Fixing slow performance issues with AWS Glue ETL jobs

Craig Godden-Payne
Feb 7 · 3 min read ★

I recently spent some time looking at an AWS Glue job which was not performing as expected.

The job was taking a file from S3, some very basic mapping, and converting to parquet format. The file was in GZip format, 4GB compressed (about 27GB uncompressed).

The job initially took in excess of 30 minutes to complete, which felt like it could be improved, as similar tools could perform a similar task in a much quicker timeframe (under 10 minutes).

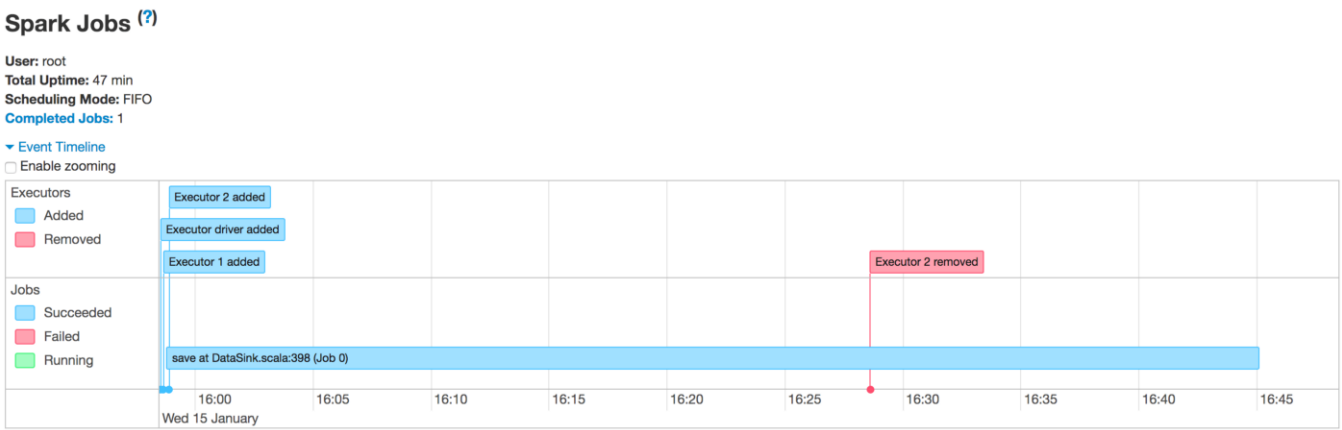
I looked at the spark output for the job, and it looked like the job was not being distributed to all the available nodes.

Description:

- Input = CSV, 100 columns
- Job: Map columns to specific type, remove nulls, save as parquet format.
- Uncompressed Size: 27GB
- Compressed Size: 4GB

The stats translated to:

- Input Size / Records: 4 GB gzipped / 22667151
- Output: 4.3 GB / 22667151
- Total Time Across All Tasks: 17.4 h
- Actual Time Taken: 46 min
- Task Split: 27
- DPUs assigned: 30
- Rows per second: ~8212



What could be the problem?

The first thing I looked at was whether the compression type for the data was the problem. GZip is a non splittable compression type, so it is likely the excess time is from uncompression of the data. One of the problems I quickly noticed, was that the source

