Simple Task Parallelism with Python

Sheri Mickelson ASAP/CISL
Task Parallel Examples

Calling the same NCL script for different input files ...

```
ncl 'srcFileName="file_1.nc"' regrid.ncl
ncl 'srcFileName="file_2.nc"' regrid.ncl
ncl 'srcFileName="file_3.nc"' regrid.ncl
ncl 'srcFileName="file_4.nc"' regrid.ncl
ncl 'srcFileName="file_5.nc"' regrid.ncl
ncl 'srcFileName="file_6.nc"' regrid.ncl
ncl 'srcFileName="file_7.nc"' regrid.ncl
ncl 'srcFileName="file_8.nc"' regrid.ncl
```

All of these lines can all be executed at the same time because they are all independent of each other.
Task Parallel Examples

Because none of these tasks rely on each other, we can run them all at the same time.

If ran serially, the cost would be $8 \times N$.
If ran in parallel and each ncl call had its own MPI task, the cost would be $N$. 

ncl 'srcFileName="file_1.nc"' regrid.ncl → MPI Task
ncl 'srcFileName="file_2.nc"' regrid.ncl → MPI Task
ncl 'srcFileName="file_3.nc"' regrid.ncl → MPI Task
ncl 'srcFileName="file_4.nc"' regrid.ncl → MPI Task
ncl 'srcFileName="file_5.nc"' regrid.ncl → MPI Task
ncl 'srcFileName="file_6.nc"' regrid.ncl → MPI Task
ncl 'srcFileName="file_7.nc"' regrid.ncl → MPI Task
ncl 'srcFileName="file_8.nc"' regrid.ncl → MPI Task
Task Parallel Examples

Calling different independent NCL scripts ...

ncl plot_SALT.ncl
ncl plot_ADVS.ncl
ncl plot_UES.ncl
ncl plot_UVEL.ncl
ncl plot_VVEL.ncl
ncl plot_BSF.ncl
ncl plot_TAUUX.ncl
ncl plot_TAUUY.ncl

All of these lines can all be executed at the same time because they are all independent of each other.
Task Parallel Examples

Calling different independent NCL scripts ...

ncl plot_SALT.ncl → MPI Task
ncl plot_ADVS.ncl → MPI Task
ncl plot_UES.ncl → MPI Task
ncl plot_UVEL.ncl → MPI Task
ncl plot_VVEL.ncl → MPI Task
ncl plot_BSF.ncl → MPI Task
ncl plot_TAUX.ncl → MPI Task
ncl plot_TAUY.ncl → MPI Task

If ran serially, the cost would be 8 x N
If ran in parallel and each ncl call had its own MPI task, the cost would be N
The Messenger Library

- Python library written by the ASAP/CISL Group
- Contains wrappers around commonly used MPI techniques eliminating the need for users to know MPI
- Contains functions for partitioning, sync, rank, master, collection
- Available in the public ASAP repository:
  svn export
  https://proxy.subversion.ucar.edu/pubasap/pyTools/tags/v0.1/
We need to do:
Regrid every NetCDF file in a directory

What the script will do:
Collect all of the file names that need to be regridded, give each MPI task an equal number of files to work on, and then call the same NCL regridding script for each file.
Task Parallel Example

Calling an NCL script several times, but just changing an input value

```python
#!/usr/bin/env python
import glob,sys,os
from messenger import create_messenger

data_dir = "/Users/test/data/"
casename_prefix = "my_case"

# First glob the directory for all NetCDF files with that prefix
glob_string = data_dir+'/'+casename_prefix+'*.nc'
global_file_list = []
for path in  glob.glob(glob_string):
    dirname,name = os.path.split(path)
    global_file_list.append(name)

# Initialize the messenger class
messenger = create_messenger(serial=False)

# Get a local file list that this rank is responsible for
local_file_list = messenger.partition(global_file_list)

# For each file in my local list, execute the regridding script
for f in local_file_list:
    command = 'ncl '+'srcFileName='+data_dir+f+'
    os.system(command)
```
Task Parallel Example
Calling an NCL script several times, but just changing an input value

#!/usr/bin/env python
import glob,sys,os
from messenger import create_messenger

data_dir = "~/Users/test/data/"
casename_prefix = "my_case"

# First glob the directory for all NetCDF files with that prefix
glob_string = data_dir+'/'+casename_prefix+'*.nc'
global_file_list = []
for path in glob.glob(glob_string):
    dirname,name = os.path.split(path)
    global_file_list.append(name)

# Initialize the messenger class
messenger = create_messenger(serial=False)

# Get a local file list that this rank is responsible for
local_file_list = messenger.partition(global_file_list)

# For each file in my local list, execute the regridding script
for f in local_file_list:
    command = 'ncl -c "srcFileName="'+data_dir+f+'"" < regrid.ncl"
    os.system(command)
Task Parallel Example
Calling an NCL script several times, but just changing an input value

