Using Julia on the Submit cluster

Workshop on Basic Computing Services
in the Physics Department

January 6, 2023

Washington (Wati) Taylor, MIT

Research problems requiring extensive computing:

(1) Exploring exponentially large space of geometries for string theory vacua, by, e.g., (1a) Systematic exploration in complexity parameter, (1b) Monte Carlo approaches

(2) Dynamics of complex ecological systems related to (2a) discrete stochastic lattice models (complicated non-deterministic CAs), (2b) generalized LV ODE systems with complex structure of equilibria ($\sim$ matrix models in glassy regime)

Computational approach:

– Run thousands of CPU-days on Submit condor cluster

– Use Julia as computing language

Note: Have been using the Tier-2/submit system sporadically over the last couple of years. With some help from Zhangqier and Christoph, everything has gone very smoothly!

Julia?

Julia is a relatively new high-performance language for scientific computing, developed in part by folks at MIT (Edelman, etc.)

– High level language, easy to program like python

– Uses “just-in-time” compilation for higher speed, high performance

– Dynamically typed, good interactive interface

– Various libraries coded at low level, compatible with parallel systems

– First use on Tier-2/Submit!

Workflow

0. Files and account based on `submit.mit.edu` and relatives (`submit01`, etc.).

```
% ssh -l wati submit.mit.edu
% pwd
/home/submit/wati
% cd example-project
```

1. Can use a script to generate submission file `submit_file` for N separate jobs

2. Submission file submitted to condor, each of N jobs runs with different inputs, generates different output files

3. Move output files to your usual machine, work with data.

A simple example: 1. Julia code

Some Julia code takes various inputs and produces an output file (`output`)

Simplest option: put all code in a single file

`dynamics.jl:`

```
# solves dynamics of LV with multiple stable equilibria
# version 3: only a single variance in off-diagonal.

import Pkg
Pkg.add("DifferentialEquations")

using DifferentialEquations
using Random
using LinearAlgebra

# pick a random matrix, with given size, mean  for diagonal, off, sigma
# for example, usage: randommatrix(5, 1, 2, 0.1)

function randommatrix(size, dm, om, sd)
 a = Matrix{Float64}(undef, size, size)
 y = fill!(a, om) + sd*randn((size, size)) + (dm-om)*I
 return y
end

...
```

Single instance of code is invoked by

```
% julia dynamics.jl <parameters>
```

Submission file: Can generate from simple script or (suggested by Qier) have parameters as (integer) functions of process number

submit-dynamics:

```
# Submit description file for dynamics-random-g.jl program
#----------------------------------------------
Executable            = run-dynamics
Requirements          = ( BOSCOCluster =!= "t3serv008.mit.edu" && BOSCOCluster =!= "ce03.cmsaf.mit.edu" && BOS
+DESIRED_Sites        = "mit_tier3"
Universe              = vanilla
#GetEnv                = True
transfer_input_files  = dynamics-random-g.jl
should_transfer_files = YES
RequestMemory         = 2000
when_to_transfer_output  = ON_EXIT
Log                   = dynamics-random-g.log
Arguments             = dynamics-random-g.jl  20 0.2 1000 0
transfer_output_files = dynamics_0.result
Output                = dynamics_0.out
Error                 = dynamics_0.err
Queue
Arguments             = dynamics-random-g.jl  20 0.4 1000 1
transfer_output_files = dynamics_1.result
Output                = dynamics_1.out
Error                 = dynamics_1.err
Queue
```

Can also use process number in arguments (tip from Qier):

```
Arguments             = "dynamics-random-g.jl  20 0.2 1000  $(Process)"
transfer_output_files = dynamics_$(Process).result
...
Queue 10
```

run-dynamics:

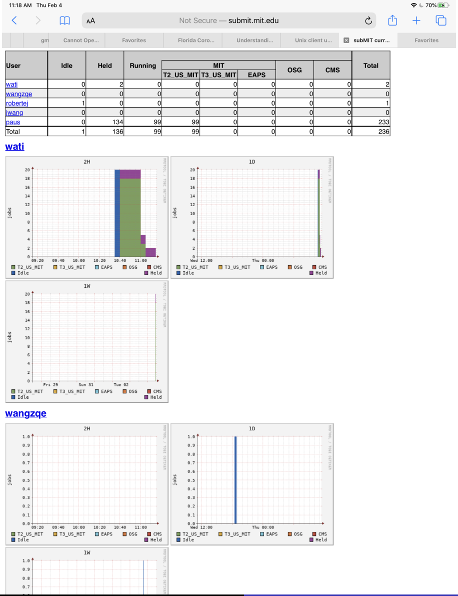
```
#!/bin/bash
#----------------------------------------------------------------------------------------------------
#
# Script to run julia. Likely you have some organizational work to do before you get to the true
# julia command. Here is the place to do it.
#
#----------------------------------------------------------------------------------------------------
# fix problem with loading new packages, as suggested by Qier

export JULIA_DEPOT_PATH="$PWD/.julia"

# show my running environment
echo ""; echo " ## Environment"
env

# show the directory
echo ""; echo " ## Directory"
pwd
echo ""
ls

# show the arguments
echo ""; echo " ## Arguments"
echo $@

# now let's do what we came for ($@ is just the full set of arguments provided to the run script)
echo ""; echo " ## Julia: julia $@"
julia $@

# let's make sure we name our output correctly
JOBID="${@: -1}"   # last value in the command line arguments
echo ""; echo " ## Julia: mv output dynamics_${JOBID}.result"
mv output dynamics_${JOBID}.result
```

