4.4. Process Flowchart

CoGeMPA basically has two main features. The first one is the metabolic profiling of gene data sets via KEGG-annotations. The second feature is the metabolic profiling, with an additional separation of input genes into the core- dispensable- and singleton categories based on EDGAR data. The most important elements which makes those features work are listed here (Fig.8):.

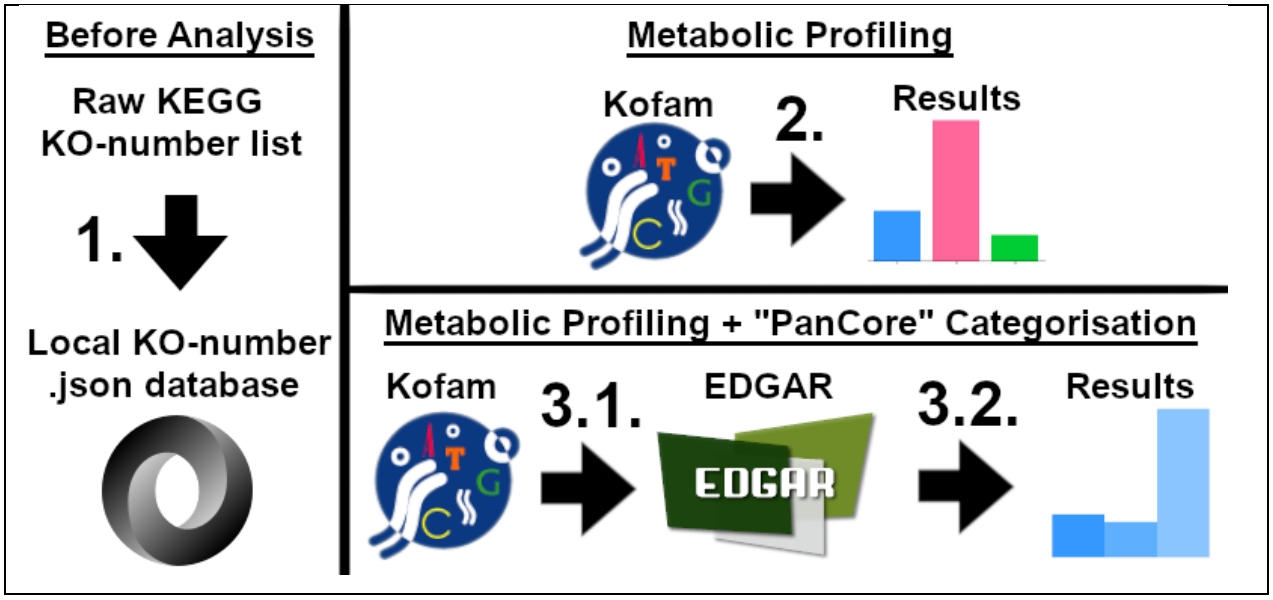

Fig.8: Simplified schema of the CoGeMPA workflow

1. → A local Ko-number .json database is constructed based on a raw list from KEGG. It is a dictionary with the KO-number as key and category info as the value.

2. → After KofamScan has annotated the amino acid sequences with KO-numbers, a lookup in the previously created database returns the matching categories information. The respective categories are counted and finally visualised via Matplotlib.

3.1. → Analogous to step 2., this step starts with a KO number assignment, but data gets not visualised right away, this is required for the downstream analysis.

3.2. → A temporary dictionary with the EDGAR-ID as the key and either 0,1 or 2 as the value is created. These numbers correspond to either the core- dispensable- and singleton “category” and are derived from the .csv downloadable with the .faa file in EDGAR. The corresponding KEGG category information, earlier linked with the EDGAR-ID, is sorted accordingly and visualised.

5. Usage

CoGeMPA is a Python application for Linux and Windows that processes and visualises KofamScan Output- and EDGAR pan-genom.csv files. It also includes functionality to directly process standard amino acid fasta's (.faa), if KofamScan is installed correctly.

Most of its features can be used under Windows, although Linux is recommended. This is because KofamScan and its dependencies require Linux to work. Here Ubuntu was used, which worked without a problem. It is also possible to acquire the KofamScan Output from the free Web-Tool "KofamKOALA" [11], but this has its limitations as mentioned earlier.

This guide assumes Python, pip and Conda are installed and added to PATH.

Installing KofamScan:

The KEGG website offers an installation guide aside the program downloadable at [47]. I followed this external guide using Conda for setup [48]. The required "KOfam"-Data Base is updated about every month, so it is recommended to keep it up to date.

Setup:

CoGeMPA and its config file (categories_configuration.json) should be placed into the same folder where the main KofamScan script resides. Its dependencies as listed in chapter 4.3. can be installed manually using **pip install .** To make the installation easier for the user, a helper script with the name "**Install Dependencies**" can be used.

