	180	COMPLETED	16/16	15:55:24	15:55:24	57ms	1.24 MB	0 B
+	179	COMPLETED	16/16	15:55:22	15:55:22	31ms	1.23 MB	0 B

Configuration Tab

Checkpointing Mode: Either Exactly Once or At least Once.

Interval: The configured checkpointing interval. Trigger checkpoints in this interval.

Timeout: Timeout after which a checkpoint is cancelled by the JobManager and a new checkpoint is triggered.

Minimum Pause Between Checkpoints: Minimum required pause between checkpoints. After a checkpoint has completed successfully, we wait at least for this amount of time before triggering the next one, potentially delaying the regular interval.

Maximum Concurrent Checkpoints: The maximum number of checkpoints that can be in progress concurrently.

Persist Checkpoints Externally: Enabled or Disabled. If enabled, furthermore lists the cleanup config for externalized checkpoints (delete or retain on cancellation).

极客时间

Checkpoint 配置参数

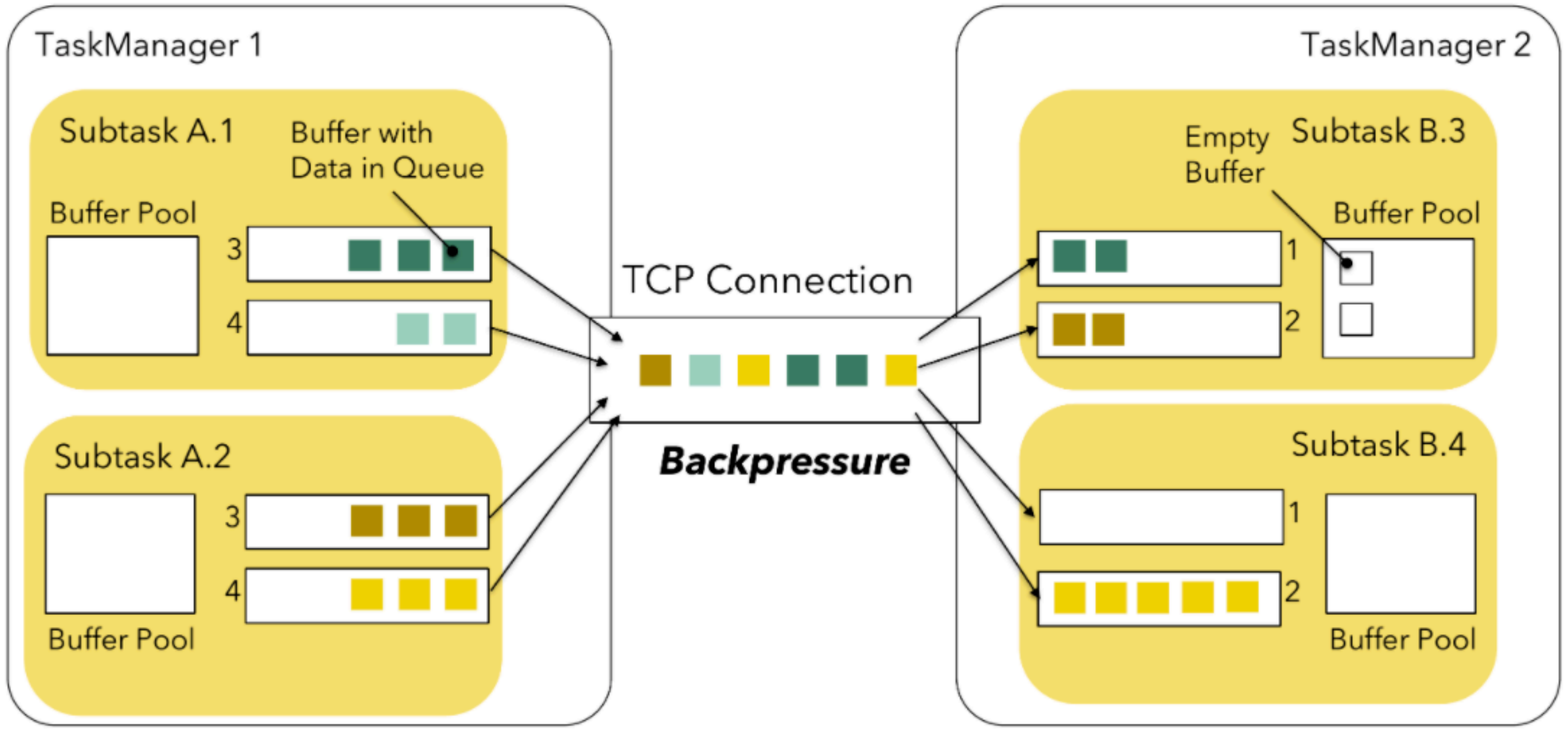
Key	Default	Type	Description
state.backend	(none)	String	StateBackend 类型
state.checkpoints.dir	(none)	String	Checkpoint数据持久化路径
state.savepoints.dir	(none)	String	Savepoint数据持久化路径
state.backend.incremental	FALSE	Boolean	是否增量Checkpoint
state.backend.local-recovery	FALSE	Boolean	是否支持本地恢复State，仅对keyed state backends有效，MemoryStateBackend不生效
state.checkpoints.num-retained	1	Integer	最大completed checkpoint保留个数
taskmanager.state.local.root-dirs	(none)	String	本地恢复需要指定的根路径

