NCAP
–
NGS Cleaner and Preprocessing

Christian Meesters
Steffen Rapp
Benjamin Rieger

Zentrum für Datenverarbeitung (ZDV)
HPC Group
University of Mainz

Institute for Molecular Genetics
NGS Facility
University of Mainz

Mainz, May 22, 2016

Contents

Chapter 1

Theory of NGS-Quality Issues

Let  be our sequence of interest. And  this sequence with adaptor;

Appendix A

NCAP – Development

The following chapters are intended for developers of NCAP. You are most welcome to participate! Contributions may not only be code-based: Any bug report, hint on how to improve the manual, or feature request is welcome.

A.1 How to retrieve a working copy of NCAP

NCAP can be retrieved from github (see `https://github.com/cmeesters/ncap`). The git versioning system is available for every operating system. Please contact your local administrator, if you don't know how the program can be installed.

A first checkout can be done as follows, provided you have a github account:

```
$ git clone git@github.com:cmeesters/ncap.git
```

A.2 Building the development version

NCAP uses `scons` (see `www.scons.org`) as a build system (UNIX's `make` is the most well-known counterpart). In order to build a git download just type

```
$ cd your-ncap-download-path
$ scons
$ scons -c # will clean up after building
```

`scons` is available for download on the `scons` web page. However, many systems provide packages for `scons`. Please consult your sys-admin if in doubt.

A.3 Packing a Release Version

Update the `ncap` version in the `SConstrust`-file, first. `ncap` follows the GNU version numbering scheme: major.minor.revision .

Prior to packing a release, please update the repository first:

```
$ git pull
```

This is because `ncap` can report is version and revision number and updating first makes sure that users reporting bugs can be sure about the release.

Typing

```
$ scons --static
# and subsequently !!!
$ scons --pack_release
```

will create a directory called `dist` with the appropriately packed files for the current system. The shown static build is optional.

The `VERSION` parameter is just the version string, which will be used for naming the download directory as a unique identifier of the NCAP version.

A file containing the md5 sums for all produced files is written out, too.

A.4 Building a Debug Version

Invoking SCONS like

```
$ scons --debug_build
```

Will compile and link with `-g`, `-pg`, and `-DDEBUG` flags. The first two flags enable using the GNU debugger (by writing a file named `gmon.out` when NCAP runs). The third flag turns internal error handling routines of NCAP off. All this is also used with regard to the debuggers we use.

A.5 Selecting Compilers

For an explicit compiler seletion try:

```
$ scons --compiler=g++-4.8.2
```

for instance.

Currently the GNU C++ compiler, version 4.8.2 and support for C++11 is the minimum requirement.

A.6 Parallel Builds with Scons

Like the infamous GNU `make` tool, SCONS allows using the `-j` option to build an application in parallel:

```
$ scons -j <number of parallel compiler calls>
```

A.7 Including Sandboxed Code

If a Developer wants to create sandbox code, that is code which should not turn up in productive versions of the program until completion of that code, the code in question can be framed with preprocessor guards like this:

```
#ifdef SANDBOX
// your sandbox code here
#endif
```

This guard can be applied to entire `cpp`-files.

Issuing the `--sandbox`-flag to scons will turn the `SANDBOX`-preprocessor variable on.

A.8 Other Options

```
$ scons --help
```

will list a bundle of additional options we included to ease the NCAP development.

A.9 Used Libraries

A.9.1 OpenMP

NCAP uses OpenMP to spawn threads on shared memory systems. This way some algorithms can reach a tremendous speed-up. Please install the appropriate header for your system using the system's packaging tools.

gcc from version 4.4 on will support OpenMP 3.0. We recommend using gcc as the compiler. However, necessary instructions for other compilers are linked here: http://openmp.org/wp/openmp-compilers/.

A.9.2 boost

NCAP makes heavy use of boost (see http://www.boost.org/ for download and instructions). In order to ease the installation process, here a shortcut for Unix-like systems:

```
$ # need to install MPI library
$ # Debian and Ubuntu users:
$ sudo apt-get install lam4-dev libopenmpi-dev
```

Or check out the git version of modular boost.