Multi-Criteria Decision Making with Consistency-Driven Pairwise Comparisons Method

by

Yingli Song

A thesis submitted in partial fulfilment
of the requirements for the degree of
Master of Science (MSc) in Computational Sciences

The Faculty of Graduate Studies
Laurentian University
Sudbury, Ontario, Canada

THESIS DEFENCE COMMITTEE/COMITÉ DE SOUTENANCE DE THÈSE
Laurentian Université/Université Laurentienne
Faculty of Graduate Studies/Faculté des études supérieures

Title of Thesis Titre de la thèse	Multi-Criteria Decision Making with Consistency-Driven Pairwise Comparisons Method	
Name of Candidate Nom du candidat	Song, Yingli (Lillian)	
Degree Diplôme	Master of Science	
Department/Program Département/Programme	Computational Sciences	Date of Defence Date de la soutenance April 16, 2021

APPROVED/APPROUVÉ

Thesis Examiners/Examinateurs de thèse:

Dr. Waldemar W. Koczkodaj
(Supervisor/Directeur(trice) de thèse)

Dr. Miroslaw Mazurek
(Committee member/Membre du comité)

Approved for the Faculty of Graduate Studies
Approuvé pour la Faculté des études supérieures
Tammy Eger, PhD
Vice-President, Research
Vice-Rectrice à la recherche

Dr. Aleksandra Kawala-Sterniuk
(External Examiner/Examinateur externe)

ACCESSIBILITY CLAUSE AND PERMISSION TO USE

Abstract

Multi-criteria decision-making is a sub-discipline of operations research. It explicitly evaluates multiple conflicting criteria in decision making. The consistency-driven pairwise comparisons method is a valuable tool for solving multi-criteria decision-making problems. It is also a powerful technique for the inference that could be used for knowledge acquisition for the knowledge management system. In fact, pairwise comparisons are basics for practically all science disciplines and have been used in many critical national projects. This thesis describes an application's implementation based on the consistency-driven pairwise comparisons method using an R package, Shiny. Shiny offers an elegant and powerful web framework for building interactive web applications straight from R. This application, RConcluder, is a standalone system and could be used as a supplement to any expert or knowledge management system.

Keywords

Multi-criteria decision making, Consistency-driven pairwise comparisons, Inconsistency analysis, R, Shiny

Acknowledgement

First and foremost, I would like to express my sincere gratitude to my supervisor, Prof. Waldemar W. Koczkodaj. He offered me so much help for my study and my family during my study at Laurentian University. For his professional advice, heart-warming encouragement and patient guidance, I will be forever grateful. It is a real honour and delight for me to meet him in this beautiful university.

I also would like to extend my sincere thanks to the review committee member, Dr.Miroslaw Mazurek and my external examiner, Dr.Aleksandra Kawala-Sterniuk for their time and effort in viewing and advising my thesis.

My thanks also go to my husband Chenguang Xu for his encouragement and support, without which this two years study would not be so smooth.

Finally, I would like to dedicate this thesis to my children, Xinyang Xu and Xinyu Xu.

Yingli Song

2021

Contents

List of Figures

List of Tables

1 Introduction

Multi-criteria decision-making (MCDM) is concerned with structuring and solving many complex problems, especially those that involved choosing alternatives. Pairwise comparisons(PC) is widely recognized as a practical and effective method for supporting the MCDM process based on subjective judgments. When the relative importance of these items can not be rated directly, the benefit of using it for building a global ranking from binary comparisons becomes apparent. One of the primary techniques for solving MCDM problems is the consistency-driven pairwise comparisons(CDPC) method. It provides an elegant means by comparing, ranking, quantizing, and offering strategies to identify and resolve inconsistencies. It is based on an inconsistency index, Koczkodaj Inconsistency Indicator, proposed and named after prof. W.W. Koczkodaj (in 1993) and is used as a technique for validation.

The hierarchical pairwise comparisons method is based on a multi-level pairwise comparisons structure. In [32], "The Analytic Hierarchy Process first decomposes the decision problem into a hierarchy of subproblems". It reduces complexity (in terms of many pairwise comparisons) from $O(n^2)$ to close to $n * log(n)$. This is a significant step forward. A hierarchical tree is the primary visualization form in this thesis.

2 Pairwise Comparisons basics

2.1 Background

The history of pairwise comparisons(PC) can be traced back to nearly 740 years ago. Ramon Lull, a Majorcan mathematician, philosopher, and logician, used PC for electoral systems in the 13th century in [29]. Later on, Marie Jean Antoine Nicolas de Caritat rediscovered it. Nicolas de Condorcet [16], a French philosopher and mathematician, had used it in its primitive form: win/ loss. Thomas L. Saaty [15] invented an Analytic Hierarchy Process method, a systematic, hierarchical analysis method that combines qualitative and quantitative analysis based on modified pairwise comparisons in 1977. Recently, there has been renewed interest in this method.

