Employing Task Parallelism to Facilitate Dynamic Comparison of Model Output

Hannah Aizenman (CUNY Graduate Center, NOAA-CREST at CCNY)

Mentors: Dave Brown, Mary Haley (NCAR-CISL/VETS)
Contributors: Sheri Mickelson (ANL), Susan Bates, Gokhan Danabasoglu (NCAR-CGD)
OMWG POP Diagnostics

\[ D_V(\phi) = \delta(z) \delta(\tau) \]

\[ = \frac{1}{dz_k} \left( \frac{k_{k-\frac{1}{2}}(\phi_{k-1} - \phi_k)}{dz_{k-\frac{1}{2}}} - \frac{k_{k+\frac{1}{2}}(\phi_k - \phi_{k+1})}{dz_{k+\frac{1}{2}}} \right). \]

\[ dV_T D[T] = \Delta_x \left[ \left( \frac{h_x h_z}{h_z} \right) \bar{K}^2 \Delta_x T \right] + \Delta_x \left[ \left( \frac{h_z}{h_z} \right) \Delta_x (\gamma h_z \bar{K} \Delta_x T) \right] + \Delta_y \left[ \left( \frac{h_z}{h_z} \right) \bar{K}^2 \Delta_y T \right] + \Delta_y \left[ \left( \frac{h_z}{h_z} \right) \Delta_y (\gamma h_z \bar{K} \Delta_y T) \right] \]
Swift POP Diagnostics

- Parallelizes execution of NCL scripts
- 3X speed up over Non-Swift version
Only supported on DASG machines
Even on DASG...
NCAR’s mission is “to support, enhance, and extend the capabilities of the university community and the broader scientific community, nationally and internationally”

- http://ncar.ucar.edu/about-ncar
The Community Earth System Model (CESM) is a fully-coupled, global climate model that provides state-of-the-art computer simulations of the Earth's past, present, and future climate states.

CESM is sponsored by the National Science Foundation (NSF) and the U.S. Department of Energy (DOE). Administration of the CESM is maintained by the Climate and Global Dynamics Division (CGD) at the National Center for Atmospheric Research (NCAR).
Service Oriented Architecture

- User's Web Browser
- User's Web Browser
- The Internet json, http
- javascript css/html
- Services/Pyramid
- data
- diagnostic
- data db
- data server (glade, hpss, etc)
- Pydiag - python interface to pop diagnostics
Why SOA?

- **Services stay independent**
  - Diagnostics can be added independent of other services
  - New services can use old ones without changing the old ones
- **Multiple clients can talk to the same server**
  - CLI, mobile, desktop, …
  - Can keep old UI up while rolling out new one
- **Clients and services do not need to be on the same machine**
- **Slow services (the diagnostics) do not interfere with fast ones (data, observations)**
RESTful SOA

- time-series
- diagnostic
- comparison

GET /diagnostic?type=popdiag

- diagnostic service

HTTP/1.1 200 OK
Vary: Accept
Content-Type: text/javascript

{"p1": 'plot 1', 'p2': 'plot 2', ...}
Start a run

POST /diagnostic

diagnostic

creates

6f6365616e73
Get the results

GET /diagnostic/6f6365616e73?format=archive

6f6365616e73.tar.gz

diagnostics

GET /diagnostic/6f6365616e73?format=json

HTTP/1.1 200 OK
Vary: Accept
Content-Type: text/javascript

[mik.ucar.edu/results/6f6365616e73/p1.gif, mik.ucar.edu/results/6f6365616e73/f2.txt, ...]
URL->File

POST /diagnostics

validates inputs and returns responses

pass data from DB into pydiag library

pydiag - interfaces with original ncl/csh/awk scripts
Why pydiag?
Automate running the diagnostics

- Turn user defined settings into inputs
  - case/run, observations, years, and plots
  - Inputs are python dictionaries

- Generate Swift configuration files
  - sites.xml changes with every new output directory
  - sites.xml and tc.data are machine dependent
  - Configs are dictionaries and lists of dicts
    - Files generate xml, tsv, cdm, and configuration

- Isolate dependencies

- Turn it into an importable library
```python
def __pm_fld3dza():
    """3D fields (zonal average)"

    Returns:
    ncl script for generating zonal average
    """
    return ['field_3d_za.ncl']

@preserve_cwd
def pm_fld3dza_no_swift(workdir, data, obs):
    """3D fields (zonal average)"

    Parameters:
    workdir: str
        /path/to/workdir
    data: dict
        {'msroot': '/path/to/input/data' 'case': input data case 'years': (YEAR0, YEAR1), 'cpl': 7 if CESM4 or higher, else 6 'resolution': the ocean model resolution 'clindir': path/to/_OBS/phc/POP_format'
    obs: dict
        {'tsobsdir': 'full/path/to/obs' 'tobsfile': 'temperature' 'sobsfile': 'salt'}
    
    tavg = futils.TAVGFILE(data['years'])
    poputils.get_za_tavg(workdir, tavg)
    os.chdir(workdir)
    for fl in [obs['tobsfile'], obs['sobsfile']]:
        if os.path.exists('za_%s' % fl):
            continue
        temp = 'za_%s_tmp' % fl
        shutil.copy(os.path.join(obs['tsobsdir'], fl), temp)
        subprocess.call(['ncks', '-A', '-v', 'UAREA', tavg, temp])
        subprocess.call([os.path.join(futils.TOOLPATH, 'za'),
                '-0', '-time_const', temp])
        if os.path.exists('za_%s_tmp' % fl):
            os.rename('za_%s_tmp' % fl, 'za_%s' % fl)
            os.remove(temp)
    poputils.do_plots(workdir, __pm_fld3dza())
    return
```
Deployment

• Requires any libraries used by the diagnostics

• Self-contained environment
  o Isolated through use of virtualenv
  o All the packages are in PyPi
  o Only other dependencies
    • libevent and access to the data servers (glade, hpss)
  o Works better on cluster

• Maintenance and extensibility:
  o Modular architecture
  o Unit, integration, and functional regression tests
  o Documentation can be compiled into web pages, books, etc

• pydiag, which is stand-alone, is compatible with Python2.4 through Python2.7
Server Side

mik.ucar.edu
Use any of the documented/sample urls
Client Side

- Observational Comparison
- Time-Series
- Model Comparison


Start: __________ End: __________

- ENSO wavelet plots
- Yeager's POP log file line plots
- CPL log entry budget line plots
- Annual mean max MOC time series
- Monthly mean max MOC time series
- generate annual means
- Regional Mean Temp & Salt (z,t) with diff&rms from obs

Submit
Acknowledgments

• Dave Brown and Mary Haly (NCAR/CISL)
• Sheri Mickelson (ANL)
• Susan Bates and Gokhan Danabasoglu (NCAR/CGD)
• Paul Goodman and Nathan Wilhelmi (NCAR-CISL)
• Wei Huang and Rick Brownrigg (NCAR/CISL)