计时器（内存 vs. RocksDB）

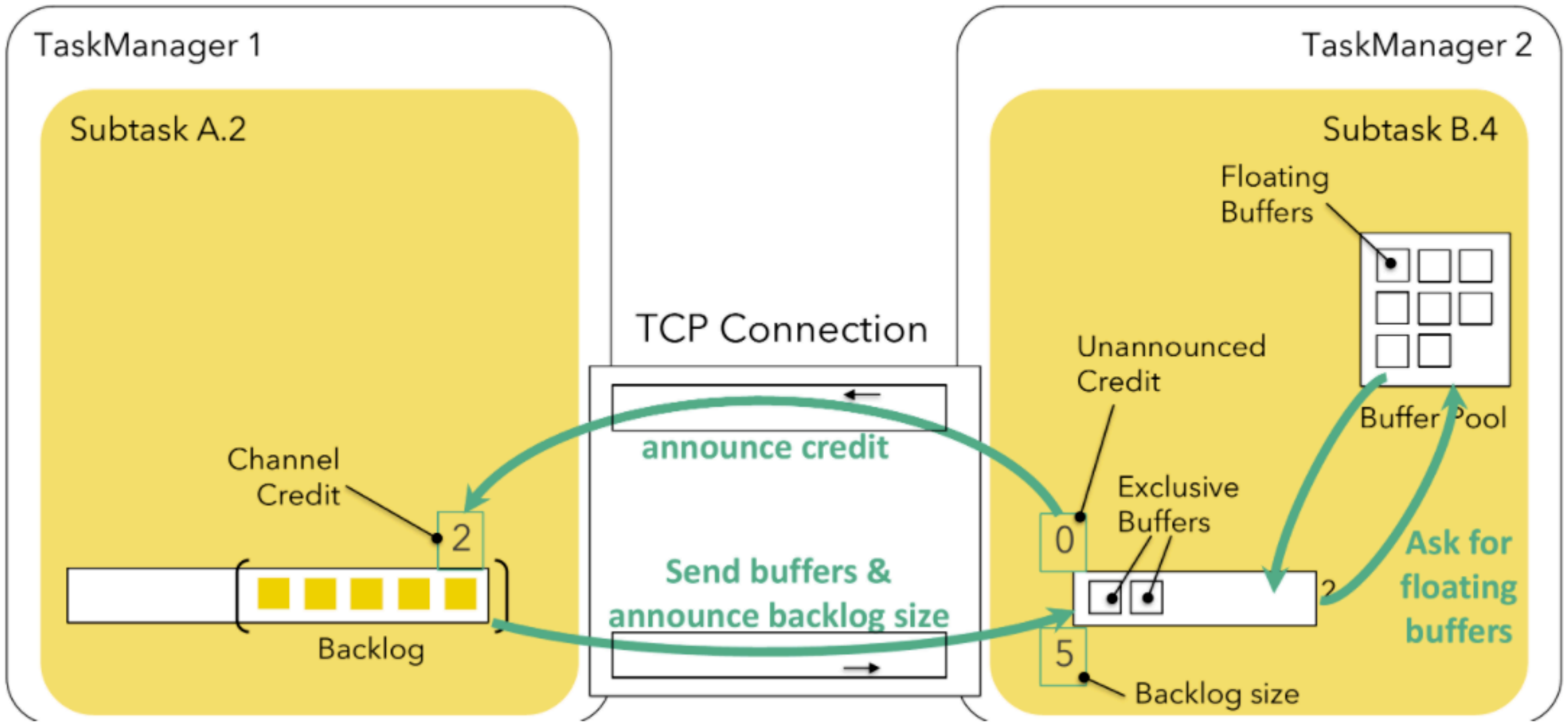
- 计时器（Timer）用于安排稍后的操作（基于事件时间或处理时间），例如触发窗口或回调 ProcessFunction。
- 当选择 RocksDBStateBackend 时，默认情况下计时器也存储在 RocksDB 中。这是一种健壮且可扩展的方式，允许应用程序使用很多个计时器。另一方面，在 RocksDB 中维护计时器会有一定的成本，因此 Flink 也提供了将计时器存储在 JVM 堆上而使用 RocksDB 存储其他状态的选项。
- 当计时器数量较少时，基于堆的计时器可以有更好的性能。
- 您可以通过将 state.backend.rocksdb.timer-service.factory 配置项设置为 heap（而不是默认的 rocksdb）来将计时器存储在堆上。

