**Crawl the Datasets**

* Setup of server, database and datasets -

By

Arapaut V. Sivaprasad

15 November 2018

**SUMMARY**

This document describes the steps involved in setting up datasets for services such as GEOGLAM. It is intended to be as complete as possible from start to finish, such as setting up the VM, creating database, crawling and ingesting the data. You need only the car key to drive, but to change a fuse the car manual is required. In a similar way, this document is required only if you run into trouble. The component programs have been streamlined so that one or a few simple “key” commands will setup the system.

**DISCLAIMER**

This is an evolving document and may go out of sync with the programs in time. Those who maintain the programs are requested to update this document as well. While every care is taken to be factually and operationally correct including the directory paths, filenames, etc., there may be instances where certain aspects are not described correctly or there may be reasons for some steps to be different.

# TL;DR

* The script to crawl and ingest the data is ‘/…/gsky/crawl/[**crawl.sh**](https://github.com/asivapra/gsky/blob/master/crawl/crawl.sh)’
  + Edit it to insert the correct pathnames as e.g.
    - export **CRAWL\_DIR**=/g/data2/tc43/modis-fc/v310/tiles/8-day/cover/
    - export **CRAWL\_OUTPUT\_DIR**=/home/900/avs900/crawl\_outputs
  + Execute it from a shell command as ‘**sudo ./crawl.sh**’
* Setting up the VM is described [here](https://github.com/asivapra/gsky/blob/master/install/README.md).

# BACKGROUND

The crawler processes the \*.nc files to take the metadata and inserts it into a PostgreSQL database table, ‘ingest’ in the database, ‘mas’. There are three columns in this table as given below.

mas=# \d ingest

Unlogged table "public.ingest"

Column | Type | Collation | Nullable | Default

---------+-------+-----------+----------+---------

in\_path | text | | |

in\_type | text | | |

in\_json | jsonb | | |

e.g.

in\_path: /g/data2/tc43/modis-fc/v310/tiles/8-day/cover/FC.v310.MCD43A4.h09v03.2001.006.nc

in\_type: gdal

in\_json: {"filename":"/g/data2/…/FC….nc","file\_type":"netCDF","geo\_metadata":…"nodata":255}]}

The ‘in\_json’ column records the details that can be overlaid as images on GEOGLAM web site.

The process of crawling and recording the above details is described in this document.

# PROCESS FLOW

The crawling process is done through a series of bash scripts as shown below.

**Fig.1**

crawl\_pipeline.sh

ingest\_pipeline.sh

crawl.sh

shard\_ingest.sh

ingest.sh

shard\_create.sh

shard\_refresh.sh

# SHORT DESCRIPTION

* ‘crawl.sh’ sets up environment variables and calls ‘crawl\_pipeline.sh’
* ‘crawl\_pipeline.sh’ reads the \*.nc files (using GDAL) to create a TSV file.
* ‘crawl.sh’ then calls ‘ingest\_pipeline.sh’ to ingest the data in the TSV into database.
* ‘ingest\_pipeline.sh’ calls ‘shard\_create.sh’ to create the required PSQL tables and functions.
* ‘ingest\_pipeline.sh’ passes the TSV filepath to ‘shard\_ingest.sh’.
* ‘shard\_ingest.sh’ calls ‘ingest.sh’ to ingest the data from the TSV file.
* ‘shard\_refresh.sh’ refreshes the tables and functions (??)

# DETAILS OF SCRIPTS

## crawl.sh

This script sets up various environment variables to be used by the scripts that follow.

* export PATH="/local/gsky/bin:/local/gsky/share/mas:/local/gsky/share/gsky:$PATH"
  + This is required to find executables
* export CRAWL\_DIR=/g/data2/tc43/modis-fc/v310/tiles/8-day/cover/
  + This is the absolute directory path for the \*.nc files.
  + There can be sub-dirs in it, which will also be crawled. (??)
  + To specify a few files, instead of a whole directory, define the variable, ‘CRAWL\_FILE\_LIST’. This, however, is not in ‘crawl.sh’, and is probably an error.
* export CRAWL\_OUTPUT\_DIR=/home/900/avs900/crawl\_outputs
  + This is where the TSV file is created.
  + If not given a ‘CRAWL\_FILE\_LIST’, a list of files will be created in this directory.
* export CRAWL\_CONC\_LIMIT=2
  + This is to control how many parallel processes must be run simultaneously.
  + The default is 16, but since the VM often has 2 to 4 CPUs, specify a number here.
* export LD\_LIBRARY\_PATH="/usr/local/lib:${LD\_LIBRARY\_PATH:-}"
  + Define the path where the libraries required by various programs.
* export PGUSER=postgres
  + This is to ensure that PSQL can be executed as user, ‘postgres’
  + The ‘root’ or any other user cannot run PostgreSQL, for security reasons.
* export PGDATA=/usr/local/pgsql/data
  + This is where the PostgreSQL data files reside.
* export GPATH=/g/data2/tc43
  + It is the base directory for the \*.nc files. Unsure why it must be defined. (??)

