

Digital Preservation at Oxford and Cambridge

A collaborative research project to evaluate and provide sustainable recommendations for our digital preservation programmes

Validating half a million TIFF files. Part Two.

Posted on [17 August, 2017](#) by [James Mooney](#)

Back in May, I wrote a blog post about preparing the groundwork for the process of validating over 500,000 TIFF files which were created as part of a Polonsky Digitization Project which started in 2013. You can read [Part One here on the blog](#).

Restoring the TIFF files from tape



— Stack of backup tapes. Photo: Amazon

For the digitization workflow we used [Goobi](#) and within that process, the master TIFF files from the project were written to tape. In order to actually check these files, it was obvious we would need to restore all the content to spinning disk. I duly made a request to our system administration team and waited.

As I mentioned in [Part One](#), we had setup a new virtualised server which had access to a chunk of network storage. The Polonsky TIFF files were restored to this network storage, however midway through the restoration from tape, the tape server's operating system crashed...disaster.

After reviewing the failure, it appeared there was a bug within the RedHat operating system which had caused the problem. This issue proved to be a good lesson, ***a tape backup copy is only useful if you can actually restore it!***

Question for you. When was the last time you tried to restore a large quantity of data from tape?

After some head scratching, patching and a review of the related systems, a second attempt at restoring all the TIFF content from

tape commenced and this time all went well and the files were restored to the network storage. Hurrah!

JHOVE to validate those TIFFs



I decided that for the initial validation of the TIFF files, checking the files were well-formed and valid, [JHOVE](#) would provide a good baseline report.

As I mentioned in another blog post [Customizable JHOVE TIFF output handler anyone?](#) JHOVE's XML output is rather unwieldy and so I planned to transform the XML using [xsltproc](#) (a command line xslt processor) with a custom XSLT stylesheet, allowing us to select any of attributes from the file which we might want to report on later, this would then produce a simple CSV output.

On a side note, work on adding a CSV output handler to JHOVE is in progress! This would mean the above process would be much simpler and quicker.

Parallel processing for the win.

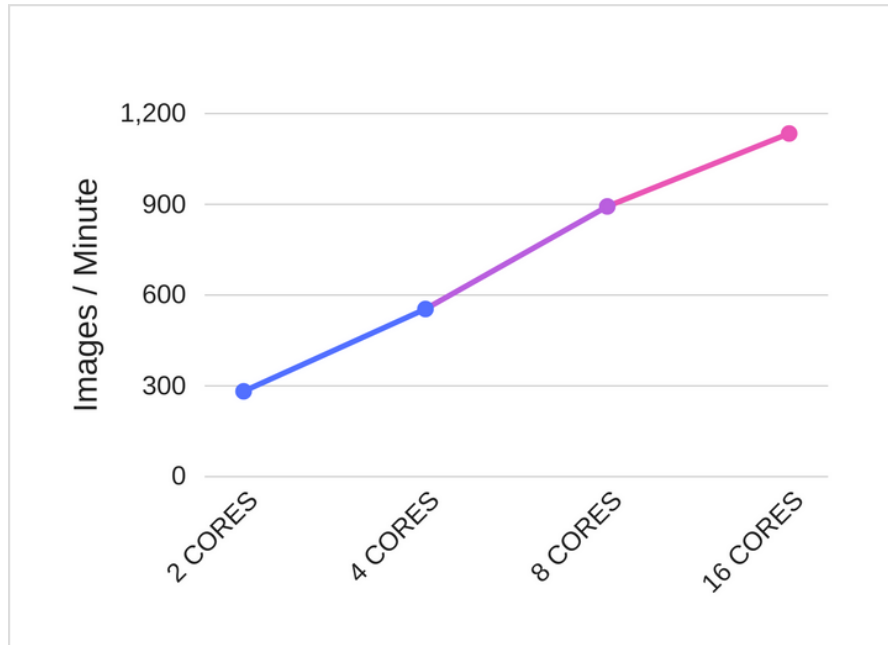
What's better than one JHOVE process validating TIFF content? Two! (*well actually for us, sixteen at once works out quite nicely.*)

It was clear from some initial testing with a 10,000 sample set of TIFF files that a single JHOVE process was going to take a long time to process 520,000+ images (around two and half days!)

So I started to look for a simple way to run many JHOVE processes in parallel. Using [GNU Parallel](#) seemed like a good way to go.

I created a command line BASH script which would take a list of directories to scan and then utilise [GNU Parallel](#) to fire off many JHOVE + XSLT processes to result in a CSV output, one line per TIFF file processed.

As our validation server was virtualised, it meant that I could scale the memory and CPU cores in this machine to do some performance testing. Below is a chart showing the number of images that the parallel processing system could handle per minute vs. the number of CPU cores enabled on the virtual server. *(For all of the testing the memory in the server remained at 4 GB.)*



So with 16 CPU cores, the estimate was that it would take around 6-7 hours to process all the Polonksy TIFF content, so a nice improvement on a single process.

At the start of this week, I ran a full production test, validating all 520,000+ TIFF files. 4 and half hours later the process was complete and 100 MB+ CSV file was generated with 520,000+ rows of data. Success!

For Part Three of this story I will write up how I plan to visualise the CSV data in [Qlik Sense](#) and the further analysis of those few files which failed the initial validation.

SHARE THIS:



This entry was posted in [digitisation](#), [digitization](#), [technology](#), [tools](#) and tagged [backups](#), [cpus](#), [gnu parallel](#), [jhove](#), [parallel](#), [processing](#), [tape](#), [validation](#), [xml](#), [xslt](#) by [James Mooney](#). Bookmark the [permalink](#) [<http://www.dpoc.ac.uk/2017/08/17/validating-half-a-million-tiff-files-part-two/>].

This site uses Akismet to reduce spam. [Learn how your comment data is processed.](#)