The program requires a custom KO-Number database which can be installed with:

```
python .\CoGeMPA.py -i
```

This automatically downloads the most current KO-number list, converts it to the local DB format and sets the path in config. It is also possible to install the DB from the GUI and a fallback is provided via -d for manually converting KO-number lists which can be downloaded at [49].

Usage:

The user has two options for controlling the program, either the command-line or the GUI, most features are available in the GUI exceptions will be discussed. Examples for both will be shown here. This chapter lists the main options of CoGeMPA, a full list of all arguments can be found at (Fig.17).

Command-line:

A typical command-line call could look something like this:

```
python .\CoGeMPA.py -a -f1 .\buchnera.txt -f2 .\xanthomonas.txt -na -per -exc
```

The output from this will look similar to (Fig.14). In this case (-a) is the selected function, (-f1 and -f2) mark the “KofamScan files” that will be analysed and (-na, -per, -exc) are optional arguments for additional fine tuning, their effects are explained at (Fig.17).

```
python .\CoGeMPA.py -c -csv .\test_buchnera.csv
```

Another example: -c is the selected function, -csv marks the .csv that will be analysed.

Available functions:

-a: For the analysis of KofamScan files. Visualises the number of genes in each respective Functional KEGG category.

→ (Required: -f1) (Optional: -f2, -f3, -exl, -na, -pie, -jf, -dbj)

-c: For the analysis of “EDGAR .csv files”. Visualises the number of genes belonging to Core- and Dispensable- genome and singletons.

→ (Required: -csv) (Optional: -pie)

-r: For “fasta amino acid files”. Removes duplicate ID’s and their sequences from the .faa file. KofamScan does not work when duplicate ID's are present.

→ (Required: -e, -o) (Optional: None)

-d: For the raw “K0-number list”, Converts it to the locally used DB format and sets the path in config.

→ (Required: -db) (Optional: -o)

-i: Identical to -d, but automatically downloads the most current K0-number list by itself. (-db) can be used to change KEGG brite type!

→ (Required: None) (Optional: -db)

-cm: Combination of -a and -c. Visualises the amount of genes in each respective Functional KEGG category, split into Core- and Dispensable- genome and singletons for a singular “KofamScan file” and its corresponding “EDGAR .csv”.

→ (Required: -f1, -csv) (Optional: -exl, -na, -pie, -jf, -dbj)

-rko: Removes multiple KO-Numbers from “KofamScan files”. (Command line only)

→ (Required: -f1) (Optional: None)

GUI:

Once the user has typed in the following command, the GUI will start (Fig.9):

```
python .\CoGeMPA.py -sg
```

