QuickProbs 2.06
User's Manual

Adam Gudyś, Sebastian Deorowicz

June 12, 2017

Institute of Computer Science, Silesian University of Technology, Akademicka 16, 44-100 Gliwice, Poland

adam.gudys@polsl.pl, sebastian.deorowicz@polsl.pl

Installation

QuickProbs 2 comes with a set of precompiled binaries for Windows and Linux and OS X. These are:

- `quickprobs-2.06.exe` — CPU executable for Windows,
- `quickprobs-cl-2.06.exe` — OpenCL executable for Windows,
- `quickprobs-2.06-linux` — CPU executable for Linux,
- `quickprobs-cl-2.06-linux` — OpenCL executable for Linux,
- `quickprobs-2.06-osx` — CPU executable for OS X,
- `quickprobs-cl-2.06-osx` — OpenCL executable for OS X.

QuickProbs 2 can be also built from the sources. They are distributed as CMake project for easy deployment under different system platforms and development environments. QuickProbs 2 uses the following external libraries:

- OpenMP for CPU parallelism.
- OpenCL for massively parallel computing (GPGPU).
- TCMalloc for fast memory management (part of Google Performance Tools).

Depending on the operating system, corresponding libraries are either integrated with QuickProbs 2 sources, or have to be downloaded separately. Below one can find detailed instructions on building QuickProbs 2 from sources under Windows, Linux, and OS X.

Windows

1. Download and install CMake: `https://cmake.org`.
2. Run CMake executable.
3. Fill *Where is the source code*—this is the path to the QuickProbs `src` directory.
4. Fill *Where to build binaries*—this is the path where project files will be placed.
5. Click *Configure* and select project generator (e.g. *Visual Studio 12 2013 Win64*) with default compiler.
6. Click *Configure* again and then *Generate*.
7. Go to output directory and open Visual Studio solution file (`QuickProbs.sln`).
8. Build the solution.
9. QuickProbs binaries will be placed in the `bin` directory next to the `src`.

Linux

1. Install CMake: `sudo apt-get install cmake-gui`.
2. Install TCMalloc: `sudo apt-get install libgoogle-perftools-dev`.
3. Run CMake executable: `cmake-gui`.
4. Fill *Where is the source code*—this is the path to the QuickProbs `src` directory.
5. Fill *Where to build binaries*—this is the path where makefile will be placed.
6. Click *Configure* and select *Unix Makefiles* with default compiler.
7. Click *Configure* again and then *Generate*.
8. Go to output directory and run `make`. GCC 4.8 or later is required for proper build.
9. QuickProbs binaries will be placed in the `bin` directory next to the `src`.

OS X

1. Install Brew package manager:
   `/usr/bin/ruby -e`
   `"$(curl -fsSL https://raw.githubusercontent.com/Homebrew/install/master/install)"`
2. Install CMake: `brew cask install cmake`
3. Install TCMalloc: `brew install google-perftools`
4. Install GCC/G++ with OpenMP support: `brew install gcc -without-multilib`
5. Establish GCC/G++ version: `gcc -v`
6. Run CMake executable: `cmake-gui`.
7. Fill *Where is the source code*—this is the path to the QuickProbs `src` directory.
8. Fill *Where to build binaries*—this is the path where makefile will be placed.
9. Click *Configure*, select *Unix Makefiles* and specify own native compiler.
10. Set `gcc-<version>` and `g++-<version>` as GCC and G++ compilers.
11. Click *Configure* again and then *Generate*.
12. Go to output directory and run `make`. GCC 4.8 or later is required for proper build.
13. QuickProbs binaries will be placed in the `bin` directory next to the `src`.

Usage

QuickProbs 2 is a fast and accurate algorithm for multiple sequence alignment suited for massively parallel architectures (compatible with OpenCL). It is provided with two binaries for CPU and OpenCL devices (like GPU). All parameters suitable for the CPU version are also supported by the OpenCL binary. The latter additionally requires an OpenCL platform and device to be specified. Below one can find typical QuickProbs 2 use cases.

`quickprobs-2.06.exe`
Displays the detailed information about QuickProbs 2 options.

`quickprobs-cl-2.06.exe`
Displays the detailed information about QuickProbs 2 options and available OpenCL devices.

`quickprobs-2.06.exe` *input*
Aligns sequences from *input* file and prints the result on the standard output (other messages are by default printed on the standard error stream). Both input sequences and output alignment are in the FASTA format. The calculations are performed on a central processor (without OpenCL) and distributed among all detected cores.

`quickprobs-2.06.exe` *input* `-o` *output*
Aligns sequences from *input* and stores the result in *output* file.

`quickprobs-2.06.exe` *input* `-o` *output* `-v`
Runs algorithm in a verbose mode (prints a lot of debug information).

`quickprobs-2.06.exe` *input* `-o` *output* `-t 6`
Distributes computations on 6 CPU threads.

`quickprobs-cl-2.06.exe`*input* `-o` *output* `-p 0 -d 1`
Runs QuickProbs in OpenCL mode on device 1 from platform 0. Use this option to perform the calculations on massively parallel devices like graphics processors. Note, that central processors are also OpenCL devices. Nevertheless, OpenCL code in QuickProbs 2 was especially optimised for GPUs architectures and was not tested on CPUs! It is suggested to use the CPU binary to perform analysis on a central processor only.

`quickprobs-2.06.exe`*input* `-o` *output* `-n`
Runs the algorithm in a nucleotide mode.

`quickprobs-2.06.exe`*input* `-o` *output* `-c 3`
Runs the algorithm using 3 consistency transformations (default: 2 for number of sequences < 50, 1 otherwise).

`quickprobs-2.06.exe`*input* `-o` *output* `-r 3`
Runs the algorithm using 60 refinement iterations (default: 30 for number of sequences < 50, 100 otherwise).

`quickprobs-2.06.exe`*input* `-o` *output* `--mem-limit 30000`
Attempts to fit computations into 30,000 MB of RAM memory (default: 55,000 MB).

Bulk mode

For convenience QuickProbs 2 has been equipped with bulk mode which allows multiple sequence sets to be processed at a single run. In bulk mode *input* and *output* parameters have to be existing directories. Software automatically process all files from *input* and stores alignments in *output* under same file names.

OpenCL configuration

GPU driver timeout under Windows

By default Windows resets GPU driver after 2 seconds of non-responsiveness. This may be insufficient for QuickProbs. To change this setting do the following:

1. Go to registry editor: Start –> Run –> Type in "regedit".
2. Go to "HKEY_LOCAL_MACHINE \SYSTEM \CurrentControlSet \Control \GraphicsDrivers".
3. Create a key of type "DWORD (32-bit)" and name "TdrDelay" with a value of 10 as Decimal value.

AMD graphics cards

In order to maximally utilise computational power of AMD graphics processors some environment variables have to be set. They are described below.

1. GPU_MAX_WORKGROUP_SIZE For unknown reasons maximum size of workgroup for AMD devices reported by OpenCL is lower than physical hardware limit. E.g. for Radeon 7970 it reports maximum size of 256 while real limit is 1024. This environment variable allows one to override the default setting. Read in the device documentation maximum physical size of workgroup and set the variable to this value.

2. GPU_MAX_ALLOC_PERCENT This value indicate maximum percentage size of single OpenCL buffer related to the total size of GPU global memory.

NVidia graphics cards

Some instabilities were experienced when running OpenCL on GeForce 980 under Windows 10 x64. In particular, from time to time the software crashed during OpenCL initialisation. Most probably this was due to bug in 359.06 driver–the same situation was observed for other OpenCL applications. If you encounter similar problem, please run QuickProbs 2 again.