反压监控与原理

TCP 自带反压的局限性

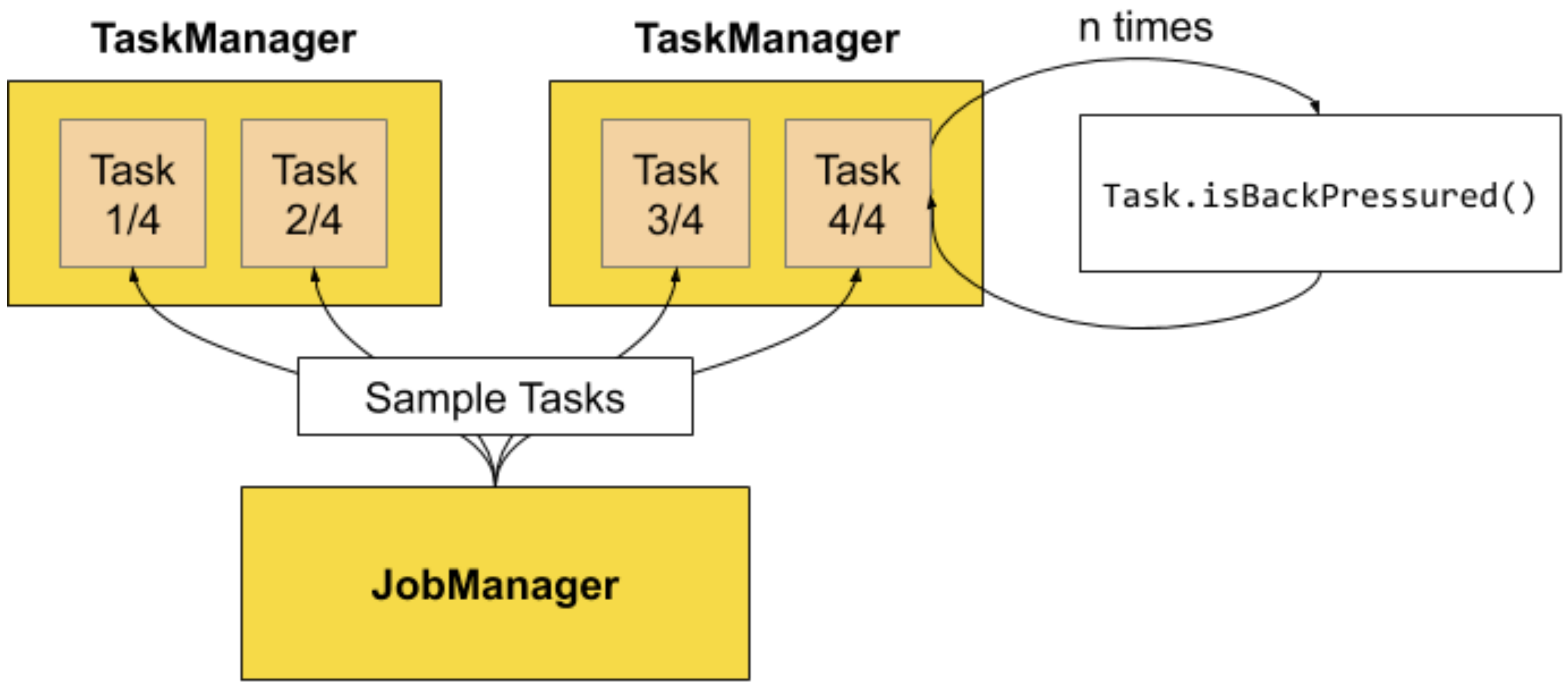


基于 Credit 反压机制

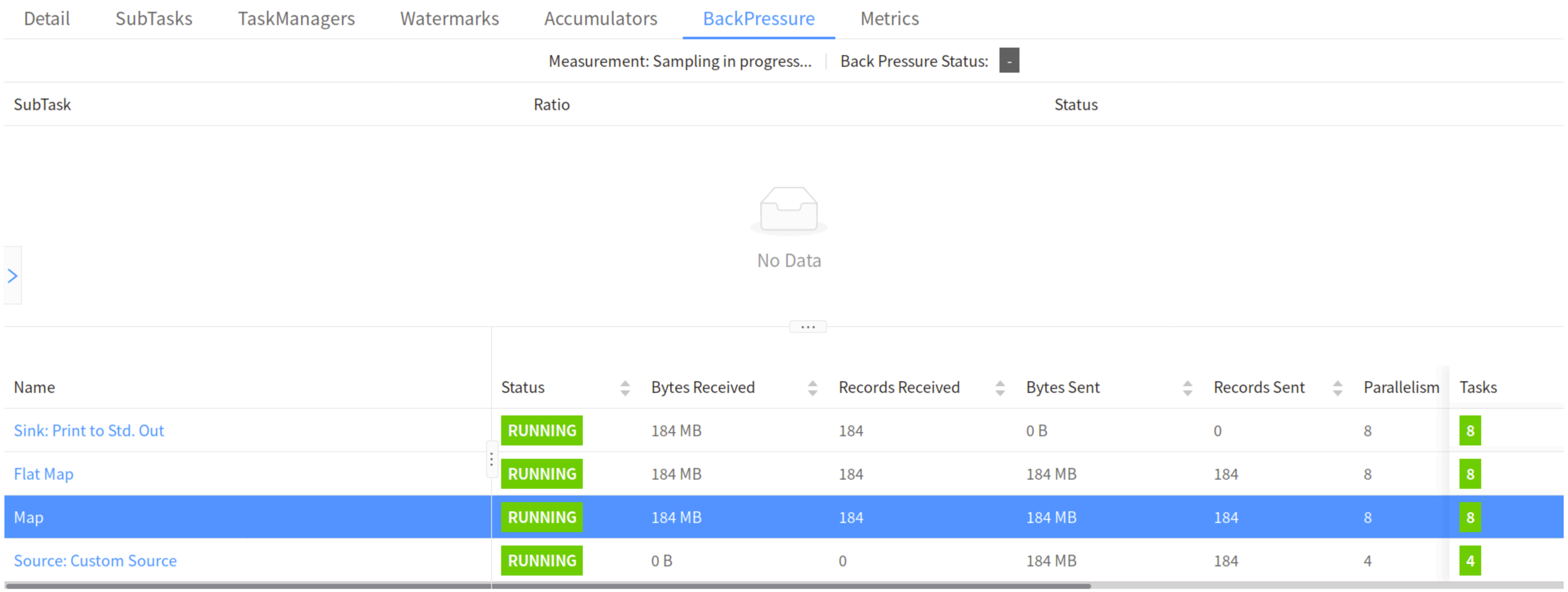


反压采样

OK: 0 <= 比例 <= 0.10
LOW: 0.10 < 比例 <= 0.5
HIGH: 0.5 < 比例 <= 1



反压采样进行中



Detail SubTasks TaskManagers Watermarks Accumulators BackPressure Metrics

Measurement: Sampling in progress... | Back Pressure Status: -

