Exercise II:

Summary:

In this exercise we built upon the knowledge we gained in exercise 1. Here, we will also manipulate the data file; however, we will have *several instances* of a particular program working on the data file *at the same time* which is a particular form of *parallel* processing. Instead of having the same program working *sequentially* through all entries of the data file, there will be several instances of the program working on smaller portions of the data file *simultaneously*. The overall processing time will be much shorter this way. It is not uncommon that sequential processing of large-scale problems takes days to weeks, yet by using parallel processing, as we do it in this example, the compute time can be cut down to just a few hours.

Below are the steps to be taken.

Open a terminal (Activities->right-click terminal icon):

Copy data package for exercise 2 to your Desktop (**type at the command prompt end enter strings in italics**):

*cp /tmp/ex2.tar.gz ~/Desktop*

Navigate to the Desktop folder:

*cd Desktop*

Unpack the package:

*tar -xzvf ex2.tar.gz*

Inspect content of the directory/folder:

*ls -la .*

Please note that Linux commands can be aggregated and scripted into a text file. The file can be executed as a new command then.

Design and configure (embarrassingly) parallel compute jobs. Use the script launcher file template to create a Matlab job (uncomment where appropriate):

1. Open the file launchScripts.sh in the gedit text editor:

*gedit launchScripts.sh*

To configure a Matlab job, uncomment the lines in the Matlab section as directed.

1. Save the file.

Submit the set of compute jobs to the wait queue:

*./launchScripts.sh*

Monitor the job wait queue and the submitted jobs:

*qstat*

*showq*

Upon completion (C status in qstat) examine the result output/error files (replace “<jobid>” with proper number from qstat):

*more ex2\_script\_1.e<jobid>*

*more ex2\_script\_1.o<jobid>*

Assess the outcome of your compute jobs. Describe how the batch scheduler was used to run the computation. What happened at each step of the procedure? Describe how a hierarchy of scripts was used to automate parallel job submission and execution.

Submit R and Python jobs separately by uncommenting the proper sections in the file launchScripts.sh.

Think about bioinformatics computations where these parallelization concepts could be useful.