February 21st, 2011 | ENG 317 Assignment 3 Spring 2011 | Colby Blair

I. Introduction

Thank you for purchasing BioBlast, a computational resource for DNA analysis on amino acid and protein sequences. This installation guide will help you understand all the underlying functionality of BioBlast, and give you resources and information on how to compile and run it to achieve maximum performance. It is important to pay close attention to these instructions and not deviate, unless you do so to gain performance, in which case you can.

II. Install Your Prerequisites

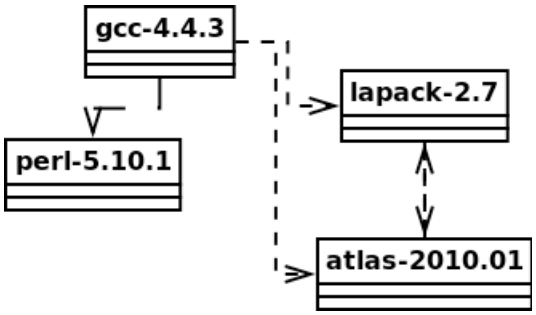

Figure 1

To install BioBlast, the administrator first needs to install some other programs that BioBlast depends on: gcc-4.4.3, perl-5.10.1, lapack-2.7, and atlas-2010.01. To install these, see the following:

1. gcc-4.4.3

To install this version of gcc, your Linux Operating System user manual must be read in the section on how to upgrade your compiler.

2. perl-5.10.1

To install this version of perl, visit http://www.cpan.org/ and download the source. Make sure to install perl as a module, and not override your default version. Once finished, this version of perl should be installed in /mnt/home/module/apps/perl, and not in the default location /usr/bin . Also the installer should set the environment variable PERLHOME to the module path above.

3. lapack-2.7

For lapack to be installed, use the source that come with BioBlast. You should compile it with gcc-4.4.3, and link the lapack compile against atlas-2010.01, which you've already installed.

Warning! Compiling with other versions of gcc may cause BioBlast to crash randomly. This will not cause you to lose your test data, unless BioBlast configure files are overwritten, in which case you may lose all your data.

4. atlas-2010.01

To install atlas, one would use the source that comes with BioBlast. Compile atlas using gcc-4.4.3, and the lapack install above.

III. Running BioBlast

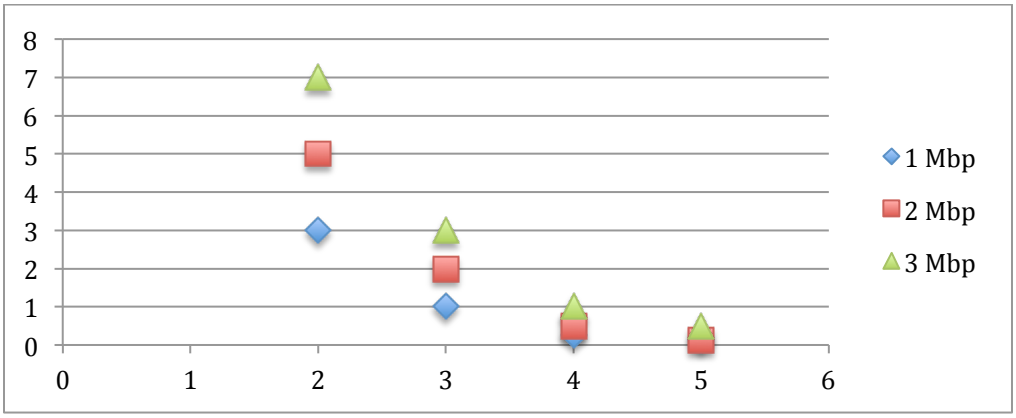

Figure 2

To run BioBlast, the biologist would type 'bb –i <input file> -o <output file>' on the command line. To get optimal performance, there are a set of optional options you can use: '-p <integer>' for running local alignments in parallel, '-w <integer>' to set w-mer size, to name a few. From Figure 2, it is clear to see how performance increases with each additional node for –p mode.

ANALYSIS

The parody violates a lot of the rules that we have learned this semester. They are unfortunately common in a lot of software documentation. Software is needed by a lot of customers that are not disciplined in software engineering. Software is also produced by many people that are not necessarily disciplined either. This attempts to relieve the large demand for different programs, but results in a lot of examples I have tried to include in this parody.

The first general problem is that the intended audience is the customer, who is probably a biologist. The parody lightly brushes over deep Linux concepts that would make no sense to the audience. Not only that, but the processes, as written, would even have experienced engineers confused.

The parody also has problems with format and diagrams. The header scheme is unclear to which labels are subsections and which are sections (I vs. 1). Figure 1 is ambiguous and doesn't help the consumer sort out dependencies. The graph seems to show an improvement in performance, but doesn't give any context to the data (missing axis labels, what are Mbp's?).

The format of the warning highlights another problem that is very common in technical books. The custom icon for warning is unnecessary versus an international symbol. The warning itself attempts to explain a situation that needs more than a short explanation, and in the end would create more frustration.

General language in the parody is also poor. The parody uses subjunctive ('biologist would type') as well as passive ('For lapack to be installed') instead of the imperative and active. The same actor, the customer installing the product, changes from administrator to biologist to installer, which is confusing. The parody also has some logic errors. Lapack and atlas both depend on the other already being installed before being installed themselves.

In conclusion, the parody was an opportunity to take some of my previous errors and examples I work with, and create a parody from them. When taken to the extreme, it is easier to remember why to avoid these problems in their subtler instances.