Final Project: Protein Docking

D. Jiménez, T. Monreal, E. Morancho

08-09-Q1

Index

1

Final Project

1.1 Objective of the project

The objective of the project is to evaluate if the student is able to apply different optimization techniques (the possible ones), using a correct optimization methodology, to a given source code.

1.2 Description

We propose you to optimize a known and used protein docking application, called `ftdock`. In fact, we propose you to optimize part of the `ftdock`. We have already modified the code so that the execution is stopped at some point with a message (*"PCA Optimizations stop here"*). At this point, the elements of a matrix are writen to stdout. You can redirect that output to a file to be able to check the correctness of your optimizations.

You should apply the techniques and methodology used in the course to re-code the parts of the code that you consider necessary to reduce the execution time of the program.

The final project should be done in pairs.

1.3 Project Description

`ftdock` performs the rigid-body docking of two proteins: one small and the another bigger than the another. Docking means the way the two proteins can be joined in the space. In this final project, we are going to focuss on the computation of the electrostatic matrix at the begining of the program. Electrostatic information is important to be able to accept or not a possible relative position of the two proteins in the space.

There are two parameters that you have to use when running `ftdock`: `-static` and `-mobile`. `-static` parameter is used in order to indicate the parsed protein that is big. `mobile` is used in order to indicate the parsed protein that is small. You can find the parsed input proteins in the `proteins.tar.gz` file.

1.3.1 Source code and input files

In order to be able to run and test your final project you need the following files:

1. `fftw-2.1.3.tar.gz` : FFTs library.
2. `gnu_licensed_3D_Dock.tar.gz`: Source code to be optimized.
3. `proteins.tar.gz`: Parsed proteins.

The application that you should optimize is ftdock, which is under the progs directory when decompressing the gnu_licensed_3D_Dock.tar.gz file.

1.3.2 Installation Steps

We ask you to use the Intel Core2 Duo of the lab in order to be able to compare the optimizations done among all the students. However, if you want do some experiments in another processor (or you are not able to use the lab computer for some reason) you should specify the processor and environment setup you have used in the final document that you have to submit to us, so that we can understand the improvents achieved by your optimizations. The optimization flags should be the ones given in these installation steps. However, if you want to use others you can do it, specifying which is the improvement achieved in this case by the compiler flags you use.

In the case of the library installation you have to follow the following steps:

```
> cp fftw-2.1.3.tar.gz PATH_LLIB
> cd PATH_LLIB
> tar -xvzf fftw-2.1.3.tar.gz
> cd fftw-2.1.3
> mkdir installation
> env CFLAGS="-O3 -march=pentium4" ./configure --prefix=$PWD/installation
> make
> make install
```

PATH_LLIB is the absolute path that you want to use for your FFTW library installation. Note that we are compiling with pentium4 because we are using a gcc compiler that does not have support for core2 march type. With gcc.4.3 of later, you may want to use core2.

Why we are using gcc.4.1.2? That is the one installed for sure in the lab and it seems that gcc.4.1.2 obtains better performance than gcc.4.3 for some studies done. In the case of the application installation you have to follow the following steps:

```
> cp gnu_licensed_3D_Dock.tar.gz PATH_APLI
> cd PATH_APLI
> tar -xvzf gnu_licensed_3D_Dock.tar.gz
> cd PATH_APLI/3D_Dock/progs/
> vim Makefile
change the FFTW_DIR variable value with that:
FFTW_DIR = PATH_LLIB/fftw-2.1.3/installation
> make
```

PATH_APLI is the absolute path that you want to use for your application (ftdock) installation.

1.3.3 How to run it?

The `ftdock` program is located in `PATH_APLI/3D_Dock/progs`. We suggest you to use the following parameters in order to analyze and optimize the `ftdock` program:

```
-static 2pka.parsed -mobile 5pti.parsed
-static 1hba.parsed -mobile 5pti.parsed
-static 4hhb.parsed -mobile 5pti.parsed
```

Redirect the standard output to a file, in order to be able to compare/check the program results.

```
./ftdock -static 2pka.parsed -mobile 5pti.parsed > absolute_path/output
```

Note that the parsed protein should be in your PATH variable or in the current directory you are running `ftdock`.

1.4 Evaluation

The evaluation of this work will be based on the optimizations you make, the source code of your optimizations, the shell scripts you have used in order to run, check and test your optimizations, and

the pdf report you submit. That document should explain all the optimizatons you have done, the methodology you have used, results, speedups, etc.

The methodology we expect you to apply is that we have explained at the beginnning of the course. Remember that the analysis that you do about the results you obtain is as important as the final speedup you get. So, for instance, we expect:

- For each optimization step, you should show the weight of the most consuming functions.
- Which optimization options you have for each part of the code.
  - It would be good to analyze them one by one (and all together is they are compatible).
- Possible optimizations that you thought but they didn't work.
  - In this case, it would be good to analyze why those optimizations didn't work.
- Figures with speedup's, time, etc.
- Assembly code analysis if it is needed.
- Etc.

1.4.1 Which documents you have to submit us?

You have to submit a .pdf file with all the analysis of the results, and the methodology of the optimizations you have done. This .pdf shouldn't have more than 15 pages. Also, you have to submit the source code, makefiles, and any scritps you have used in order to do your work.

Enumeration of the documents, etc. you have to submit:

1. .pdf document
2. scripts used
3. execution output results
4. patches or source code of your optimizations
5. makefiles
6. REAMDE to run your scripts
7. INSTALL guide to install whatever you have done

How to submit it?

You will have to create a .zip file with all the documents explained above and submit it via the RACO of PCA.