Cadi Ayyad University
Faculty of Science Semlalia
Computer Science Departement
-Marrakesh-

Final Year Project

BACHELOR'S DEGREE

Branch : SMI

INFLUENCE MAXIMISATION IN COMPLEX NETWORKS USING CENTRALISATION OF NODES AND AHP ALGORITHM

Realised by :

Mohammed Ez-zakkar
Azddine Elmoumny

Supervised By :
Dr.Tarik Agouti , Professor, Head of Computer science department, FSSM
Presented on July 22nd 2020 in FSSM in front of the jury :
Dr. Jihad Zahir, Professor,Computer science department, FSSM

University Year
2019/2020

Abstract

Influence Maximisation has been an area of active research in recent years, due to its large applications in many network-related fields like viral marketing, recommendation systems and also election campaigns. And so many research has addressed this problem which is proven to be NP-hard, which is obvious because of the complexity of networks which include individuals and their interactions and the variety of networks. In This project we suggest an approach to address this problem, this approach combines Complex Network Analysis techniques (CNA) - specifically Centrality measures- and AHP Algorithm (Analytic Hierarchy Process), one of the most common Multi-criteria Decision making algorithm by Saaty. The purpose of this paper is to provide a different method for defining influential individuals (nodes) in a network or key-players with a given budget, k-number of nodes, inside a network and most important is to make it compatible with many types of networks and this is done by giving the agent the ability to tweak the algorithm parameters..

Keywords: Complex Networks Analysis,Influence Maximisation ,Centrality measures,AHP algorithm

Acknowledgement

I would like to express my gratitude and appreciation to all those who gave the possibility to complete this report. A special thanks to our final year project coordinator, Dr Agouti, who gave the golden opportunity to do this wonderful project on the topic "INFLUENCE MAXIMISATION IN COMPLEX NETWORKS USING CENTRALISATION OF NODES AND AHP ALGORITHM", which also help us in doing loads of research and come to know about so many new things related and unrelated to the topic, we are thankful for him. I would like also to acknowledge his help, stimulating suggestions, comments and encouragement and not to forget to thank our kindest Professor Dr. Jihad Zahir for her great support and guidance through the realisation of this project. Finally, a sincere gratitude to all professor and administrator in my university.

List of Figures

Contents

Introduction

In recent years, complex network theory with advances in the understanding of the highly interconnected nature of various social, biological, and communication systems has gained much attention. Transferring information, trust, ideas, diseases and influences between any two nodes is the key function of complex networks. The information can spread rapidly to a large number of nodes begin with an influential node. Hence, evaluating the influence of the nodes is a significant issue in complex networks, such as in the control of the disease and rumour dynamics, research on public opinion, and creating new marketing tools. Many measurements of node centrality have been used commonly such as Degree centrality (DC), Betweenness centrality (BC), Closeness centrality (CC) and so on. The DC method is very simple but of little relevance, since the measure does not consider the global structure of the network. BC and CC are global metrics which can better identify influential nodes, but they are difficult to apply in large-scale networks due to their computational complexity. Another limitation of CC is the lack of applicability to networks with disconnected components: two nodes that belong to different components but do not have a finite distance between them. Several spectral centrality measures are also available, such as semi-local centrality(SLC), eigenvector centrality(EC), PageRank (PR), and LeaderRank (LR). In SLC the topological connections among the neighbours are neglected, only the number of the nearest and the next nearest neighbours of a node is taken into account. EC can not be applied to asymmetric networks in which some positions are unchosen. PR, as well as LR, only affects directed networks, it will degenerate to DC in undirected networks. Multiple Attribute Decision Making (MADM) (or called Multi-Criteria Decision-Making, MCDM) methods have been proposed to decide a preferred alternative, classify alternatives in a small number of categories, and prioritise alternatives in subjective preference order. It has been applied to many fields. Some mathematical tools, such as fuzzy sets, evidence theory and D numbers, are widely used to address MADM. In this paper, we try to explore how to identify influential nodes by employing MADM methods. Du et al. used DC, CC, BC as the multi-attribute in AHP to generate the ranking list to evaluate the node spreading influence. Among numerous MADM methods developed to solve real-world decision problems, the Analytic Hierarchy Process (AHP) continues to work satisfactorily across different application areas. Thomas Saaty originally introduced AHP to deal with complex decision making in a systematic and structured way and may help the decision-maker to set priorities and make the best decision with a finite number of criteria. By reducing complex decisions to a series of pairwise comparisons, and then synthesising the results, the AHP helps to capture both subjective and objective aspects of a decision. Besides, the AHP incorporates a useful technique for checking the consistency of the decision maker's evaluations, thus reducing the bias in the decision making process. AHP makes full use of attribute information, provides a cardinal ranking of alternatives and it's the inability to adequately handle the inherent uncertainty and imprecision in the data is also often criticised. As a well-known classical MADM method, AHP has gained much interest from researchers and practitioners. It is the first time for applying the AHP to identify influential nodes in complex networks. Firstly, we calculate the value of different centrality measures of the network. Notice that in different networks in which the information is transmitted in different ways, the different centrality measures need to be used. Then, AHP is utilised to evaluate the importance of nodes and identify influential nodes by regarding the centrality measures as the multi-attribute of complex networks.

Chapter 1

Complex network analysis CNA

Social systems, the human brain, the Internet and the World Wide Web are all examples of complex networks, i.e. systems composed of a large number of units interconnected.

The study of complex systems is a new science, and so a commonly accepted formal definition of a complex system is still missing. We can roughly say that a complex system is a system made by a large number of single units (individuals, components or agents) interacting in such a way that the behaviour of the system is not a simple combination of the behaviours of the single units. In particular, some collective behaviours emerge without the need for any central control. This kind of behaviour is what we find in human societies at various levels, where the interactions of many individuals give rise to the emergence of civilisation, urban forms, cultures and economies. Analogously, animal societies such as, for instance, ant colonies, accomplish a variety of different tasks, from nest maintenance to the organisation of food search, without the need for any central control.

The standard tools to study complex networks are a mixture of mathematical and computational methods. They require some basic knowledge of graph theory, probability, differential equations, data structures and algorithms.

1.1 Graphs and Graph Theory

Graphs are the mathematical objects used to represent networks, and graph theory is the branch of mathematics that deals with the study of graphs.

1.1.1 what is Graph ?

A graph is a pictorial representation of a set of objects where some pairs of objects are connected by links. The interconnected objects are represented by points termed as vertices, and the links that connect the vertices are called edges. Formally, a graph is a

pair of sets (V, E), where V is the set of vertices and E is the set of edges, connecting the pairs of vertices. Have a look at the following graph in figure 1.1 above
$V = \{a, b, c, d\}$ $E = \{\{a, b\}, \{a, c\}, \{b, c\}, \{c, d\}\}$

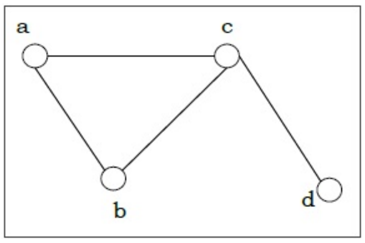


Figure 1.1: Graph

1.1.2 Undirected, Directed and Weighted Graphs

Definition 1 ***(Undirected graph):***

A graph, more specifically an undirected graph,$G = (N, L)$, consists of two sets, $N \neq \emptyset$ and L. The elements of $N = \{n_1, n_2, ..., n_N\}$ are distinct and are called the nodes (or vertices, or points) of the graph G. The elements of $L = \{l_1, l_2, ..., l_K\}$ are distinct unordered pairs of distinct elements of N and are called links (or edges, or lines).

Example 1 *Undirected graph $G = (N, L)$*
$N = \{1, 2, 3, 4\}, L = \{\{0, 1\}, \{0, 3\}, \{0, 4\}, \{1, 2\}, \{1, 3\}, \{2, 3\}, \{3, 4\}\}$

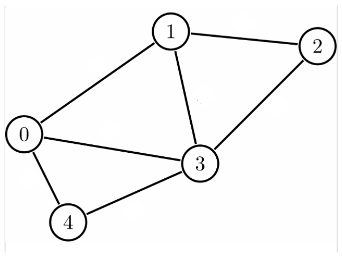


Figure 1.2: Undirected graph

Definition 2 ***(Directed graph):***

A directed graph $G = (N, L)$ consists of two sets, $N \neq \emptyset$ and L. The elements of $N = \{n_1, n_2, ..., n_N\}$ are the nodes of the graph G. The elements of $L = \{l_1, l_2, ..., l_K\}$ are distinct ordered pairs of distinct elements of N and are called directed links, or arcs.

