**Workflow for Running Monarch Model using Slurm on ISU HPC machines**

*October 2019*

*With some edits December 2020*

This assumes you have created a shapefile that you want to run monarch agents on. Shapefiles must have polygons smaller than monarch step length. Shapefiles must have columns in the .dbf (in the attribute table) that correspond to the column names in the Repast Simphony Java code. These columns are CLASS\_NAME, Shape\_Area, ProbEggs, probMove (Yes, for the last two one is capitalized and one is not. This is my little joke.), and GISPolyID. I won’t go into any more detail here about creating the shapefile.

*Getting the Shapefile into the Monarch Model in Repast Simphony*

Now you need to get the shapefile into Repast Simphony and test it there, then create a .jar file to send to the ISU HPC machines to run there.

1. On completed shapefiles that are large and have been through a lot of processing, run the Repair Geometry tool (Data Management Tools | Features). You also may need to run the Integrate tool (Data Management Tools | Feature Class) with a 1 m tolerance. These two tools should get rid of any self-intersections and other problem shapes that will make area calculations difficult. (Integrate may throw an error if you run it before repair geometry.) Then be sure the Shape\_Area column in the shapefile is updated with accurate info, because it is used for calculations elsewhere. Update it BEFORE you change the coordinate system to lat/long, or it will be in nonsensical units (lat/long area units). Check probEggs and probMove to see if there any bad values (sort ascending and descending). If you did any complicated procedures to make the map, be sure there are no stacked polygons or overlapping polygons.
2. You should also strongly consider creating a field/column with a unique ID for each polygon. The FID in GIS often gets divided into multiple rows/polygons while processing, so each one has the same FID. RS assigns a PolygonID to each polygon, but that seems to be just numbering them from 1-X in whatever order they are. I have always been a little bit nervous about that. Though to date it has always worked out, I have been confused more than once trying to match the output from RS with the original shapefile. Each time (at least 2 x I can think of), the order of the PolygonID matches the order of the rows in the shapefile, but if you make your own uniqueID it will save some angst about how RS assigns the PolygonID. The last time I did this, I created field called GISPolyID (with precision 8 for large numbers) with sequential numbers using the Python codeblock in the ArcMap help file for calculating fields. The code in RS gives each polygon an incremental ID based on the order they were added, so appears to be the order they occur in the .dbf. I have added a new field in RS for GISPolyID, that is brought in from ArcMap.
3. Copy relevant shapefile files (should be 6-8 files – not all are necessary, but I forget which ones are not, so just copy all) from the current working folder to C:\Users\tgrant\eclipse-workspace\Monarchs\data. Note that all files in /data get copied in the model .jar, so it’s a good idea to keep this folder lean, to keep file the .jar file size down. If you don’t know what I mean, watch when the model jar is unzipped on the HPC drive. I now move lat/long files that I have run models on, to keep the exact shapefiles I ran the model on, to “C:\Users\tgrant\Documents\Monarchs GIS\Shapefiles\Analysis Shapefiles moved from data”.
4. Shapefiles MUST be in lat/long projection, not UTM. I keep a .mxd under in the Monarchs GIS folder called Analysis Shapefiles.mxd that has shapfiles in the right projection. Typically, I open the shapefile there from C:\Users\tgrant\eclipse-workspace\Monarchs\data and change the projection there. So, the shapefiles in the ToxSims folders are not in the right projection, only the shapefiles in the Monarchs\data folder are in the right projection. Change the projection using the tool Data Management>Projections and Transformations>Project (not Define Projection). Output coordinate system should be the Geographic Coordinate System under World folder called WGS 1984. Save it to C:\Users\tgrant\eclipse-workspace\Monarchs\data instead of the default .gdb. I moved shapefiles I’m not using to Monarchs GIS\Shapefiles\Analysis Shapefiles.
5. Normally you need to change the model Java code several places. (Ideally these would be an input box but for now they are hard-coded.) The code uses lat/longs coordinates to do several things, like form a bounding box, randomly choose initial agent locations, etc. I’m just using the Story Co area again for the landscape toxicological analysis, so only need to change one thing (number 3 below – make sure to load the shapefile that is in lat/long, not UTM projection). If using a shapefile from a different area, need to find the 4 corners for two bounding boxes around the shapefile, one slightly inside for placing random locations and one slightly outside for keeping monarch agents inside the shapefile. You can use ArcMap to get the coords (under View > Data Frame Properties > General tab there is a box for units – change display units to decimal degrees). The places to change code every time you use a new shapefile are:
   1. Travel network code (line 100, Contextbuild.java – I believe this code only matters for the Repast Simphony display)
   2. Generate random initial locations (line 286, Contextbuild.java – lat/long coords should be slightly inside edges of shapefile). This part is obsolete now with addition of random location code for irregular shapefiles. Numbers should be changed at line 351.
   3. Load features from shapefile (line 377, Contextbuild.java) – note that after adding code to randomly add monarch agents in irregular shapefiles, now has to be added at **line 88**.
   4. Bounding box (line 113, Monarch.java – lat/long coords should be outside edges of shapefile)
   5. Bounce back coordinates (line 332, Monarch.java – should be same coords as 5.2)
6. Save the model code with changes you made in step 3 above. I have most or all previously run maps commented out, so they are easy to use in the future. No need to delete anything, though it’s getting a bit cluttered.
7. Run the model. The window pops up. It’s normal for it to take a few seconds or minutes to load.
8. It’s good practice to run the model with a few agents to make sure there are no errors. Running it with more than a few agents may take longer than you would like. Change to 3 agents in the parameters window (see tabs at lower left of left pane). Hit initialize run (blue button at top, looks like a power button). It’s also normal at this point to take a few seconds to initialize. Run one step (there is a button for “step run”). If it runs that far with no errors, it’s good to go.

