ReadMe

*Welcome to my master thesis repository*

*In this repo you will find all the supplementary material of my master thesis as well as PDF versions of my project proposal and master thesis.*

# output-descriptive-analysis/general-descriptive-analysis

This map contains the interaction datasets which are the result of the public data extraction and clean up process. These contain our interaction data as well as the metadata per species.

The map also contains an excel file in which the general descriptive analysis results can be found (corresponds to table 01, 02 and 03 on page 19 of my master thesis

# output-descriptive-analysis/network-descriptive-analysis

This map contains the full results of the network descriptive parameter calculation, which was performed using the NetworkAnalyzer tool from cytoscape, as explained in the master thesis. It also contains separate excel files which contain the extracted top 20 highest degree vertices from each separate bacterial HPI network, which corresponds to table A (also table 05 p21) , B and C on page 37-39

# output-goa

This map contains the full results of the goa analysis per species, as well as the summarising excel file which contains the summarized table (corresponding to table 06 p22) and the filtered top 20 results (table D-K, p40-47)

# output-subgraph-mining

This map contains a map per subgraph mining run. For each run you will find a maximum of 4 files:

* an “xxx-translated.txt” file which contains the motifs which were found to be frequent by the algorithm. This file is called “translated” because we used the GO IDas labels that we used during the subgraph mining analysis, thus the algorithm presented the resulting motifs with these GO IDs. We used a script to translate those GO IDs to their names for interpretability.
* an “xxx\_edge1\_groups.xlsx” file which contains the results after the grouping step which sorted the motifs based on their first edge in the building pattern.
* an “xxx\_edge2\_groups.xlsx” file which contains the results after the grouping step which sorted the motifs based on their first edge in the building pattern.
* an “xxx\_edge2\_top.txt” file which contains the extracted top 10 most frequent motifs based on the counts that can be found in the “xxx\_edge2\_groups.xlsx” file.

Notes:

Run 11 couldn’t be completed due to insufficient memory to run the analysis and therefore there are no results. Consequently, there is no results map present. For run 8 and 9 there is no results map as well, because we did not find any frequent motifs during these analysis runs.

Run 6 only has the translated file as the analysis only produced 2 motifs. Thus, no grouping was needed to summarise the results.

Furthermore, this map contains an excel file containing the table that presents number of results per analysis (corresponding to table 11 p29) as well as an excel file containing all the top 10 most frequent motifs based on their counts after the grouping step based on the first and second edge of the building edge was done. (corresponding to table M-U, p49-56)

Lastly, subgraph-grouping.ipynb is a python notebook which is an interactive python script which eases the writing process of python scripts using a cell-based input-output system (You can enter code in a cell and run it, without running the entire script, essentially allowing you to build and test your script piece by piece). This notebook was used to perform the grouping procedure of the subgraph mining results.

# Python-scripts

*Contains all the scripts that are used during our workflow both directly and indirectly. Directly used scripts are called using the bash command line.*

## Extraction.py

This script is used to filter, clean and merge the raw interaction data that was acquired after querying IntAct, HPIDB2.0 and PHISTO for PPIs of *F. tularensis*, *B. anthracis* and *Y. pestis*. It is also used to add the metadata (GO and IPR terms) to the proteins in the interaction datasets.

## Frequent-motifs.py

This script can be used to extract the top X most frequent motifs from an output file that is created after a subgraph mining run is performed. This script was eventually not used in the workflow of this thesis but can be useful nonetheless and is therefore included in this supplementary material.

## GOparsing.py

This script is used to define functions to parse the GAF files and create dictionaries from them. These dictionaries can then be used to search the GO terms of a protein.

This script was never used directly in our workflow. Instead, the extraction.py and remapping.py scripts use the functions that are defined in this script to add GO terms to the proteins of the interaction dataset. The subgraphprep.py script makes use of these functions as well to create the label files for the subgraph mining. And lastly, the goa-tools.py script also calls the functions of this script to make the association file.

## GOremapping.py

This script is used to define functions that translate the ID of GO terms into the respective name or vice versa as well as functions that eventually lead to the remapping of GO terms to a given specificity depth. This script was also never used directly in the workflow but rather called upon by the Subgraphprep.py and the goa-tools.py scripts.

## IPRparsing.py

This script is used to define functions to parse the IPR datasets and create dictionaries from them. These dictionaries can then be used to search the IPR terms of a protein.

This script was again never used directly in our workflow. Instead, the extraction.py and remapping.py scripts use the functions that are defined in this script to add the IPR terms to the proteins of the interaction dataset.

## Remapping.py

This script is used to tackle the Uniprot ID problem (as described in the master thesis p14) by attempting to remap the UniprotIDs after which the GO and IPR terms of the remapped ID are searched and, if found, were added to the interaction dataset.

## Subgraphprep.py

This script is used to create the files that are needed to perform the subgraph mining analysis, i.e. the network file (containing all the PPIs) and the label file (containing all the protein-label combinations). Depending on some modifying parameters of the bash command that calls this script, the label file is created either using the GO terms as they are present in the interaction datasets or remapping the GO terms to a specified specificity depth.

## association-file-fix.sh

Not a python file but linked to the goa-tools.py script. The resulting association file is formatted incorrectly to be used with the GOA tool. The problem is that every line in the association file has to end with a semicolon, while the goa-tools.py script ends every line with a comma. This is easily fixed using this bash script in the map where the association files are present.

## goa-tools.py

This script is used to create the files need for the GOA analysis, i.e. the study file (containing the set proteins which we are interested in), the population file (containing the set of proteins we consider as background, or in other words, we want to evaluate if certain terms are significantly more or less present in our study set vs this set) and an association file (containing all the proteins of the population and study file and their respective GO terms). This script can create the association file using GO terms as they are present in the datasets or remapped to a certain depth.

## sugraph-output-parser.py

If the label file used in a subgraph mining run contains GO IDs as labels, then the resulting motifs will also have GO IDs as labels. To make the output more interpretable these IDs have to be translated to their respective names. This script can be used to parse through the output, replacing all the IDs with the corresponding names. In addition, it will only translate motifs that are fully labelled. This was done to implement an extra checkpoint that only fully labelled motifs were included in the results. However, this also prevents this script to be used for translation of not fully labelled files. Nevertheless, this can be easily solved with small modifications to this script.

## Files to be ignored

***Python-notebooks****:* contains iPython notebooks which are interactive python scripts used to ease the writing process of the python scripts. These notebooks are the predecessors of the abovementioned final python scripts.

***\_ipynb\_checkpoints***: linked with the python-notebooks (contain saves?)

***.gitignore:*** determines which files of the local copy (on my pc) of this repository can be ignored during synchronization.