Example 2 *Directed graph $G = (N, L)$*
$N = \{1, 2, 3, 4\}, L = \{(0, 1), (0, 3), (0, 4), (1, 2), (1, 3), (3, 2), (4, 3)\}$

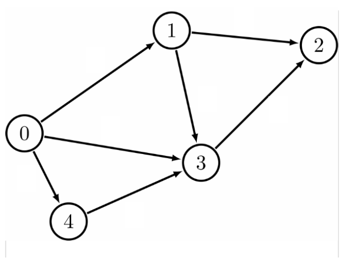


Figure 1.3: Directed graph

Remark 1 *A weighted graph is just a graph in which a numerical value is associated with each link, i.e a graph whose edges have weights.*

Example 3 *The same examples 1,2 With Edges' Weights*

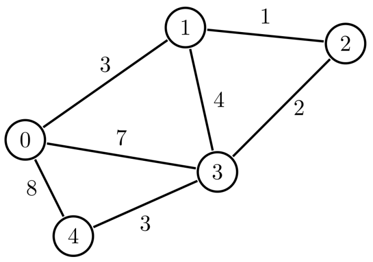


Figure 1.4: Undirected and Weighted Graph

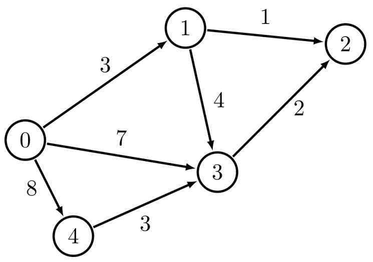


Figure 1.5: Directed and Weighted Graph

1.1.3 Basics Definitions

Definition 3 ***(Node degree):***

The degree k_i of a node i is the number of edges incident in the node. If the graph is directed, the degree of the node has two components: the number of outgoing links k_i^{out} , referred to as the out-degree of the node, and the number of in-going links k_i^{in} , referred to as the in-degree of node i. The total degree of the node is then defined as $k_i = k_i^{out} + k_i^{in}$.

Definition 4 ***(Walks, trails, paths and geodesics)***

A walk $W(x, y)$ from node x to node y is an alternating sequence of nodes and edges (or arcs) $W = (x = n_0, e_1, n_1, e_2, ..., e_l, n_l = y)$ that begins with x and ends with y, such that $e_i = (n_{i-1}, n_i)$ for $i = 1, 2, ..., l$. Usually a walk is indicated by giving only *the sequence of traversed nodes:* $W = (x = n_0, n_1, .., n_l = y)$*. The length of the walk,* $l = l(W)$*, is defined as the number of edges (arcs) in the sequence. A trail is a walk in which no edge (arc) is repeated. A path is a walk in which no node is visited more than once. The shortest path (or geodesic) from node* x *to node* y *is a walk of the minimal length from* x *to* y*, and in the rest of this document it will be denoted as P(x,y).*

Definition 5 ***(Graph distances)***

In an undirected graph the distance between two nodes x *and* y *is equal to the length of a shortest path P(x, y) connecting* x *and* y*. In a directed graph the distance from* x *to* y *is equal to the length of a shortest path P(x,y) from* x *to* y*.*

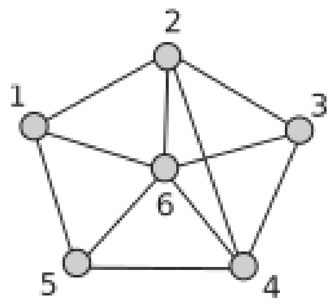


Figure 1.6: Graph (a)

Example 4

Let us consider the graph shown in Figure 1.6 . The sequence of nodes(5, 6, 4, 2, 4, 5) is a walk of length 5 from node 5 back to node 5. This sequence is a walk, but not a trail since the edge (2,4) is traversed twice. An example of a trail on the same graph is instead (5, 6, 4, 5, 1, 2, 4). This is not a path, though, since node 5 is repeated. The sequence (5, 4, 3, 2) is a path of length 3 from node 5 to node 2. However, this is not the shortest path. We can go from node 5 to node 2 in two steps in three different ways: (5, 1, 2), (5, 6, 2), (5, 4, 2), these are the three shortest paths from 5 to 2.

Definition 6 ***(Circuits and cycles)***

A circuit is a closed trail, i.e. a trail whose end vertices coincide. A cycle is a closed walk, of at least three edges (or arcs) $W = (n_0, n_1, .., n_l)$*,* $l \geq 3$*, with* $n_0 = n_l$ *and* n_i *, 0 < i < l, distinct from each other and from* n_0 *. An undirected cycle of length k is usually said a k-cycle and is denoted as* C_k *.* C_3 *is a triangle* ($C_3 = K_3$)*,* C_4 *is called a quadrilater,*C_5 *a pentagon, and so on.*

Example 5

An example of circuit on graph Figure 1.6 is W = (5, 4, 6, 1, 2, 6, 5). This example is not a path on the graph, because some intermediate vertices are repeated. An example of cycle on graph Figure 1.6 is (1, 2, 3, 4, 5, 6, 1). Roughly speaking a cycle is a path whose end vertices coincide.

1.2 Centrality measures

Centrality measures represent one of the most important techniques in CNA, due to the huge amount of information which can be extracted from networks using such a centrality measure. Therefore, we are going through definitions of some popular measures.

Network as we discussed previously, is theoretically a graph. Let G=(V, E) where is n = |V| is the number of Vertices (nodes) and m = |E| the number of edges of the network. Now we are all set to discuss various Centrality measures by giving definitions and mathematical expressions and we let the examples to discuss later in this report.

Definition 1:

Degree centrality is one of the easiest to calculate. The degree centrality of a node is simply its degree, the number of edges it has. The higher the degree is, the more central the node is. The DC of node i, denoted as $C_D(i)$ is defined as follows:

$$C_D(i) = \sum_{n=1}^{N} x_{ij}$$

where i is the focal node, j represents all other nodes,N is the total number of nodes, and x_{ij} represents the connection between node i and node j. The value of x_{ij} is defined as 1 if node i is connected to node j, and 0 otherwise.

Definition 2 :

The Closeness Centrality CC, denoted as $C(i)$, describes the closeness of a node i to other nodes j, the more a node is close to others the high central is. It stands on shortest paths between the node and the rest of nodes in a network.CC differs from a network to a another either it's connected or not.
For connected graphs CC is defined as the reciprocal of the sum the length of the shortest paths between the nodes in the graph. Mathematically speaking, we express CC as follows:

$$C(i) = \frac{1}{\sum_j d(i,j)}$$

where $d(i,j)$ is the distance between vertices i and j. But when speaking about CC practically, we usually refer to its normalised form, which is the average length of the shortest paths instead of their sum. This form is deduced from the previous formula with multiplying by $N-1$, N is the number of vertices inside the graph. For large networks 1 is neglected so we reformulate CC :

$$C(i) = \frac{N}{\sum_j d(i,j)}$$

For disconnected graphs CC follows a different form from the one we discussed above. Instead of the reciprocal of the sum of distances, it uses the sum of reciprocal of distances with the convention of $1/\infty = 0$:

$$C(i) = \sum_{j \neq i} \frac{1}{d(i,j)}$$

These are the main forms of CC, there are many variants of this method which are customised for specific use, but in this paper we are going to stick to those forms.

Definition 3:

Betweenness Centrality BC, as the name suggest, captures how much a node u in between other nodes v . This metric is measured with the number of shortest paths (between any couple of nodes in the graphs) that passes through the target node u, denoted as $\sigma_{v,w}(u)$. the node in question would have a higher BC if it appears in many shortest paths.

$$B_c(u) = \sum_{u \neq v \neq w} \frac{\sigma_{vw}(u)}{\sigma_{vw}}$$

where σ_{vw} is the total number of shortest paths from node v to node w and $\sigma_{vw}(u)$ is the number of those paths that pass through u . Note that $B_c(u)$ represent the probability of u being in between u and w. But what about a weighted network?

In a weighted network the edges connecting the vertices are no longer treated as binary interactions but are weighted in proportion to their capacity, influence frequency, etc, which adds another dimension of heterogeneity within the network beyond the topological effects. A node's strength in a weighted network is given by the sum of the weights of its adjacent edges.

$$s_i = \sum_{j=1}^{N} a_{ij} w_{ij}$$

with a_{ij} and w_{ij} being adjacency and weight matrices between nodes i and j.

This measure essentially highlights which nodes are **bridges** between nodes inside a network or which node is the most involved in transactions between nodes, In terms of our topic, BC finds the individuals who influence the flow of information. Notice that BC can be applied to edges as well.

Definition 4:

Where BC finds the influential nodes, The Eigenvector Centrality EC measures the influence of a node in a network. EC is a generalisation of DC where we assign relative scores to all nodes in the network based on the concept that connections to high-scoring nodes contribute more to the score of the target node than equal connections to low-scoring nodes. Hence, a node with a higher *EC* score is more connected to other many nodes who themselves have high scores.