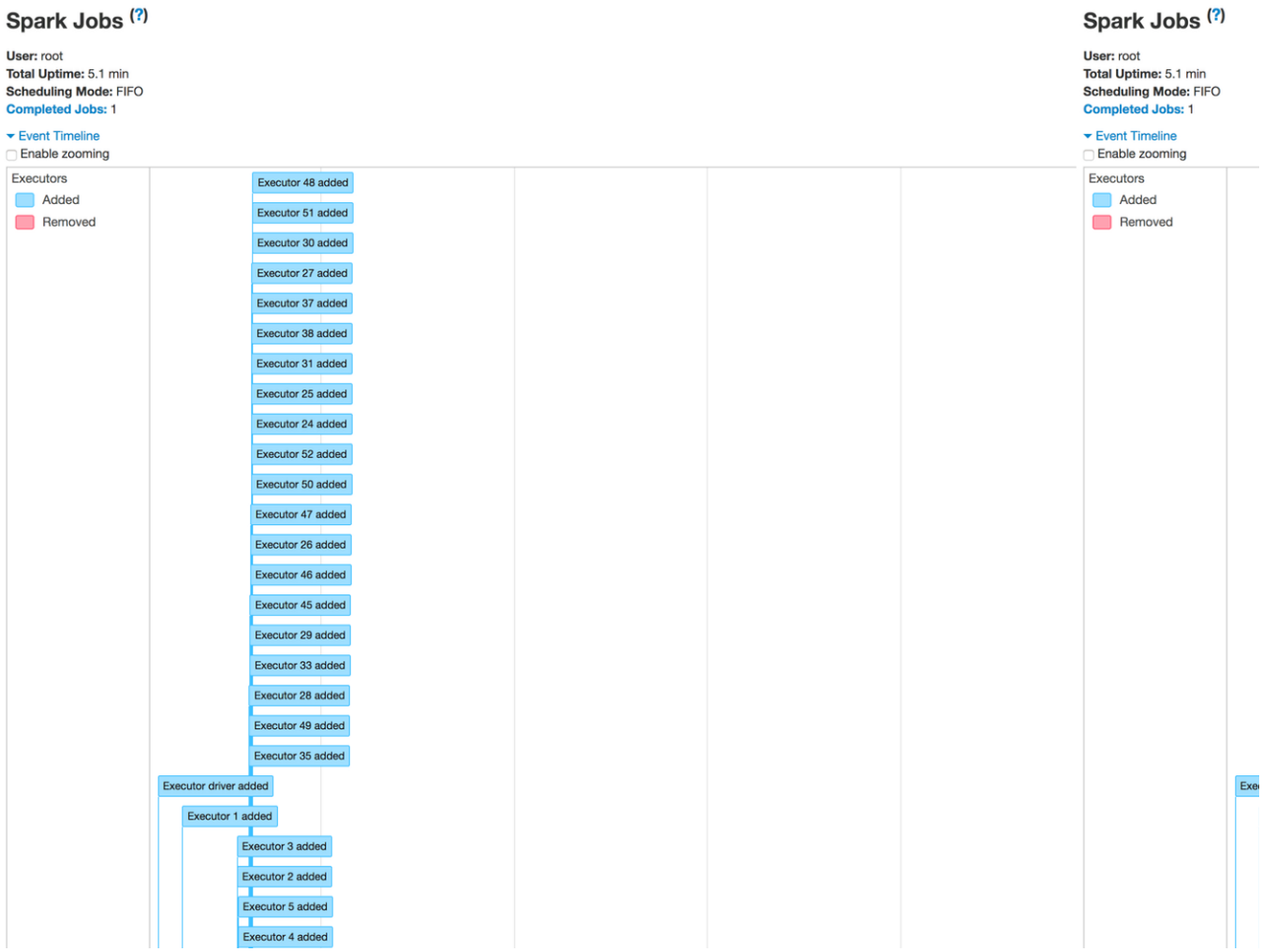
system producing the data could not efficiently product a splittable compressed file type, such as BZip2

