Performance Analysis of Large-Scale Online Data Processing Applications like Apache Spark

POLYTECHNIQUE MONTRÉAL

UNIVERSITÉ D'INGÉNIERIE

Reza Rouh

December 7th, 2023

TABLE OF CONTENTS

Introduction/ Tools

Apache Spark

Apache Hadoop

Apache Flink

Apache Storm

Apache Kafka

Apache Beam

Introduction/ Why spark?

Speed

Fast for both batch and interactive queries

Ease of Use

Consistent APIs in Python, Java, Scala, and R

Unified Engine

Combine SQL, streaming, and complex analytics

In-Memory Processing

The RDD, allows for in-memory processing

Fault Tolerance

Can recover the lost data automatically

ML Libraries

Spark's MLlib offers a powerful set of ML

Introduction/ Apache tools logging system

	Spark	Hadoop	Flink	Storm	Kafka	Beam
Support log4j ?	✓	✓	✓	✓	✓	✓

They share similar logging mechanisms!

Introduction/Log4j

Type

Logging framework for Java logging ecosystem

Functionality

ERROR, WARN, INFO, DEBUG, TRACE

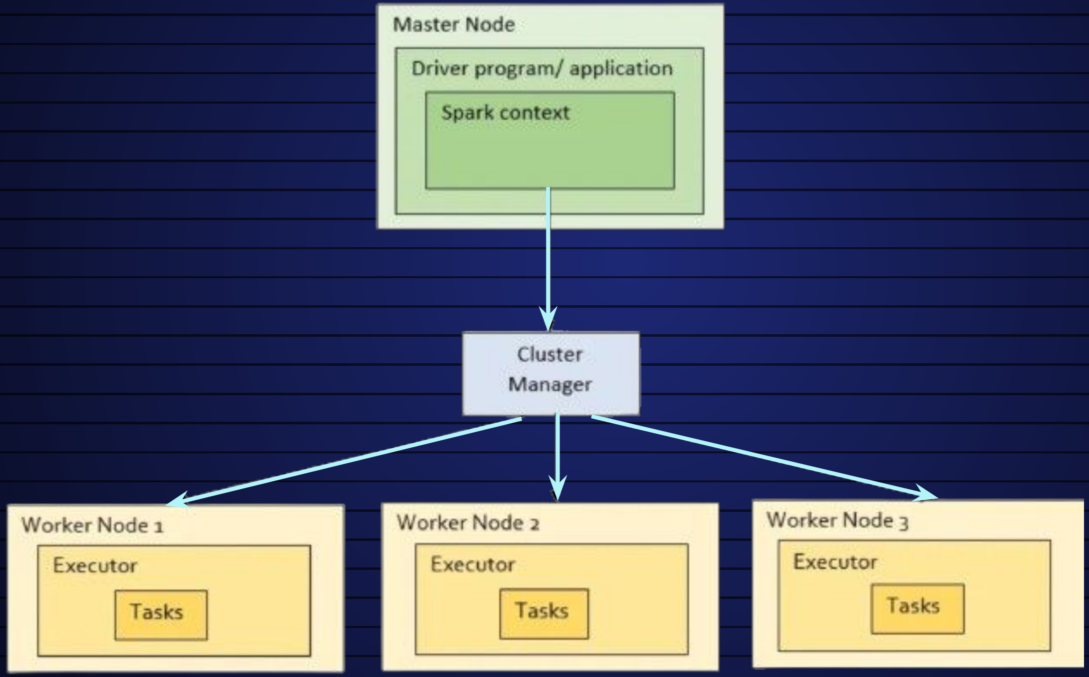
Customization

Allows detailed customization

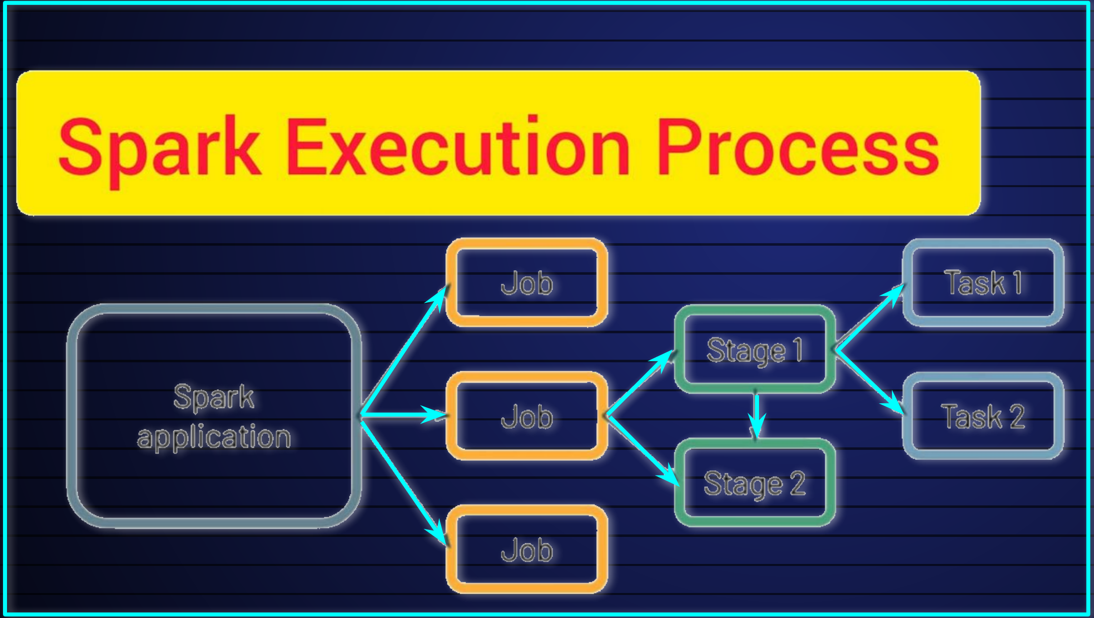
Versions