The script then does the following before calling ‘crawl\_pipeline.sh’

* Check whether \*.nc files are present in the ‘$CRAWL\_DIR’
  + Exit if none.
* Create the $CRAWL\_OUTPUT\_DIR
* Run the ‘crawl\_pipeline.sh’
* Run the ‘ingest\_pipeline.sh’
* Exit

## crawl\_pipeline.sh

This script checks the defined environment variables, sets up a file list of the \*.nc files and calls ‘GDAL’ via an executable, ‘gsky-crawl’ to read each \*.nc file and add its data to a TSV file. This TSV file is the one used by the ingest scripts later on.

* Check whether $CRAWL\_OUTPUT\_DIR exists and, if not, create as a sub-dir of the script dir.
  + here="$( cd "$( dirname "${BASH\_SOURCE[0]}" )" && pwd )"
  + data\_dir=$here/crawl\_tsv/`date +'%Y-%m-%d\_%H-%M-%S'`
* Check if $CRAWL\_FILE\_LIST is defined.
  + If yes, create a ‘$file\_list’ with its values.
  + If not, get a listing of the \*.nc files in $CRAWL\_DIR to make the $file\_list.
    - job\_id="${find\_dir//[\/]/\_}"
    - file\_list=$data\_dir/${job\_id}.filelist
* Stream the names of files in $file\_list to ‘gsky-crawl’ to create the TSV file
  + cat $file\_list | concurrent -i -l $conc\_limit xargs bash -c 'gdal\_json "$@"' \_ | gzip > $crawl\_file
  + The above runs ‘'gdal\_json’ in parallel on $conc\_limit of CPUs
    - in turn calls ‘gsky-crawl’
  + The output is added to $crawl\_file and zipped.
    - I think it is wrong to gzip at this stage, as the text file is required later. (??)
    - I think it must be kept as plain TSV file, used later and deleted. (??)

## ingest\_pipeline.sh

This script takes the data stored in the TSV file (see previous section) and add it to the database. Though this script looks long and complex, all it does is setting up the environment for ingesting the data and then calling another script to ingest it. There are some functions in this script that apparently can be eliminated by simply defining one variable, ‘gpath’. This script of ~140 lines can probably be reduced to less than 10 lines.

* Check whether the environment variable, $GPATH, has been set.
  + If not, then construct the variable, $gpath, from $CRAWL\_DIR
    - Can eliminate two functions, get\_gpath() and assert\_gpath() (62 lines), by defining $GPATH
* Calls ‘shard\_create.sh’ to create a database schema.
* Calls ‘shard\_ingest.sh’, which in turn calls ‘ingest.sh’ to ingest the data.
  + Into the ‘mas’ database table, ‘ingest’.
* Calls ‘shard\_refresh.sh’
  + Purpose is unknown! (??)

## shard\_create.sh

The primary purpose of this script is just to create a schema named “$shard”. Before creating the schema, it does some checking to see if it already exists. There was an error in this step which prevented the creation of the schema. Disabling this bit of code appears to have no adverse effect, but it must be verified. (??)

## shard\_ingest.sh

This script just calls another, ‘ingest.sh’, and is puzzling why the latter cannot be called directly from the ‘ingest\_pipeline.sh’.

## ingest.sh

This script reads the TSV file and adds the data into a database table (mas:ingest).

* There were two apparent errors that prevented this step.
  + The TSV file is in gzip (\*.gz) format and cannot be read by the script.
  + No file is being read and, hence, nothing can go into the DB table. Is it an error or my mis-understanding?
  + Even when the TSV file is unzipped, nothing gets added to the table due to some error in the triggers attached to the table.
    - It worked when the triggers were removed.
    - Unsure what impact it will have to the data. (??)

## shard\_refresh.sh

All that it does is the following. Not knowing what the downstream programs do, I am not sure of its purpose. (??)

* select refresh\_views();
* select refresh\_polygons();
* select refresh\_caches();

# OUTCOME

If everything goes to completion, the ‘mas:ingest’ database table will be populated with the data as…

* in\_path text
  + /g/data2/tc43/modis-fc/v310/tiles/8-day/cover/FC.v310.MCD43A4.h09v03.2001.006.nc
* in\_type text
  + gdal
* in\_json jsonb
  + {"filename":"/g/data2/.../FC…2001.006.nc","file\_type"… "nodata":255}]}

The TSV file appears to be transient and is not required to be kept. Another transient file, “\*.filelist” is also not required to be kept, but both these are being retained.

# ENHANCEMENT

For the crawling to work, there needs to be some pre-requisite steps such as creating the databases, functions and tables. Putting them all together into the same script is worth looking into. Also, instead of having 7 separate scripts, combining them all into one is probably feasible. See below.

crawl.sh

**Fig.2**

At this stage I am not sure how it will affect the crawling and ingesting process or, more importantly, the format of the ingested data.