QCB Midterm I, November 7th, 2023

Instructions

1. Download the sciprog-qcb-07-11-2023-FIRSTNAME-LASTNAME-ID.zip file and extract it on your desktop.

2. Rename sciprog-qcb-07-11-2023-FIRSTNAME-LASTNAME-ID folder:

   Replace FIRSTNAME, LASTNAME, and ID with your first name, last name and student id number. Failure to comply with these instructions will result in the loss of 1 point on your grade.

   like, sciprog-qcb-07-11-2023-john-doe-432432

   From now on, you will be editing the files in that folder.

3. Edit the files following the instructions.

4. At the end of the exam, compress the folder in a zip file whose name will look like:

   sciprog-qcb-07-11-2023-john-doe-432432.zip

   and submit it. This is what will be evaluated. Please, include in the zip archive all the files required to execute your implementations!

Exercise 1

Text rotations are used in the calculation of the Burrows-Wheeler Transform, a widely used approach that rearranges strings into runs of similar characters to allow the compression of biological sequences (e.g. the BWA and Bowtie algorithm make use of this technique).

You are asked to compute the n-th rotation of an input string and evaluate its goodness, by implementing these functions:

computeRotation(*text, offset*) that should work with positive and negative offsets, and also with offsets that are greater than the length of the string.

computeCompressibility(*text*) that should compute the "potential for compression" of the text as the number of different characters over the total length of the text.

Some examples of what the output should look like are provided here:

Text: Hello World
Rotation 2 of the text: llo WorldHe
Compressibility: 0.27

Text: lorem ipsum dolor sit amet consectetur adipiscing elit
Rotation -26 of the text: g elitlorem ipsum dolor sit amet consectetur adipiscin
Compressibility: 0.7

Text: Sometimes it's better to light a flamethrower than curse the darkness
Rotation 123 of the text: rse the darkness.Sometimes it's better to light a flamethrower than cu
Compressibility: 0.67

Exercise 2

The *hs_annotation.gtf* file is a GTF-format file containing a subset of the annotations for the human transcriptome from the GENCODE database. A sample line is reported below:

@ chr1 HAVANA gene 11869 14409 . + . gene_id "ENSG00000223972.5"; gene_type "transcribed_unprocessed_pseudogene"; gene_name "DDX11L1";

The columns schema is as follows:

1. **seqname** - name of the chromosome;
2. **source** - name of the program that generated this feature, or the data source
3. **feature** - feature type name, e.g. Gene, Variation, Similarity
4. **start** - Start position of the feature, with sequence numbering starting at 1.
5. **end** - End position of the feature, with sequence numbering starting at 1.
6. **score** - A floating point value.
7. **strand** - defined as + (forward) or - (reverse).
8. **frame** - One of '0', '1' or '2'. '0'.
9. **attribute** - A semicolon-separated list of tag-value pairs, providing additional information about each feature

Implement the following functions:

1. ***loadAnnotations****(filename, source = “”)*: loads the annotation file specified by the *filename* parameter and store the resulting annotations in a matrix-format object with row and column names. Note that the annotations in the “*attribute*” column must be splitted into as many columns as its individual parts (e.g., referring to the sample line above, *gene_id* is a part, *gene_type* is another, etc).

The function must:
- filter to keep only annotations coming from the given *source* type (if *source* != “”).
- output basic information about the loaded annotations, namely the number of annotation items and the list of the different sources and feature types represented in the file.
- return the produced data structure so that it can be used by subsequent functions.

2. ***plotGeneTypes***(*annotations*): plots a barplot with frequencies of each *gene_type* in the *annotation* object, as produced by *loadAnnotations*.

The function must:
- return as output an object containing the plotted frequencies.
- if there are multiple sources in the *source* column, the function must output one plot and one object per source type present in that column.

3. ***computeGeneTypeOccupancy***(*annotation*, *gene_type, relative=False*): computes how much genome is occupied by genes of the given *gene_type* according to the provided *annotation* (as produced by *loadAnnotations*). If *relative* = False, the occupancy is the number of occupied nucleotides; if *relative* = True, the occupancy is the percentage of occupied nucleotides over the total for all genes of all gene types in the *annotation* object.