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`

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

Contents

PPGtk is a set of tools written in C++ for population genetic/genomic analyses with polyploids using high-throughput sequencing data. It was designed for research in non-model taxa (i.e., no reference genome), and provides a number of functions for conducting population genomics analyses in these taxa.

1   Installing from source

Installing PPGtk follows the typical workflow for Unix-based operating systems using a Makefile and the commands 'make && sudo make install'.

1.1   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")
```

1.2   Boost C++ Libraries

1.3   OpenMP

can be used for parallelization.

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.

If you are on a Mac, type:

```
$ make darwin
```

If you are on a Linux computer, type:

```
$ make linux
```

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

```
$ ppgtk -n 100 -l 2000 -p 4 -t total.txt -r reference.txt -q
```

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 . . .
```