Exercise 'Protein Prediction II'
Winter Term 2010/11

Sheet 2

General information

- Our course homepage, containing lecture slides and exercise sheets: http://rostlab.informatik.tu-muenchen.de/cms/pp2/

- Time and place: Friday, 13:30 – 15:00, room MI 00.11.038 ('John von Neumann')

- Contact: hampt@rostlab.org, schaefer@rostlab.org, vicedo@rostlab.org

- Send an email (one per group) to us (!) including the paths to your results (answers, scripts, datasets) until Friday November 12, 9:00 am. Scripts should be executable for us so that we can reproduce your results. Please do not send files as email attachments, the paths to your solutions are totally sufficient. Again, everything has to be readable by us, so please check the permissions of your directories/files! Otherwise we cannot consider your solutions.

Exercise 2: Redundancy reduction and BLAST database preparation

In the last exercise, we parsed the SwissProt database for GO-annotated protein sequences. Usually, protein sequence databases contain a lot of redundancy in terms of identical or similar sequences. Since our performance results should not be biased by those over-represented sequences, the next step would be to reduce this redundancy. A common tool for this purpose is CD-HIT. It takes a sequence database and filters out those sequences that are above a given sequence identity threshold.

You will find CD-HIT here:

/mnt/opt/data/pp2_exercise/cdhit/cdhit
Familiarize yourself with the command line options of CD-HIT. Now, take the GO-annotated sequences from

/mnt/opt/data/pp2_exercise/sprot_go.f

and run the redundancy reduction on a 80% threshold. Store the reduced sequence set in a fasta file called sprot_go_80.f.

Hint: Overall three CD-HIT parameters are necessary for that task. Leave everything else on its default values.

Additionally, answer the following questions:
How many sequences do you find now in your redundancy-reduced set? What is the ratio compared to the unreduced set?

In order to efficiently search against a sequence set using BLAST, that set has to be prepared at first.

Use the tool /usr/bin/formatdb with the only parameter -i sprot_go_80.f. This should result in three additional files sprot_go_80.f.phr, sprot_go_80.f.pin, sprot_go_80.f.psq

Exercise 3: BLAST parser

BLAST is a standard tool in bioinformatics so you are probably already familiar with it. In short, BLAST is used to efficiently find significant hits of similar sequences to a given query sequence in a sequence database. Since our objective will be to find similar sequences to a query sequence and infer their GO annotations, we will use BLAST here, too.

Make a sample BLAST run against our prepared database (s. above) and familiarize yourself with the output. The query sequence is one of the GO-annotated sequences and its fasta file could be found here:

/mnt/opt/data/pp2_exercise/test.f

Run BLAST like this in the same directory as your BLAST database:

/usr/bin/blastpgp -i /mnt/opt/data/pp2_exercise/test.f -d sprot_go_80.f -e0.1 -h0.1 -j2

Familiarize yourself with the command line options that we use here (man blastpgp).

What is the first hit BLAST found? Is that surprising to you? Give a short explanation in your answers.

Now write a program that parses the BLAST output. Your program should return the identifiers of the BLAST hits (in our case the GO-terms, comma-separated) along with their e-values ('Expect'). We leave the specific output format to you.

Hint: Each hit is introduced with a line that starts with:
>GO:......

We are only interested in the results from the second run ('Results from round 2' in the output)!

Hand in your parsed results.