

# Final Presentation

Team Red

# How to Run

- Master and worker are distributed as .jar files.

```
java -jar master.jar 2
```

```
java -jar worker.jar 141.223.91.80:5000 -I /data1/input /data2/input -O /home/red/data
```

# Results: Small Settings

- Our implementation works well with **small** dataset
  - with 5 machines:
    - Without fault: around 60s
    - With fault: +10~20s
  - with 20 machines:
    - around 20 minutes
    - log shows one or two machines takes unusually long time to sort data  
(most likely disk I/O issue)

Does the master print a sequence of workers? - 

Is the output sorted in each workers? - 

# of input records == # of records in the output? - 

# Results: Large Settings

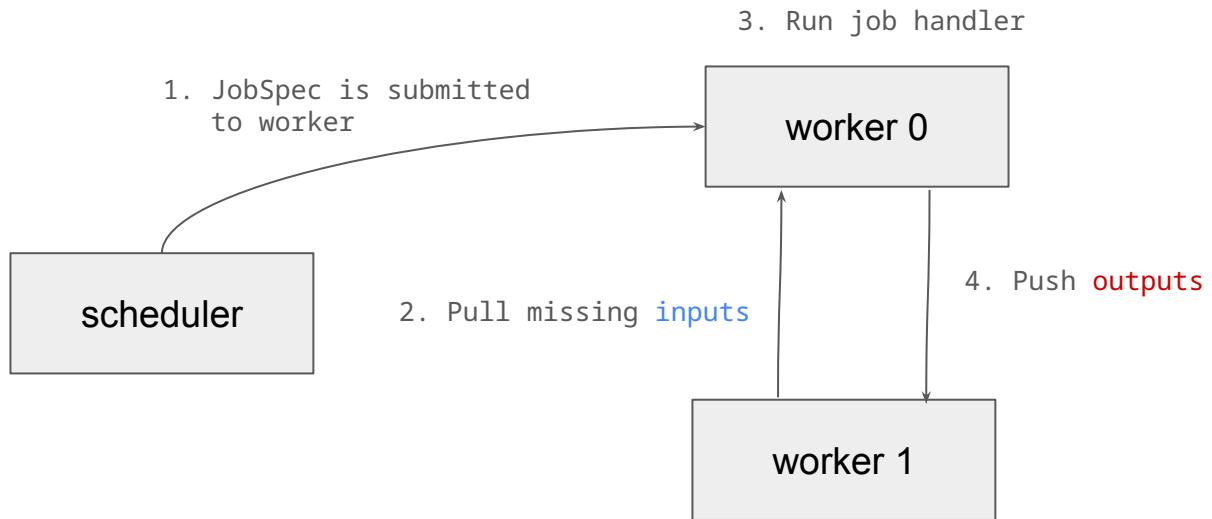
- However, it crashes with **large** dataset.
  - postmortem analysis will follow after system architecture.
- We couldn't test **big** dataset due to limited cluster usage time.

# System Architecture

Three subprojects:

- **jobs** - job running infrastructure library, provides:
  - **Scheduler** for dispatching jobs to workers
  - **Worker** for running job when requested by **Scheduler**
- **master** and **worker** - builds on **jobs** library

# System Architecture: jobs package



```
JobSpec(  
    name = "sort",  
    args = [],  
    inputs = [  
        FileEntry(  
            path = "@{input}/part1",  
            replicas = [0]  
        )  
    ],  
    outputs = [  
        FileEntry(  
            path = "@{working}/sorted.0",  
            replicas = [0]  
        )  
    ]  
)
```

# System Architecture: Fault Tolerance

- gRPC keepalive for fault detection
  - will raise transport error when keepalive timeout fires
- retry job for fault tolerance - input must be available somewhere!

```
[ INFO ] r.jobs.scheduler.SchedulerFiber -- <worker 1,0> completed job: pending: 0, completed: 1  
ERROR redsort.jobs.RPCHelper -- transport error: io.grpc.StatusRuntimeException: UNAVAILABLE: Network closed for unknown reason  
ERROR redsort.jobs.RPCHelper -- RPC call failed due to transport error, will retry in 1 seconds...  
ERROR redsort.jobs.RPCHelper -- transport error: io.grpc.StatusRuntimeException: UNAVAILABLE: io exception  
ERROR redsort.jobs.RPCHelper -- RPC call failed due to transport error, will retry in 1 seconds...  
ERROR redsort.jobs.RPCHelper -- transport error: io.grpc.StatusRuntimeException: UNAVAILABLE: io exception  
ERROR redsort.jobs.RPCHelper -- RPC call failed due to transport error, will retry in 1 seconds...  
] INFO r.jobs.scheduler.SchedulerRpcService -- new worker registration from NetAddr(172.26.4.93,9101)  
] DEBUG r.jobs.scheduler.SchedulerFiber -- got event WorkerRegistration(WorkerHello(0,Some(LocalStorageInfo(None,-1,Map(@{input}/h  
ERROR redsort.jobs.RPCHelper -- transport error: io.grpc.StatusRuntimeException: UNAVAILABLE: io exception  
ERROR redsort.jobs.RPCHelper -- RPC call failed due to transport error, will retry in 1 seconds...  
] DEBUG r.jobs.scheduler.SchedulerFiber -- got event JobCompleted(JobResult(true,None,None,None,Vector(FileEntryMsg(@{working}/len
```

transport error detected, retry

worker comes back online (re-register)

job completes

# Why unsuccessful?

We underestimated **network congestion**.

## Assumptions on Network

We assume there is a reliable and ordered network channel between any pairs of machines available unless one of machine is down.

Wrong assumption in design phase



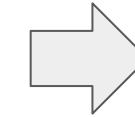
network congestion due to replication

```
.keepAliveTime(10, TimeUnit.SECONDS)  
.keepAliveTimeout(5, TimeUnit.SECONDS)
```

Misconfiguration of keepalive



insufficient testing on large-scale settings



# Why unsuccessful?

What actually happened:

- Our program builds on assumption that network congestion does not happen.
- But with large number of workers, this assumption breaks.

replicating large files across workers

→ network congestion

→ gRPC keepalive packet drops

→ keepalive timeout fires (false fault detection)

→ job resubmission

→ **exception**

# Lessons Learned: Good Parts

## 1. Design more, suffer less

: Thanks to detailed design, implementation phase went smoother than we expected.

## 2. Collaboration works quite well once all team members get used to the workflow.

- Program crashes if all input directories are empty bug #51 · by DanaKharaz was closed 4 days ago 11 1 3
- Record count check in the local integration test helper -- DistributedSortingTestHelper enhancement #49 · by DanaKharaz was closed 5 days ago 11 1 1
- Partition indices do not match the order of worker machines given by the master bug #47 · by DanaKharaz was closed last week 11 1 1

## Lessons Learned: Bitter Parts

**Always test your program on actual environment.**

- The bug we described earlier was not reproducible on local settings.
- Indeed, the bug can be fixed by modifying just a few lines of codes.