We define *EC* based on adjacency matrix. Let $G = (V, E)$ with n=|V| vertices and $A = (a_{ij})$ be the adjacency matrix ,i.e $a_{ij} = 1$ if vertex i is linked to j ,and $a_{ij} = 0$ otherwise. The relative centrality score ,x, score of vertex i can be defined as:

$$x_i = \tfrac{1}{\lambda} \textstyle\sum_{t \in M(i)} x_t = \tfrac{1}{\lambda} \textstyle\sum_{t \in G} a_{vt} x_t$$

where $M(i)$ is a set of the neighbours of v and λ is a constant. With small mathematical rearrangement this can be reformulated in vector notation as the eigenvector equation

$$AX = \lambda X$$

1.3 Python Implementation

NetworkX is a Python package for the creation, manipulation, and study of the structure, dynamics, and functions of complex networks.

```python
import networkx as nx

def get_Centrality(filepath, DiGraph = False, Weighted = False):

    # load data from csv file
    data = open(filepath, 'rb')

    # Construct the graph
    if DiGraph:
        G = nx.DiGraph()
    else:
        G = nx.Graph()

    if Weighted:
        G = nx.read_weighted_edgelist(data,delimiter=',',create_using=G
    )
    else:
        G = nx.read_edgelist(data,delimiter=',',create_using=G)


    # computing the Centrality Measures
    DC = nx.degree_centrality(G)
    CC = nx.closeness_centrality(G)
    BC = nx.betweenness_centrality(G)
    if DiGraph:
        G = G.reverse()
    EC = nx.eigenvector_centrality(G,1000)

    return {"DC":DC, "CC":CC, "BC": BC, "EC":EC}
```

Example

source	target
A	B
A	C
D	A
D	E
E	F
E	G

Figure 1.7: File data.csv

```
>>> from get_centrality import get_Centrality
>>> get_Centrality("data.csv")
{'DC': {'A': 0.5, 'B': 0.16, 'C': 0.16, 'D': 0.33, 'E': 0.5, 'F': 0.16,
    'G': 0.16},
'CC': {'A': 0.54, 'B': 0.37, 'C': 0.37, 'D': 0.6, 'E': 0.54, 'F': 0.37,
    'G': 0.37},
'BC': {'A': 0.6, 'B': 0.0, 'C': 0.0, 'D': 0.6, 'E': 0.6, 'F': 0.0, 'G':
    0.0},
'EC': {'A': 0.49, 'B': 0.25, 'C': 0.25, 'D': 0.49, 'E': 0.49, 'F':
   0.25, 'G': 0.25}}
```

Chapter 2

Analytic hierarchy process AHP

2.1 Introduction

The analytic hierarchy process AHP is a widely used $MCDA$ methodology proposed by *Saaty*. AHP is based on relative measurement theory, where we are not interested in the exact measurement of the alternatives performances over the criteria. AHP uses pairwise comparisons among the different alternatives to produce a relative rating of the alternatives. Sometimes when we are making a decision our aim is the find best choice, here AHP is the best suited for this case.

In group decision making AHP has a particular application and is used widely in a variety of decision situations, in fields such as government, business. AHP has been used successfully in many institutions and companies. Although the method is so universal, it is still simple enough to execute in Excel. One of AHP's great advantages is the ability to use it for group decisions, in which all participants evaluate pairs and the group result is determined as the mathematically optimum consensus. In practice the solutions arrived by the method are well accepted, since the results are objective and free of political influence. industry, healthcare, shipbuilding and education.

2.2 Methodology

Let us assume that the decision making problem consists of m alternatives and n criteria. AHP does not use exact evaluations to form a decision matrix as in the other $MCDM$ methods. Psychologists argue that it is easier and more accurate to use scales in order to evaluate an alternative over another. It is also easier for a decision maker to express an opinion on only two alternatives rather than on all the alternatives simultaneously. Therefore, AHP uses a ratio scale that does not require any units in the comparison. A judgement is a relative value of two quantities having the same units. In AHP, the decision maker does not provide a numerical judgement, but a relative verbal comparison of one alternative over another. Hence the criteria weights and the decision matrix were calculated by the decision maker. Instead of evaluating all alternatives with a numerical judgement over all criteria, matrices with pairwise comparisons are formed in AHP.

$$X = \begin{bmatrix} 1 & x_{12} & \cdot\cdot & x_{1k} & x_{1n} \\ \frac{1}{x_{12}} & 1 & \cdot\cdot & x_{2k} & x_{2n} \\ \frac{1}{x_{1k}} & \cdot\cdot & \frac{1}{x_{k(k-1)}} & 1 & x_{kn} \\ \frac{1}{x_{1n}} & \cdot\cdot & \cdot\cdot & \frac{1}{x_{(n-1)n}} & 1 \end{bmatrix}$$

Matrix X is a positive reciprocal square matrix, where x_{ij} is the comparison between element i and j . Naturally, $x_{ij} = \frac{1}{x_{ji}}$ and $x_{ji} = \frac{1}{x_{ij}}$. For instance, if site 1 has three times more employment needs than site 2, i.e., $x_{12} = 3$, then site 2 has three times less employment needs than site 1, i.e., $x_{21} = \frac{1}{3}$. In addition, $x_{ii} = 1$ since each alternative is equally important to itself. If the matrix is perfectly consistent, then the transitivity rule holds for all comparisons. We can summarise AHP procedure in the next diagram:

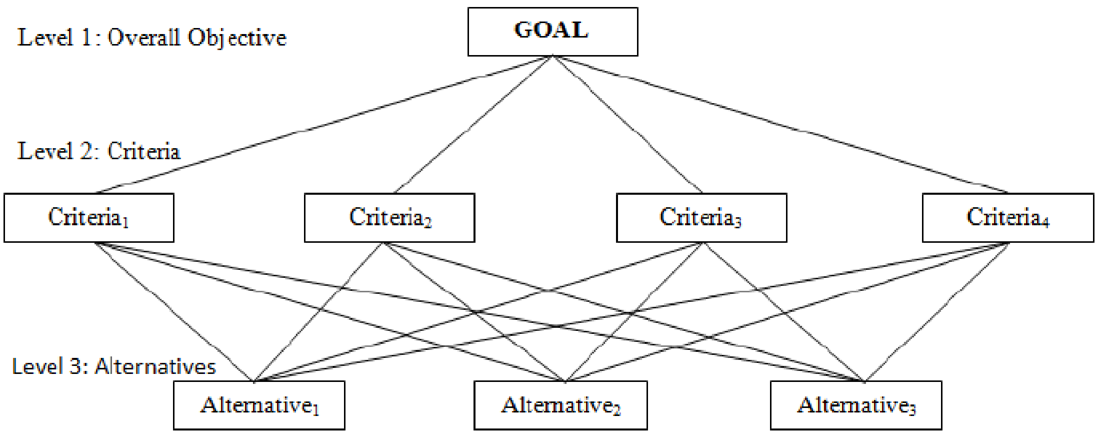


Figure 2.1: AHP Model

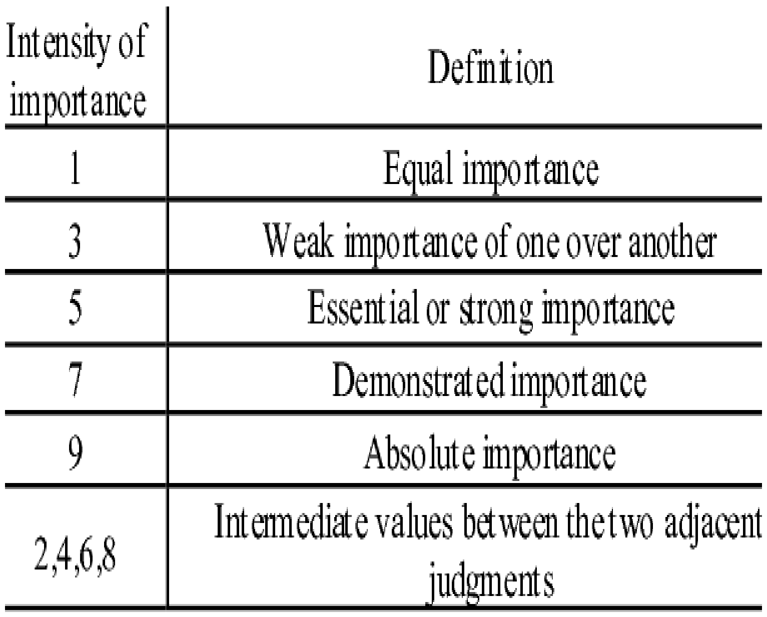
Now we go over AHP procedure by walking through these steps:

Step 1 ***Build the hierarchy for the decision***

Goal	*Select the best alternative*
Criteria	$c_1, c_2, c_3, \ldots, c_m$
Alternatives:	$a_1, a_2, a_3, \ldots, a_n$

Step 2 ***Judgements and Comparison***

Build a numerical representation using a 9-point scale in a pairwise comparison for the attributes criterion and the alternatives. The goal, in AHP, is to obtain a set of eigenvectors of the system that measures the importance with respect to the criterion. We can put these values into a matrix or table based on the values from Saaty's 9-point scale :



Intensity of importance	Definition
1	Equal importance
3	Weak importance of one over another
5	Essential or strong importance
7	Demonstrated importance
9	Absolute importance
2,4,6,8	Intermediate values between the two adjacent judgments

We must ensure that this pairwise matrix is consistent according to Saaty's scheme to compute the Consistency Ratio CR. The value of CR must be less than or equal to 0.1 to be considered valid.

