ITX Integrative Transcriptomics
Dr. K. Nieselt, Faculty of Science

EBERHARD KARLS
UNIVERSITÄT
TÜBINGEN

Lecture: Bioinformatics I — WS 2015/16

Assignment No. 1 (8 points)

Hand out: Monday, October 12
Hand in due: Monday, October 19, 10:15
Direct inquiries to `alexander.seitz@uni-tuebingen.de`

Theoretical Assignments

1. **Smith-Waterman today** (2P, max. 2h)

   The Smith-Waterman (SW) algorithm was published in 1981, that is more than 30 years ago. Because of its time (and memory) complexity it is not really feasible to apply it for example when a homologous sequence is compared to a large-scale database (such as GenBank).

   Give an example for a modern bioinformatics problem and discuss briefly (half a page) where the SW algorithm (maybe modified) nowadays still has an important role. Find one or two current algorithms / methods / implementations that use the SW algorithm (maybe modified)?

2. **DP Algorithm for finding motifs** (3P, max. 3h)

   The local alignment gives the best match of a subsequence of X to a subsequence of Y. An extended question is to find local regions of Y that appear more than once in X. For example, one sequence may contain a *motif* and the algorithm shall compute all copies of (parts of) the motif in the other sequence. Here is an example:

   ```
   X=AAAAATCCAAAAATGCAAAAATCC
   Y=TCC
   ```

   Let Y be the sequence that contains a motif and let X denote the sequence in which we want to find copies of the motif.

   Set up a dynamic programming algorithm that finds one or more non-overlapping copies of sections in sequence Y in sequence X. The algorithm shall output all local alignments whose score is above some threshold T.

   Hand in a pseudo-code and/or the DP initialization and recursion.

3. **Programming assignment: Needleman-Wunsch alignment** (3P, max. 3h)

In "Grundlagen der Bioinformatik" you had to implement the Needleman-Wunsch algorithm (in Java). Make sure that you still have this program available and adapt it if necessary such that it

- accepts command-line switches for input files and parameters
- reads two sequences from FastA files
- computes a Needleman-Wunsch alignment (with linear gap cost) for the two sequences
- outputs the best alignment together with its score
- gives helpful usage information when called with the parameter `-?`

If you do not want to use your own code, you can use the one provided. In order to accomplish all these tasks you then have to fix the `TODO`s in the provided java source files!

Furthermore, make sure your program is modular, we will expand it in the next assignments. If you are not unsure what 'modular' code is, ask your tutor.

Apply it to the sequences in the material file that can be downloaded from the ILIAS website. Try different parameters for the scoring and gap penalty. Which parameters work 'best'?

Write a short report of the results of your program as well as of the changes you made and why you think those changes had been necessary!

In addition, hand in the program itself as a zip file of the java source files.

Please read the questions carefully. If there are any questions, you may ask them during the tutorial session or via e-mail to your tutor. You will usually get an answer in time, but late e-mails (e.g. on Monday morning before class) might not be answered in time. Please send all your electronic solutions to `alexander.seitz@uni-tuebingen.de`. Please pack both your source code as well as the theoretical part into one single archive file. Source code should compile correctly. Make sure, that you export the source code and not only the binaries. Handwritten assignment solutions (e.g. for the theoretical part) can be turned in during the lecture.