Brief introduction to the use of SNAP Graph Processing Tool

ESA/CONAE L/C/X band SAR Training

Course (12 -17 November 2018)

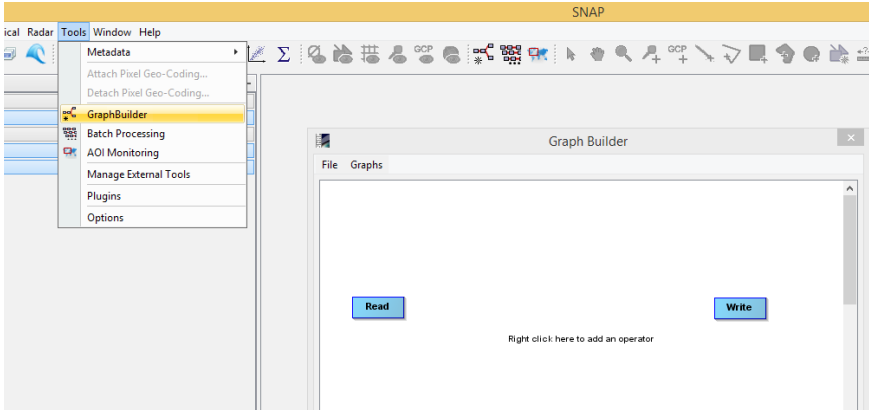
Mario Camuyrano (CONAE) mcamuyrano@conae.gov.ar

- Graph Processing Tool (gpt)

The gpt command is a tool that allows us to execute a number of SNAP operations in batch mode without opening the java graphical console GUI. That improves notably the performance of the operations and allows us to incorporate the SNAP programs in our own code developed with low or high level programming languages.

The gpt command can be found at the ***bin*** directory of the SNAP installation home, usually it is located at ***/opt/snap*** in linux and ***C:\Program Files\snap*** in Windows, but that it can change according to installation parameters.

gpt can run a list of tasks that have to be passed to it in XML format which can be developed using the graph-builder interface of SNAP.



- As an example we are going to show how to calibrate and do a multilook of a COSMO image.

  First we generate a ***CalandML.xml*** using the SNAP graphbuilder, after that we have to edit it to insert the variables that we are going to pass as parameters, in this example we are going to use three variables, ***input***, ***output*** and ***nl***. The variables have to be inserted with a ***$*** prefix as shown.

`CalandML.xml`

```
<graph id="Graph">
  <version>1.0</version>
  <node id="Read">
    <operator>Read</operator>
    <sources/>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <file>$input</file>
    </parameters>
  </node>
  <node id="Calibration">
    <operator>Calibration</operator>
    <sources>
      <sourceProduct refid="Read"/>
    </sources>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <sourceBands/>
      <auxFile>Latest Auxiliary File</auxFile>
      <externalAuxFile/>
      <outputImageInComplex>false</outputImageInComplex>
      <outputImageScaleInDb>false</outputImageScaleInDb>
      <createGammaBand>false</createGammaBand>
      <createBetaBand>false</createBetaBand>
    </parameters>
  </node>
  <node id="Multilook">
    <operator>Multilook</operator>
    <sources>
      <sourceProduct refid="Calibration"/>
    </sources>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <sourceBands/>
      <nRgLooks>$nl</nRgLooks>
      <nAzLooks>$nl</nAzLooks>
      <outputIntensity>false</outputIntensity>
    </parameters>
  </node>
  <node id="Write">
    <operator>Write</operator>
    <sources>
      <sourceProduct refid="Multilook"/>
    </sources>
    <parameters class="com.bc.ceres.binding.dom.XppDomElement">
      <file>$output</file>
      <formatName>BEAM-DIMAP</formatName>
    </parameters>
  </node>
  <applicationData id="Presentation">
    <Description/>
    <node id="Read">
            <displayPosition x="37.0" y="134.0"/>
    </node>
    <node id="Calibration">
      <displayPosition x="170.0" y="134.0"/>
    </node>
    <node id="Multilook">
      <displayPosition x="314.0" y="136.0"/>
    </node>
    <node id="Write">
            <displayPosition x="455.0" y="135.0"/>
    </node>
  </applicationData>
</graph>
```

To run SNAP with those parameters we have to write:

```
$SNAP_HOME/bin/gpt CalandML.xml -Pintput=~/Imagenes/CosmoSkymed/cosmo1.h5 -Pnl=4
-Poutput=~/Imagnes/CosmoSkymed/
```