According to [17], "There are two types of pairwise comparisons that are commonly considered: additive and multiplicative." From the mathematic perspective, the multiplicative PC is more often used than additive PC. In [1], a naming problem with PC was mentioned. PC has been misused in its singular form. It is mistaken to replace the plural form in "women's world cup" with the singular form (woman). Similarly, The plural form "comparisons" in PC is needed in front of a method or a matrix. So the term PC should be distinguished with the term "paired comparison," which is commonly used for binary entries.

2.2 Pairwise Comparisons Matrix

Pairwise comparisons is based on the construction of a pairwise comparisons matrix (for short, PC matrix). In [2], asssuming there are n alternative items need to be compared. Using m_{ij} to present a relative quantity, intensity, or preference of an entity E_i over E_j. A $n \times n$ PC matrix simply as a square matrix $M = [m_{ij}]$ such that $m_{ij} > 0$ for every $i, j = 1, ..., n$ is defined as

$$M = \begin{bmatrix} 1 & m_{12} & \cdots & m_{1n} \\ \frac{1}{m_{12}} & 1 & \cdots & m_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{1}{m_{1n}} & \frac{1}{m_{2n}} & \cdots & 1 \end{bmatrix} \tag{1}$$

A PC matrix M is called reciprocal if $m_{ij} = \frac{1}{m_{ji}}$, for every $i, j = 1, ..., n$. And automatically $m_{ii} = 1$, for every $i = 1, ..., n$ [3]. The entity compared can be considered as an object, attribute of it or a stimulus. It can also be abstract and subjective, which means there is no well-established measure such as a second or meter. "Life happiness" or "Social safety" are examples of such attributes or entities to be compared in pairs.

Ratios often express two entities' subjective preferences, but it does not mean that it can be done by division. It is unacceptable that equalizing the ratios with the division (e.g., $\frac{E_i}{E_j}$). The division operation has no mathematical meaning when the entities are subjective, for example, the reliability and robustness of software (commonly used in a software development process as product attributes). However, we can still get the priority of them for a

given project. The symbol"/" is not the division of two numbers but in the context of "related to".

2.3 Triad

Given $n \in N$, we define $T(n) = \{(i, j, k) \in \{1, ..., n\} : i < j < k\}$, as the set of all PC matrix indexes of all permissible triads in the upper triangle [3].

$$M = \begin{bmatrix} 1 & m_{12} & \cdots & m_{1n} \\ \frac{1}{m_{12}} & 1 & \cdots & m_{2n} \\ \vdots & \vdots & \vdots & \vdots \\ \frac{1}{m_{1n}} & \frac{1}{m_{2n}} & \cdots & 1 \end{bmatrix},$$

Figure 1: A triad in pairwise comparisons matrix

For example,in the upper picture, matrix M is a PC matrix, (m_{12}, m_{2n}, m_{1n}) is a triad of M. Triad is the basic unit for inconsistency analysis.

2.4 Consistency and Inconsistency of PC Matrix

According to [3], a PC matrix $M = [m_{ij}]$ is called consistent (or transitive) if, for every $(i, j, k) \in T(n)$:

$$m_{ij} \times m_{ik} = m_{ik} \tag{2}$$

Equation (2) also can be referred to as a "consistency condition". Although every consistent PC matrix is reciprocal, the converse is false in general. If the consistency condition does not hold, the PC matrix is inconsistent(or intransitive) [3].

Consistency is an important index when be used for representing real-life problems using scales. In fact, in most comparison cases, inconsistency is more common than consistency. As per Koczkodaj et al.:, "one of the important aspects of collaboration is inconsistency arising from different points of view on the same issue" [4]. That is because our comparisons are subjective, even highly personal. Often we are not even sure about them. It nicely illustrates what are "incomplete knowledge" and "approximate reasoning". The simplest case of inconsistency occurs when three items need to be compared. Take the exchanging of fox, rabbit and fish, for example.

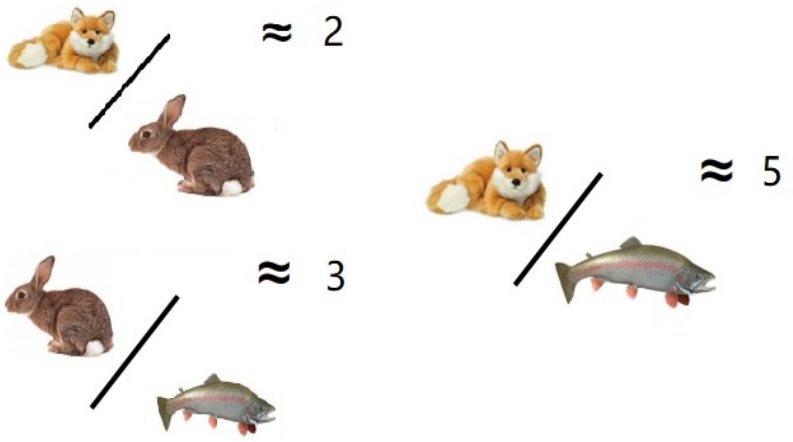

Figure 2: Inconsistency in Pairwise Comparisons

Assuming that one fox needs to be exchanged with two rabbits, one rabbit's value is as same as three fishes, so it is very clear that one fox should be equal to six fishes. Claiming then that one fox is similar to five fishes is inconsistent with our earlier assumptions. In brief, inconsistency represents the degree of contradictions existing on the assessments. Awareness of measure and improve the inconsistency should be arising.