Log4j 1.x and Log4j 2.x

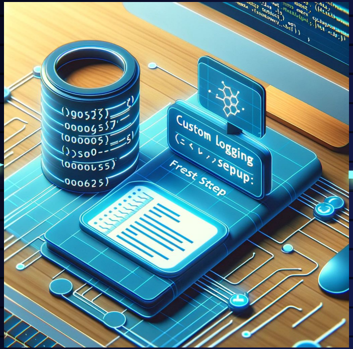
Introduction/ Spark Architecture

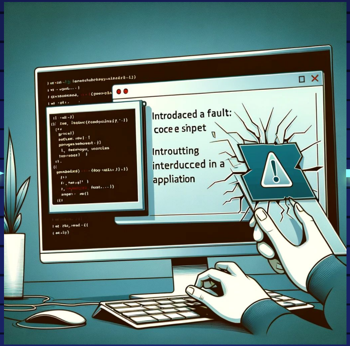


Introduction/ Spark Arcitecture



Objectives and Scope





Set up

Logging During Execution

Analyze Traces

Methodology

Installing Tools

Spark, Lttng, Trace compass, …

Configuration

LTTng sessions, choose ust and kernel events, spark setting

Get logs

From spark logs, custom logs, listeners, …

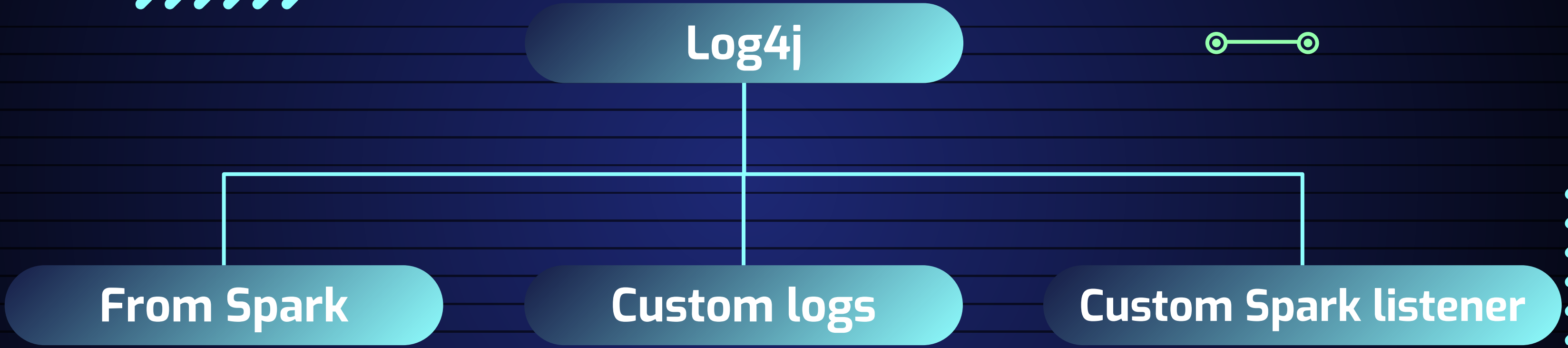
Running examples

Execution of Normal and Abnormal Behavior in Spark

Analyze and Visualization

Find fault reasons and visualize in trace compass

Methodology/ capturing logs



Methodology/ capturing logs

1) Log4j from Spark

1. Adding log4j2.xml
2. Create a pattern for logs in Console
3. There is no pattern layout setting here from lttng
4. Add Thread_ID to LTTNG UST source code
5. Enable log4j logs and run Spark example

```xml
<?xml version="1.0" encoding="UTF-8"?>
<Configuration status="WARN" strict="true">
 <Appenders>
   <Lttng name="Lttng1" domain="LOG4J">
   </Lttng>
