Alex Kotz
09/12/23

CPBS 7711 – Module 1, HW 3

1. **Motivating Problem from Domain**
   - There are 12 known genes that are associated with Fanconi anemia. This disease is caused by genomic instability. Creating a functional network of all Fanconi Anemia (FA) genes may assist in new findings related to this disease.
2. **Computational Problem**
   - Given a network of gene-gene relationships, visualize a subnetwork of FA gene-gene relationships with their associated edges.
3. **Specific Approach**
   - The approach taken to address the problem stated above, is as follows: First, create an object that contains all of the known FA genes. Then utilize the FA genes object to filter out rows from the given gene network. When all of the gene-gene objects are stored in an additional object (subnetwork), generate a subnetwork file, based on that subnetwork object, of which can be visualized in Cytoscape.
4. **Specific Implementation**
   - The current implementation generates a sub-network of Fanconi Anemia related genes from the given STRING 1.txt network. The script first reads in the Input.gmt.txt file and isolates the genes into a separate list. The list is used in a separate function in which the STRING 1.txt file is read in and for each row in the STRING 1.txt file, the program checks the first two columns (both genes) of the row to see if they exist in the list generated from Input.gmt.txt; if so, they are added to the results list. After the results list is complied (containing gene-gene mapping with associated edge weight property), the list is written to a separate file. That file is then used in another function that checks if there are any duplicate rows (e.g. row1 = [A,B,3] | row2 = [B,A,3]), and removes the duplicated row from the results. When the final, un-duplicated subnetwork is constructed, a subnetwork file (in STRING format) is generated.

Assumptions:

- The data provided for the larger subnetwork (STRING 1.txt):
    - Does not contain missing data elements.
    - Is programmatically parsable.
    - Is presented in a consistent 3-column string format with no deviations.
- The data provided for the Fanconi Anemia genes (Input.gmt.txt):
    - Accurately represents the most up-to-date dataset for FA related genes.
    - Is programmatically parsable.
    - Does not contain any genes that do not exist in STRING 1.txt
        - I.e. A one-one relationship can be made using both input files
- Output:
    - The output file is in STRING 3-column format
    - The output file does not contain any missing data
    - The output file can be imported into a visualization aid application/software
    - The output data is tab delimited
    - The output data is a subset of the data in the STRING 1.txt input file (does not contain extra data from an external source)

Pseudocode:

Input: a text file containing network of connected genes (STRING 1.txt); a text file containing genes and loci known to be associated with Fanconi Anemia (Input.gmt.txt)
Output: text file containing subnetwork of connected genes associated with Fanconi Anemia

