Quick and easy parallel computing on your laptop or William and Harvey

Daniel Reuman
QB and BIG

You learned how to use lapply in lecture. This guide is a quick introduction to mclapply in the parallel package, which is very similar to lapply except it runs on multiple cores. This approach to parallelization only works on linux. Well, it “works” on windows, too, but you have to set mc.cores=1, which means you are only using one core, which means you are effectively not doing parallel computing at all!

There are other ways to do parallel computing. There are several alternative options for your laptop or for small multi-core servers like William and Harvey. There are also large-scale options in the form of queuing systems on large clusters like the Imperial HPC cluster. This is just one option, but it is a particularly convenient one for typical biology-scale computing.

At the top of your code, where you name your other packages (if any) put:
require(parallel)
This loads the parallel package.

The command you need in the package is **mclapply**. As usual, you can type
?mclapply
Into the R console to get the help file. The main arguments needed are **X**, **FUN**, and **mc.cores**. Usually you can accept default values for the other arguments.

The **mc.cores** argument is simply how many cores to use. To determine this, you ought to know how many cores there are on the machine you are on, and how many of them are currently occupied by other users. The command
cat /proc/cpuinfo | grep processor | wc –l
will tell you how many cores. You can see usage statistics on multi-user machines with
top –d 60
for instance.

The **X** argument is a list of any description. The **FUN** argument is a function you have written which accepts as its single argument something of the format of the entries of **X**. The machine will divide up the entries of **X** across the allocated cores, run **FUN** on each element, and assemble results into another list, which it returns. For instance, try the code in ExampleParallel.R.