   <Lttng name="Lttng2" domain="LOG4J2">
   </Lttng>
 </Appenders>
 <Loggers>
   <Root level="INFO">
     <AppenderRef ref="Lttng1"/>
     <AppenderRef ref="Lttng2"/>
   </Root>
 </Loggers>
</Configuration>
```

Methodology/ a little change in LTTNG-UST

Logs result in lttng session

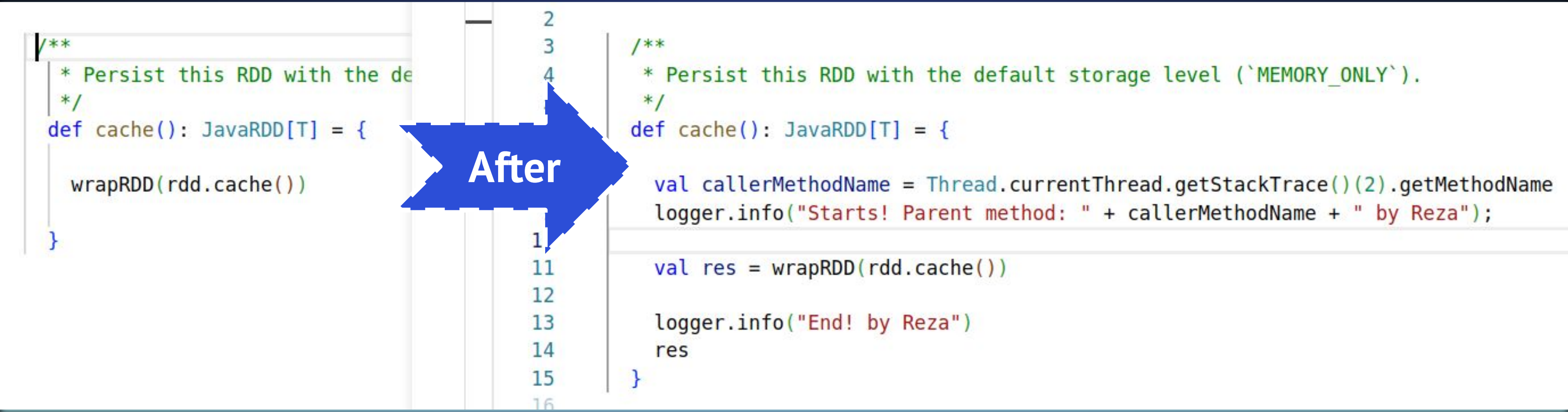
LTTNG UST add new code to LttngLogAppender class

```
[19:51:23.497752838] (+0.018001959) Rezghool
lttng_log4j:event: { cpu_id = 7 },
{ msg = "Starts! Parent method: main by Reza",
logger_name = "org.apache.spark.sql.SparkSession",
class_name = "org.apache.spark.sql.SparkSession",
method_name = "read", filename = "SparkSession.scala",
line_number = 725, timestamp = 1700268683497,
int_loglevel = 20000,
thread_name = "main +
1 +
null +
org.apache.spark.sql.SparkSession.read(SparkSession.scala:725) +
5 +
null + org.apache.logging.slf4j.Log4jLogger" }
```

```
          event.getThreadName() +
" + " + event.getMarker()  +
" + " + event.getSource() +
" + " + event.getThreadPriority() +
" + " + event.getThrown() +
" + " + event.getLoggerFqcn()
```