For large shapefiles, however, you may get an out of memory error. This is may be an error in the Repast Simphony display, not the model. Run the model on the HPC machines and see if you get the same error. Error messages for the HPC machines are in the .err or .out files created when the model runs.

*Making the Model .Jar*

Now you can make the model .jar that will be sent to the HPC machines to run. Close the ArcMap Analysis Shapefiles.mxd because it will lock the shapefiles in the eclipse workspace so they can’t be copied into the .jar.

To make model .jar, from the same pop-up window, hit the yellow button with lightning bolt (you don’t have to initialize and run the model to use this button – if you know the model works fine for your shapefile, you can just go straight to this button). Specify parameters. Don’t forget to specify maximum probEggs. Note the model still uses the term perception distance, which I later changed to perceptual range in the manuscript. Save coordinates saves the lat/long coordinates for each monarch each step. It creates a massive, massive output file, so only do this if you really want these. Generate the batch parameters file by hitting the generate button. Wait until the console says completed building model archive.

The number of instances in Repast Simphony is usually the number of tasks in the repast.sbatch file, unless you want to get slightly more complicated. But it seems most simple to keep instances the same as lines in the unrolledparm.txt file, which is the same as tasks. So the space-separated list of agents should equal the number of tasks. So, for example, I want to run 10,000 monarchs in Story Co. I can run 50 tasks. So that means I need to run 200 monarchs per task. So I enter a list of 200, 50 times, in the agent list, as seen at the end of this document. I created it at the end of this document, double checked it in Excel, and copied it to RS.

Be sure you hit generate batch parameter file, then just hit “create model archive for batch runs” button at top (two yin-yang arrows). As the console tab shows, the archive is at C:\Users\tgrant\eclipse-workspace\Monarchs\output\complete\_model.jar. Wait until it says it’s done, I tend to X out before it’s done.

*Moving the .Jar to the HPC Machines*

My research drive is [\\phantoon.las.iastate.edu\work\LAS\spbrad-lab\tgrant](file:///\\phantoon.las.iastate.edu\work\LAS\spbrad-lab\tgrant) It can be accessed from Windows Explorer by typing it in the address bar or clicking the link above. It can also be accessed with PuTTY. It’s better to use the address [\\prontodtn.las.iastate.edu](file:///\\prontodtn.las.iastate.edu), which runs through a faster pipeline for file transfer. But using phantoon drive in Windows Explorer is the same thing as using protodtn.

Move the complete\_model.jar to my research folder above. I can do this in Windows Explorer because it’s hooked up to that drive.

