Table of Contents

How to run a simple experiment

Before you start this tutorial make sure that WRF4G is correctly installed on your machine. To do so follow the Installation instructions

Start wrf4g_framework and list computing resources

Firstly, check if WRF4G framework is running:

```
[user@localhost ~]$ wrf4g_framework status
DRM4G (GridWay) is NOT running
WRF4G_DB (MySQL) NOT running
```

if WRF4G framework is not running, you should execute:

```
[user@localhost ~]$ wrf4g_framework start
Starting DRM4G (GridWay) .... OK
Starting WRF4G_DB (MySQL) ... OK
```

By doing this, you will start the services that manage the CR and keep track of the experiments.

Note that if you do not start `wrf4g_framework`, you won not be able to work with WRF4G.

By default, WRF4G framework is configured to use the computer where WRF4G has been installed. In order to list the CR available, run the wrf4g_resources command:

```
[user@localhost ~]$ wrf4g_resources
HID PRIO OS              ARCH   NODES(U/F/T) LRMS       HOSTNAME
0   1    GNU/Linux2.6.32 x86_64        0/1/1 FORK       mycomputer
```

If you want to modify your CRs see Computing Resources section, which is located in `$HOME/WRF4G/etc/framework4g.conf` file.

Prepare and submit the test experiment

Go to the directory where the experiment configuration files are located:

```
[user@localhost ~]$ cd $HOME/WRF4G/experiments/single_test
```

Edit experiment.wrf4g and check its configuration. If you want to know more information about this experiment, you should see resources.wrf4g as well.

Run wrf4g_prepare to prepare the experiment:

```
[user@localhost ~]$ wrf4g_prepare
Preparing namelist...
WRFV3/run/namelist.input
WRF Check Warning: CAM radiation selected but paerlev/levsiz/cam_abs_dim1/cam_abs_dim2 was not set. Fixing...
WRF Check Warning: radt is shorter than dx (0.500000)

---> Single params run
---> Continuous run
       ---> cycle_chunks: test 2011-08-28_12:00:00 2011-08-30_00:00:00
              ---> chunks 1: test 2011-08-28_12:00:00 2011-08-29_00:00:00
              ---> chunks 2: test 2011-08-29_00:00:00 2011-08-29_12:00:00
              ---> chunks 3: test 2011-08-29_12:00:00 2011-08-30_00:00:00
```

Check the experiment status with wrf4g_status. You will see that the experiment is in P (Prepared) status.

```
[user@localhost ~]$ wrf4g_status
Experiment P   W   R   D   F
```

```
test       1   0   0   0   0

[user@localhost ~]$ wrf4g_status --long
Realization        GW  Stat Chunks Comp.Res   WN         Run.Sta       ext   %
test               -     P     0/3        -    -        Prepared       -  0.00
```

Submit the experiment with wrf4g_submit and check the status for a minute. You will see how the status of the experiment change into: submitted, downloading input data, ungrib, metgrid, real and wrf.

```
[user@localhost ~]$ wrf4g_submit
Submitting realization: "test"
       Submitting Chunk 1:     2011-08-28_12:00:00     2011-08-29_00:00:00
       Submitting Chunk 2:     2011-08-29_00:00:00     2011-08-29_12:00:00
       Submitting Chunk 3:     2011-08-29_12:00:00     2011-08-30_00:00:00

[user@localhost ~]$ wrf4g_status
Experiment P   W   R   D   F
test       0   1   0   0   0

[user@localhost ~]$ wrf4g_status --long
Realization        GW  Stat Chunks Comp.Res   WN         Run.Sta       ext   %
test               0     W  1/3    -          -          Submitted       - 0.00

[user@localhost ~]$ wrf4g_status --long
Realization        GW  Stat Chunks Comp.Res   WN         Run.Sta       ext   %
test               0     W  1/3    -          localhost  ungrib          - 0.00

[user@localhost ~]$ wrf4g_status --long
Realization        GW  Stat Chunks Comp.Res   WN         Run.Sta       ext   %
test               0     W  1/3    -          localhost  metgrid         - 0.00

[user@localhost ~]$ wrf4g_status --long
Realization        GW  Stat Chunks Comp.Res   WN         Run.Sta       ext   %
test               0     R  1/3    mycomputer localhost  real            - 0.00

[user@localhost ~]$ wrf4g_status --long
Realization        GW  Stat Chunks Comp.Res   WN         Run.Sta       ext   %
test               0     R  1/3    mycomputer localhost  WRF             - 0.00

[user@localhost ~]$ wrf4g_status --long
Realization        GW  Stat Chunks Comp.Res   WN         Run.Sta       ext   %
test               0     R  2/3    mycomputer localhost  Submitted       - 33.33


[user@localhost ~]$ wrf4g_status --long
Realization        GW  Stat Chunks Comp.Res   WN         Run.Sta       ext   %
test               0     R  3/3    mycomputer localhost  Finished        0 100.00
```

