TikaEvalOnVM

How to Run tika-eval-app on the VM

While users can run tika-eval-app on their own machines with their own documents, the Apache Tika, Apache PDFBox and Apache POI communities have gathered >1TB of documents from govdocs1 and from Common Crawl to serve as a regression testing corpus. Before a release, we'll run the last release against the candidate release to identify potential regressions.

This page is intended for committers/PMC members with access to the VM who want to run the regression tests. The example focuses on testing a SNAPSHOT version of PDFBox, but the steps are nearly identical for the full Tika eval or for sub projects. See TikaEval for more information on the `tika-eval-app` module by itself. See this blog for a description of running this project on Tika's VM.

The driver `appBatchExecutor.sh`, the various configuration files and the file lists for PDFs are available here: batch-scripts.tgz.

If you haven't done so in your .bashrc file, make sure to `umask g+rw` before running anything

The main working directory is: `/data1/tools/tika/batch`

An Example with Apache PDFBox

1. Clean up from any previous runs
    a. Remove tika-app-X-Y.jar from /data1/tools/tika/batch/bin – make sure to leave in the other "optional" jars: `jai-imageio-jpeg2000-1.4.0.jar, sqlite-jdbc-3.45.3.0.jar and zstd-jni-1.5.6-2.jar`
    b. Remove or rename `/data1/tools/tika/batch/logs`
    c. Remove or rename `/data1/tools/tika/batch/nohup.out`
2. Run the current "A" version
    a. Place the "A" version of tika-app-X.Y.jar in `/data1/tools/tika/batch/bin`
    b. Modify `appBatchExecutor.sh` to
        i. put the output in a new output directory `-o /data1/extracts/pdfboxA`
        ii. if using a file list, confirm that the correct file list is specified `-fileList fileLists/ccAndBugTracker_pdfs.txt`
    c. Execute: `nohup ./appBatchExecutor.sh &`
    d. Wait for the "A" version to complete before starting the "B" version
3. Build and run the "B" version
    a. Update PDFBox from SVN, `mvn clean install`
    b. Update the PDFBox, Fontbox and jbig2-imageio versions in the Tika project tika-parsers/pom.xml
    c. Run `mvn clean` on the whole Tika project and make sure that your IDE has picked up the changes
    d. Run the PDFParser tests in tika-parsers/src/test/java/o.a.t.parsers.pdf.* to make sure that at least the Tika unit tests work.
    e. Build the entire Tika project (even though you'll only use tika-app.jar): `mvn clean install`
    f. On the VM, remove the tika.app-A.jar from `/data1/tools/tika/batch/bin`, rename the existing `nohup.out` to `nohup-A.out`, rename `logs/` to `logs-A/`
    g. Drop the new tika-app-B.jar into (you guessed it!): `/data1/tools/tika/batch/bin`
    h. Modify `appBatchExecutor.sh` to
        i. put the output in a new output directory `-o /data1/extracts/pdfboxB`
        ii. if using a file list, confirm that the correct file list is specified `-fileList fileLists/ccAndBugTracker_pdfs.txt`
    i. Execute: `nohup ./appBatchExecutor.sh &`
    j. Wait for the "B" version to complete before starting the comparisons and reports
4. Make the comparisons and report
    a. In `/data1/tools/tika/eval`, remove the existing db file `pdfboxAvsB.mv.db` if you don't want to rename it.
    b. `nohup java -jar tika-eval-app-X.Y.jar Compare -extractsA /data1/extracts/pdfboxA -extractsB /data1/extracts/pdfboxB -db pdfboxAvsB&`
    c. When that completes,
        i. Remove any files left over from the last run in `reports/`: `rm -r reports`
        ii. Write the reports `java -Djava.io.tmpdir=tmp -jar tika-eval-app-X.Y.jar Report -db pdfboxAvsB` – **Note the -Djava.io.tmpdir=tmp – need to set the tmp directory to something writeable by 'collab'**

When this process completes, you'll have all of the reports written to `/data1/tools/tika/eval/reports/`.

H2 to Postgresql and Reports

With the expansion of the regression corpus, I'm finding that H2 isn't able to write the reports – no matter the -Xmx, even after a few hours, it doesn't even get to the point of creating the *reports* directory.

I should set up postgres on our VM, but I haven't gotten around to that yet. For now, I'm copying the H2 db to Postgresql and then writing the reports from there. The code to copy H2->postgres is available here: tika-addons.

I had to modify the report SQL slightly to work with Postgresql, and I stripped out some of the reports/calculations that aren't critical to the full regression tests. The modified report SQL is available comparison-reports-pg.xml