*Running the Model on the ISU HPC Machines*

You can work with the files in Windows Explorer, but you’ll need to ssh to the HCP machines using PuTTY to run commands. I have a profile set up in PuTTY where you just double click to ssh to pronto.las.iastate.edu. It will ask for your login password that you use for all computers on campus.

(If you’ve never used PuTTY, or SSH, or Linux, it’s not that hard, but you may want to read up on it or get someone to show you the basics. If your HPC system is different, you may use a different program, like GitBASH, to ssh to the big machines. Typically these big machines are running some kind of Linux, which is a command line system, not a GUI like Windows.)

If you’ve already run a model, then there will be a bunch of instance folders. Delete these in Windows Explorer or PuTTY, or the outcombiner.sh file won’t work.

PuTTY commands:

rm -r instance\_\*

rm –r combined\_data

rm –rf data/data

or rm -r data

Might be a good idea to delete lib folder, too. I never do though and never seems to cause a problem, it just overwrites it.

Be sure to save repast.sbatch and slurm\_repastwrapper.sh when deleting files. They don’t written over, but if you delete everything they will be gone. There is no recycle bin in Linux. Sometimes I move them to the root (spbrad-lab) to keep copies. I also keep copies elsewhere so I don’t lose them.

Open PuTTY if not yet open, log in to pronto (I have a profile set up in PuTTY, but all it is is ssh [tgrant@pronto.las.iastate.edu](mailto:tgrant@pronto.las.iastate.edu), see <https://researchit.las.iastate.edu/pronto>), navigate to my drive using “cd /work/LAS/spbrad-lab/tgrant”. (You can use “pwd” to see where you are in the file system. “ll” or “ls” show files in current folder.)

Use “unzip complete\_model.jar” command while in the tgrant folder (complete\_model.jar should be in this folder, obviously). It will ask if you want to overwrite. Say A for all - unless you want a separate folder for every single run, but its far more efficient to save output separately.

***Repast.sbatch*** *and* ***Slurm\_repastwrapper.sh***

Repast.sbatch is the file that tells the SLURM how to run the model (SLURM is the scheduling program that schedules and runs all the different jobs being run on the HPC machines). I find it easiest to edit in Notepad ++, which IT can quickly install for you. Repast.sbatch calls the file slurm\_repastwrapper.sh.

Next edit repast.sbatch as needed for number of tasks, time, mem, etc. For Story County, 50 tasks with 400G memory seems best (but I learned later that the memory variable in repast.sbatch doesn’t matter as much as the memory variable in slurm\_repastwrapper.sh.). The repast.sbatch folders calls the slurm\_repastwrapper.sh file. In slurm\_repastwrapper.sh (also easiest to edit with Notepad++), you can change the memory allocated to each instance. I have been allocating 8092M. For Des Moines Lobe Scenario 2, IT told me to quadruple memory in slurm\_repastwrapper.sh to 32G and set the mem in repast.sbatch to 500G, because I think it’s 32G per instance/line, and I have 100 lines. It definitely worked when I used 64g and 700G. The time option is used by SLURM to try to plan for scheduling, so as far as I know it doesn’t kill the job if it goes over the time, it’s just for planning.

The repast.sbatch file I have seems to equate tasks from SLURM that are defined with #SBATCH –ntasks with cores. So if I enter 50 tasks, it prints out “Total Cores: 50”. Then for some DUMB reason, it divides cores by 2, for no reason that I can figure. So then it decides to run 2 lines/tasks for instance. Why not just run 1 line per 1 instance? So, to get it to do what I want, without fussing with the sbatch script, just run twice as many tasks as lines. Except I can’t request 100 tasks, it throws error that node configuration not available. So running 50 tasks, which means 25 instances and 2 lines per instance. By the way, this is the file I got from USGS. The original repast.pbs seems to make much more sense.

I emailed the Repast-L listserv about this in Feb 2020 and Nick Collier wrote some new SLURM files for the HPC situation, but I haven’t used the new ones yet. I did download the files.

Shane Nehring in research IT said should only need to ever request 1 node, but I requested 2 nodes for the Des Moines Lobe model.