The experiment has to finish with **ext** option **0** which means it has finalized correctly. If **ext** has a different code, you can look up more information about it either in WRF4G errors or in `$HOME/WRF4G/lib/bash/wrf4g_exit_codes.sh` file

Check the experiment output and log

The location where output and log files are stored is defined with the `WRF4G_BASEPATH` variable in resources.wrf4g. Check `$HOME/WRF4G/etc/resources.wrf4g` to find where `$WRF4G_BASEPATH` is pointing. You will see that it is pointing to `$HOME/WRF4G/repository/output`. Go to `$HOME/WRF4G/repository/output/test/test` and see the directory structure and the output files.

```
[user@mycomputer~]$ ls -lh $HOME/WRF4G/repository/output/test/test/output
total 420K
```

```
-rw-rw-r-- 1 user user 6.2K 2011-10-28 12:40 wrf24hc_d01_20110828T120000Z_20110828T120000Z.nc
-rw-rw-r-- 1 user user 6.2K 2011-10-28 12:41 wrf24hc_d01_20110829T000000Z_20110829T000000Z.nc
-rw-rw-r-- 1 user user 6.2K 2011-10-28 12:43 wrf24hc_d01_20110829T120000Z_20110829T120000Z.nc
-rw-rw-r-- 1 user user 14K 2011-10-28 12:40 wrfout_d01_20110828T120000Z_20110828T180000Z.nc
-rw-rw-r-- 1 user user 11K 2011-10-28 12:40 wrfout_d01_20110828T210000Z_20110829T000000Z.nc
-rw-rw-r-- 1 user user 14K 2011-10-28 12:41 wrfout_d01_20110829T000000Z_20110829T060000Z.nc
-rw-rw-r-- 1 user user 11K 2011-10-28 12:41 wrfout_d01_20110829T090000Z_20110829T120000Z.nc
-rw-rw-r-- 1 user user 14K 2011-10-28 12:43 wrfout_d01_20110829T120000Z_20110829T180000Z.nc
-rw-rw-r-- 1 user user 11K 2011-10-28 12:43 wrfout_d01_20110829T210000Z_20110830T000000Z.nc
-rw-rw-r-- 1 user user 3.4K 2011-10-28 12:40 wrfrain_d01_20110828T190000Z_20110828T190000Z.nc
-rw-rw-r-- 1 user user 3.4K 2011-10-28 12:41 wrfrain_d01_20110829T070000Z_20110829T070000Z.nc
-rw-rw-r-- 1 user user 3.4K 2011-10-28 12:43 wrfrain_d01_20110829T190000Z_20110829T190000Z.nc
-rw-rw-r-- 1 user user 99K 2011-10-28 12:40 wrfxtrm_d01_20110828T120000Z_20110829T000000Z.nc
-rw-rw-r-- 1 user user 99K 2011-10-28 12:41 wrfxtrm_d01_20110829T000000Z_20110829T120000Z.nc
-rw-rw-r-- 1 user user 99K 2011-10-28 12:43 wrfxtrm_d01_20110829T120000Z_20110830T000000Z.nc

[user@localhost ~]$ tree $HOME/WRF4G/repository/output/test/test
.
??? log
?   ??? log_1_1.tar.gz
?   ??? log_2_2.tar.gz
?   ??? log_3_3.tar.gz
??? namelist.input
??? output
?   ??? wrfout_d01_19830825T120000Z_19830825T233000Z.nc
?   ??? wrfout_d01_19830826T000000Z_19830826T000000Z.nc
?   ??? wrfout_d01_19830826T000000Z_19830826T113000Z.nc
?   ??? wrfout_d01_19830826T120000Z_19830826T120000Z.nc
?   ??? wrfout_d01_19830826T120000Z_19830826T233000Z.nc
?   ??? wrfout_d01_19830827T000000Z_19830827T000000Z.nc
?   ??? wrfrain_d01_19830826T000000Z_19830826T000000Z.nc
?   ??? wrfrain_d01_19830826T120000Z_19830826T120000Z.nc
?   ??? wrfrain_d01_19830827T000000Z_19830827T000000Z.nc
?   ??? wrfxtrm_d01_19830825T120000Z_19830825T233000Z.nc
?   ??? wrfxtrm_d01_19830826T000000Z_19830826T000000Z.nc
?   ??? wrfxtrm_d01_19830826T000000Z_19830826T113000Z.nc
?   ??? wrfxtrm_d01_19830826T120000Z_19830826T120000Z.nc
?   ??? wrfxtrm_d01_19830826T120000Z_19830826T233000Z.nc
?   ??? wrfxtrm_d01_19830827T000000Z_19830827T000000Z.nc
??? realout
??? restart
   ??? wrfrst_d01_19830826T000000Z.nc
   ??? wrfrst_d01_19830826T120000Z.nc
   ??? wrfrst_d01_19830827T000000Z.nc
```

In order to improve your knowledge about **WRF4G**, move on to the second part of this tutorial