Next, we approximate the largest eigenvalue, λ, using the power method (see Burden and Faires). We compute the consistency index, CI, using the formula:

$$CI = \frac{(\lambda - n)}{(n-1)}$$

Then, we compute the CR using:

$$CR = \frac{CI}{RI}$$

If $CR \leq 0.1$, then our pairwise comparison matrix is consistent and we can continue the AHP process. If not, we must go back to our pairwise comparison and fix the inconsistencies until the $CR \leq 0.1$. In general, the consistency ensures that if $A > B$, $B > C$, that $A > C$ for all A, B, and C all of which can be criteria.

Step 3 *Finding all the eigenvectors combined in order to obtain a comparative ranking. Various methods are available to achieve that.*

2.2.1 Method to solve for Decision-Maker Weights

We suggest the method from Burden and Faires using the power method as it is straightforward to implement using technology.

Definition of a dominant eigenvalue and dominant eigenvector: Let $\lambda_1, \lambda_2, ..., \lambda_n$ be the eigenvalues of a $n \times n$ matrix A,λ_1 is called the dominant eigenvalue of A if $|\lambda_1|$ $> |\lambda_i|$, for $i = 2, ..., n$. The eigenvectors corresponding to λ_1 are called the dominant eigenvectors of A. The power method to find these eigenvectors is iterative. First, assume that the matrix A has a dominant eigenvalue with corresponding dominant eigenvectors, then choose an initial nonzero vector in R^n as the approximation, x_0 , of one of the dominant eigenvectors of A. Finally,form the iterative sequence.

$$\begin{gathered} x_1 = A_{x_0} \\ x_2 = A_{x_1} = A^2_{x_0} \\ x_3 = A_{x_2} = A^3_{x_0} \\ . \\ . \\ . \\ x_k = A_{x_{k-1}} = A^k_{x_0} \end{gathered}$$

Step 4 *After the $m \times 1$ criterion weights are found and the $n \times m$ matrix for n alternatives by m criterion, we use matrix multiplication to obtain the $n \times 1$ final rankings.*

Step 5 *We order the final ranking.*

2.3 Python Script

```
import numpy as np

RI =[0,0,0.58,0.9]

def Make_pairewiseMatrix(d):

    def i(x,rec=1):
        x = float(d[x])
        if x < 1:
            x = np.reciprocal(-x+2)
        if rec:
            return np.reciprocal(x)
        return x
    matrix = np.array([[1,i('1',0),i('2',0),i('3',0)],
                      [i('1'),1,i('4',0),i('5',0)],
                      [i('2'),i('4'),1,i('6',0)],
                      [i('3'),i('5'),i('6'),1]])

    return matrix

def Consistency_Ratio(data,N=4):

    Weighted = np.array([])
    Pairwise_Matrix = np.matrix(Make_pairewiseMatrix(data))

    eigvals , eigvects = np.linalg.eig(Pairwise_Matrix)
    lamb = eigvals.max()
    index = np.where(eigvals == lamb)
    for el in eigvects:
        Weighted = np.append(Weighted,el[0,index[0]])

    CI=(lamb-N)/(N-1)

    CR=CI/RI[N-1]

    return [lamb, Weighted, CR]
```

Example

```
>>> from CR import Consistency_Ratio
>>> PM={'1':1,'2':1,'3':1,'4':1,'5':1,'6':1}
>>> Consistency_Ratio(PM)
[3.9999999999999996, array([0.5, 0.5, 0.5, 0.5]),-1.644774851296528e
   -16]
>>> PM={'1':1/2,'2':1/3,'3':1/3,'4':1/2,'5':1/2,'6':1}
>>> Consistency_Ratio(PM)
[4.011,array([0.331,0.428,0.594,0.594]),0.004]
>>> PM={'1':2,'2':3,'3':1,'4':1/2,'5':2,'6':3}
>>> Consistency_Ratio(PM)
[4.497,array([-0.723,-0.373,-0.492,-0.307]), 0.184]
```

Chapter 3

Address IM problem with CNA+AHP

Networks were always complex so they require special addressing. AHP is known as a good algorithm for MCDM, but it requires defining the criteria for the decision you are willing to make, thus the diversity of network types and dimensions make it hard to define a set of criteria to generalise all networks types. Therefore comes the essential role of centrality measures which act like a multi-criteria for the AHP algorithm.
$CNA - AHP$ essentially combine the network analysis techniques and AHP and get the best out of both worlds to ensure this approach become flexible and customisable and can handle different types of networks, This method uses CM as a multi-criteria to AHP and gives the user the ability to prioritise any CM he wants and set the scale range based on his preferences. The following diagram displays the method proposed:

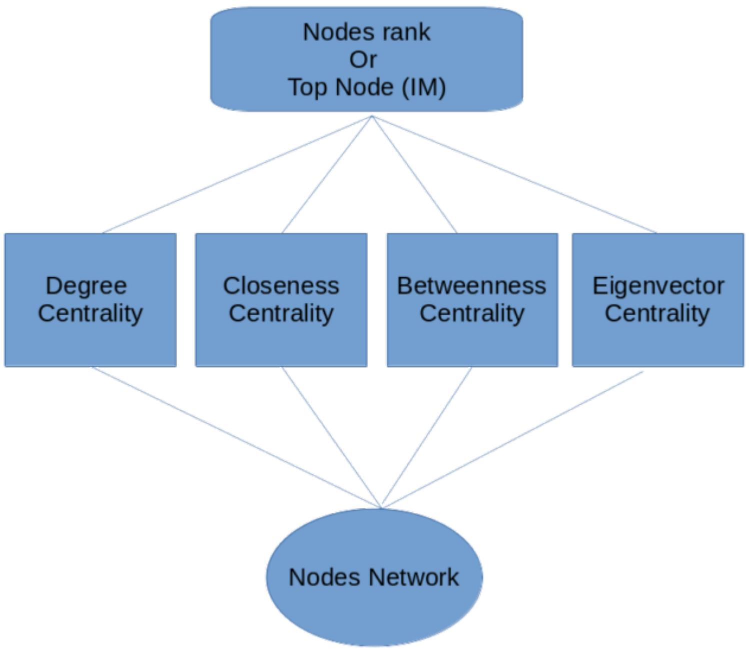


Figure 3.1: AHP Model with Centarlity Measures as a Multi-Criteria

From the diagram above we can say that this method an AHP-based with the use of CM as a multi-criteria, this simple tweaking can highly improve the flexibility of the method when working with different types of networks, because of the pairwise

matrix which allows us to change the priority of each CM depends on the network we have and the outcome we expect. The consistency ratio will ensure there's no bias during the process.
With a CSV file containing the network data the web application implemented using the method discussed in this paper, will provide the customisability to the agent to set the preferences for his decision, The application will process the data and identify the influential nodes (individuals) within the network given and shows the results to the agent. The full process of the application can be summarised in those figures:

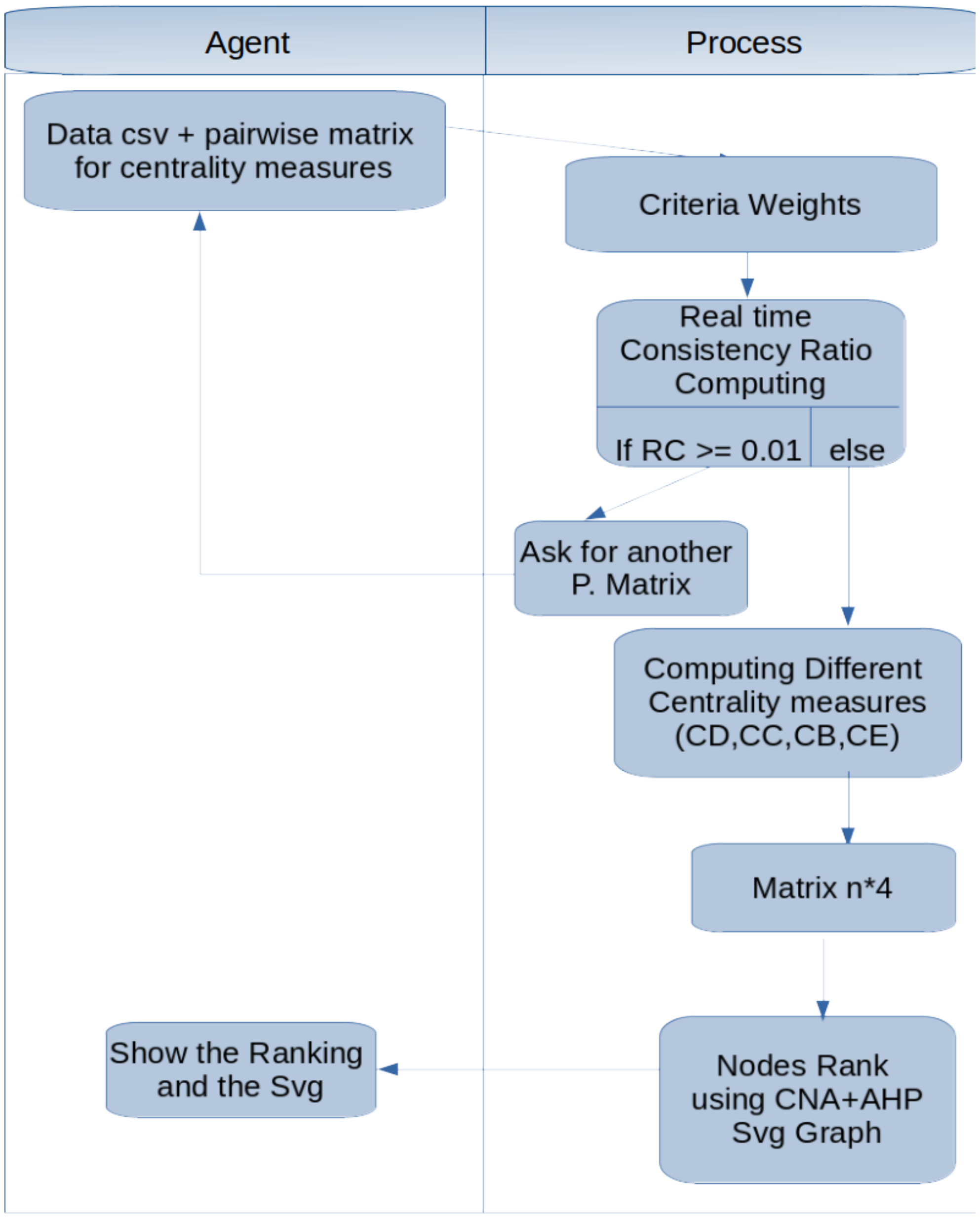
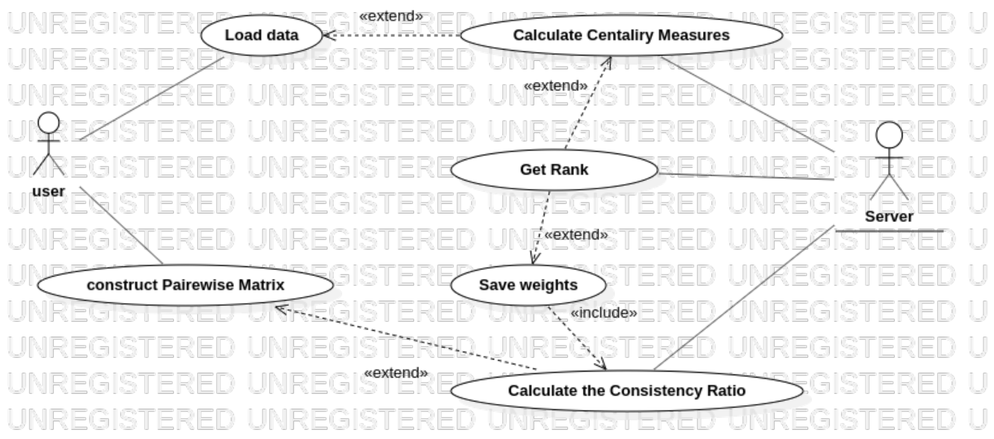

Agent
Process
Data csv + pairwise matrix for centrality measures
Criteria Weights
Real time Consistency Ratio Computing
If RC >= 0.01
else
Ask for another P. Matrix
Computing Different Centrality measures (CD,CC,CB,CE)
Matrix n*4
Nodes Rank using CNA+AHP Svg Graph
Show the Ranking and the Svg

Figure 3.2: CNA-AHP approach diagram



Load data
«extend»
Calculate Centaliry Measures
«extend»
user
Get Rank
Server
«extend»
construct Pairewise Matrix
Save weights
«include»
«extend»
Calculate the Consistency Ratio

Figure 3.3: Use Case Diagram

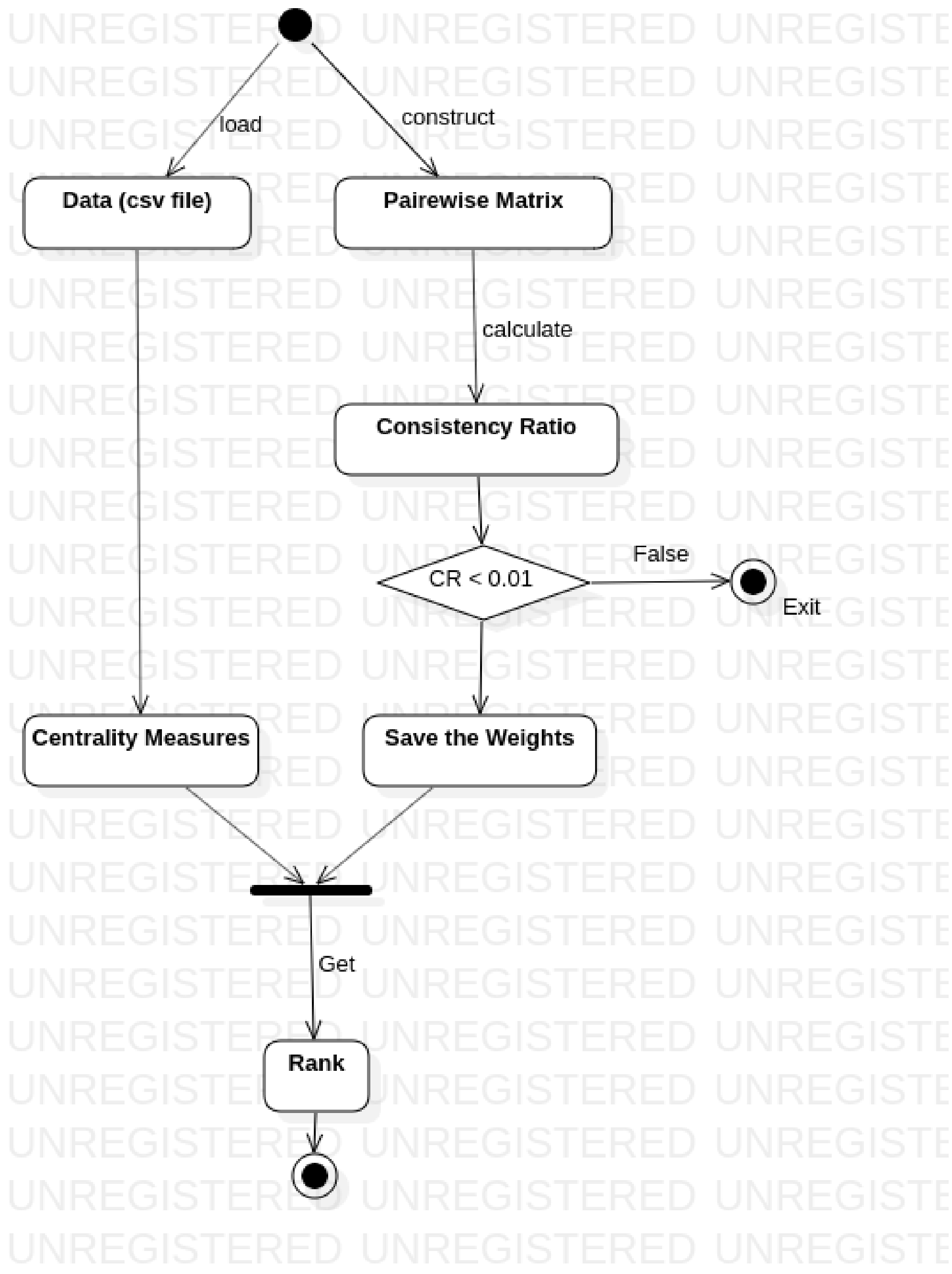


Figure 3.4: Activity Diagram

3.1 Python Script

```
from get_centrality import get_Centrality
from CR import Consistency_Ratio

def get_Ranking(csvpath,PM, DiGraph = False, Weighted = False):

    Rank = []

    CM   = get_Centrality(csvpath,DiGraph,Weighted)
    lamb,w,cr = Consistency_Ratio(PM)
    if cr > 0.01:
        return "Inconsistent System"

    for el in CM["DC"]:
        score=w[0]*CM["DC"][el]+w[1]*CM["CC"][el]+w[2]*CM["BC"][el]+w
    [3]*CM["EC"][el]
        Rank.append({"name":el,"score":score,"DC":CM["DC"][el],"CC":CM[
    "CC"][el],"BC":CM["BC"][el],"EC":CM["EC"][el]})

    Rank=sorted(Rank, key=lambda node: node.get("score"), reverse=True)
    return Rank
```