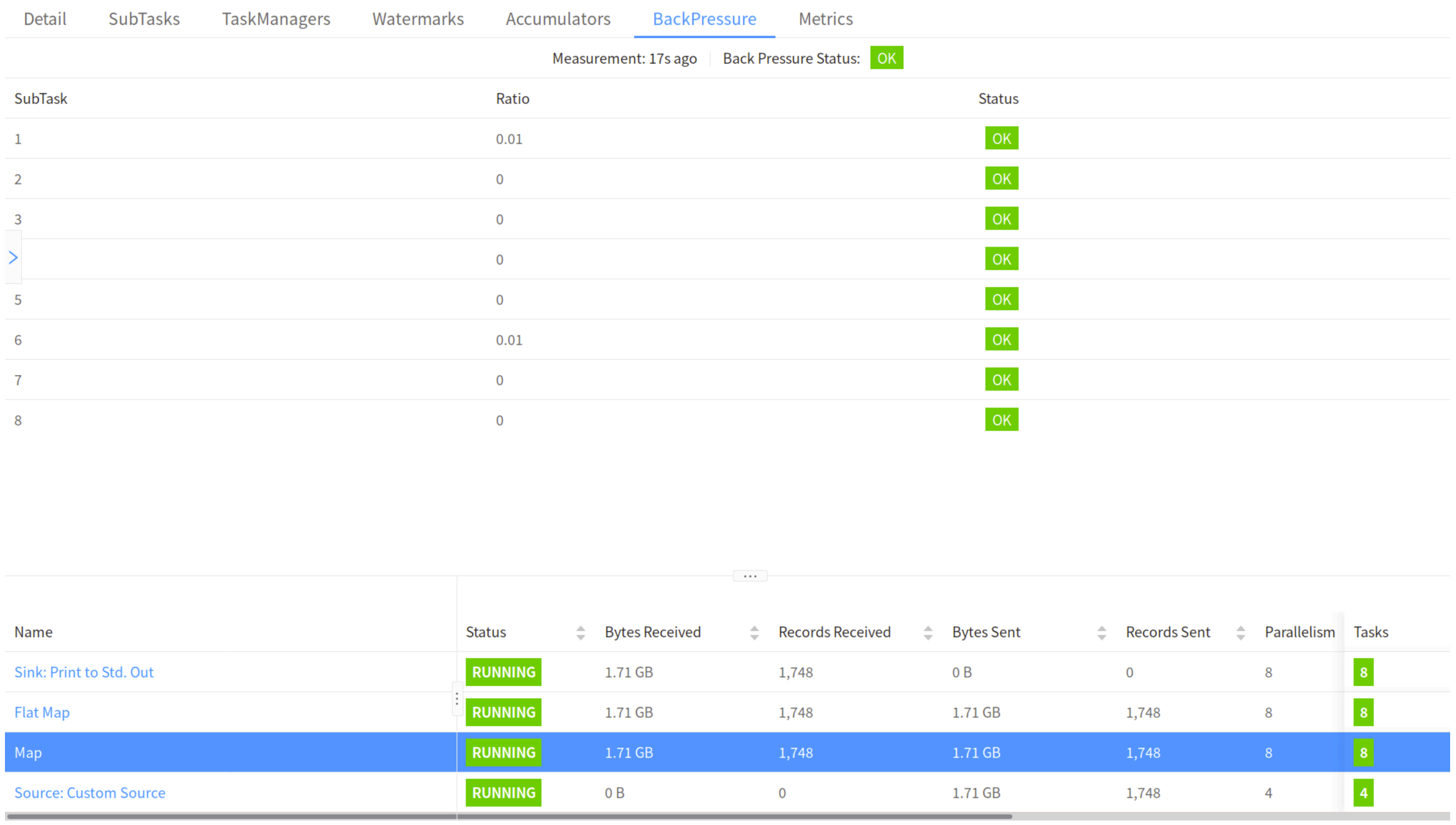
SubTask	Ratio	Status

No Data

Name	Status	Bytes Received	Records Received	Bytes Sent	Records Sent	Parallelism	Tasks
Sink: Print to Std. Out	RUNNING	184 MB	184	0 B	0	8	8
Flat Map	RUNNING	184 MB	184	184 MB	184	8	8
Map	RUNNING	184 MB	184	184 MB	184	8	8
Source: Custom Source	RUNNING	0 B	0	184 MB	184	4	4