Inconsistent analysis of pairwise comparisons is essential since it can locate the most inconsistent inputs and optimize them to improve the inconsistency. A lot of work had been done for computing the priorities of the comparison items [33]. For example, the Geometric mean Method [18], the Eigenvector Method [19], the Least Squares Method [20], Logarithmic Least Squares Method [21–23], Weighted Least Squares Method [24] and Singular Value

Decomposition [25].

In contrast to the methods mentioned before, Prof. W.W. Koczkodaj has proposed a consistency measure [26] in 1993. Later, it was illustrated and checked as an inconsistency index (Ix) in [27]. Finally, It is well known as the name of Kii, a function from the set of all finite-dimensional reciprocal matrices. It aims to measure how a given matrix differs from a consistent matrix. In other words, for a given reciprocal matrix M, it measures the inconsistency in M.

The Kii of a triad $T = (x, y, z)$ is defined as follow:

$$Kii(x, y, z) = 1 - \min\{\frac{y}{x \cdot z}, \frac{x \cdot z}{y}\} \tag{3}$$

The Kii of a given reciprocal matrix M could be illustrated by the following: For every $(i, j, k) \in T(n)$.

$$Kii(M) = \max_{i<j<k}(1 - \min\{\frac{m_{ik}}{m_{ij} \cdot m_{jk}}, \frac{m_{ij} \cdot m_{jk}}{m_{ik}}\}) \tag{4}$$

It is apparent that for the following PC matrix:

$$A = \begin{pmatrix} 1 & x & y \\ \frac{1}{x} & 1 & z \\ \frac{1}{y} & \frac{1}{z} & 1 \end{pmatrix}$$

we have:

$$Kii(A) = Kii(x, y, z)$$

Especially when the PC matrix M is consistent, the inconsistency indicator of M equals zero. Simultaneously, for each triad of M, every inconsistency indicator should have a value of 0. It could thus be concluded that this is an entirely consistent system.

The following example illustrates the inconsistency in a triad:

This is an inconsistent 3×3 matrix A, with one triad $(2,2,2)$, the corresponds index (i,j,k) is $(1,2,3)$ which is marked by the bold font below:

$$A = \begin{pmatrix} 1 & \mathbf{2} & \mathbf{2} \\ \frac{1}{2} & 1 & \mathbf{2} \\ \frac{1}{2} & \frac{1}{2} & 1 \end{pmatrix}$$

Because $a_{12}\cdot a_{23} \neq a_{13}$, The inconsistency of this triad can be measured using Kii where:

$$Kii = 1 - \min\left\{\frac{a_{13}}{a_{12}\cdot a_{23}}, \frac{a_{12}a_{23}}{a_{13}}\right\} = 1 - \frac{2}{2\times2} = 0.5$$

For the case of $n = 3$, Kii is illustrated by the following 3D plot for the mean value of the triad set to 1.5 on the scale of 1 to 3, which is strongly recommended by the research of prof. W.W. Koczkodaj, The variable (y) of a triad(x,y,z) was set to 1.5, so plot 3D could be gotten for illustration purposes (mainly to show that Kii is not an entirely trivial function).

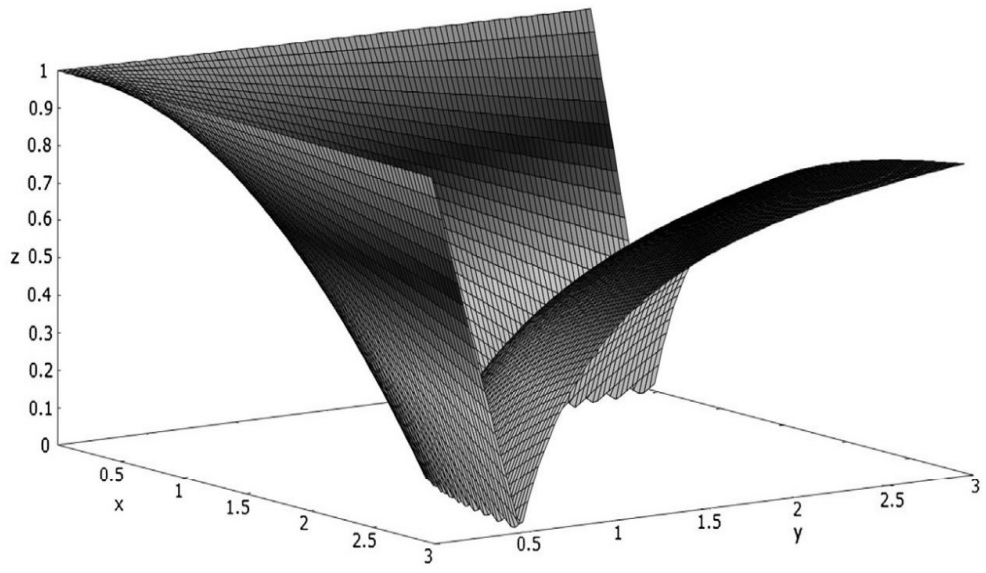

Figure 3: Kii inconsistency indicator 3D section of plot for y = 1.5. [3]