#!/usr/bin/env python
import glob, sys, os
from messenger import create_messenger

data_dir = "/Users/test/data/"
casename_prefix = "my_case"

# First glob the directory for all NetCDF files with that prefix
glob_string = data_dir+"/"+casename_prefix+"*.nc"
global_file_list = []
for path in glob.glob(glob_string):
    dirname, name = os.path.split(path)
    global_file_list.append(name)

# Initialize the messenger class
messenger = create_messenger(serial=False)

# Get a local file list that this rank is responsible for
local_file_list = messenger.partition(global_file_list)

# For each file in my local list, execute the regridding script
for f in local_file_list:
    command = "ncl srcFileName="+data_dir+f" < regrid.ncl"
    os.system(command)
Task Parallel Example
Calling an NCL script several times, but just changing an input value

#!/usr/bin/env python
import glob,sys,os
from messenger import create_messenger

data_dir = "/Users/test/data/"
casename_prefix = "my_case"

# First glob the directory for all NetCDF files with that prefix
glob_string = data_dir+'/'+casename_prefix+'*.nc'
global_file_list = []
for path in glob.glob(glob_string):
    dirname,name = os.path.split(path)
    global_file_list.append(name)

# Initialize the messenger class
messenger = create_messenger(serial=False)

# Get a local file list that this rank is responsible for
local_file_list = messenger.partition(global_file_list)

# For each file in my local list, execute the regridding script
for f in local_file_list:
    command = "ncl 'srcFileName="'+data_dir+f+'"'+' < regrid.ncl"
    os.system(command)
Task Parallel Example
Calling an NCL script several times, but just changing an input value

```python
#!/usr/bin/env python
import glob, sys, os
from messenger import create_messenger

data_dir = "~/Users/test/data/"
casename_prefix = "my_case"

# First glob the directory for all NetCDF files with that prefix
glob_string = data_dir+'/'+casename_prefix+'*.nc'
global_file_list = []
for path in glob.glob(glob_string):
    dirname,name = os.path.split(path)
    global_file_list.append(name)

# Initialize the messenger class
messenger = create_messenger(serial=False)

# Get a local file list that this rank is responsible for
local_file_list = messenger.partition(global_file_list)

# For each file in my local list, execute the regridding script
for f in local_file_list:
    command = 'ncl '+'\'"'+data_dir+f+'\'"'+" < regrid.ncl"
    os.system(command)
```
Task Parallel Example
Calling an NCL script several times, but just changing an input value

#!/usr/bin/env python
import glob,sys,os
from messenger import create_messenger

data_dir = "~/Users/test/data/"
casename_prefix = "my_case"

# First glob the directory for all NetCDF files with that prefix
glob_string = data_dir+'/'+casename_prefix+'*.nc'
global_file_list = []
for path in  glob.glob(glob_string):
dirname,name = os.path.split(path)
global_file_list.append(name)

# Initialize the messenger class
messenger = create_messenger(serial=False)

# Get a local file list that this rank is responsible for
local_file_list = messenger.partition(global_file_list)

# For each file in my local list, execute the regridding script
for f in local_file_list:
    command = 'ncl '+'"srcFileName="'+data_dir+f+'""\"+" < regrid.ncl"
    ### Command will look like:
    ### ncl 'srcFileName="/Users/test/data/my_case_1.nc"' < regrid.ncl
    os.system(command)
A More Complicated Workflow

We need to do:
Create NetCDF files for each variable in a list and then create plots that use these new NetCDF files.

What the script will do:
Divide a variable list between the MPI tasks, loop over each variable to create NetCDF files, sync all MPI tasks, and then loop over each variable to create plots.
#!/usr/bin/env python
import glob,sys,os
from messenger import create_messenger

# Initialize the messenger class
messenger = create_messenger(serial=False)

# Create some sort of list
var_list = ["SALT","ADVS","UES","UVEL","VVEL","BSF","TAUX","TAUY"]

# Divide the list up and print out how everything was divided
local_var_list = messenger.partition(var_list)

# Create NetCDF Files
for var in local_var_list:
    # Command will look like: ncl 'var_name="BSF"' < createNetcdf.ncl
    os.system("ncl 'var_name=""+var+""' < createNetcdf.ncl")

# Wait for all procs to get to this point
messenger.sync()

# Create Plots
for var in local_var_list:
    ncl_file = 'plot_'+var+'.ncl'
    os.system("ncl < " + ncl_file)
A More Complicated Workflow

#!/usr/bin/env python
import glob,sys,os
from messenger import create_messenger

# Initialize the messenger class
messenger = create_messenger(serial=False)

# Create some sort of list
var_list = ["SALT","ADVS","UES","UVEL","VVEL","BSF","TAUX","TAUY"]

# Divide the list up and print out how everything was divided
local_var_list = messenger.partition(var_list)

# Create NetCDF Files
for var in local_var_list:
    # Command will look like: ncl 'var_name="BSF"' < createNetcdf.ncl
    os.system("ncl 'var_name="\""+var+"\""' < createNetcdf.ncl")

