Reflective Essay

Biocomputing II Project

MSc Bioinformatics and system Biology

Sungjae Song

kardo41@gmail.com

1. Approach to the project

1.1 Interaction with the team

We had meetings once a week (sometimes twice) regularly, talking about our project. Greg took database tier, I took business logic tier and Rudo took web-front tier. Since my part was middle layer, I tried to make connection between the layers, understanding how I could load data from mysql database and how my calculation will be imbedded into web front page.

I had some conversations with Greg about talking the type of database I will get (such as gene ID, coding region, exon starts ..etc) and features of each types (for example in coding region sequence, I will get 4 alphabetical format , 'ACTG' mainly, however we found that character N exist and it is one of ambiguous nucleotides. This make me be more careful when I treat with sequence data. With Rudo, I explained that how my function (definitions) will work and how the result will be shown. Unfortunately, she could not join us on last days of our project so that I could not show her my updated version of functions.

1.2 Overall project requirements

Main aim of this programming project is creating genome browser, especially in human chromosome 3 for our group. 3 layers of tier are introduced, Database tier, Business logic tier and front-end tier. We need to load the Genbank data into the database and web front page will show the results of search, containing (1) details of protein (ex. Product name, gene ID etc.), (2) complete DNA seqcences with coding region highlighted, (3) codon usage frequencies, (4) finding restriction enzyme site and identifying this enzyme is good or bad.

First, we discussed about types of file that we will use, we agreed that the database will be stored in SQL format as table, and then the database will be imported into the python. The calculation of imported data (including finding restriction enzyme site) will be done in python file, and finally with cgi script, the information would be delivered into the website, by using html file format.

1.3 Requirements for my contribution

The purpose of middle layer is implementing data access tier to middle tier, making logic tier which provides modules including calculation. So I imported data access tier, and then calculate with them. I made several python files for calculation, each definitions in file will calculate the information that you need. Codon Usage, finding restriction enzyme site and identifying whether this enzyme is good or bad, is main calculation of my files, it is stored as codoncal.py, res.py, goodbad.py individually. The pre-calculation of codon usage of human chromosome 3 is done with codoncal.py, the data is stored in csv file.

2. Performance of development cycle

First, we decided to do our task individually. For database tier, greg started generating a table and in middle layer, I started making calculation file with python and Rudo started to build the website. Our plan was after we finished our task first and then was going to make a bridge between them and It. While we are meeting once/twice a week, we had conversation about how much we did it until we met and if there are any problems occurred, we helped to each other. Development was going really well, each team mates has done their task with their enthusiasm.

3. Development process

In middle layer, there are 3 calculation python files; codoncal.py, goodorbad.py and res.py

- Codoncal.py

I started the project with designing codon frequencies. In codon usage pages, we need to calculate (1) percentage of each codon appearance in the sequence. (2) relative ratio of each codons in amino acids. In codoncal.py, first, I stored all codons and their counterparts of amino acids into codon_aa dictionary.

In cds(coding region sequence), sometimes they have ambiguous nucleotide, such as N(any nucleotide) or Y (T or C), so we had to remove codons which contains these nucleotides. Rem_vague() has the function that get rid of these, to prevent any error occurrence in calculation

Count_codons() divides the whole coding sequences into 3 (triplets) and it will count each codons respectively. This calculation is needed for Count_perce_codons(), which will calculate whole percentage of codons. Count_perce_codons() definitions, it actually have function which divides count_codons() by whole length of amino acid sequence(len(cds)/3),

Count_relev_perce_codons() will show relative ratio of each codons in one amino acid, for example (‘AAA’,’K’.’1.0’). 1.0 is maximum value for relative ratio, it means in that amino acids, only one type of codon (there is no AAG for K) was translated into that amino acid.