2.5 The Rating Scale Paradox

A rating scale is a set of categories designed to elicit data about a qualitative or a quantitative attribute. Many researchers try to solve the issue of offering a proper subjective scale for PC judgments. In [30], Thurstone defined a psychological continuum with scale values. His approach was later analyzed and elaborated by Luce and Edwards in [28]. Saaty introduced a "scale of 1 to 9" in [6] in 1977. Koczkodaj, W. W proposed the scale of 1 to 5 in [26]. Using rating scales for the input data is a fundamental method in most pairwise comparisons methods to collect graded assessments. According to [31], the paradox may occur when the acquired data are using in a PC matrix

without prior processing.

For example, for the value, 2 on the rating scale "1 to 100" is not the same 2 as on the scale of 1 to 10. When inputting the PC matrix entries, it is necessary to clarify the rating scale upper limit and then incorporate m into the PC matrix. A normalizing mapping is usually used to solve the rating scale paradox [30].

A small rating scale, 1 to 3 for pairwise comparisons, is recommended in [4] for the impossibility of expressing subjective assessments with high accuracy.

3 Inconsistency Improvement Method

3.1 Inconsistency Tolerance

There is a well-known computer adage GIGO (garbage in – garbage out). It summarizes what has been known for a long time: getting good results from "dirty data" is unrealistic, and indeed, can not be guaranteed. Therefore, an approximation of a PC matrix is meaningful if the inconsistency level of it is acceptable. The approximation could be made by targeting the inconsistency and updating it to a certain predefined threshold.

The threshold is usually defined as $\frac{1}{3}$ which is suggested in [7] or any other arbitrary value on a case-to-case basis.

3.2 The Distance-based Inconsistency Reduction Algorithm

The inconsistency reduction problem in pairwise comparisons is one of the most fundamental issues. The inconsistency of a PC matrix indicates the perturbations from consistency. If the inconsistency indicator is large than an acceptable inconsistent tolerance, it is not advisable to rely on the answers generated from unreliable preference information.

A lot of inconsistent research works had been done in [8–10]. The consistency condition illustrated by (2) was introduced and analyzed for the first time in [10]. A distance-based inconsistency indicator is used for the distance-based

inconsistency reduction algorithm. Unlike the eigenvalue-based inconsistency indicator introduced in [11], which has the global characteristic, The distance-based inconsistency indicator targets the most inconsistent triad (or triads). The maximum overall triads in a PC matrix of their inconsistency indicators are defined by (3).

A fast convergence of distance-based inconsistency improvement method was demonstrated in [5]. Considering instead of a full inconsistent PC matrix A, for a given triad (a_{ik}, a_{ij}, a_{kj}) of A, assume $a_{ik} \cdot a_{kj} \neq a_{ij}$, we could use $(\tilde{a}_{ik}, \tilde{a}_{kj}, \tilde{a}_{ij})$ to replace the value of (a_{ik}, a_{ij}, a_{kj}) separately, then a more consistent triad will be obtained. $(\tilde{a}_{ik}, \tilde{a}_{kj}, \tilde{a}_{ij})$ are generated by the following formulas:

$$\tilde{a}_{ik} = a_{ik}^{2/3} \cdot a_{kj}^{-1/3} \cdot a_{ij}^{1/3}$$
$$\tilde{a}_{ij} = a_{ik}^{1/3} \cdot a_{kj}^{1/3} \cdot a_{ij}^{2/3}$$
$$\tilde{a}_{kj} = a_{ik}^{-1/3} \cdot a_{kj}^{2/3} \cdot a_{ij}^{1/3}$$

A complete convergence proof was provided in [12], According to the result of the Monte Carlo study demonstrating the convergence speed of inconsistency reduction in pairwise comparisons in [5], it converges very quickly. Bringing matrices to an inconsistency below 1/3 usually occurs in no more than ten iterations for the worst randomly generated case.

The flow chart followed has illustrated this distance-based inconsistency improvement method briefly.

