Toy coupled model for C-Coupler2 with DiRong User's Guide

Hao Yu, Li Liu, Ruizhe Li

yuh18@mails.tsinghua.edu.cn
liuli-cess@tsinghua.edu.cn
Ministry of Education Key Laboratory for Earth System Modeling,
Department for Earth System Science (DESS),
Tsinghua University, Beijing, China

4th, August, 2020

1 Run the C-Coupler evaluation code

1.1 Download source code and unzip

Before the evaluation, users can download the platform code using the DOI address mentioned in the manuscript. After downloading, users can get the compressed package "*DiRong_test_platform-v1.x.zip*" and run command "*unzip DiRong_test_platform-v1.x.zip*". The *DiRong_test_top_model* in unzipped directory is the working directory of the C-Coupler platform.

1.2 Directory structure

The working directory of the C-Coupler platform consists of three directories: the *model platform, input data* and *working directory*. The *model platform* contains three main directories: *models*, *scripts*, and *config,* to store the configuration information, source code and scripts for operating the simulations. The *input data* contains several sub directories to respectively store the input data for component models. The *working directory* (here called *model_experiments*) is the directory for users to configure, compile and run the toy coupled model.

1.3 Configure, compile and run the toy coupled model

1.3.1 Configure the toy coupled model

Before configure the toy coupled model, users should run command "*source source_env.sh*" under the working directory. *source_env.sh* encapsulates two scripts "*model_platform/scripts/register_platform.sh*" and "*inputdata/register_inputdata.sh*" to register some setting environment variables of the C-Coupler platform.

To configure the toy coupled model, users should run command "*./configure*" under the working directory. During the execution of the script, users may need to enter the path of netcdf and mpi.

1.3.2 Compile the toy coupled model

Users need to first modify the mpi and netcdf paths in the *build.sh* in the *model_platform/models/libs/c_coupler/build*. This is the build script used to compile C-Coupler2. To compile the toy coupled model, users should run the command "*./compile*" under the working directory. The log files of the compilation are stored under the sub directory *run* of the working directory.

Users can run the command "*./clean para*" to remove the files produced by the compilation, where *para* means the parameter for *clean*. The parameter can be the name of a component model or a library, or *all*. For command "*./clean all*", all files produced by compilation will be removed.

1.3.3 Run the toy coupled model

To run the toy coupled model, users should run the command "*./runcase*" under the working directory. Note that, "*./runcase*" is common to any kind of simulations on any kind of hardware platforms.

1.4 Evaluation settings

Our evaluation parameters mainly involve *grid size* and *number of processor cores (processes)*. Variables *num_total_proc* in the section *model* of file *config/common/case.conf* specify the processor cores of the corresponding component model. User can modify the variables *num_total_proc* when you want to change the parallel setting of the corresponding component model, as shown in Figure 1. Please configure the model simulation **after** modifying the processor cores.

```
atm_demo : atm : atm_demo
{
        cpl_interface_time_step=1200
        num_thread=1
        num_total_proc=8        // parallel setting of atm_demo
        stop_latency_seconds=0
}
ocn_demo : ocn : ocn_demo
{
        cpl_interface_time_step=1200
        num_thread=1
        num_total_proc=8        // parallel setting of ocn_demo
        stop_latency_seconds=0
}
```

Figure 1. part of *config/common/case.conf*

Variables *num_point* in the file *run/atm/atm_demo/data/atm_demo.nml* and *run/ocn/ocn_demo/data/ocn_demo.nml* specify the grid size of the corresponding component model. User can modify the variables *num_point* when you want to change the grid size of the corresponding component models, as shown in Figure 2.

There are some other parameters in the file that can be modified, such as *time_step, decomp_type_id*, *coupling_freq*, and *overlapping_rate*. *time_step* specify the time step of each component model. *decomp_type_id* specify the parallel decomposition type, such as 2-D decomposition. *coupling_freq* specify the coupling frequency of coupled model. *overlapping_rate* specify the overlapping rate of grids between different processes within the same component model. *read_ncfile* specify whether component model read grid file in netcdf format. If *read_ncfile* is true, the *grid_file_name* specify the grid file. Users can modify parameters as needed to evaluate DiRong algorithm.

```
& atm_demo_nml
  time_step = 1800
  decomp_type_id = 2
  coupling_freq = 1800
  num_point = 4000000
  overlapping_rate = 1
  read_ncfile=false
  grid_file_name=lonlat_grid_0.3.dat
  /
```

Figure 2. The contents of *run/atm/atm_demo/data/atm_demo.nml*

2 Run the MCT evaluation code

The *mct_test_model* in unzipped directory is the working directory of the MCT platform. It consists of three directories: the *model_platform, weight* and *log*. The *model_platform* contains the source code of component models. The *weight* contains several weight files for component models (Due to the file size limitation of the Github platform, the weight file is not uploaded. Users can use the weight file generated by C-Coupler to run the MCT code). The *log* is the directory to store log files.

2.1 Compile and run the MCT coupled model

Before compiling MCT coupled model, users need to compile MCT and generate *libmct.a* in *mct_test_model/model_platform/models/libs/MCT/mct*. Users can run the command *build.sh* to build the MCT coupled model and run the command *run.sh* to run the model in *mct_test_model*.

2.2 Evaluation Settings

The MCT evaluation parameters mainly involve *grid_size* (for atm_demo), *grid_file_name* (for ocn_demo) in name list and *number of processor cores* in bash script.

Variable *num_point* in the file *atm_demo.nml* specifies the grid size of the atm_demo component model. Variable *grid_file_name* in the file *ocn_demo.nml* specifies CUBE grid file used in ocn_demo component model. User can modify the corresponding variables when you want to change the grid size of the corresponding component models, as shown in Figure 3.

There are some other parameters in the file that can be modified, such as *time_step, decomp_type_id*, *coupling_freq*, and *overlapping_rate*. *time_step* specify the time step of each component model. *decomp_type_id* specify the parallel decomposition type, such as 2-D decomposition. *coupling_freq* specify the coupling frequency of coupled model. *overlapping_rate* specify the overlapping rate of grids between different processes within the same component model. *read_ncfile* specify whether component model read grid file in netcdf format. If *read_ncfile* is true, the *grid_file_name* specify the grid file. *weight_file_index* specify the index of weight file. Users can modify parameters as needed to evaluate MCT coupled model.

```
& atm_demo_nml
  time_step = 1800
  decomp_type_id = 2
  coupling_freq = 1800
  num_point = 4000000
  overlapping_rate = 1
  read_ncfile = false
  grid_file_name=lonlat_grid_0.3.dat
  weight_file_index=32
  /
```

Figure 3. The contents of *atm_demo.nml*