Compile your MATLAB scripts and submit as a batch job

This tutorial details the procedure to compile your MATLAB script and launch it as an OAR batch job on IGRIDA.

Logging In

Start by connecting the IGRIDA frontend using SSH:

```
mymachine%ssh mylogin@igrida-oar-frontend
```

When at the IGRIDA front-end, start an interactive job:

```
igrida-oar-frontend%oarsub -I
```

Preparing the Scripts

Let us assume that your main MATLAB function is called `main_foo`. This function has two input parameters and calls another function called `sec_foo`. These two functions are stored in your home directory `/udd/mylogin`. However, any output must me stored (for example in .mat files) on `/temp_dd/igrida-fs1/mylogin/` and never on your NFS home directory.

main_foo.m

```
function main_foo(x,y)
a=x+y;
c=sec_foo(a,x);
matFilename = sprintf('/temp_dd/igrida-fs1/mylogin/results.mat');
save(matFilename,a,b,c);
```

sec_foo.m

```
function c = sec_foo(a,x)
c=a*x;
```

In order to compile your scripts, you can use the `mcc` compiler. Note that you should limit MATLAB to a single computational thread (by default, MATLAB makes use of the multithreading capabilities of the computer on which it is running). All your MATLAB scripts should be given as arguments starting with `main_foo.m`. The syntax of the `mcc` command is given hereafter while in the interactive OAR session:

```
igrida01-01%module load matlab
```

```
igrida01-01%/soft/matlab_hd/R2012b/bin/mcc -R -singleCompThread -m
main_foo.m sec_foo.m
```

Now, you have got an executable called main_foo that you test on the command line by typing:

```
igrida01-01%./main_foo 10 20
```

You can now exit the interactive OAR session and get back to the frontend.

Preparing the OAR jobs

Let us suppose that you need to run hundreds of simulations of your MATLAB compiled script. You can choose to modify your MATLAB scripts and add iterating loops (then recompile). Then, you can follow this tutorial to run your job. However, with this method, you launch a single OAR job and do not take full advantage of the computing facilities on IGRIDA. In order to run multiple jobs, you can write a launcher script as in `oar_launcher.sh`. This script outputs a set of jobs in separated script files called job*.sh. In each job file, you call the compiled main_foo with incrementing parameters. Note that you may need to change the walltime and the number of cores. The launcher ends by submitting all the created jobs.

oar_launcher.sh

```
#!/bin/bash
for i in {1..10};
do
    for j in {1..10};
    do
        echo "#!/bin/bash " > job$i$j.sh
        echo "source /etc/profile.d/modules.sh" >> job$i$j.sh
        echo "module load matlab" >> job$i$j.sh
        echo "#OAR -l core=10,walltime=3:00:00" >> job$i$j.sh
        echo "#OAR -O /temp_dd/igrida-
fs1/mylogin/log/job.%jobid%.output" >> job$i$j.sh
        echo "#OAR -E /temp_dd/igrida-
fs1/mylogin/log/job.%jobid%.error" >> job$i$j.sh
        echo 'echo "My job was ran on these nodes:"' >> job$i$j.sh
        echo 'cat $OAR_NODEFILE' >> job$i$j.sh
        echo "#Setup MCR cache directory locally" >> job$i$j.sh
        echo 'export
MCR_CACHE_ROOT=/tmp/mcr_cache_${USER}_OAR_JOBID_${OAR_JOBID}' >>
job$i$j.sh
        echo 'mkdir -p $MCR_CACHE_ROOT' >> job$i$j.sh
        echo "#cd to your execution directory first" >> job$i$j.sh
        echo "cd /udd/mylogin/" >> job$i$j.sh
        echo "./main_foo $i $j" >> job$i$j.sh
        echo "#Remove temporary MCR cache directory" >> job$i$j.sh
        echo '/bin/rm -rf $MCR_CACHE_ROOT' >> job$i$j.sh
    done;
done;
```

```
for k in job*;
do
    chmod +x $k;
    oarsub -S $k;
done;
```

Do not forget to create your log directory:

```
igrida-oar-frontend%mkdir -p /temp_dd/igrida-fs1/mylogin/log/
```

Prepare you OAR launch script to be executed:

```
igrida-oar-frontend%chmod +x oar-launcher.sh
```

Launching the Job

You can now launch your jobs by simply calling your launcher script as in the following:

```
igrida-oar-frontend%./oar-launcher.sh
[ADMISSION RULE] Modify resource description with type constraints
[ADMISSION RULE] Job walltime greater than 4 hours, adding property
duration_weight > 2 (large job).
Generate a job key...
OAR_JOB_ID=666666
[ADMISSION RULE] Modify resource description with type constraints
[ADMISSION RULE] Job walltime greater than 4 hours, adding property
duration_weight > 2 (large job).
Generate a job key...
OAR_JOB_ID=777777
```

Step back and watch your jobs being successively created then executed.

Verifying the Output

If you need to follow the execution of your job, you can check the status of all your jobs:

```
igrida-oar-frontend%oarstat -f | grep mylogin
```

and verify the output or the errors generated using a specific job ID:

```
igrida-oar-frontend%more /temp_dd/igrida-fs1/mylogin/log/job.jobID.output
igrida-oar-frontend%more /temp_dd/igrida-fs1/mylogin/log/job.jobID.error
```