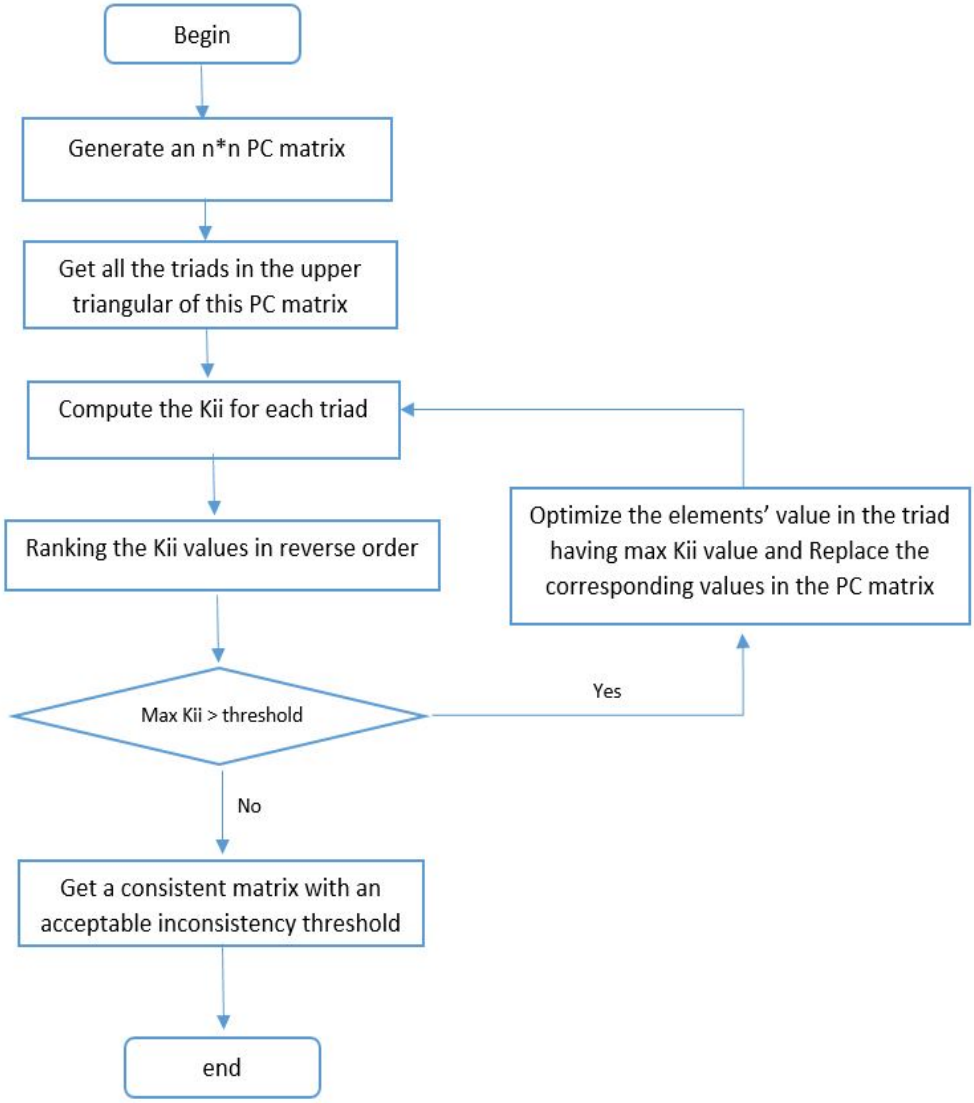

Figure 4: The flowchart of the distance-based inconsistency reduction algorithm

3.3 Examination of Inconsistency Reduction Algorithm

To examine the efficiency and limitation of the distance-based inconsistency reduction algorithm, a basic experiment was conducted in MATLAB.

3.3.1 The Experiment Procedure

First, to perform this experiment, an 8×8 reciprocal pairwise comparisons matrix with the randomly generated elements for the upper triangular part is constructed. The random value is between one to three. This scale (1 to 3) demonstrated in [4], rather than larger ones, has ethical mathematical foundations.

```
m=8;
%generate a random pc matrix
A=eye(m,m);
for i=1:m
    for j = i+1:m
       A(i,j) =randi(3);
    end
    for j = 1:i-1
       A(i,j) =1/A(j,i);
    end
end
A;
```

Figure 5: MATLAB Codes to Generate a Random 8×8 Reciprocal Pairwise Comparisons Matrix

As the followed picture shows, a PC Matrix A is generated as the experi-

ment's input.

A =

1.0000	3.0000	3.0000	1.0000	3.0000	2.0000	1.0000	1.0000
0.3333	1.0000	2.0000	3.0000	3.0000	1.0000	3.0000	3.0000
0.3333	0.5000	1.0000	2.0000	3.0000	1.0000	2.0000	3.0000
1.0000	0.3333	0.5000	1.0000	3.0000	3.0000	2.0000	1.0000
0.3333	0.3333	0.3333	0.3333	1.0000	3.0000	3.0000	3.0000
0.5000	1.0000	1.0000	0.3333	0.3333	1.0000	3.0000	3.0000
1.0000	0.3333	0.5000	0.5000	0.3333	0.3333	1.0000	2.0000
1.0000	0.3333	0.3333	1.0000	0.3333	0.3333	0.5000	1.0000