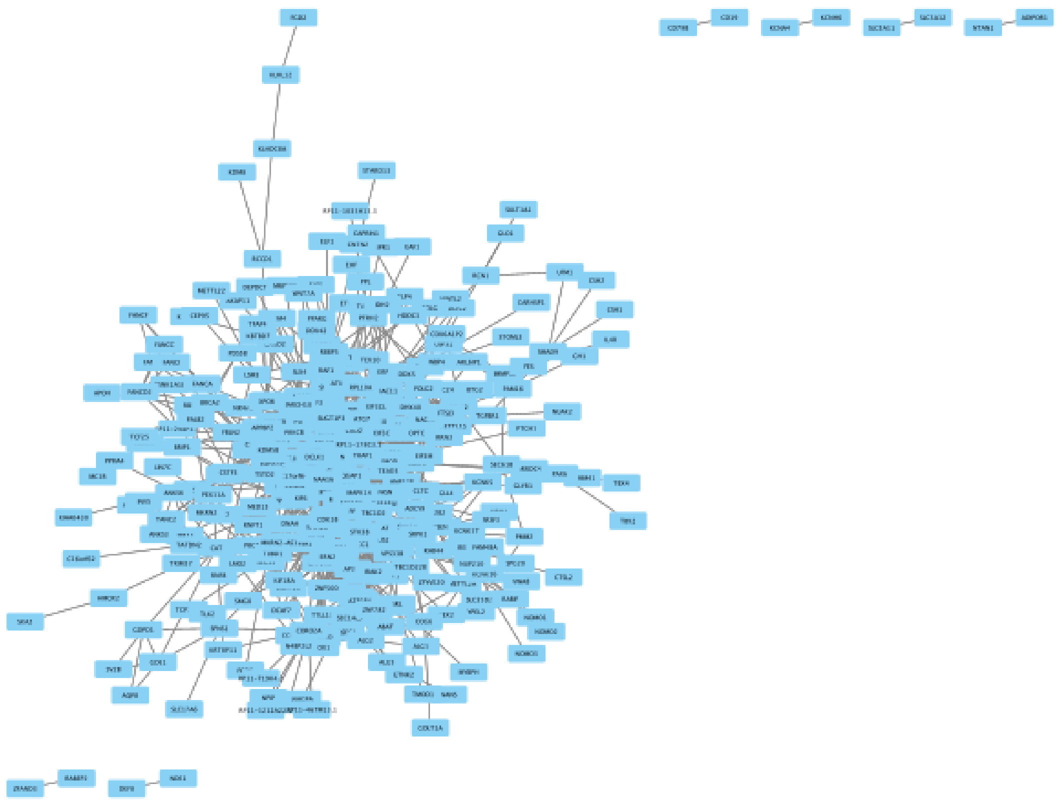
```
Function: fanconi_anemia_genes(inputFile)
    1. empty list → faData
    2. empty list → faGenes
    3. empty list → results
    4. for each row in inputFile:
        - add contents of each row to anemiaData, split each row on '\t'
        - remove newline character from the end of the last element in each
          anemiaData sub-list
        - remove the first two elements of each row (labels for each loci)
    5. write the contents of anemiaData into a single list anemiaGenes

Function: create_subnetwork(inputFile, faGenes):
    1. empty list → results
    2. for each row in inputFile:
        - create a list for each row, split on '\t'
    3. if the first item in row exists in anemiaGenes:
        - if the second item in the row exists in anemiaGenes:
            - add row to results (results become a list of lists)
    4. write contents of results into a new file, adding a tab between each item in each
       sub-list

*Function: check_duplicate(resultsFile):
    1. empty set → results
    2. empty set → seen
    3. empty set → duplicates
    4. for each row in resultsFile:
        - split each row on '\t'
        - add each row to a tuple orderdRow and sort
    5. if orderedRow in seen:
        - add the row to duplicates (dummy set)
    6. else:
        - add the row to seen and results
    7. write contents of results to subnetwork file in STRING 3-Column format
```

* Optional function, included in main.py (uncommented), but can be commented out in main()

Discussion and Results:

Discussion:

After visualizing the finalized subnetwork, three unconnected groups of mapped genes were observed.



This observation prompted a need for added functionality; specifically, a feature to connect the two outlier groups (small groups seen on the top right and bottom left of the image above) to the main group. The approach that was used to attempt to solve this issue is described below:

Using the subnetwork generated from the current state of the python script: first, identify the two outlier groups; second, find non-associated Fanconi Anemia genes that can be used as intermediaries between the separate groups; third, append those gene connections to the subnetwork. An attempt was made to resolve this issue, but with the limited amount of time remaining before the required submission of this project, no resolution was proposed.

Pseudocode for Outlier issue:

```
function: check_outliers(resultsFile)
    1. object containing subnetwork → results
    2. empty object → seen
    3. empty object → outliers
    4. empty object → possibleConnections
    5. empty object → nonFAGenes
    6. for each row in results:
        - if row not in seen:
            - add to seen
        - else:
            - continue
        - * seen now contains unique rows
    7. for row in seen:
        - if row[0] (geneA) & row[1] (geneB) pair only exist once in seen:
            - add row to outliers
            - *indicating the group is an outlier
    8. for row in STRING 1.txt:
        - find all rows that have geneA or geneB, regardless of the connection to column 2
          of the row.
        - add row to possibleConnections
    9. for row in possibleConnections:
        - identify each gene that is not an FA_gene
        - add to nonFAGenes
    10. for row in STRING 1.txt:
        - if (row[0] in nonFAGenes and row[1] in faGenes) or (row[0] in faGenes and
          row[1] in nonFAGenes):
            - add row to connections
    11. append connections to subnetwork file.
```

Results:

The expected results of this program would generate of subnetwork of FA related genes, where each gene is represented as an individual node and each node is directly connected via a weighted edge. There would be one network where all genes are linked together. However, the actual results from this program yield a different subnetwork. This subnetwork presents a visual where each node is a FA related gene and not every gene has a direct edge to another gene. Eventhough the results of the current state of this program do not meet the expected results, adding a visual aid to the set of partially connected, FA related genes could assist in further investigation of the Fanconi Anemia disease.