Example

```
>>> from get_ranking import get_Ranking
>>> get_Ranking("data.csv",{'1':2,'2':3,'3':1,'4':1/3,'5':1,'6':1})
'Inconsistent  System'
>>> get_Ranking("data.csv",{'1':1,'2':1,'3':1,'4':1,'5':1,'6':1})
[{'name': 'b', 'score': 1.34, 'DC': 0.66, 'CC': 0.75, 'BC': 0.66, 'EC':
    0.60}, {'name': 'c', 'score': 1.34, 'DC': 0.66, 'CC': 0.75, 'BC':
   0.66, 'EC': 0.60}, {'name': 'a', 'score': 0.60, 'DC': 0.33, 'CC':
   0.5, 'BC': 0.0, 'EC': 0.37}, {'name': 'd', 'score': 0.60, 'DC':
   0.33, 'CC': 0.5, 'BC': 0.0, 'EC': 0.37}]
```

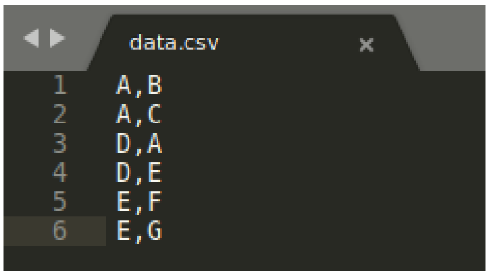
Chapter 4

Web-based implementation of the CNA-AHP Method

In this chapter we move to understand how the implementation of the method suggested in this paper works, it just takes three steps from the agent to get the ranking of nodes:

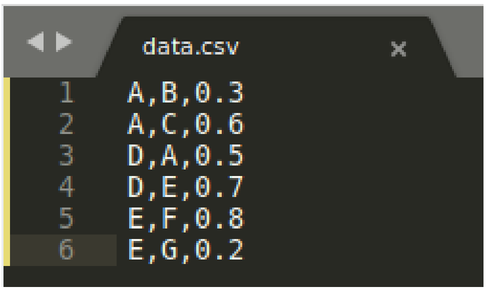
Step 1: Load data and construct the pairwise Matrix (PM):

The network's data should contain the source (node), the target (node) and weight of links, we don't need to put the weight in the data file (CSV file) if all the links in the network have the same weight which equals to 1, for that reason an option to tick if the graph is weighted (ticked if the CSV file contain Weights), and another option for Directed Graph (ticked if a Directed Graph). This next CSV file describes how the data set should be formatted :



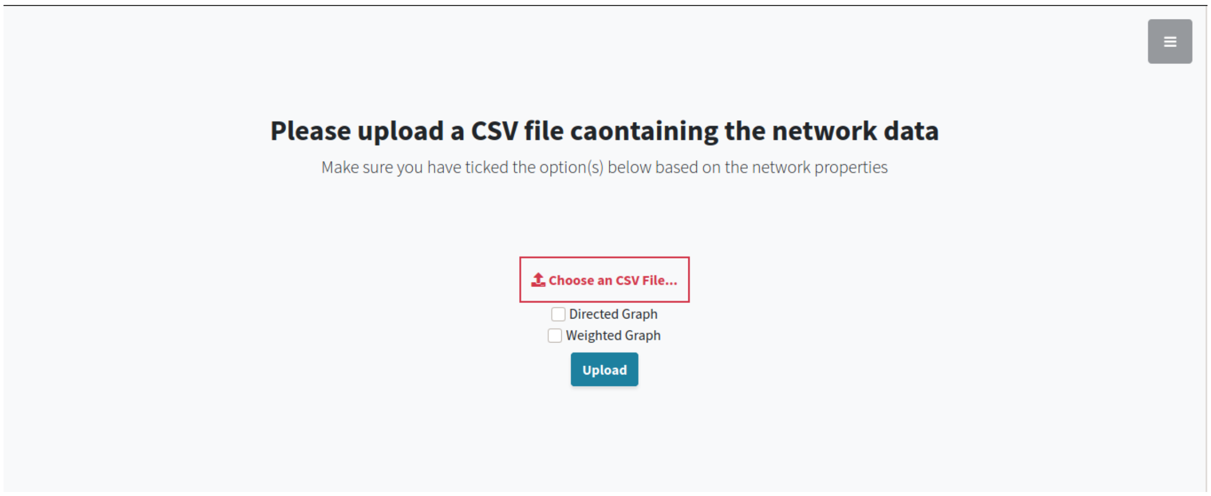
```
A,B
A,C
D,A
D,E
E,F
E,G
```

Figure 4.1: CSV file (Unweighted Graph)



```
data.csv
A,B,0.3
A,C,0.6
D,A,0.5
D,E,0.7
E,F,0.8
E,G,0.2
```

Figure 4.2: CSV file (Weighted Graph)

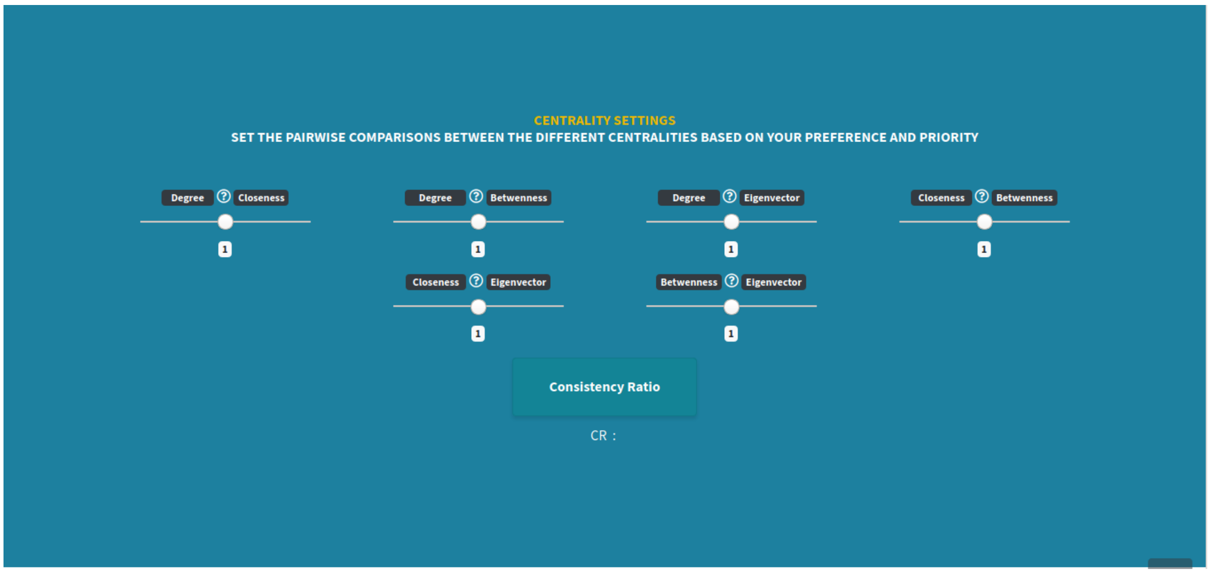


Please upload a CSV file caontaining the network data

Make sure you have ticked the option(s) below based on the network properties

Figure 4.3: Upload File Capture

To construct the Pairwise Matrix for the AHP method to get the Consistency ratio and the weights we need just to establish six comparisons between the four centrality measures (Degree, Betweenness, Closeness and Eigenvector), we provide a section within the application containing six sliders to help you customise your PM :