DON’T FORGET TO CHANGE MEM REQUEST IN HEADER AND IN SRUN LINE AT BOTTOM.

Finally, type “sbatch repast.sbatch” into the PuTTY command line to queue the model.

Research IT recommended array scripts for running a lot of instances (when I said I needed 400 instances): <https://researchit.las.iastate.edu/how-create-slurm-job-array-script>. But not sure if outputcombiner.sh would work in that case.

*Checking the status of the model run*

“squeue” is the most basic command to look at all jobs in the system. Look down and find your username.

“squeue -u tgrant” shows jobs by user tgrant

Save job ID and use “scontrol show job XXXXX” to see more details about the job. Doesn’t really show how much memory is being used.

Also try “sidle”.

I used to look at how much memory each core was using, using atop or htop or top, but don’t know how to do that on new node configuration. The command sstat and sacct should tell how much RAM and CPU is being used, but I can’t them to work well. After asking ResearchIT, I can get sstat to work by using -a in the invocation, e.g., “sstat 182351 --format=jobid,AveVMSize,MaxVMSize -a”.

I asked ResearchIT how I could check if all the cores/nodes were full, i.e., if there was no space. Shane Nehring said:

“Your job will run if there is space for it eventually. It's probably better to give it the resources it needs and wait for it to run. That said, you can use the 'scontrol show node <nodename here>' to see what resources have been allocated to a node and what it has.

If you want to instead monitor your own jobs for adequate core usage, you can ssh to the node you were assigned and run htop -u <your username> to see your running jobs.”

You can use “sacct --jobs=179664 --format=elapsed” to see how long the job took.

*Compiling Results*

When the model is done running, the output files are spread over a bunch of folders. RS has a nice program called an output combiner that combines everything into one output file.

On ISU’s HPC, you can’t run programs directly on pronto, which is the “head node.” Everything you run on pronto has to be through slurm. So, we need to request an “interactive session” through slurm to use the HPC resources to run the output combiner. This is described here: <https://researchit.las.iastate.edu/slurm-basics>

Type this command to request resources for an interactive session:

srun --time=00:15:00 --nodes=1 --cpus-per-task=8 --pty /usr/bin/bash

It should load up right away. Load the java module. I have been using:

module load jdk/8u172-b11-rnauqmr

Navigate to the appropriate directory (/work/LAS/spbrad-lab/tgrant, in my case) and type:

./outputcombiner.sh

Or, if you get a permission denied error, try

bash outputcombiner.sh

It will run and put the combined output files in the “combined\_data” folder. Type exit to stop the interactive session.

When it’s done running, you can move the combined output files in the combined data folder using Windows Explorer to wherever you want. You’ll want to move the output files before you run the model again, because all these steps will overwrite the output files. Keep all the output files, including the batch\_param\_map files – these tell what parameter settings were and will help you keep your datasets straight.

You can use “sacct --jobs=179664 --format=elapsed” to see how long the job took.

*Data Analysis*

The model has been run and you have combined output files. Now we will be moving to R for the next steps. I have several R files I have used for analysis of the output data. They serve different purposes, but someone may find useful examples of code there: ToxSims2.R has my code for landscape scale toxicology analysis. Also see the R code for Grant et al. 2018 and the Des Moines Lobe simulations.

You may have a problem joining .csv files in ArcMap. For some reason, I couldn’t get the density field to join. So I calculated density in ArcMap with field calculator.

Space-separated list of 100x100 instances – i.e., 100 instances/tasks of 100 monarchs each, for 10,000 monarchs.

100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100 100

200x50:

200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200

500x100:

500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500

400x100:

400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400 400

500x50:

500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500 500

200x20:

200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200 200

*MORE Linux commands*

hostnamectl – what is the operating system or version of Linux. Pronto at ISU is using Red Hat Enterprise Linux.

To paste to the PuTTY command line, Ctrl-V won’t work. Right click.

“module spider” to see all the available modules. “module spider jdk” returns the different versions of java. There’s apparently no way to check if a particular module is loaded or not. You can try lsmod, but I haven’t been able to find the jdk module in there. You can always load the module again if you’re not sure it’s loaded.

“free” shows memory available. “free -h” shows memory with units.