Analyzing large radar datasets using Python

Robert Jackson¹, Scott Collis¹, Zach Sherman¹, Giri Palanisamy², Scott Giangrande³, Jitendra Kumar², Joseph Hardin⁴

UCAR Software Engineering Assembly 2018,
April 2, 2018
Boulder, CO

1. Argonne National Laboratory, Argonne, IL
2. Oak Ridge National Laboratory, Oak Ridge, TN
3. Brookhaven National Laboratory, Upton, NY
4. Pacific Northwest National Laboratory, Richland, WA
Motivation

- Ground based scanning radars collects years of data:
  - Datasets useful for statistical analysis of convection, GCM validation
- Datasets are on order of >100,000 files, >10 TB of data:
  - Need to map radar processing onto supercomputing cluster
- Tools exist in Python to map problem to supercomputing cluster.

Which tools do we use?
The Python ARM Radar Toolkit, Py-ART, is an open source Python module containing a growing collection of weather radar algorithms and utilities build on top of the Scientific Python stack and distributed under the 3-Clause BSD license.

Py-ART is used by the Atmospheric Radiation Measurement (ARM) Climate Research Facility and by others in the scientific community for working with data from a number of weather radars and instruments.

Py-ART is open source and hosted on GitHub at http://arm-doe.github.io/pyart/
print(__doc__)

# Author: Jonathan J. Helmus (jhelmus@anl.gov)
# License: BSD 3 clause

import matplotlib.pyplot as plt
import pyart

filename = 'XSW110520105408_RAW7HHF'

# create the plot using RadarDisplay (recommended method)
radar = pyart.io.read_rsl(filename)
display = pyart.graph.RadarDisplay(radar)
fig = plt.figure()
ax = fig.add_subplot(111)
display.plot('reflectivity', 0, vmin=-32, vmax=64.)
display.plot_range_rings([10, 20, 30, 40])
display.plot_cross_hair(5.)
plt.show()
Corrected Moments in Antenna Coordinates 2.0

- Python ARM Radar Toolkit/CSU Radar Tools used for visualization, processing

CMAC2.0 processes data + provides quicklooks from XSAPR @ ARM SGP:

<table>
<thead>
<tr>
<th>Operation</th>
<th>Methodology</th>
</tr>
</thead>
<tbody>
<tr>
<td>Hydrometeor ID</td>
<td>CSU fuzzy logic</td>
</tr>
<tr>
<td>Clutter filter</td>
<td>Texture + reflectivity stats</td>
</tr>
<tr>
<td>Phase processing</td>
<td>Giangrande et al. (2014)</td>
</tr>
<tr>
<td>Dealiasing</td>
<td>Region based from Py-ART</td>
</tr>
<tr>
<td>Attenuation correction</td>
<td>Gu et al (2011) Z-phi</td>
</tr>
<tr>
<td>Rainfall rate retrieval</td>
<td>Ryzhkov et al. (2014)</td>
</tr>
</tbody>
</table>
CMAC2.0 available at: https://github.com/EVS-ATMOS/cmac2.0
Stratus cluster

Cluster hosted at Oak Ridge National Laboratory

- 1080-core computing cluster for ARM investigators and users of ARM data
- 241-node Cray cluster w/7.68 GB DDR4 memory per core.
- Two Intel Xeon E5-2697V4 processors/core (18 cores per processor, 36 cores per node).
- 57.6 TB fast Solid State Drive (SSD) storage
- 100 TB parallel Lustre filesystem storage.
MapReduce

CMAC2.0

Node → Radar file @ t₀
Node → Radar file @ t₁
Node → Radar file @ t₂
Node → Radar file @ t₃
Node → Radar file @ t₄
Node → Radar file @ tₙ

Desired statistics
Designed to interact with NumPy/SciPy/Pandas.

Advantages:

• Easy integration w/CMAC2.0 using a *bag* to do MapReduce.
• Low overhead/latency

Disadvantages:

• Limited to Python, no high level optimization

Source: dask documentation
import dask.bag as db
from distributed import Client
client = Client(scheduler_file='myfile.json')
the_bag = db.from_sequence(radar_files)
run_cmac = lambda file_name:
run_cmac_and_plotting(file_name,sounding_time,args)
result = the_bag.map(run_cmac).compute()
CMAC2.0 performance (go to dask profiler)

CMAC2.0 run on 1 month (~4200 files) of PPIs of XSAPR i5 data
Gridding run on 4,000 files from KHGX NEXRAD radar using Bebop (36 cores/node – 4 GB ram/node). Dask used.
Example of a reduction!

18 years of CPOL data! ETHs > 7 km in convection (~250,000 radar files, 4 yrs 9 mos)
Conclusions/Future work

Python can be used to easily processes thousands of radar files within hours!

250,000 radar files analyzed using Python and Dask!

Evaluate performance with quasi-vertical profiles

Contact: rjackson@anl.gov