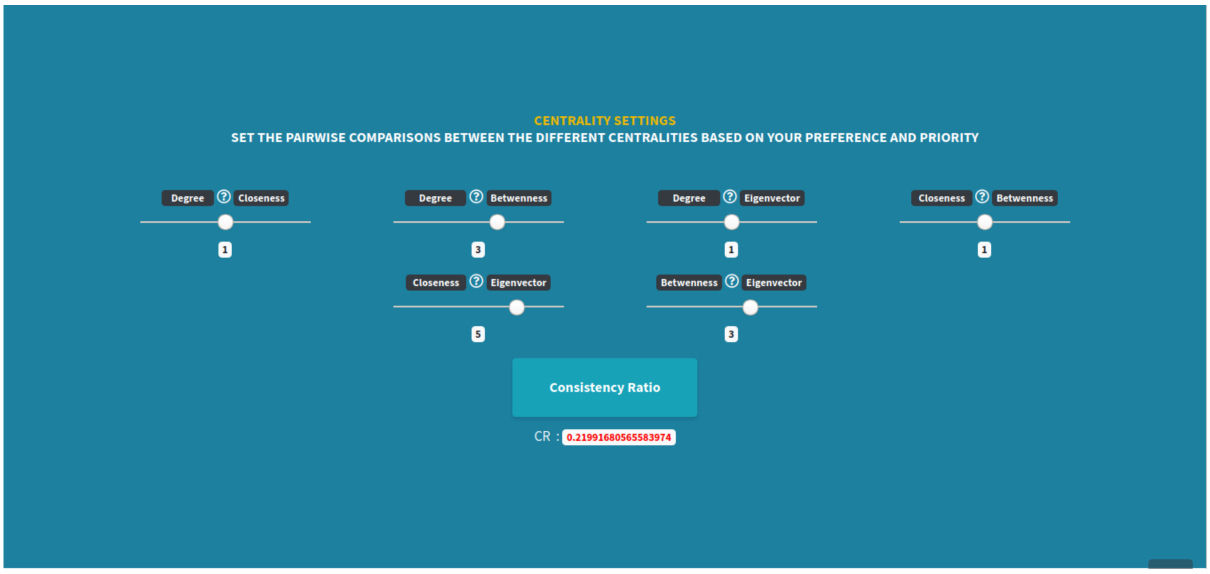
CENTRALITY SETTINGS

SET THE PAIRWISE COMPARISONS BETWEEN THE DIFFERENT CENTRALITIES BASED ON YOUR PREFERENCE AND PRIORITY

Degree ? Closeness: 1
Degree ? Betweenness: 1
Degree ? Eigenvector: 1
Closeness ? Betweenness: 1
Closeness ? Eigenvector: 1
Betweenness ? Eigenvector: 1

Figure 4.4: Pairwise Matrix Sliders Capture

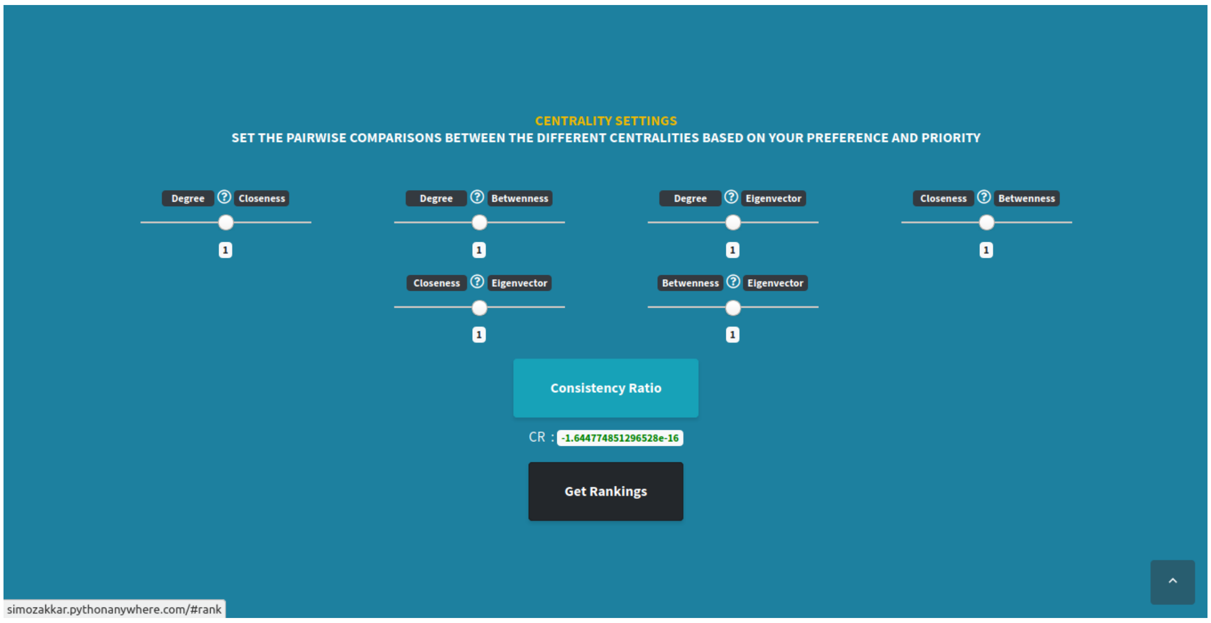
Notice: Make sure your PM is consistent (the value of CR < 0.01),i.e if we have A>B and B>C then should be A>C


CENTRALITY SETTINGS
SET THE PAIRWISE COMPARISONS BETWEEN THE DIFFERENT CENTRALITIES BASED ON YOUR PREFERENCE AND PRIORITY

Degree ⓘ Closeness 1
Degree ⓘ Betwenness 3
Degree ⓘ Eigenvector 1
Closeness ⓘ Betwenness 1
Closeness ⓘ Eigenvector 5
Betwenness ⓘ Eigenvector 3

CR : 0.21991680565583974

Figure 4.5: Inconsistent System


CENTRALITY SETTINGS
SET THE PAIRWISE COMPARISONS BETWEEN THE DIFFERENT CENTRALITIES BASED ON YOUR PREFERENCE AND PRIORITY

Degree ⓘ Closeness 1
Degree ⓘ Betwenness 1
Degree ⓘ Eigenvector 1
Closeness ⓘ Betwenness 1
Closeness ⓘ Eigenvector 1
Betwenness ⓘ Eigenvector 1

CR : -1.644774851296528e-16

simozakkar.pythonanywhere.com/#rank

Figure 4.6: Consistent System

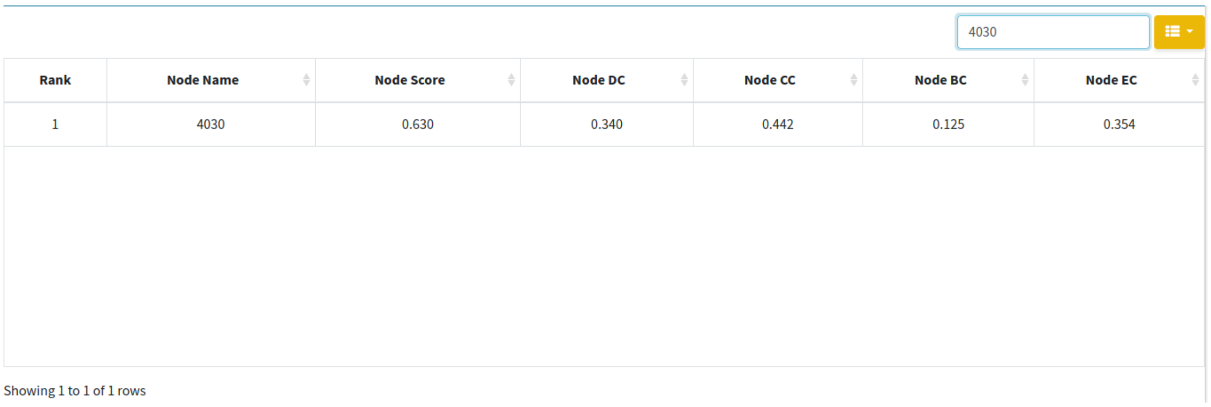
Step 2: The CSV file and PM in the Server:

When the agent finishes the step 1 and asks for the ranking of nodes, the application run the python scripts in flask (a web framework written in Python) server and send the result to the browser agent .

Step 3: Visualising the ranking of nodes:

The ranking of nodes are visualised in the application in three forms :

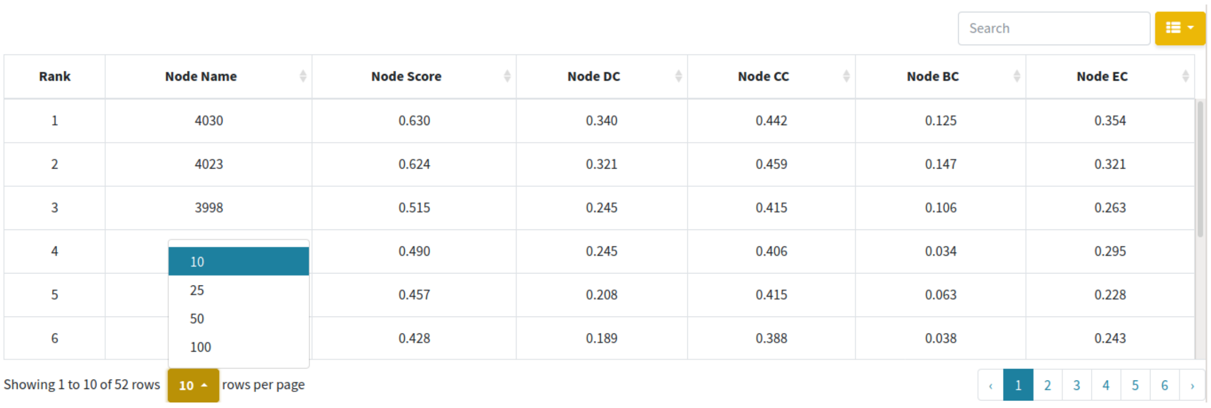
Table: A table containing the Rank, Name, Score, Degree, Betweenness, Closeness and Eigenvector of nodes, with amazing options, the first one is the search, you can search by any item in the table (by name, by score ,...),The second is the pagination, in other words, the number of rows you want to see in the table, The next option is the column view .