Methodology/ capturing logs

2) Custom logs



```
/**
 * Persist this RDD with the de
 */
def cache(): JavaRDD[T] = {

  wrapRDD(rdd.cache())

}
```

```
/**
 * Persist this RDD with the default storage level (`MEMORY_ONLY`).
 */
def cache(): JavaRDD[T] = {

  val callerMethodName = Thread.currentThread.getStackTrace()(2).getMethodName
  logger.info("Starts! Parent method: " + callerMethodName + " by Reza");

  val res = wrapRDD(rdd.cache())

  logger.info("End! by Reza")
  res
}
```

Methodology/ capturing logs

2) Custom logs Result:

```
lttng_log4j:event: { cpu_id = 7 },
{ msg = "Starts! Parent method: main by Reza",
logger_name = "org.apache.spark.sql.SparkSession",
class_name = "org.apache.spark.sql.SparkSession",
method_name = "read", filename = "SparkSession.scala",
line_number = 725, timestamp = 1700268683497,
int_loglevel = 20000,
thread_name = "main +
1 +
null +
org.apache.spark.sql.SparkSession.read(SparkSession.scala:725) +
5 +
null + org.apache.logging.slf4j.Log4jLogger" }
```

Methodology/ capturing logs

3) Logs result in lttng session

```
msg = "Task end failed with error reason: Task
end info
- Stage ID: 35, Task ID: 65,
Executor ID: driver,
Duration: 93,
Task end reason::
ExceptionFailure(java.lang.NegativeArraySizeExcep
tion,
-727379968,[Ljava.lang.StackTraceElement;@5ed6f37
1,java.lang.NegativeArraySizeException:
-727379968
```

3) New Spark Listener

```
public class MyCustomSparkListener extends SparkListener {}
```