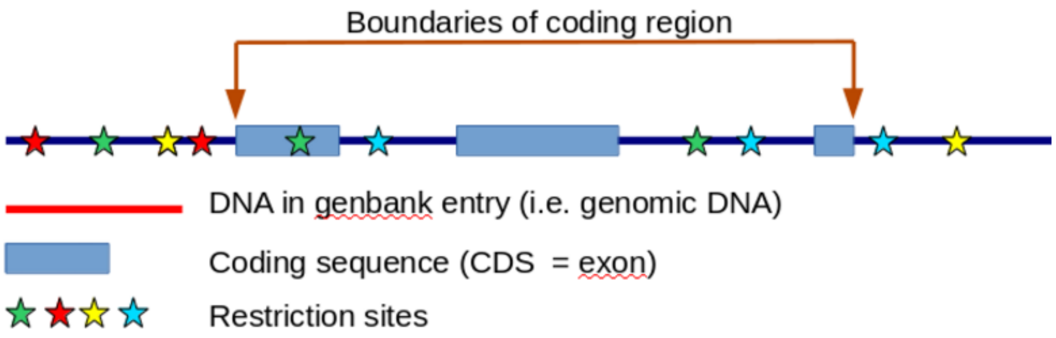
- Res.py

In res.py, name of the enzyme and its sequence are stored in dictionary called ’enzymes’. I have 3 different function in this file, giveme_res() will search the location of whole restriction enzyme site, BamHI, EcoRI and BsuMI. If users want to search more restriction enzyme site, they can simply add information of restriction enzyme sequence in dictionary ‘enzymes’

If you want to search single restriction enzyme site, you could use store_list_single() it will show you the multiple restriction enzyme site of BamHI or EcoRI or BsuMI. In addition, I made custom search function, which will allow users could search with their own restriction enzyme sequence. After they typed the sequence of their own restriction enzyme, custom_search_res() will find the location.

- goodorbad.py

First of all, sequence of the coding region will be imported from data_access.py file. It contains start and end site of coding region and information of whether 5' and 3' is partial or not. Given sequence will be stored in extract_seq and restriction enzyme site will be calculation with single_res function in res.py. Good_bad() check if there is any partial end first, if there is, we could not figure it out it is good or bad enzyme, however if both are non-partial end, the next step will go on. The next step is checking any of enzyme site is located between coding region.


Boundaries of coding region
DNA in genbank entry (i.e. genomic DNA)
Coding sequence (CDS = exon)
Restriction sites

If any of restriction enzyme site is located between coding region, it means bad, since they cut off the coding region. So the function good_bad() will check the restriction enzyme site(int(k)) is between the coding site (coding _s and cooding_e)

4. Coding test

At the first time when I was making python file for calculation the name of the data access python file was not decided, so I just made the function(definition) and test it with dummy data. Greg and I had some conversation of data access tier, and he told me the format of the data coming from the data access tier, so I made up some dummy data with the same format and I tested with it.

Also, if you see the python files in middle layer, each calculation file has #test part, it could let you check whether this function is working properly or not. Normally it is tested with single simple

5. Known Issue

At the beginning, I spent lot of times to make a function of calculating percentage of codons and its relative ratio and there were some problems occurred in goodbad.py (you could check trash folder), however everything was solved in the end. However, even if I finished making all my calculation file, we need to call(imbed) this function into the website, however, Rudo could not test with it so that I do not know it will work properly in webpage. (She made her own cgi script with dummy data, but did not imbed my function since she could not join us at the last days of projects).

Also in precalculation of human chromosome 3, greg told me that there was some error occurred in data access layer so that could not retrieve full sequences. Therefore, I pre calculated it with part of sequences of chrosome3 instead (dummy data) , even if It is small amount.

6. What worked and what didn't - problems and solutions

Every calculation is working correctly, however there was some error occurred in data access layer, so that I did not get full sequences of chromoseome 3, so instead I calculated with dummy data(part of sequence of chromosome3) and made csv file.

- What was good?

it was good to think about exsitence of ambiguous nucleotides(N,Y etc.), so that make sure that prevent any errors might occur. Also after we realised the format of coding location will include information whether it is 3’ or 5’ partial(< or >), which make us think critically. (if one of them/both are partial, we cannot judge whether some restriction enzyme is good or bad.)

7. Alternative strategies

If you see res.py, I actually wrote down the dictionary of ‘enzyme’. Alternatively I could make csv file first, which will stored the all(name of enzyme: sequence) information, for future added one. In this case, I will load csv file first, with open('./enzyme.csv', 'rb') as csvfile: and it will used as dictionary. This way is better for fluid dictionary, if you have to change/add types of enzyme continuesly. Similarly, in codoncal.py, I wrote down all (codons : amino acids) , however, alternatively I could store it csv file and then load it.

8. Personal insights

This project was good opportunity for me to understand how gene search engine is working and learn how database is stored and imported and calculated. At the first time, there were many errors occurred on calculation function, but it makes me more critically think about the class of information (str,int. etc.) and loops algorithm. Even if we could not successfully make the proper search engine site, I am satisfied that my calculation(middle layer) has tested and it worked successfully. This experience will help me when I am making this kind of database search website in the future and I want to try again making proper working search website in future if I have another chance.