Rank	Node Name	Node Score	Node DC	Node CC	Node BC	Node EC
1	4030	0.630	0.340	0.442	0.125	0.354

Showing 1 to 1 of 1 rows

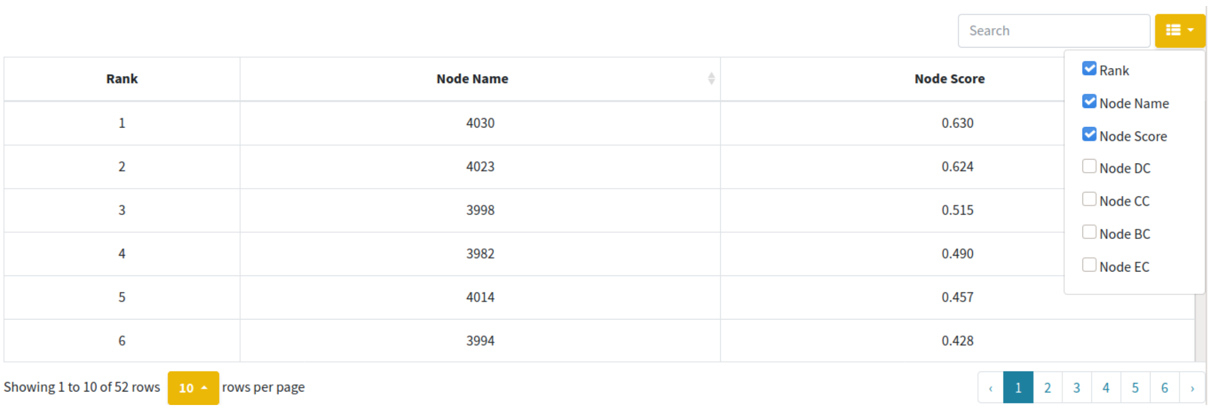
Figure 4.7: Table Search Option



Rank	Node Name	Node Score	Node DC	Node CC	Node BC	Node EC
1	4030	0.630	0.340	0.442	0.125	0.354
2	4023	0.624	0.321	0.459	0.147	0.321
3	3998	0.515	0.245	0.415	0.106	0.263
4		0.490	0.245	0.406	0.034	0.295
5		0.457	0.208	0.415	0.063	0.228
6		0.428	0.189	0.388	0.038	0.243

Showing 1 to 10 of 52 rows 10 rows per page

Figure 4.8: Table Pagination Option



Rank	Node Name	Node Score
1	4030	0.630
2	4023	0.624
3	3998	0.515
4	3982	0.490
5	4014	0.457
6	3994	0.428

✓ Rank
✓ Node Name
✓ Node Score
☐ Node DC
☐ Node CC
☐ Node BC
☐ Node EC

Showing 1 to 10 of 52 rows 10 rows per page

Figure 4.9: Table Column View Option

Chart Bar: Chartjs is a Javascript framework (official website) which increases the data readablity:

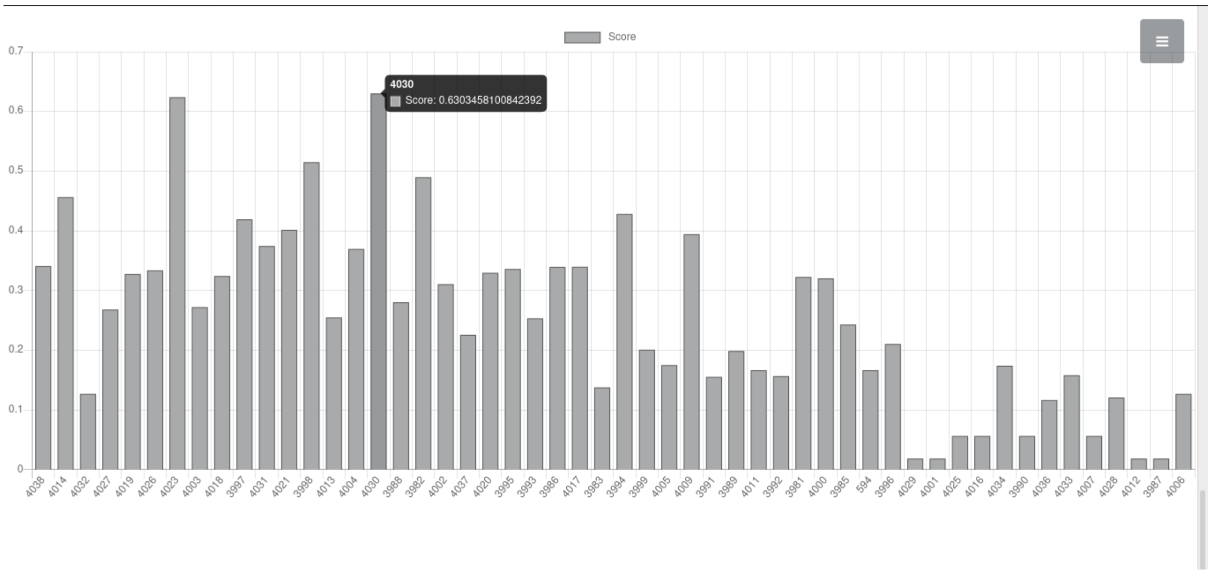


Figure 4.10: Chart Bar Capture

Graph : D3js is a Javascript framework (official website) used to draw graph representing the network in question. The radius and the colour of each node depends on its score which highlights the key-nodes in the networks.

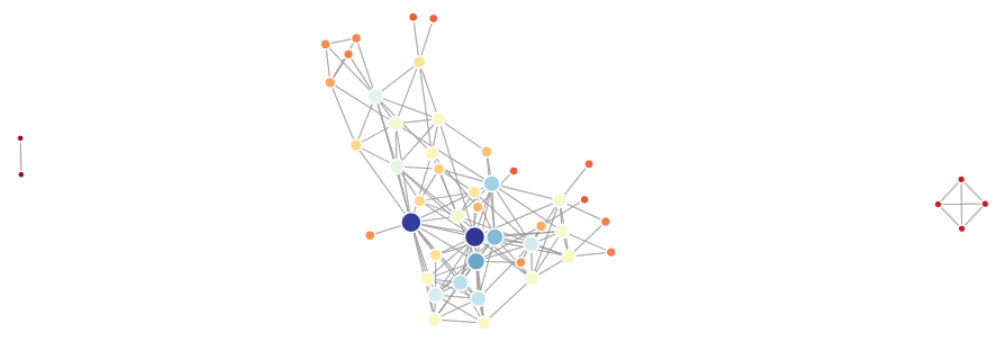


Figure 4.11: D3js Graph Capture

Conclusion

In this project, a different method is proposed to identify the influential nodes in a complex network based on the AHP. In our method, we consider several different centrality measures as the multi-attribute of a complex network in AHP application and give the corresponding weights to each attribute according to the matching degree. AHP is used to aggregate the multi-attribute to evaluate the importance of each node, which can comprehensively consider different centrality measures and prioritise them based on the user preferences and controlling the bias. The implementation is of our method is a Flask web-based application built using python as server-side programming language and the popular front-end web development languages (HTML5, CSS3,JS).

To try our application you are welcome see https://improblemandahp2020.herokuapp.com/
A github repository is available to have a look on the code source: https://github.com/Aelmouny11/IMproblem-using-AHP-and-centralisation-of-nodes

References

Saaty-1987-The analytic hierarchy process-what it is and how it used
Complex Network Analysis in Python: Recognize - Construct - Visualize - Analyze - Interpret · 2018 · Author : Dmitry Zinoviev

Multiple Criteria Decision Aid Methods, Examples and Python Implementations · 2018 · Authors: Papathanasiou, Jason, Ploskas, Nikolaos

Multiple Criteria Decision Making and Aiding Cases on Models and Methods with Computer Implementations -2019- Authors: Huber, Sandra, Geiger, Martin Josef, de Almeida, Adiel Teixeira (Eds.)
https://en.wikipedia.org/wiki/Complex_network
https://www.researchgate.net/publication/271453981_Key_nodes_evaluation_with_multi-criteria_in_complex_networks_based_on_AHP_analysis
https://www.researchgate.net/publication/314132299_Identifying_influential_nodes_in_complex_networks_based_on_AHP
Power method(Burden and Faires): https://fac.ksu.edu.sa/sites/default/files/numerical_analysis_9th.pdf
http://dx.doi.org/10.1016/j.physa.2017.02.085
https://www.stat.uchicago.edu/~lekheng/meetings/mathofranking/slides/saaty.pdf
https://online-journals.org/index.php/i-jet/article/view/10779
https://www.coursera.org/learn/social-network-analysis