Figure 6: An Randomly Generated 8×8 Reciprocal Pairwise Comparisons Matrix

To calculate the Koczkodaj inconsistency indicator(Kii) of triads, Finding all the triads located in the upper triangular part of the PC matrix must be carried out. After that, the program computes the Kii for each triad. The indexes values (i, j, k), the values of the corresponding index positions in the matrix, and the Kii value of each triad are organized in a row of a summary matrix. This summary matrix has seven columns (The first three columns hold the triad index, the second three columns store this triad's value, the last column keeps the triad's Kii value). It could be straightforward to rank

this summary matrix's rows by the Kii value column in reverse order.

```
function [ST] = findAllTrials(A)
row=1;
S = size(A);% get the row number of input matrix
m = S(1);
for i= 1:m-1
    for j = i+1:m
        x = A(i,j);
        for k=j+1:m
            z = A(j,k);
            y = A(i,k);
            % first three columns store the index of traid
            T(row,1)=i;
            T(row,2)=j;
            T(row,3)=k;
            % second three columns store the value of traid
            T(row,4)=x;%a_ij
            T(row,5)=y;%a_ik
            T(row,6)=z;%a_jk
            % the last column store the Kii value
            T(row,7)=1-min(y/(x*z), (x*z)/y);
            row = row+1;
        end
    end
end
% sort by the Kii value desc
ST=sortrows(T,-7);
```

Figure 7: Function for Finding All Trails in Pairwise Comparisons Matrix

From the output of the $findAllTraids$ function, it could be seen that there are 56 triads in the upper triangle of this 8×8 PC Matrix in total. The most massive Kii value is 0.8889. The indexes are(3, 5, 6), the corresponding values are m_{35} (value is 3) , m_{56} (value is 1), m_{36} (value is 3).

i	j	k	m_{ij}	m_{ik}	m_{jk}	Kii
3.0000	5.0000	6.0000	3.0000	1.0000	3.0000	0.8889
1.0000	5.0000	7.0000	2.0000	1.0000	3.0000	0.8333
1.0000	6.0000	7.0000	3.0000	1.0000	2.0000	0.8333
1.0000	6.0000	8.0000	3.0000	1.0000	2.0000	0.8333
3.0000	5.0000	8.0000	3.0000	1.0000	2.0000	0.8333
2.0000	5.0000	6.0000	3.0000	2.0000	3.0000	0.7778
2.0000	5.0000	7.0000	3.0000	2.0000	3.0000	0.7778
1.0000	2.0000	4.0000	2.0000	1.0000	2.0000	0.7500
1.0000	2.0000	7.0000	2.0000	1.0000	2.0000	0.7500
1.0000	2.0000	8.0000	2.0000	1.0000	2.0000	0.7500
1.0000	5.0000	8.0000	2.0000	1.0000	2.0000	0.7500
4.0000	5.0000	8.0000	2.0000	1.0000	2.0000	0.7500
4.0000	6.0000	8.0000	2.0000	1.0000	2.0000	0.7500
1.0000	2.0000	5.0000	2.0000	2.0000	3.0000	0.6667
1.0000	3.0000	6.0000	1.0000	3.0000	1.0000	0.6667
1.0000	3.0000	7.0000	1.0000	1.0000	3.0000	0.6667
1.0000	4.0000	7.0000	1.0000	1.0000	3.0000	0.6667
2.0000	4.0000	7.0000	2.0000	2.0000	3.0000	0.6667
2.0000	5.0000	8.0000	3.0000	2.0000	2.0000	0.6667
3.0000	5.0000	7.0000	3.0000	3.0000	3.0000	0.6667

3.0000	7.0000	8.0000	3.0000	1.0000	1.0000	0.6667
4.0000	5.0000	6.0000	2.0000	2.0000	3.0000	0.6667
4.0000	7.0000	8.0000	3.0000	1.0000	1.0000	0.6667
5.0000	6.0000	8.0000	3.0000	2.0000	2.0000	0.6667
1.0000	2.0000	3.0000	2.0000	1.0000	1.0000	0.5000
1.0000	5.0000	6.0000	2.0000	3.0000	3.0000	0.5000
2.0000	3.0000	4.0000	1.0000	2.0000	1.0000	0.5000
2.0000	3.0000	6.0000	1.0000	2.0000	1.0000	0.5000
2.0000	3.0000	8.0000	1.0000	2.0000	1.0000	0.5000
2.0000	4.0000	6.0000	2.0000	2.0000	2.0000	0.5000
2.0000	6.0000	7.0000	2.0000	2.0000	2.0000	0.5000
2.0000	6.0000	8.0000	2.0000	2.0000	2.0000	0.5000
3.0000	4.0000	6.0000	1.0000	1.0000	2.0000	0.5000
3.0000	6.0000	8.0000	1.0000	1.0000	2.0000	0.5000
4.0000	5.0000	7.0000	2.0000	3.0000	3.0000	0.5000
5.0000	6.0000	7.0000	3.0000	3.0000	2.0000	0.5000
1.0000	3.0000	5.0000	1.0000	2.0000	3.0000	0.3333
1.0000	4.0000	6.0000	1.0000	3.0000	2.0000	0.3333
2.0000	3.0000	7.0000	1.0000	2.0000	3.0000	0.3333
3.0000	4.0000	5.0000	1.0000	3.0000	2.0000	0.3333
3.0000	6.0000	7.0000	1.0000	3.0000	2.0000	0.3333