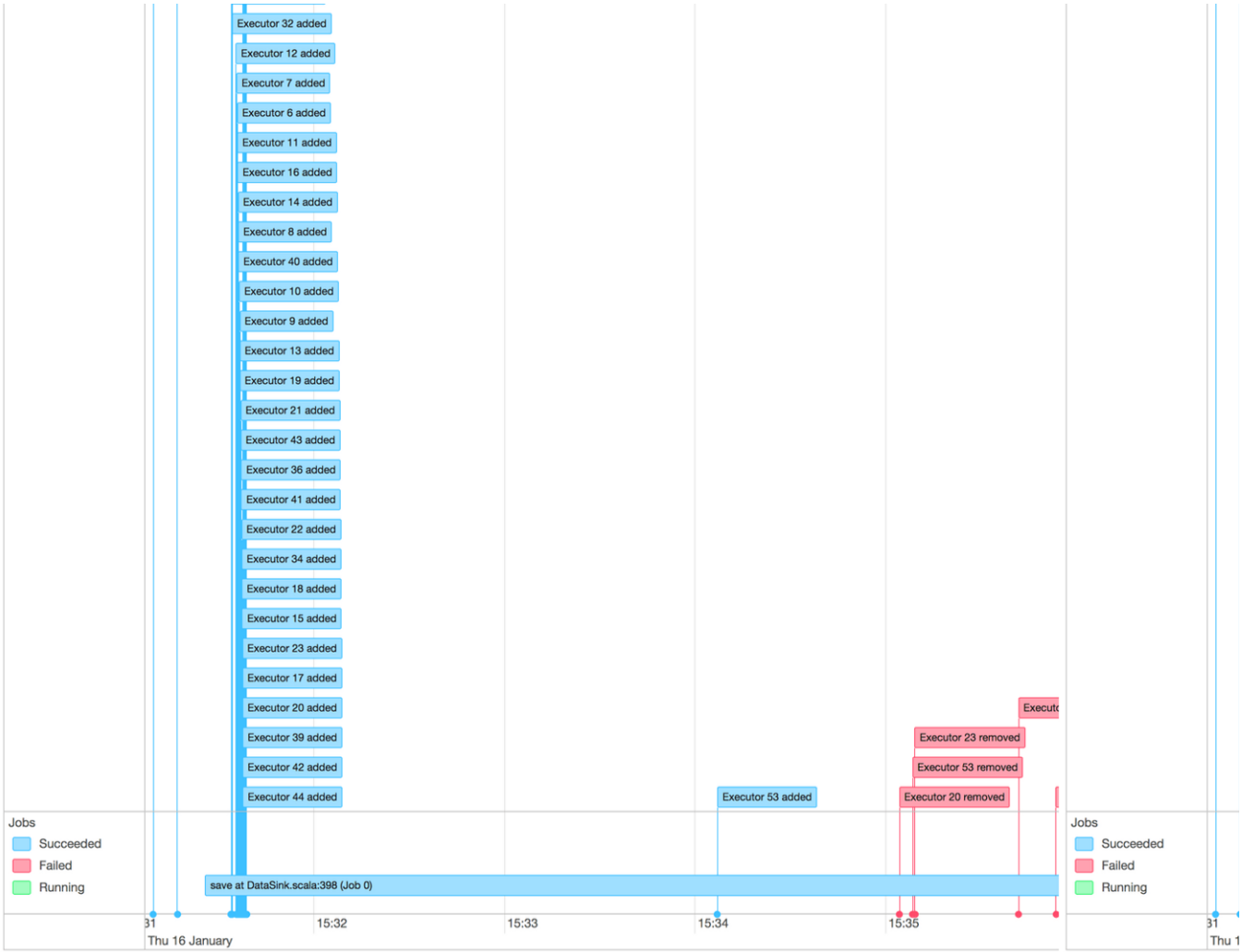
I ran the job with the data uncompressed in S3, so the full 27GB file

The results were:

- Input Size / Records: 26.7 GB uncompressed / 22667151
- Output: 4.3 GB / 22667151
- Total Time Across All Tasks: 13.7 h
- Actual Time Taken: 5.1 min
- Task Split: 428
- DPUs assigned: 30
- Rows per second: ~74075

This was almost ten times increase on performance!





The conclusion was after decompressing the files prior to glue processing, the parallel processing worked. Uncompressing gzip files using the spark server not only seems like an expensive way to uncompress files, but it also seems to use one DPU, only allowing parallel processing of 2 tasks when uncompressing a large file.

Subsequent testing went on, using many smaller GZip files (the uncompressed data was split into 256MB size files, then GZipped). This also had a very positive effect

A solution to our problem was to either uncompress gzip files using S3 event hooks, prior to them being processed with Glue, or to use smaller GZip files to get over these performance barriers.

The end solution was to use the small GZip files, as it had the least disruption on the existing process, and also meant that the transfer to S3 was quicker.

Performance AWS Glue