sequence-assembler

This project is a program to assemble a given set of DNA-Sequences into one single strand by using common Algorithms. Overall the Project contains three main components core.py, doubleHelix.py and substitution.py which can be executed indivudally.

core.py

This file contains the main components of the project. It can generate a graph based on a given data set. This graph has weighted edges, which are also generatet based on the data set. After the generation, the program uses a simple algorithm to assemble the graph of sequences into one strand. The algorithmen simply considers the highest weighted edges and unifies the vertices which are connected with it. The target vertex is merged into the source vertex, based on the suffix of the source vertex and the prefix of the target vertex. After that the graph gets *refactored* and not newly generated. This algorithm is repeated until no edges are left.

doubleHelix.py

This file is an addition to the core.py. The original algorithm gets extended and considers the fact, that the Sequences could be from a double Helix and not a single Strand. With that in mind, the algorithm takes the base data and checks in which strand a single sequence is probably contained.

substitution.py

This file is also an extention to the core.py. The original algorithm gets extended and coniders, that there are substitution errors. With that in mind, the two sequences getting checked, consider an edit distanz. If this edit distance is smaller then the defined error Quota, the two sequences are still considered to have an edge with the given weight, even if the suffix of the source and the prefix of the target is slightly different.

Install

Install all base dependencies via "pip3 install .", if you want to export the graphs as raster graphics you have to install pycario as shown here. If pycario is not installed, do not try to export a plot directly, but use the generated graphviz file and the corresponding online viewer. To generate a graph just copy the contents of the graphiz file into the left column of the viewer.

Usage

The interactive program can be started with "python3 run.py" if you are in the root directory of this project. If you do not enter a value when prompted, No or a default value is assumed. The default values can be safely edited directly in the run.py.

Parameters

- **Path**: Path to the fragment file to be processed.
- **MinWeight**: minimum necessary overlap between two fragments to join them.
- **BatchExecute**: if yes, then several calculations are performed(DEFAULT=10) and the best one is returned. The result with the fewest fragments is assumed to be the best.
- **PrintPlot**: if yes, then raster graphics are created for each slice. use only if pycairo is installed.
- **CoreAssembler**: if yes, the file is processed with a simple assembler (task 1)
- **DoubleHelixAssembler**: if yes, the file will be processed with a DoubleHelixAssembler (Task 2)
- **SubsitutionAssembler**: if yes, the file will be processed with a SubsitutionAssembler (Task 3)

Git branching model

The branching model of this Git is based on Vincent Driessen proposal from 2010

Python style guide

As a reference style guide, the Google Python Style Guide is used for this project

Comments and Docstrings

As already proposed by Google Python Style Guide this project uses docstrings to document the source-code