# Wait for all procs to get to this point
messenger.sync()

# Create Plots
for var in local_var_list:
    ncl_file = 'plot_'+var+'.ncl'
    os.system("ncl < " + ncl_file)
A More Complicated Workflow

#!/usr/bin/env python
import glob,sys,os
from messenger import create_messenger

# Initialize the messenger class
messenger = create_messenger(serial=False)

# Create some sort of list
var_list = ["SALT","ADVS","UES","UVEL","VVEL","BSF","TAUX","TAUY"]

# Divide the list up and print out how everything was divided
local_var_list = messenger.partition(var_list)

# Create NetCDF Files
for var in local_var_list:
    # Command will look like: ncl 'var_name="BSF"' < createNetcdf.ncl
    os.system("ncl 'var_name=""+var+""' < createNetcdf.ncl")

# Wait for all procs to get to this point
messenger.sync()

# Create Plots
for var in local_var_list:
    ncl_file = 'plot_'+var+'.ncl'
    os.system("ncl < " + ncl_file)
A More Complicated Workflow

#!/usr/bin/env python
import glob,sys,os
from messenger import create_messenger

# Initialize the messenger class
messenger = create_messenger(serial=False)

# Create some sort of list
var_list = ["SALT","ADVISING","UES","UVEL","VVEL","BSF","TAUX","TAUY"]

# Divide the list up and print out how everything was divided
local_var_list = messenger.partition(var_list)

# Create NetCDF Files
for var in local_var_list:
    # Command will look like: ncl 'var_name="BSF"' < createNetcdf.ncl
    os.system("ncl 'var_name=""+var+""' < createNetcdf.ncl")

# Wait for all procs to get to this point
messenger.sync()

# Create Plots
for var in local_var_list:
    ncl_file = 'plot_'+var+'.ncl'
    os.system("ncl < " + ncl_file)
#!/usr/bin/env python
import glob, sys, os
from messenger import create_messenger

# Initialize the messenger class
messenger = create_messenger(serial=False)

# Create some sort of list
var_list = ["SALT", "ADVS", "UES", "UVEL", "VVEL", "BSF", "TAUX", "TAUY"]

# Divide the list up and print out how everything was divided
local_var_list = messenger.partition(var_list)

# Create NetCDF Files
for var in local_var_list:
    # Command will look like: ncl 'var_name="BSF"' < createNetcdf.ncl
    os.system("ncl 'var_name="\""+var+"\""' < createNetcdf.ncl")

    # Wait for all procs to get to this point
    messenger.sync()

# Create Plots
for var in local_var_list:
    ncl_file = 'plot_' + var + '.ncl'
    os.system("ncl < " + ncl_file)
#!/usr/bin/env python
import glob,sys,os
from messenger import create_messenger

# Initialize the messenger class
messenger = create_messenger(serial=False)

# Create some sort of list
var_list = ['SALT', 'ADV', 'UES', 'UVEL', 'VVEL', 'BSF', 'TAUX', 'TAUY']

# Divide the list up and print out how everything was divided
local_var_list = messenger.partition (var_list)

# Create NetCDF Files
for var in local_var_list:
    # Command will look like: ncl 'var_name="BSF"' < createNetcdf.ncl
    os.system("ncl 'var_name=""+var+""' < createNetcdf.ncl")

# Wait for all procs to get to this point
messenger.sync()

# Create Plots
for var in local_var_list:
    ncl_file = 'plot_' + var + '.ncl'
    os.system("ncl < " + ncl_file)
#!/usr/bin/env python
import glob, sys, os
from messenger import create_messenger

# Initialize the messenger class
messenger = create_messenger(serial=False)

# Create some sort of list
var_list = ["SALT", "ADV", "UES", "UVEL", "VVEL", "BSF", "TAUX", "TAUY"]

# Divide the list up and print out how everything was divided
local_var_list = messenger.partition(var_list)

# Create NetCDF Files
for var in local_var_list:
    # Command will look like: ncl 'var_name="BSF"' < createNetcdf.ncl
    os.system("ncl 'var_name=""+var+""' < createNetcdf.ncl")

# Wait for all procs to get to this point
messenger.sync()

# Create Plots
for var in local_var_list:
    ncl_file = 'plot_'+var+'.ncl'
    os.system("ncl < " + ncl_file)
#!/bin/csh
#BSUB -n 4
#BSUB -q geyser
#BSUB -N
#BSUB -W 1:00
#BSUB -R "span[ptile=8]"
#BSUB -P MY_PROJ_
#BSUB -o python_regrid.%J.out
#BSUB -e python_regrid.%J.err

module load python
module load all-python-libs

mpirun.lsf ./your_python_script.py
Questions?

svn export https://proxy.subversion.ucar.edu/pubasap/pyTools/tags/v0.1/

Thanks to Kevin Paul and John Dennis (CISL/ASAP) for their help in creating the Messenger Python Library