极客时间

没有出现反压情况



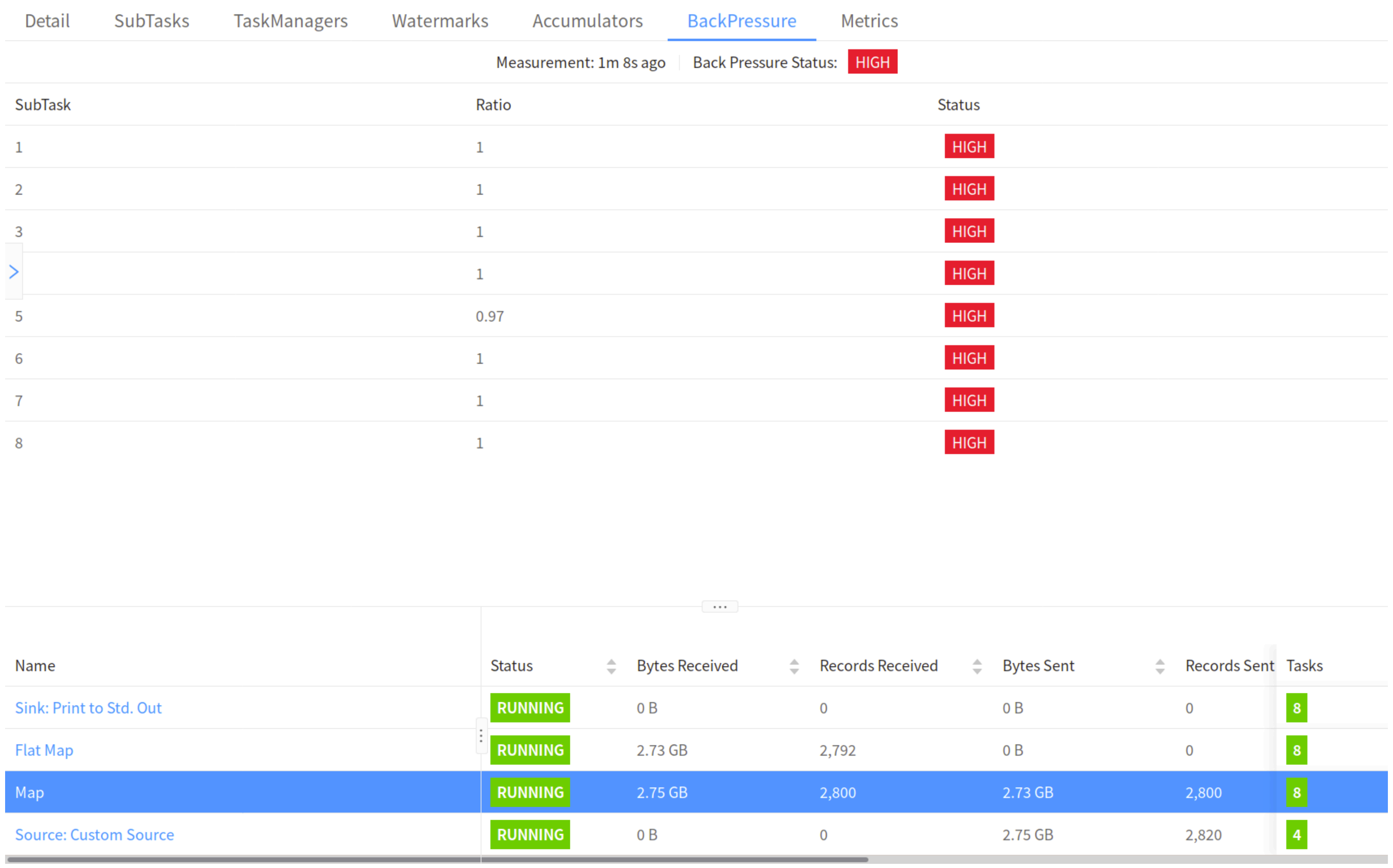
Detail | SubTasks | TaskManagers | Watermarks | Accumulators | BackPressure | Metrics

Measurement: 17s ago | Back Pressure Status: OK

SubTask	Ratio	Status
1	0.01	OK
2	0	OK
3	0	OK
>	0	OK
5	0	OK
6	0.01	OK
7	0	OK
8	0	OK

Name	Status	Bytes Received	Records Received	Bytes Sent	Records Sent	Parallelism	Tasks
Sink: Print to Std. Out	RUNNING	1.71 GB	1,748	0 B	0	8	8
Flat Map	RUNNING	1.71 GB	1,748	1.71 GB	1,748	8	8
Map	RUNNING	1.71 GB	1,748	1.71 GB	1,748	8	8
Source: Custom Source	RUNNING	0 B	0	1.71 GB	1,748	4	4

出现反压情况



Detail | SubTasks | TaskManagers | Watermarks | Accumulators | BackPressure | Metrics

Measurement: 1m 8s ago | Back Pressure Status: HIGH

SubTask	Ratio	Status
1	1	HIGH
2	1	HIGH
3	1	HIGH
	1	HIGH
5	0.97	HIGH
6	1	HIGH
7	1	HIGH
8	1	HIGH