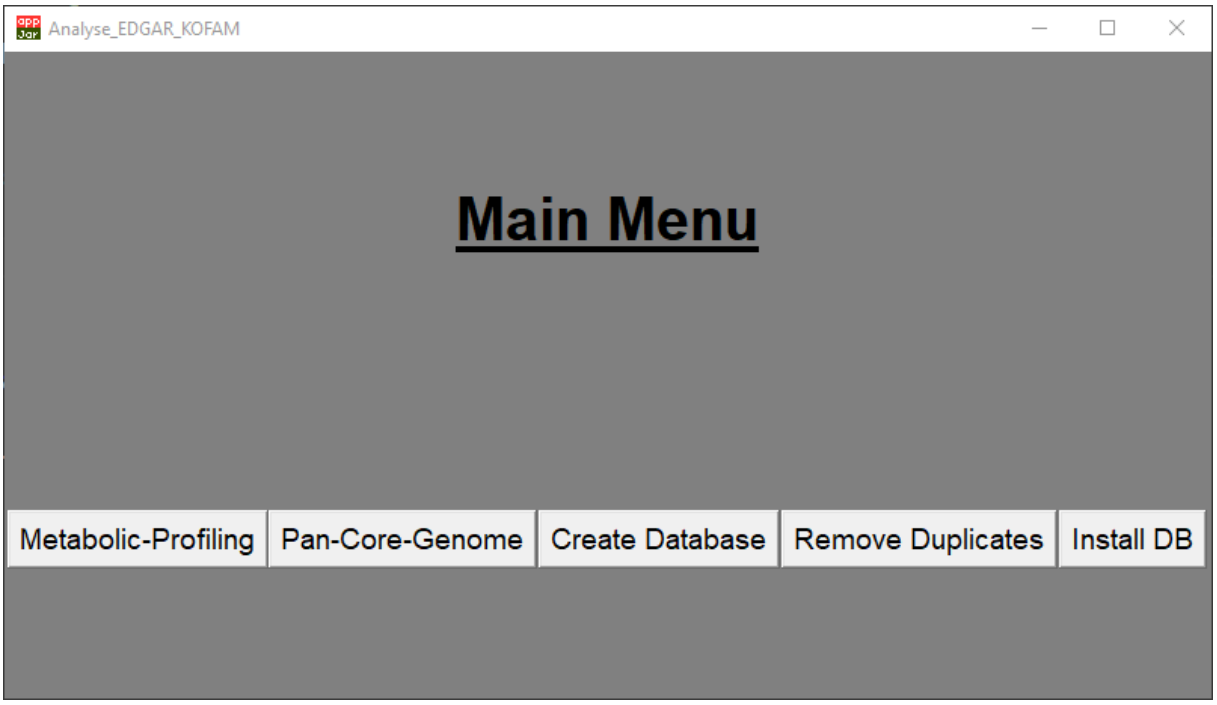

Fig.9: GUI main menu of CoGeMPA. The buttons correspond to discrete functions.

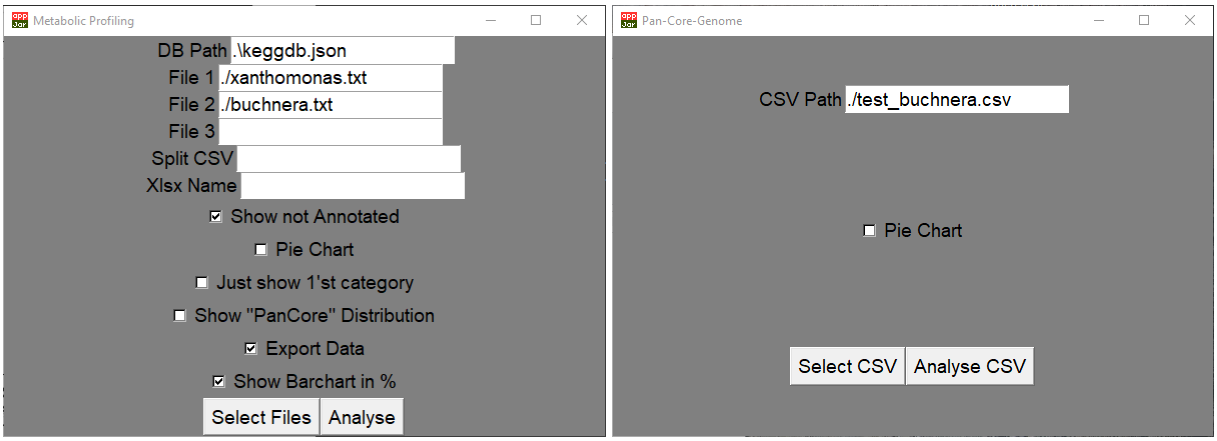

Fig.10: Two GUI windows corresponding to the command-line calls above.

The GUI is operated by clicking the provided buttons and Checkboxes. Everything is labelled according to its function and should offer a valid alternative to the command-line (Fig.10). Paths can be set either by typing or via the “Select”-button which will open the OS explorer.

Important optional arguments (command line only):

The optional arguments -fa (full auto) and -ra (remove auto) in connection with either -a or -cm can be used for directly analysing fasta amino acid files, without having to manually operate KofamScan. Here -fa engages the KofamScan Script and -ra acts as an automatic pre-processing step analogous to -r if needed.

Clickable graphs:

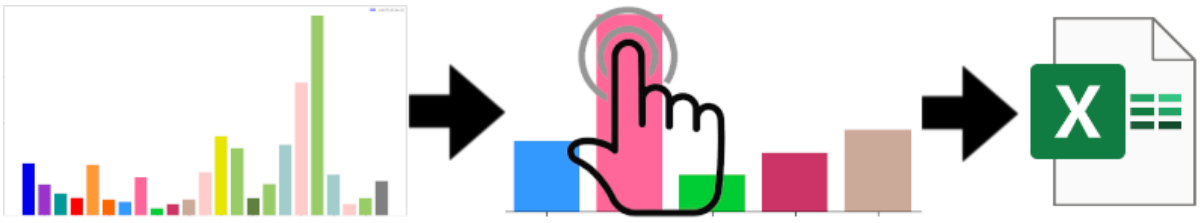

Fig.11: Flowchart shows attaining bar specific gen-, annotation data by clicking.

It is possible to export bar specific data as a xlsx. file by clicking the bar of interest. (Fig.11) The name is set with a timestamp. A line in this file might look like this:

XAFCFBP3836_RS0100575 ___ aldB; aldehyde dehydrogenase [EC:1.2.1.-]

The format is: ID from .faa file ___ KEGG DB Details. An excerpt from such a file to see formatting can be found at (Fig.17).