Simple example: 3. Running the submission file

Now that we have everything together it's off to the races!

```
% condor_submit submit-single-5
Submitting job(s)....................
20 job(s) submitted to cluster 378098.
% condor_q

-- Schedd: SUBMIT.MIT.EDU : <18.4.134.251:9615?... @ 02/04/21 10:41:47
OWNER    BATCH_NAME    SUBMITTED   DONE   RUN    IDLE  TOTAL JOB_IDS
wati     ID: 378098   2/4  10:41      _      _     20     20 378098.0-19

Total for query: 20 jobs; 0 completed, 0 removed, 20 idle, 0 running, 0 held, 0 suspended
Total for wati: 20 jobs; 0 completed, 0 removed, 20 idle, 0 running, 0 held, 0 suspended
Total for all users: 254 jobs; 0 completed, 0 removed, 21 idle, 113 running, 120 held, 0 suspended

% condor_q

-- Schedd: SUBMIT.MIT.EDU : <18.4.134.251:9615?... @ 02/04/21 11:01:11
OWNER    BATCH_NAME    SUBMITTED   DONE   RUN    IDLE   HOLD  TOTAL JOB_IDS
wati     ID: 378098   2/4  10:41      _     18      _      2     20 378098.0-19

Total for query: 20 jobs; 0 completed, 0 removed, 0 idle, 18 running, 2 held, 0 suspended
Total for wati: 20 jobs; 0 completed, 0 removed, 0 idle, 18 running, 2 held, 0 suspended
Total for all users: 254 jobs; 0 completed, 0 removed, 1 idle, 125 running, 128 held, 0 suspended

% !!
condor_q

-- Schedd: SUBMIT.MIT.EDU : <18.4.134.251:9615?... @ 02/04/21 13:09:03
OWNER    BATCH_NAME    SUBMITTED   DONE   RUN    IDLE   HOLD  TOTAL JOB_IDS
wati     ID: 378098   2/4  10:41     18      _      _      2     20 378098.2-13

Total for query: 2 jobs; 0 completed, 0 removed, 0 idle, 0 running, 2 held, 0 suspended
Total for wati: 2 jobs; 0 completed, 0 removed, 0 idle, 0 running, 2 held, 0 suspended
Total for all users: 98 jobs; 0 completed, 0 removed, 1 idle, 79 running, 18 held, 0 suspended

%
```

Nice graphical interface to track your jobs: condormon



User	Idle	Held	Running	MIT			OSG	CMS	Total
				T2_US_MIT	T3_US_MIT	EAPS			
wati	0	2	0	0	0	0	0	0	2
wangzqe	0	0	0	0	0	0	0	0	0
robertej	1	0	0	0	0	0	0	0	1
jwang	0	0	0	0	0	0	0	0	0
paus	0	134	99	99	0	0	0	0	233
Total	1	136	99	99	0	0	0	0	236

wati

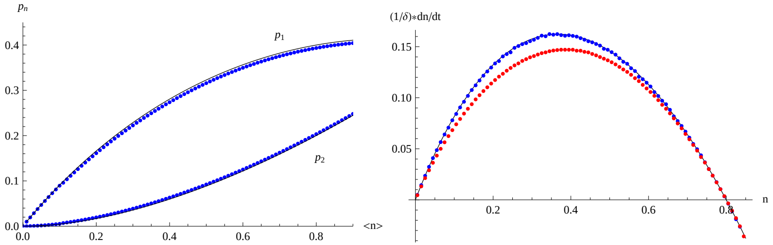
wangzqe

After jobs finish, have separate results/shell output/error files

```
% ls
base-single        single_409.result  single_501.out     single_513.err
generate_sub-single    single_410.err   single_501.result  single_513.out
generate_sub-single-1  single_410.out   single_502.err     single_513.result
generate_sub-single-2  single_410.result  single_502.out     single_514.err
generate_sub-single-3  single_411.err   single_502.result  single_514.out
generate_sub-single-4  single_411.out   single_503.err     single_514.result
generate_sub-single-5  single_411.result  single_503.out     single_515.err
output-test        single_412.err   single_503.result  single_515.out
run-single        single_412.out   single_504.err     single_515.result
single_401.err        single_412.result  single_504.out     single_516.err

...
```

Just move output files to your usual machine, and analyze data!

Results: lots of data → good match between “experiment” and theory!



Summary:

– Fairly easy to get up and running, run large jobs

– Works with Julia, familiar linux tools, can remotely mount file system,

```
sshfs submit05.MIT.edu:/home/submit/wati
/var/tmp/wati
```

– Issues encountered (and fixed): Julia/packages only on some systems, selecting memory