NETLOGO-CLUSTER

Base folder structure

File/Dir	Description	Example
Etc/	Define configuration data	
Etc/netlogo-cluster.conf	Configuration parameters. Those parameters rarely need to be modified. Only when install for the first times.	
Scripts/	Scripts to launch netlogo to the cluster and process data	
netlogo/	Soft Link that points to Netlogo	
ProjectName_Version	As a standard, when running a program it's only necessary to call to projects folder.	EV_4.0.0

Example:

Ev_4.0.0/ (Detailed but same scheme with projects)
- experiments
- Ev_4.0.0.nlogo
- extra.data
- output_rep1000_test1_1_17_04_2013/
  - output
  - output.err
  - Ev_4.0.0.nlogo
  - experiments
  - extra.data (link)
- output_rep1000_test3_2_17_04_2013/
  - output
  - output.err
  - Ev_4.0.0.nlogo
  - experiments
  - extra.data (link)
- output_rep1000_test1_3_17_04_2013/
  - output
  - output.err
  - Ev_4.0.0.nlogo
  - experiments
  - extra.data (link)
- output_rep100_test1_4_18_07_2013/
  - output
  - output.err
  - Ev_4.0.0.nlogo
  - experiments
  - extra.data (link)

Ev_4.0.1/
Ev_3.0.0/
scripts/
etc/

How to launch a job to cluster

In the base of netlogo-cluster:

./cluster-launcher.sh <Project's name>

Example: ./cluster-launcher.sh Ev_4.0.1

At this point, cluster-launcher is going to:
1-Read Ev_4.0.1/experiments
2-Read Ev_4.0.1/Ev_4.0.1.nlogo
3-Launch each experiment found in “experiments” to the cluster.

Project's folder

Each project folder contains:

Name	Description
projects_name.nlogo	It must have the same name as the folder.
experiments	Contains all experiments to run in the cluster.
output_...	Each run generates an output. All files are setted inside.
Dependent nlogo data files	Nlogo file requires other data. They must be placed in base directory project's folder.

Output folders

Once the experiment's execution is done you will find in the output folder several files/folders explained a few lines below. It's interesting to know that it's not possible to override an output folder. It will always create a new one in order to avoid the lose of data.

Format:

Parameter	Description
output	Fixed
Test runned	Experiment's name runned of nlogo file
[parallel version] Parallelized job	For a parallel job. To differentiate each job there is an identification number for each.
[parallel version] Variable value 1, Variable value 2, ...	For a parallel job. Variable value is added in order to make it easier to understand results. There will as many values as parallelizations in experiments file.
Total number of execution	Accumulated counter of all executions in the current project
#repetitions	Number of repetitions to run. It's defined in nlogo experiments
Date	Day, month and year

output_totalOfExecutions_testRunned_#repetitions_DD_MM_YY/

Parallelized version:
output_totalOfExecutions.ParallelizedJob_VariableValue1_VariableValue2_...3..._testRunned_#repetitions_DD_MM_YY/

Files created:

File	Description
output	Cluster data output
Output.err	Cluster error output
<Project's name>.nlogo	Nlogo file used
experiments	Execution file used
Link to extra data	All those files added to base folder are considered to be necessary for nlogo. A Symbolic link is created to output. It is required when job is launcher.

Experiments file
This file describes how many jobs lunch to cluster. The purpose is to automatize the launch of a large number of experiment's.

Format:

Column	Description
Experiment's name	Must be defined inside Nlogo
# repetitions	It ignores #repetitions inside Nlogo and it uses this one when launches the job.
Total execution time expected	Cluster needs to know in which queue (short, long) launch the job. For this reason it's necessary to specify an estimation. If it's not too much it could delete the job too soon (then err file would be empty).
[optional] -parallelize <Variable Name> <start> <step> <end>	Parallelize each value experiment in the cluster. Each value represents an independent job in the cluster. In the case there is more than 1 variable to parallelize, they will be combined. Note: variable must be defined under behaviour space as **constant**.

Example:

test 20 05:00:00
2-newTest 100 05:00:00 -parallelize PrbStopTravelling 0 0.1 1
3-test 50 04:00:00 -parallelize PrbStopTravelling 0 0.1 1 -parallelize PrbToBecomeAsymp 0 0.1 1
test1 10 05:00:00
test 20 05:00:00

For experiment 2:
In this case 11 (0, 0.1, 0.2, … , 1.0) jobs will be launch. One for each value “PrbStopTravelling” is

generated.

For experiment 3:
There are two variable to combine (PrbToBecomeAsymp and PrbStopTravelling). Each one has 11 values to test. As said before, there will be combined. This means there will be 11 x 11 = 121 jobs to launch.
Once all done, netlogo-plotter has an option (-jid or --job-id) to plot identifying the output by its id. So, it won't be necessary to specify 121 output files.