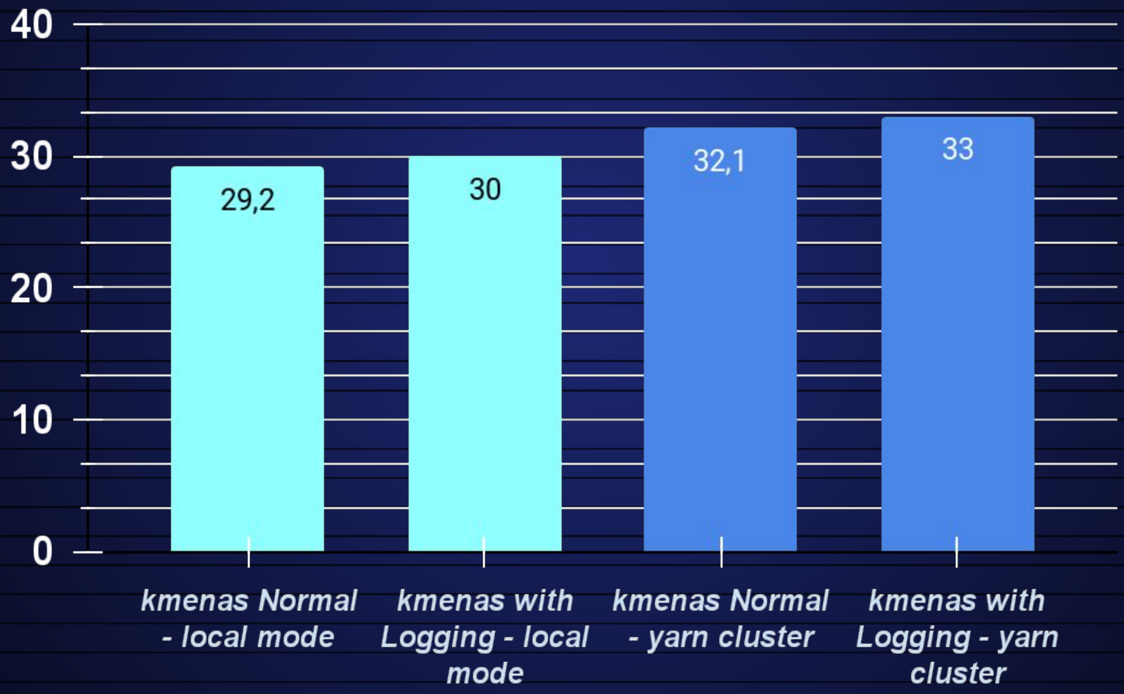
```
public void onJobEnd(SparkListenerJobEnd jobEnd) {
     String jobInfo = jobEnd.jobId() + ", Result: " + jobEnd.jobResult();
     if (!jobEnd.jobResult().toString().equals("JobSucceeded")) {
        logger.error("jobs end failed with error with Job ID: " + jobInfo);
     } else {
        logger.info("Job ended successfully with Job ID: " + jobInfo);
     }

  }
```

Key Findings/ Running the Spark example

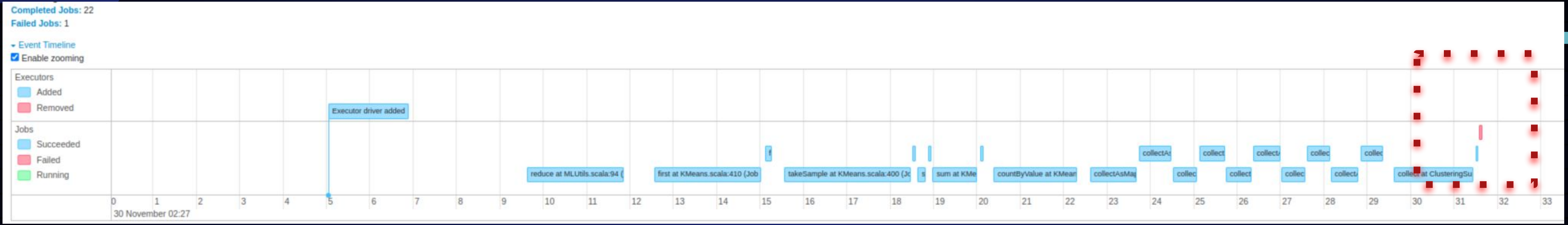
1. Using kmeans ML example with a 10GB data input
2. Restricting Acphe resources
3. Input some other bottleneck in the code
4. Compare time overhead

spark.executor.memory	512M
total.executor.cores	1
spark.driver.memory	512M
Spark.yarn.am.memory	512m

