INFO-F-438 – Algorithms in Computational Biology

Assignment 1: Multi-digest

Sébastien Collette (sebastien.collette@ulb.ac.be)

1 Original PDP

As discovered by H. Smith in the 70s, DNA molecules can be split in fragments using the restriction enzyme HindII. This enzyme breaks a long chain at occurrences of sequences GTGCAC and GTTAAC, resulting in a set of fragments whose length can be measured. Positions in the chain where one of these sequences appear are called *sites*.

Two processes have been used :

Complete digest, where there are no remaining GTGCAC or GTTAAC in any fragment. In this case, the multi-set of lengths obtained is exactly the set of distances between two consecutive sites.

Partial digest where we obtain all possible fragments between any two sites, i.e. for a molecule containing s sites, we get $\binom{s}{2}$ fragment lengths.

To formalize the problem, we will represent sites in a DNA molecule by integers on the real line. Given the multi-set L of $\binom{s}{2}$ integers, the **Partial Digest Problem** or **PDP** consists in finding a set of sites X so that

$$|X| = s$$

$$\Delta X = L$$

where ΔX is the multi-set of all distances between all $\binom{s}{2}$ pairs of items of X.

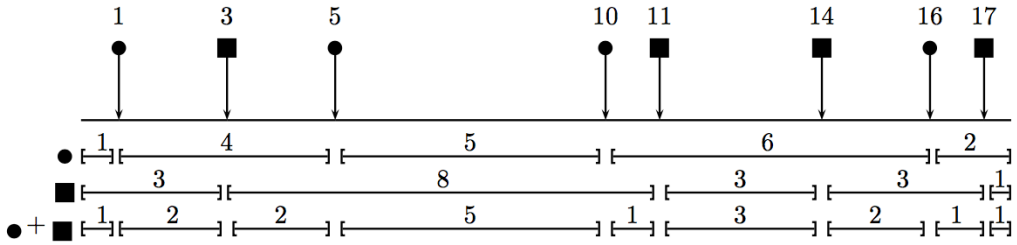
This problem was presented in details during lectures, and is extensively covered in chapter 4 of the reference book.

2 Multi-Digest

Reconstructing the postions of sites given fragment lengths – also know as *restriction mapping* – turned out to be a very important process, because it was a first approach towards DNA sequencing, multiple decades earlier.

Getting a map of a single type of sites was interesting, but other restriction enzymes could be combined. In the **Double-** or more generally **Multi-Digest** Problem, DNA molecules are exposed to k different enzymes, first individually, and then combined. We end up with k complete digests for each of the enzymes, and then one complete digest for all combined enzymes.

Here is a textbook example with $k = 2$:



Given the complete digests $\{1, 2, 4, 5, 6\}$ and $\{1, 3, 3, 3, 8\}$ for each enzyme, as well as $\{1, 1, 1, 1, 2, 2, 2, 3, 5\}$ for all combined enzymes, the goal is tho reconstruct a set of k compatible restriction site maps, in this case $\{1, 5, 10, 16\}$ for enzyme A, and $\{3, 11, 14, 17\}$ for enzyme B.

3 Your work

The first step of your work consists in rewriting the Multi-Digest Problem above as a **well-defined** problem (problem formulation with input and output, see example on page 86 of the book for the original PDP).

The second step is to write a program which can solve the Multi-Digest problem for any $k \geq 2$, where k is the number of restriction enzymes used. Provided Multi-Digest with large k often have multiple solutions, your algorithm only needs to produce a single set of sites compatible with the input. Therefore, make sure that your algorithm stops running as soon as a solution is found.

For this purpose, you can use an exhaustive search algorithm. You should also think about making your algorithm as efficient as possible. We've studied multiple variants of the algorithm for the original PDP, and we saw that the way an exhaustive search is written can drastically improve the performance of the algorithm.

4 Requirements

1. The source code of a program performing the above task. Use your favorite programming language among Python, C++, C, Java.
2. A number of relevant, non-redundant comments and explanations about your code, either in the form of comment lines, or in a separate report.

3. The result output by your program on the following input examples :
   $k = 2$, individual digests $\{1, 2, 4, 5, 6\}, \{1, 3, 3, 11\}$, and combined $\{1, 1, 1, 1, 2, 3, 4, 5\}$

   $k = 3$, individual digests $\{2, 7, 9\}, \{4, 4, 5, 5\}, \{2, 16\}$, and combined $\{2, 2, 2, 2, 2, 3, 5\}$

5 Deadline and Evaluation

Your work must be submitted by e-mail to `sebastien.collette@ulb.ac.be`, by March 7th, 2017. The evaluation will be based on the following criteria :

1. the general understanding of the instructions,
2. the proper use of the programming language,
3. the efficiency and correctness of the implemented algorithm,
4. the clarity and relevance of the comments and explanations

Plagiarism will be severely sanctioned. Plagiarism cases include reusing someone else's written or drawn material, or any kind of work, without an explicit quote or reference.