PPGtk: the polyploid pop-gen toolkit

Written and maintained by Paul Blischak

E-mail: blischak.4@osu.edu

Source code: https://github.com/pblischak/ppgtk

Docker image: https://hub.docker.com/r/pblischak/ppgtk

Online docs: http://pblischak.github.io/ppgtk

Contents

PPGtk is a set of tools written in C++ for population genetic/genomic analyses of high-throughput sequencing (HTS) data collected in polyploid organisms. It was designed for research in non-model polyploids (i.e., no reference genome), and aims to provide models that facilitate the discovery of patterns of genetic diversity.

1 Installing from source

Installing PPGtk from source requires the Boost C++ Libraries. PPGtk also has multithreading capabilities through the OpenMP libraries, which will require an OpenMP compatible C++ compiler (such as the GNU g++ compiler). As a special note for Mac OSX users, the "g++ compiler" that comes with a Mac is not actually the GNU g++ compiler. Typing `g++ -v` in a terminal window should confirm this. The native Apple/Mac compilers do not support OpenMP threading, so you will need to get the GNU versions. This link (http://hpc.sourceforge.net/) has instructions for downloading and installing gcc on a Mac. Alternatively, you could use Homebrew (http://brew.sh/).

If you would like to use PPGtk but don't want to compile it from source, we have built a Docker image with the software and dependencies already installed. Information for using PPGtk with Docker can be found in the the next section: PPGtk Docker image.

1.1 Boost C++ Libraries

```
$ cd boost_1_60_0
$ ./bootstrap.sh --help
$ ./bootstrap.sh --prefix=/usr/local --with-libraries=program_options
$ sudo ./b2 install
```

1.2 OpenMP

can be used for parallelization.

```
$ export OMP_NUM_THREADS=4
```

1.3 Obtaining PPGtk

A "bleeding edge" version of the source code for PPGtk can be cloned from GitHub using:

```
$ git clone https://github.com/pblischak/ppgtk.git
```

from the command line. You can also download the latest stable release from GitHub by following the **releases** link in the PPGtk repository.

```
$ cd ppgtk-v1.0.0
[R]> dat <- read.table("tot-reads.txt")
```

1.4 Compiling the executable

To enable parallelization with OpenMP, you will need to change the `OPENMP` variable in the Makefile to `yes`, like so:

```
OPENMP ?= yes
```

If you installed Boost in `/usr/local/bin`, then the paths to the header files and libraries should already be correctly specified in the Makefile. If you installed it somewhere else, then you will need to change the `BOOST_LIB` and `BOOST_INC` variables to point to where it was installed.

Then type `make` to compile the executable. After successfully compiling the program, typing `sudo make install` will install the `ppgtk` executable to be used from anywhere on your computer by copying it into `/usr/local/bin` (this is why you need to use `sudo`).

2 PPGtk Docker image

If you would like to run PPGtk, but don't want to go through the processing of compiling it from source, we have built a Docker image of a Ubuntu-based environment with the PPGtk program already installed. If you are unfaniliar with Docker, please read over some of the introductory documentation on their website. We will cover the basic steps that will be necessary to use PPGtk in this way.

2.1 Downloading Docker

2.2 Obtaining the PPGtk image

Launch the Docker Quickstart Terminal and type the following command:

```
$ docker pull pblischak/ppgtk
```

To run an instance of the image as a container, type:

```
$ docker run -it pblischak/ppgtk
```

Now that you are operating within the container, you can check to see if PPGtk is working by typing:

```
$ ppgtk -v
```

This should result in the following output (the version number may be different):

```
   ____  ____   ____ _   _
  |  _ \|  _ \ / ___| |_| | __
  | |_) | |_) | |  _| __| |/ /
  |  __/|  __/| |_| | |_|   <
  |_|   |_|    \____|\__|_|\_\

********************************************************************************
**  This is PPGtk version 0.1.0 (May 2016)
**  For help using the sofware type: ppgtk -h or ppgtk --model <model-name> -h
**  Documentation can be also found online at http://pblischak.github.io/ppgtk/
********************************************************************************
```

2.3 Making your data accessible within a container

```
$ docker run -it -v /Users/path/to/data:/home/analysis pblischak/ppgtk
```

This step will launch a Bash shell within a Docker container running a basic Ubuntu operating system. You will enter the container in a folder named `/home/analysis`, which is also the folder that you attached your data to. If you type `ls`, you should see the data files that you had on you own machine. With the docker container running, you can analyze the data that you attached just like you would run it from any other terminal window.

```
$ ppgtk --model freqs -c config.txt -q
```

2.4 Creating a multicore Docker Machine

```
$ docker-machine create --driver virtualbox --virtualbox-cpu-count <#cpus> <name>
```

```
$ docker-machine create --driver virtualbox --virtualbox-cpu-count 4 multicore
```

```
$ docker-machine env multicore
```

Whenever you start a Docker Quickstart Terminal, you will be using the default machine. To switch to the new `multicore` machine run the following:

```
$ eval $(docker-machine env multicore)
```

3 Getting Started

For a general help message:

```
$ ppgtk -h
```

For a help message specific to a particular model:

```
$ ppgtk --model <model-name> -h
```

To specify and run an analysis, supply the `model` and `config` arguments:

```
$ ppgtk --model freqs --config freqs.txt
```

The full `model` and `config` argument flags can also be shortened to just the first letter:

```
$ ppgtk -m freqs -c freqs.txt
```

`total.txt`:

```
24 39 12 0 63 0 33 45 . . .
20 33 21 0 0 78 27 11 . . .
.
.
.
7 4 80 22 26 22 18 4 . . .
```