Key Findings/ Running the Spark example

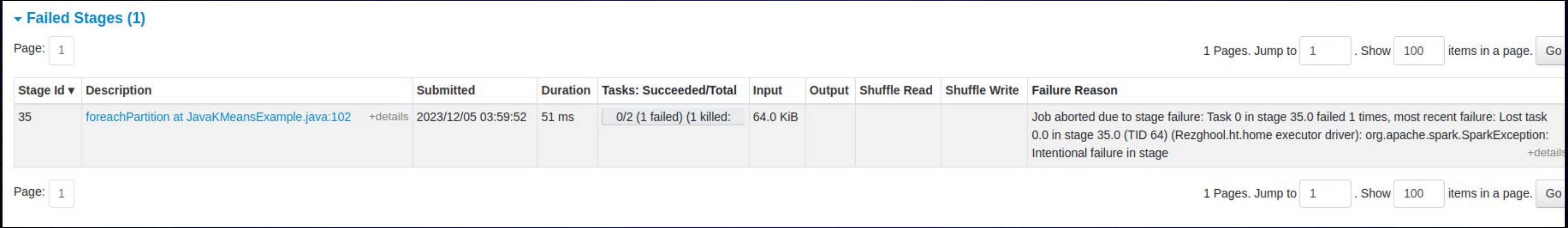


Key Findings/ Showing the fault in Spark UI



Completed Jobs: 22
Failed Jobs: 1

Event Timeline
☑ Enable zooming

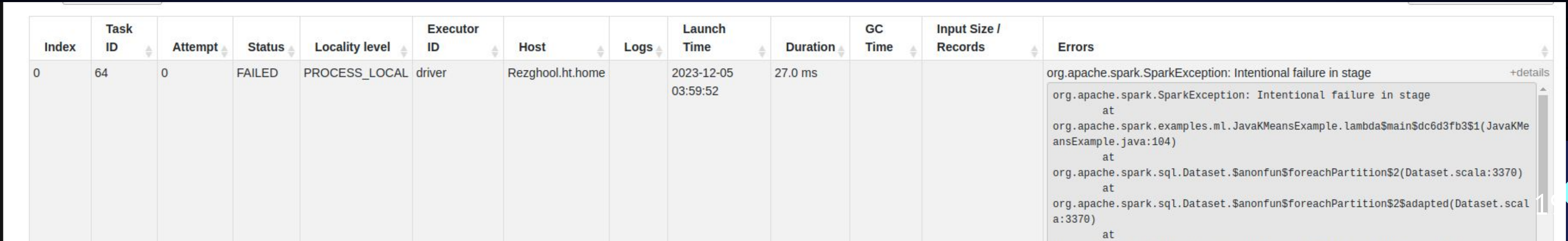


Failed Stages (1)

Page: 1 — 1 Pages. Jump to 1 . Show 100 items in a page. Go

Stage Id ▾	Description	Submitted	Duration	Tasks: Succeeded/Total	Input	Output	Shuffle Read	Shuffle Write	Failure Reason
35	foreachPartition at JavaKMeansExample.java:102 +details	2023/12/05 03:59:52	51 ms	0/2 (1 failed) (1 killed:	64.0 KiB				Job aborted due to stage failure: Task 0 in stage 35.0 failed 1 times, most recent failure: Lost task 0.0 in stage 35.0 (TID 64) (Rezghool.ht.home executor driver): org.apache.spark.SparkException: Intentional failure in stage +details