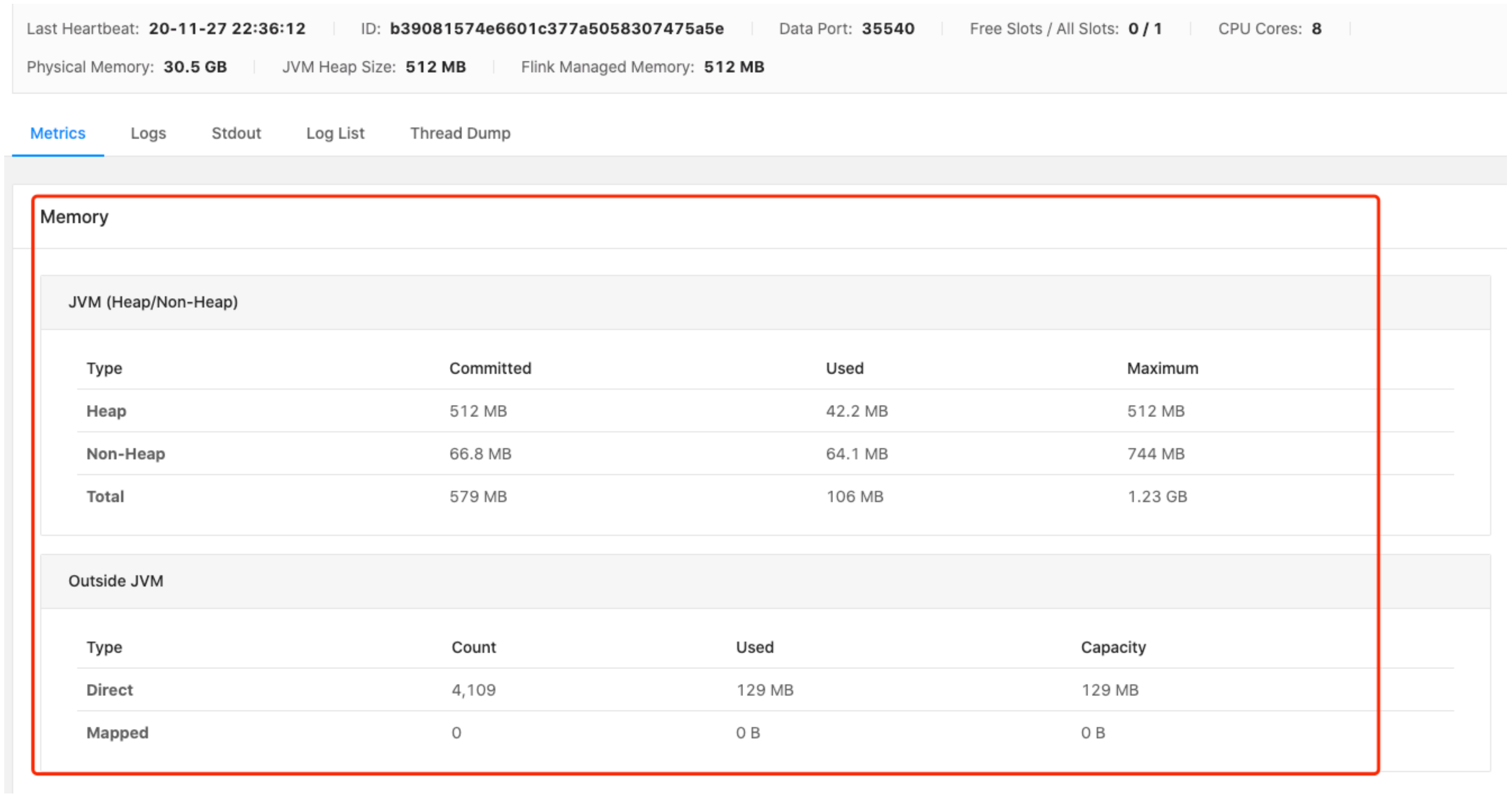
Name	Status	Bytes Received	Records Received	Bytes Sent	Records Sent	Tasks
Sink: Print to Std. Out	RUNNING	0 B	0	0 B	0	8
Flat Map	RUNNING	2.73 GB	2,792	0 B	0	8
Map	RUNNING	2.75 GB	2,800	2.73 GB	2,800	8
Source: Custom Source	RUNNING	0 B	0	2.75 GB	2,820	4

反压参数配置

- web.backpressure.refresh-interval:
  - 有效的反压结果被废弃并重新进行采样的时间 (默认: 60000, 1 min)。
- web.backpressure.num-samples:
  - 用于确定反压采样的样本数 (默认: 100)。
- web.backpressure.delay-between-samples:
  - 用于确定反压采样的间隔时间 (默认: 50, 50 ms)。

Flink 内存配置与调优

TaskManager 内存指标监控



Last Heartbeat: **20-11-27 22:36:12** | ID: **b39081574e6601c377a5058307475a5e** | Data Port: **35540** | Free Slots / All Slots: **0 / 1** | CPU Cores: **8**

Physical Memory: **30.5 GB** | JVM Heap Size: **512 MB** | Flink Managed Memory: **512 MB**

Memory

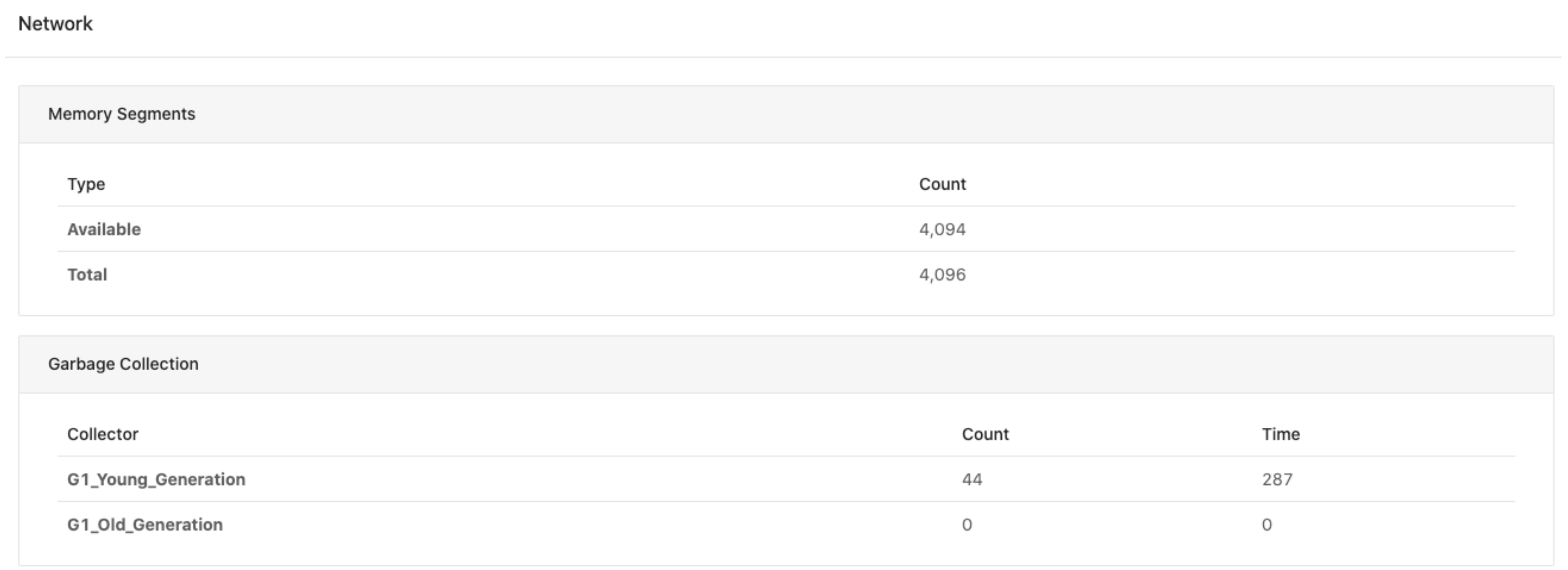
JVM (Heap/Non-Heap)