5.0000	7.0000	8.0000	3.0000	2.0000	1.0000	0.3333
1.0000	2.0000	6.0000	2.0000	3.0000	2.0000	0.2500
2.0000	4.0000	5.0000	2.0000	3.0000	2.0000	0.2500
4.0000	6.0000	7.0000	2.0000	3.0000	2.0000	0.2500
1.0000	3.0000	4.0000	1.0000	1.0000	1.0000	0
1.0000	3.0000	8.0000	1.0000	1.0000	1.0000	0
1.0000	4.0000	5.0000	1.0000	2.0000	2.0000	0
1.0000	4.0000	8.0000	1.0000	1.0000	1.0000	0
1.0000	7.0000	8.0000	1.0000	1.0000	1.0000	0
2.0000	3.0000	5.0000	1.0000	3.0000	3.0000	0
2.0000	4.0000	8.0000	2.0000	2.0000	1.0000	0
2.0000	7.0000	8.0000	2.0000	2.0000	1.0000	0
3.0000	4.0000	7.0000	1.0000	3.0000	3.0000	0
3.0000	4.0000	8.0000	1.0000	1.0000	1.0000	0
6.0000	7.0000	8.0000	2.0000	2.0000	1.0000	0

Table 1: Result Table of $findAllTrials$ Function

After getting the summary table data, we know that the number of Kii whose value is larger than the set tolerance level (1/3) is 42. For the time being, the inconsistency threshold is arbitrary or nominated by a heuristic since there is no theory to find it. It is usually defined as 1/3, which is suggested in the research of prof. W.W. Koczkodaj or any other arbitrary value depends on

each case.

The next step is crucial since we have got all the inconsistency triads, then an improvement needs to be carried out. Here the fast convergence of distance-based inconsistency improvement method demonstrated in the former part is used. The corresponding codes are as below shows.

```
%find all trials; compute Kii; Sort by Kii desc; Get a summary matrix
ST = findAllTrials(A);
%set an inconsistency threshold
eps=1/3;
% get the number of kii which value is larger than eps
N = sum(ST(:,7) > eps);
% get the number of triads
SST=size(ST);
r=SST(1);
%initial the repeat time value
repeat_time = 1;
% execute until all the Kii value is smaller than eps
while N ~= 0
    %get the value of a triad with the largest Kii
    x = ST(1,4);
    y = ST(1,5);
    z = ST(1,6);

    %improve the value
    new_x = x ^(2/3)*z^(-1/3)*y^(1/3);
    new_y = x ^(1/3)*z^(1/3)*y^(2/3);
    new_z = x ^(-1/3)*z^(2/3)*y^(1/3);

    % update matrix using new x,y,z
    A(ST(1,1), ST(1,2)) = new_x;
    A(ST(1,1), ST(1,3)) = new_y;
    A(ST(1,2), ST(1,3)) = new_z;

    A(ST(1,2), ST(1,1)) = 1/new_x;
    A(ST(1,3), ST(1,1)) = 1/new_y;
    A(ST(1,3), ST(1,1)) = 1/new_z;

    %recalculate the Kii of all triads and sort by Kii desc
    ST = findAllTrials(A);

    % get the number of Kii which value is larger than eps
    N = sum(ST(:,7) > eps);
    repeat_time = repeat_time+1;
end
repeat_time;
A;
ST;
```

Figure 8: Inconsistency Improvement Method Implementation Code

It can reduce the triad's inconsistency by optimizing the elements' value in the "highly" inconsistent triads. Next, the corresponding value in the pc matrix is replaced by the improved value. Repeat getting all the triads, calculating all the Kii and getting the most significant amount, then improving the corresponding triad value until there is no Kii larger than the tolerance level defined.

```
N =

42


repeat_time =

12


A =

1.0000  0.9335  1.0000  1.0000  1.6255  1.9674  2.2120  1.8171

1.0712  1.0000  1.0000  1.3867  1.7046  2.0000  2.3695  2.0000

1.0000  1.0000  1.0000  1.0000  1.4422  1.6510  2.6207  1.4422

1.0000  0.7211  1.0000  1.0000  1.2599  1.7371  2.3949  1.5874

0.6152  0.3333  0.6934  0.7937  1.0000  1.2103  1.6510  1.2599

0.8262  0.5000  0.6057  0.5757  0.3333  1.0000  1.3787  1.1006

0.4220  0.4807  0.6300  0.7253  0.3333  0.5000  1.0000  0.6934

0.9086  0.5000  1.4422  0.7937  0.5000  0.5000  1.0000  1.0000


ST =

2.0000  4.0000  7.0000  1.3867  2.3695  2.3949  0.2865
```

