**Large Genomic Data Sets and the Issue of File Size Limitations for Pushing to Github**

**A Step-by-Step Guide**

**Michael Hall**

**Plant Breeding and Genetics Laboratory**

**FAO/IAEA Joint Division**

**Seibersdorf, Austria**

Created: January 2022

Last updated: 27 January 2022

**Please note:** *This is an important note at the bottom of the title page.*

Contents

**1 Background**

**2 Installing Git extension for Versioning large files**

**3 Creating a new repository on Github**

**4 Creating a new local Git Repository**

**5 Creating a GitHub repository on your local machine**

**6 Push your project to Github**

**7 Cloning a Large File Repository**

**Diagram

Description automatically generated**

1 Background

**There is an issue when a project has large files in excess of the limitation imposed on it by Github. This limitation is 0.10 Gigabytes of information. When working in the field of bioinformatics chances are you will encounter a relatively large file that exceeds this limitation. In fact, if the genome file, like a fasta file is not compressed, it is more than likely too big for GitHub to accept as a simple push. In this case I have a FASTA file that is .3791 GB and a Annotation file that is .183 GB. Therefore, the solution presents itself in the form of an open-source git extension for versioning large files. This tutorial should provide necessary steps to ensure any file no matter the size can be versioned in such a way to satisfy the upper limitations of Github. The end result being you can have your large files included in an open-source project on Github free for anyone to clone etc.**

2 Installing Git Extension for Versioning Large Files

First it is necessary to install on your local machine the git extension for versioning Large Files using a super user command.

**Follow this link to download it for Linux Operating Storage**

<https://github.com/git-lfs/git-lfs/releases/download/v3.0.2/git-lfs-linux-amd64-v3.0.2.tar.gz>

**Follow this link to download for Windows Operating System**

https://github.com/git-lfs/git-lfs/releases/download/v3.0.2/git-lfs-windows-v3.0.2.exe

You should have a compressed tar ball file that you need to unpack in your downloads folder. Its extension is **tar.gz.** Right click on the file and select Extract Here. If you are on a windows operating storage Right click on the file, select 7zip and select Extract Here.

Navigate to the **install.sh** invoke the super user command to install

JohnDoe@JD-ubuntu: ~ /Downloads/git-lfs-linux-amd64-v3.0.2$ Sudo ./install.sh

Once it is installed you can be in any environment and call it.

3 Creating a new Repository on Github

I’ll assume you have a GitHub account and not worry about making an account for it now. Next create a new repository on GitHub where all of your project documents will be stored. Sign into your account and navigate to your repositories tab. From there select the green button “New”. Give the repository a name and initialize it with a README file and if you want to add .gitgnore which allows you to choose which files to “ignore” and not track and finally choose a license. After clicking the green button, “Create repository”, congratulations you have successfully created something that exists now which did not before and can be found at

<https://github.com/YOUR_GITHUB_USER_NAME/NEW_REPOSITORY_Name>

Doing this automatically creates two files a README.md and LICENSE file.

4 Creating a GitHub repository on your local machine

Another step is to create an EMPTY identical local Github repository. By identical I mean the file pathways from your empty project directory must match exactly otherwise there are going to be issues. For the sake of simplicity lets assume you made a new repository and named it **NewtonsLaws**. If this is true, then there should be a path from the internet to that repository. However, we are going to clone the repository first into the empty project directory and then copy the actual project files into it.

https://github.com/JohnDoe/NewtonsLaws

Install Git on Debian based platform

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ Sudo apt install git-all

Clone your Repository by invoking the following command.

JohnDoe@JD-ubuntu: ~ /Desktop/$ git clone <https://github.com/JohnDoe/NewtonsLaws.git>

These two files we discussed earlier, the README.md and LICENSE file should now have filled your empty project directory. Now we need to create or reinitialize the local git hub repository by invoking the following command.

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ git init.

To confirm that the hidden Github files are there invoke the following command.

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ ls -la

Now copy your project files into the reinitialized local GitHub repository.

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ cp /home/JohnDoe/BIGFASTA.fna /home/JohnDoe/Desktop/NewtonsLaws

5 Track large file extensions

In step 2 we installed a large file versioning storage. We are now going to call this into action by invoking the following command.

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ git lfs track “\*.fna”

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ git lfs track “\*.gff”

Now any file extension with that ending that is pushed to Github will pass through the limitation imposed on it. Next, add a .gitattributes file that will be included in the push.

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ git add .gitattributes

6 Push your Project to Github

At this point you should check the status of your repository.

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ git status

You will see in red that the branch is up to date but you have untracked files.

Now add all files of the project.

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ git add .

Recheck the status to confirm the untracked files are staged to be committed.

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ git status

You will see in green the files are ready to be committed. Commit the files now.

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ git commit -m “First Commit”

Next, Invoke the migratory command.

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ git lfs migrate import –include=”\*.fna,\*.gff”

Finally, push all the files in the project to Github.

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ git push -u origin --all

7 Cloning a Large File Storage Repository

To clone an existing Large File Storage Repository, even for one on Github, use the following command.

JohnDoe@JD-ubuntu: ~ /Desktop/NewtonsLaws$ git lfs clone https://github.com/YOUR\_GITHUB\_USER\_NAME/NEW\_REPOSITORY\_Name

So you should have all your big files back to where the clone happened.