- SNAP has a list of examples of tasks implemented. They can be found at:
  `C:\Users\u1\.snap\graphs\`
  `~/.snap/graphs/`

```
+---internal
|   |   BackGeocodingGraph.xml
|   |   CoregistrationGraph.xml
|   |   CreateStackGraph.xml
|   |   DataConvertGraph.xml
|   |   DEMAssistedCoregistrationGraph.xml
|   |   DEMBasedCoregistrationGraph.xml
|   |   importGraph.xml
|   |   InSARCoregistrationGraph.xml
|   |   InSARCoregistrationGraph2.xml
|   |   MosaicGraph.xml
|   |   MultiInputStackAveragingGraph.xml
|   |   MultiOutputCoregister.xml
|   |   OilSpillDetectionGraph.xml
|   |   ReadWriteGraph.xml
|   |   SARSimTCGraph.xml
|   |   Sentinel1-TOPS-Coregistration.xml
|   |   Sentinel1SLCtoGRDGraph.xml
|   |   Sentinel1SliceAssemblyGraph.xml
|   |   ShipDetectionGraph.xml
|   |   SLCtoPRIGraph.xml
|   |   SnaphuExportGraph.xml
|   |   SnaphuImportGraph.xml
|   |   StackSplitGraph.xml
|   |   StitchTileGraph.xml
|   |   TOPSARMergeGraph.xml
|   |   UnwrapTileGraph.xml
|   |   WindFieldEstimationGraph.xml
|   |   WSSDetectGraph.xml
|   |   WSSMosaicGraph.xml
|   |
```

```
+---internal
|   +---classification
|   |       KDTreeKNNClassifierGraph.xml
|   |       KNNClassifierGraph.xml
|   |       MaximumLikelihoodClassifierGraph.xml
|   |       MinimumDistanceClassifierGraph.xml
|   |       RandomForestClassifierGraph.xml
|   |       SVMClassifierGraph.xml
|   +---coregistration
|   |       CoregistrationGraph.xml
|   |       CreateStackGraph.xml
|   |       DEMAssistedCoregistrationGraph.xml
|   |       DEMAssistedCoregistrationXCorrGraph.xml
|   |       IntegerInterferogramGraph.xml
|   |       MultiInputStackAveragingGraph.xml
|   |       MultiOutputCoregister.xml
|   |       MultiOutputCreateStack.xml
|   +---FeatureExtractors
|   |       FloodDetectionGraph.xml
|   |       OilSpillDetectionGraph.xml
|   |       ShipDetectionGraph.xml
|   |       UrbanDetectionArchiveWriter.xml
|   |       UrbanDetectionGraph.xml
|   |       WindFieldEstimationGraph.xml
|   +---insar
|   |       SnaphuExportGraph.xml
|   |       SnaphuImportGraph.xml
|   |       stampsExportGraph.xml
|   |       StitchTileGraph.xml
|   |       UnwrapTileGraph.xml
```

```
+---internal
|   +---sentinel1
|   |       Sentinel1-BackGeocodingGraph.xml
|   |       Sentinel1-TOPS-Coregistration.xml
|   |       Sentinel1-TOPS-ESD-Coregistration.xml
|   |       Sentinel1-TOPSARMergeGraph.xml
|   |       Sentinel1SLCtoGRDGraph.xml
|   |       Sentinel1SliceAssemblyGraph.xml
|   +---wizards
|           Cal_ML_TC.xml
|           filtered_Interferogram.xml
|           T3_Spk_WishartGraph.xml
|           TerrainFlattenedT3.xml
|           unwrapped_phase_dem.xml
+---Radar
    +---ALOS Graphs
    |       ALOS_Cal_DSK_ML.xml
    |       ALOS_DSK_ML_SPK_RTC.xml
    +---InSAR Graphs
    |       BandSelect-Coreg-Interferogram-Filter.xml
    |       Deformation-Pre-Processing-Snaphu.xml
    |       Deformation-Pre-Processing.xml
    |       DEM-Generation-Pre-Processing-Snaphu.xml
    |       DEM-Generation-Pre-Processing.xml
    |       TOPSAR Coreg Interferogram IW All Swaths.xml
    |       TOPSAR Coreg Interferogram.xml
    |       TOPSAR Slices Coreg Interferogram.xml
    +---PolSAR Graph
    |       H-a Alpha Classification.xml
    +---Standard Graphs
            ApplyOrbit.xml
            Calibrate.xml
            Coregister.xml
            Multilook.xml
            Orthorectify.xml
            WishartClassifier.xml
```

- Linux, to run SNAP with those parameters, open a terminal and run

```
$SNAP_HOME/bin/gpt CalandML.xml -Pintput=~/Imagenes/CosmoSkymed/CS1.h5 -Pnl=4
-Poutput=~/Imagnes/CosmoSkymed/Calibradas/CS1.dim
```

  Where SNAP_HOME has the location of the SNAP home (For example `/opt/snap/`)

- Windows, open a terminal (*Command Prompt*) and execute the following

```
C:\Users\u1\gpt.bat CalandML.xml -Pintput="E:\Imagenes\CosmoSkymed\CS1.h5" -Pnl=4
 -Poutput="E:\Imagnes\CosmoSkymed\Calibradas\CS1.dim"
```

  The quotation marks are mandatory if there are space characters in the variables

- This command can be also executed from Matlab, Python, C++, batch script, etc

- For instance a batch DOS script ***calandml.bat***

```
rem Ejemplo de ejecucion de GPT en un Batch Script
set dirloc=%cd%
set input="E:\Imagenes\CosmoSkymed\CS1.h5"
set output="E:\Imagnes\CosmoSkymed\Calibradas\CS1.dim"
call gpt.bat "%dirloc%\CalyMul.xml" -Pinput=%input% -Poutput=%output% -Pnl=4
```

- A Linux shell script ***calandml.sh***

```
#!/bin/sh
# Ejemplo de ejecucion de GPT en un Shell Script
dirloc=`pwd`
snap_home=/opt/snap
input=$dirloc/Imagenes/CosmoSkymed/CS1.h5
output=$dirloc/Imagnes/CosmoSkymed/Calibradas/CS1.dim

$snap_home/bin/gpt "$dirloc/CalyMul.xml" -Pinput="$input" -Poutput="$output" -Pnl=4
```

- To find more information about gpt use:

```
$SNAP_HOME/bin/gpt -h
C:\Users\u1\gpt -h
```