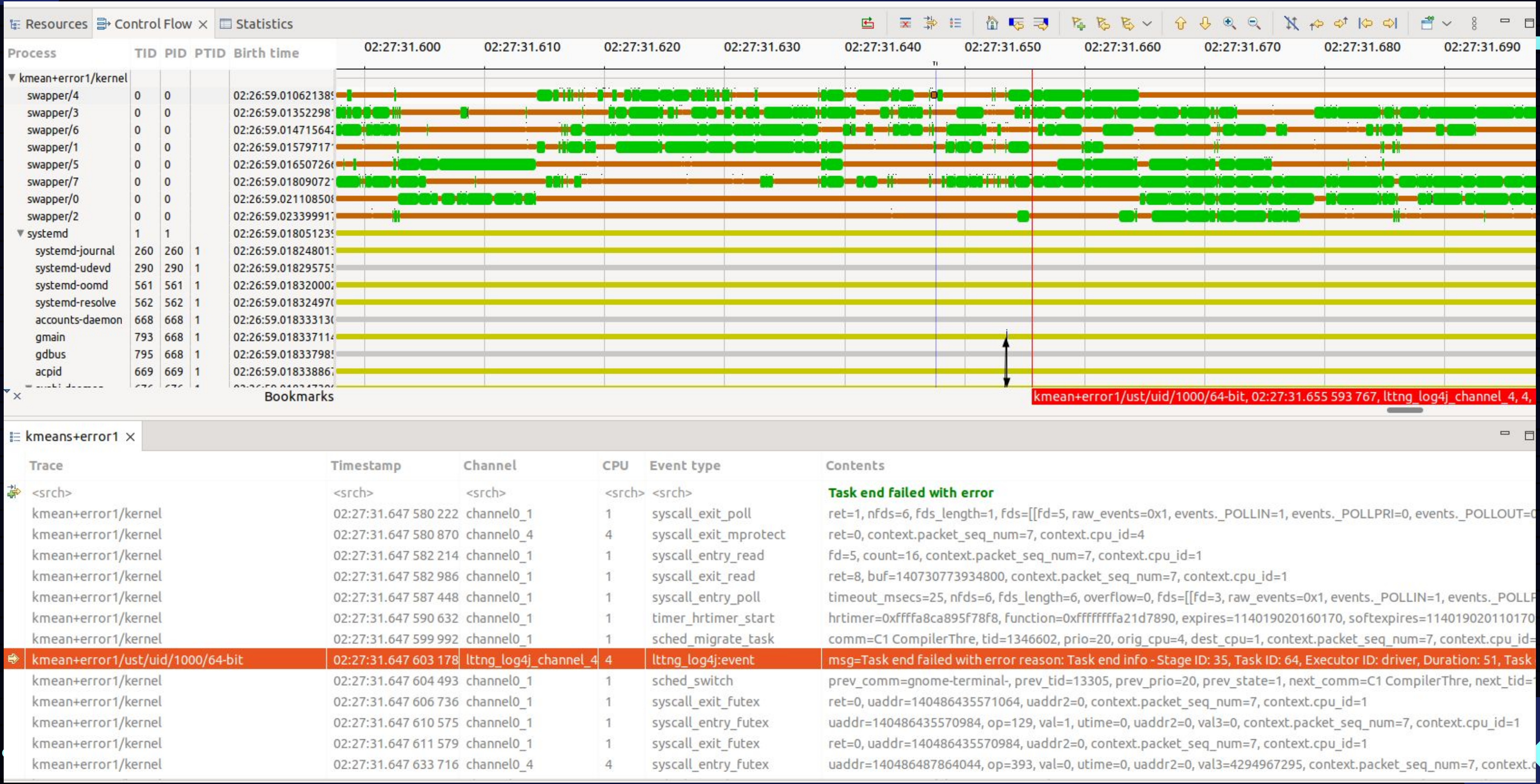
Page: 1 — 1 Pages. Jump to 1 . Show 100 items in a page. Go



Index	Task ID	Attempt	Status	Locality level	Executor ID	Host	Logs	Launch Time	Duration	GC Time	Input Size / Records	Errors
0	64	0	FAILED	PROCESS_LOCAL	driver	Rezghool.ht.home		2023-12-05 03:59:52	27.0 ms			org.apache.spark.SparkException: Intentional failure in stage +details

```
org.apache.spark.SparkException: Intentional failure in stage
	at
org.apache.spark.examples.ml.JavaKMeansExample.lambda$main$dc6d3fb3$1(JavaKMe
ansExample.java:104)
	at
org.apache.spark.sql.Dataset.$anonfun$foreachPartition$2(Dataset.scala:3370)
	at
org.apache.spark.sql.Dataset.$anonfun$foreachPartition$2$adapted(Dataset.scal
a:3370)
	at
```

Key Findings/ Showing the fault in Trace Compass



Process	TID	PID	PTID	Birth time
▼ kmean+error1/kernel				
swapper/4	0	0		02:26:59.01062138
swapper/3	0	0		02:26:59.01352298
swapper/6	0	0		02:26:59.01471564
swapper/1	0	0		02:26:59.01579717
swapper/5	0	0		02:26:59.01650726
swapper/7	0	0		02:26:59.01809072
swapper/0	0	0		02:26:59.02110850
swapper/2	0	0		02:26:59.02339991
▼ systemd	1	1		02:26:59.01805123
systemd-journal	260	260	1	02:26:59.01824801
systemd-udevd	290	290	1	02:26:59.01829575
systemd-oomd	561	561	1	02:26:59.01832000
systemd-resolve	562	562	1	02:26:59.01832497
accounts-daemon	668	668	1	02:26:59.01833313
gmain	793	668	1	02:26:59.01833711
gdbus	795	668	1	02:26:59.01833798
acpid	669	669	1	02:26:59.01833886