Type	Committed	Used	Maximum
Heap	512 MB	42.2 MB	512 MB
Non-Heap	66.8 MB	64.1 MB	744 MB
Total	579 MB	106 MB	1.23 GB

Outside JVM

Type	Count	Used	Capacity
Direct	4,109	129 MB	129 MB
Mapped	0	0 B	0 B

TaskManager 内存指标监控


Network

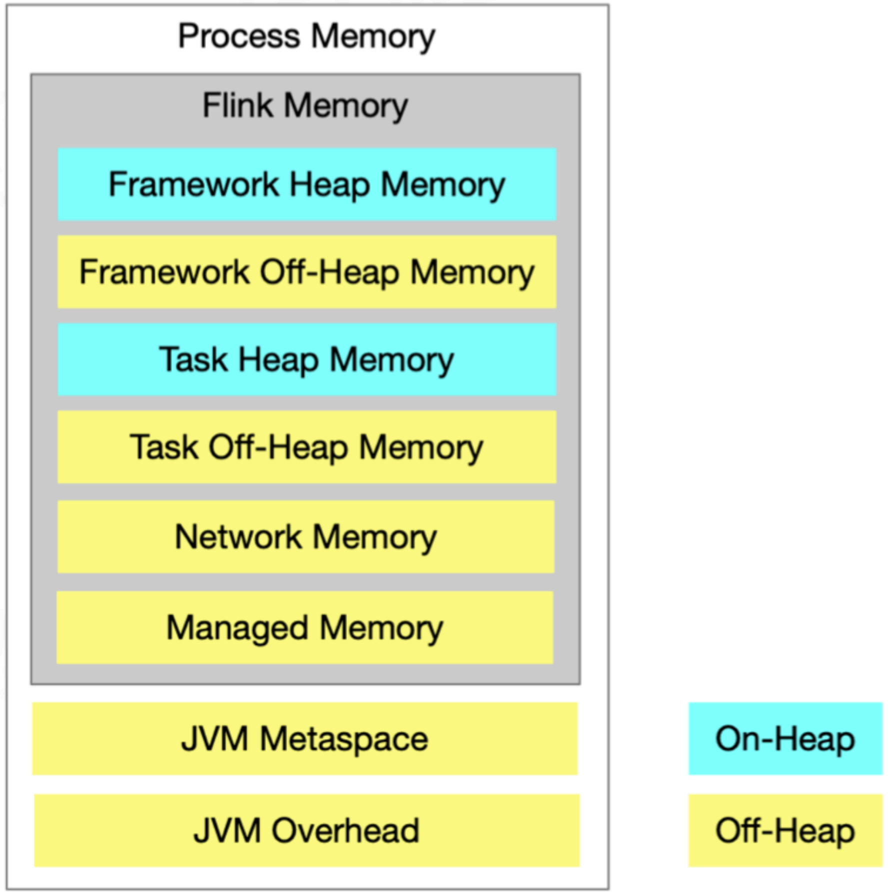
Memory Segments

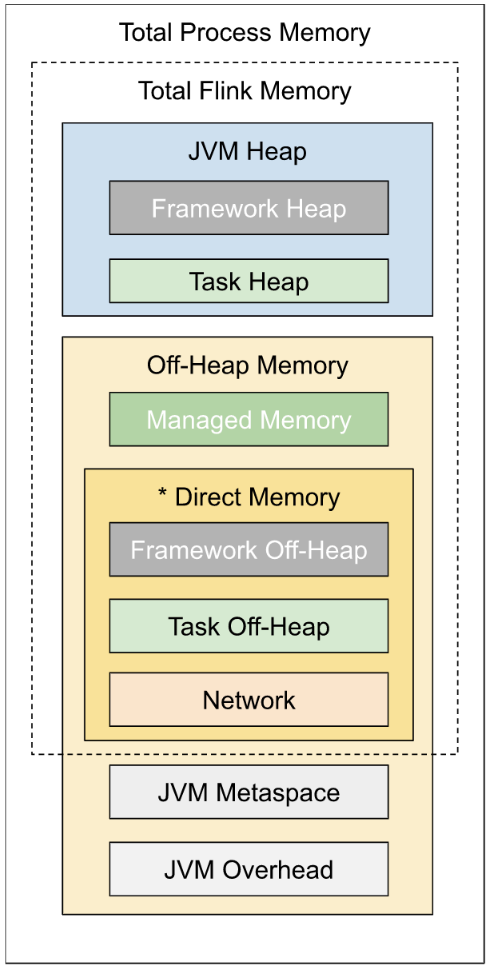
Type	Count
Available	4,094
Total	4,096

Garbage Collection

Collector	Count	Time
G1_Young_Generation	44	287
G1_Old_Generation	0	0

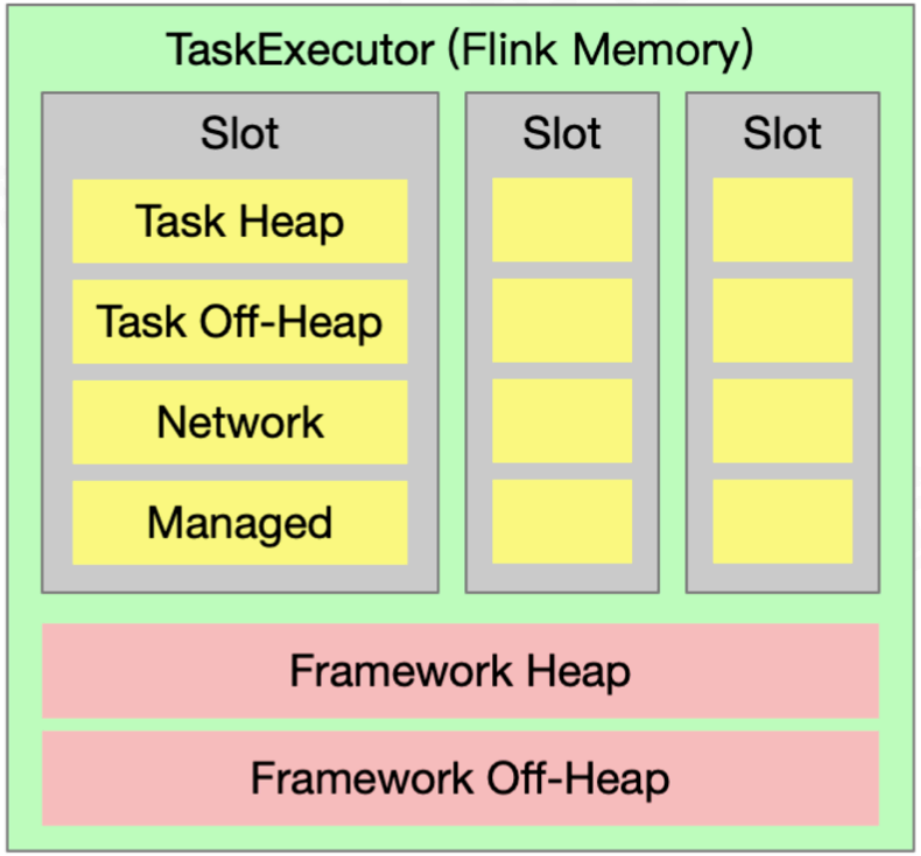
TaskManager 内存模型





Framework vs Task Memory

- 区别：是否计入 Slot 资源
- 总用量受限：
  - -Xmx = Framework Heap + Task Heap
  - -XX:MaxDirectMemorySize= Framewok Off-Heap + Task Off-Heap
- 无隔离
  - 后续社区会实现动态资源隔离（flip-56）



Heap VS Off-Heap Memory

- Heap
  - 堆内存，Java 对象数据
  - HeapStateBackend
- Off-Heap
  - Direct
    - DirectByteBuffer
      - ByteBuffer.allocateDirect()
    - MappedBytebuffer
      - FileChannel.map()
  - Native
    - JNI，C/C++，Python
  - 不区分 Direct 和 Native

Network Memory

- Direct Memory
- 主要用于网络数据传输
- 特点：
  - TaskManager 的各个 Slot 之间 没有隔离
  - 根据作业的拓扑确定 Network Memory
  - 主要决定于 Buffer数量

Managed Memory

- Native Memory 类型
- 主要用于
  - RocksDBStateBackend
  - Batch Operator
- 特点：
  - 同一 TaskExecutor 的各个 Slot 之间严格隔离
  - 多点少点都能跑，与性能挂钩

- RocksDB 内存限制