Trace	Timestamp	Channel	CPU	Event type	Contents
<srch>	<srch>	<srch>	<srch>	<srch>	**Task end failed with error**
kmean+error1/kernel	02:27:31.647 580 222	channel0_1	1	syscall_exit_poll	ret=1, nfds=6, fds_length=1, fds=[[fd=5, raw_events=0x1, events._POLLIN=1, events._POLLPRI=0, events._POLLOUT=
kmean+error1/kernel	02:27:31.647 580 870	channel0_4	4	syscall_exit_mprotect	ret=0, context.packet_seq_num=7, context.cpu_id=4
kmean+error1/kernel	02:27:31.647 582 214	channel0_1	1	syscall_entry_read	fd=5, count=16, context.packet_seq_num=7, context.cpu_id=1
kmean+error1/kernel	02:27:31.647 582 986	channel0_1	1	syscall_exit_read	ret=8, buf=140730773934800, context.packet_seq_num=7, context.cpu_id=1
kmean+error1/kernel	02:27:31.647 587 448	channel0_1	1	syscall_entry_poll	timeout_msecs=25, nfds=6, fds_length=6, overflow=0, fds=[[fd=3, raw_events=0x1, events._POLLIN=1, events._POLL
kmean+error1/kernel	02:27:31.647 590 632	channel0_1	1	timer_hrtimer_start	hrtimer=0xffffa8ca895f78f8, function=0xffffffffa21d7890, expires=114019020160170, softexpires=114019020110170
kmean+error1/kernel	02:27:31.647 599 992	channel0_1	1	sched_migrate_task	comm=C1 CompilerThre, tid=1346602, prio=20, orig_cpu=4, dest_cpu=1, context.packet_seq_num=7, context.cpu_id=
kmean+error1/ust/uid/1000/64-bit	02:27:31.647 603 178	lttng_log4j_channel_4	4	lttng_log4j:event	msg=Task end failed with error reason: Task end info - Stage ID: 35, Task ID: 64, Executor ID: driver, Duration: 51, Task
kmean+error1/kernel	02:27:31.647 604 493	channel0_1	1	sched_switch	prev_comm=gnome-terminal-, prev_tid=13305, prev_prio=20, prev_state=1, next_comm=C1 CompilerThre, next_tid=
kmean+error1/kernel	02:27:31.647 606 736	channel0_1	1	syscall_exit_futex	ret=0, uaddr=140486435571064, uaddr2=0, context.packet_seq_num=7, context.cpu_id=1
kmean+error1/kernel	02:27:31.647 610 575	channel0_1	1	syscall_entry_futex	uaddr=140486435570984, op=129, val=1, utime=0, uaddr2=0, val3=0, context.packet_seq_num=7, context.cpu_id=1
kmean+error1/kernel	02:27:31.647 611 579	channel0_1	1	syscall_exit_futex	ret=0, uaddr=140486435570984, uaddr2=0, context.packet_seq_num=7, context.cpu_id=1
kmean+error1/kernel	02:27:31.647 633 716	channel0_4	4	syscall_entry_futex	uaddr=140486487864044, op=393, val=0, utime=0, uaddr2=0, val3=4294967295, context.packet_seq_num=7, context.

Conclusion and Future Work

1. I have all required logs from Apache spark logs with a little overhead in my lttng session
2. Adding more information from kernel events

1. Visualize the logs like Spark UI
2. Show the user which part of spark have issue related to their code

Thanks!

Do you have any questions?

Credits: This presentation template was created by **Slidesgo**, including icons by **Flaticon**, infographics & images by **Freepik**