  - state.backend.rocksdb.memory.managed(default:true)
  - 设定RocksDB使用内存为Managed Memory 大小
  - 目的：防止容器内存超限
  - Standalone 可关闭限制

JVM Metaspace & Overhead

- JVM Metaspace
  - 存放 JVM 加载类的元数据
  - 加载的类越多需要的内存空间越大

- JVM Overhead
  - Native Memory
  - 用于其他 JVM 内存开销
    - Code Cache
    - Thread Stack

Flink 内存模型

组成部分	配置参数	描述
框架堆内存（Framework Heap Memory）	taskmanager.memory.framework.heap.size	用于 Flink 框架的 JVM 堆内存（进阶配置）。
任务堆内存（Task Heap Memory）	taskmanager.memory.task.heap.size	用于 Flink 应用的算子及用户代码的 JVM 堆内存。
托管内存（Managed memory）	taskmanager.memory.managed.size taskmanager.memory.managed.fraction	由 Flink 管理的用于排序、哈希表、缓存中间结果及 RocksDB State Backend 的本地内存。
框架堆外内存（Framework Off-heap Memory）	taskmanager.memory.framework.off-heap.size	用于 Flink 框架的堆外内存（直接内存或本地内存）（进阶配置）。
任务堆外内存（Task Off-heap Memory）	taskmanager.memory.task.off-heap.size	用于 Flink 应用的算计及用户代码的堆外内存（直接内存或本地内存）。
网络内存（Network Memory）	taskmanager.memory.network.min taskmanager.memory.network.max taskmanager.memory.network.fraction	用于任务之间数据传输的直接内存（例如网络传输缓冲）。该内存部分为基于 Flink总内存的受限的等比内存部分。
JVM Metaspace	taskmanager.memory.jvm-metaspace.size	Flink JVM 进程的 Metaspace。
JVM 开销	taskmanager.memory.jvm-overhead.min taskmanager.memory.jvm-overhead.max taskmanager.memory.jvm-overhead.fraction	用于其他 JVM 开销的本地内存，例如栈空间、垃圾回收空间等。该内存部分为基于进程总内存的受限的等比内存部分。