2.0000	3.0000	4.0000	1.0000	1.3867	1.0000	0.2789
2.0000	3.0000	8.0000	1.0000	2.0000	1.4422	0.2789
1.0000	2.0000	4.0000	0.9335	1.0000	1.3867	0.2275
1.0000	4.0000	5.0000	1.0000	1.6255	1.2599	0.2249
3.0000	5.0000	8.0000	1.4422	1.4422	1.2599	0.2063
3.0000	6.0000	8.0000	1.6510	1.4422	1.1006	0.2063
1.0000	3.0000	8.0000	1.0000	1.8171	1.4422	0.2063
3.0000	7.0000	8.0000	2.6207	1.4422	0.6934	0.2063
1.0000	6.0000	7.0000	1.9674	2.2120	1.3787	0.1845
2.0000	7.0000	8.0000	2.3695	2.0000	0.6934	0.1785
1.0000	5.0000	7.0000	1.6255	2.2120	1.6510	0.1758
2.0000	3.0000	6.0000	1.0000	2.0000	1.6510	0.1745
2.0000	4.0000	6.0000	1.3867	2.0000	1.7371	0.1697
4.0000	6.0000	8.0000	1.7371	1.5874	1.1006	0.1697
1.0000	6.0000	8.0000	1.9674	1.8171	1.1006	0.1608
1.0000	3.0000	6.0000	1.0000	1.9674	1.6510	0.1608
2.0000	5.0000	7.0000	1.7046	2.3695	1.6510	0.1580
1.0000	3.0000	7.0000	1.0000	2.2120	2.6207	0.1560
1.0000	7.0000	8.0000	2.2120	1.8171	0.6934	0.1560
2.0000	3.0000	5.0000	1.0000	1.7046	1.4422	0.1539
2.0000	6.0000	7.0000	2.0000	2.3695	1.3787	0.1407
3.0000	6.0000	7.0000	1.6510	2.6207	1.3787	0.1315
4.0000	5.0000	7.0000	1.2599	2.3949	1.6510	0.1315
6.0000	7.0000	8.0000	1.3787	1.1006	0.6934	0.1315

3.0000	4.0000	5.0000	1.0000	1.4422	1.2599	0.1264
1.0000	4.0000	8.0000	1.0000	1.8171	1.5874	0.1264
4.0000	5.0000	6.0000	1.2599	1.7371	1.2103	0.1222
1.0000	4.0000	6.0000	1.0000	1.9674	1.7371	0.1171
1.0000	3.0000	5.0000	1.0000	1.6255	1.4422	0.1128
1.0000	5.0000	8.0000	1.6255	1.8171	1.2599	0.1128
2.0000	3.0000	7.0000	1.0000	2.3695	2.6207	0.0958
2.0000	4.0000	8.0000	1.3867	2.0000	1.5874	0.0914
3.0000	4.0000	8.0000	1.0000	1.4422	1.5874	0.0914
2.0000	6.0000	8.0000	2.0000	2.0000	1.1006	0.0914
5.0000	7.0000	8.0000	1.6510	1.2599	0.6934	0.0914
3.0000	5.0000	7.0000	1.4422	2.6207	1.6510	0.0914
3.0000	4.0000	7.0000	1.0000	2.6207	2.3949	0.0862
1.0000	4.0000	7.0000	1.0000	2.2120	2.3949	0.0764
2.0000	5.0000	8.0000	1.7046	2.0000	1.2599	0.0687
1.0000	2.0000	3.0000	0.9335	1.0000	1.0000	0.0665
5.0000	6.0000	8.0000	1.2103	1.2599	1.1006	0.0542
3.0000	5.0000	6.0000	1.4422	1.6510	1.2103	0.0542
1.0000	2.0000	6.0000	0.9335	1.9674	2.0000	0.0510
3.0000	4.0000	6.0000	1.0000	1.6510	1.7371	0.0496
4.0000	7.0000	8.0000	2.3949	1.5874	0.6934	0.0441
2.0000	5.0000	6.0000	1.7046	2.0000	1.2103	0.0306
1.0000	2.0000	8.0000	0.9335	1.8171	2.0000	0.0267
2.0000	4.0000	5.0000	1.3867	1.7046	1.2599	0.0244

```
1.0000  2.0000  5.0000  0.9335  1.6255  1.7046  0.0211
5.0000  6.0000  7.0000  1.2103  1.6510  1.3787  0.0106
1.0000  2.0000  7.0000  0.9335  2.2120  2.3695  0.0000
4.0000  6.0000  7.0000  1.7371  2.3949  1.3787  0.0000
1.0000  5.0000  6.0000  1.6255  1.9674  1.2103  0.0000
1.0000  3.0000  4.0000  1.0000  1.0000  1.0000       0
4.0000  5.0000  8.0000  1.2599  1.5874  1.2599       0
```

The upper output gets the results, including the number of Kii, which is more significant than the tolerance level (eps = 1/3), N in the output, is 42. After 12 times repetition, we get a consistent matrix A with an acceptable tolerance level. Through the corresponded summary matrix (ST in the output), we can see that all the Kii value in the seventh column is smaller than the tolerance level (eps = 1/3).

3.3.2 Experiment Result and Conclusion

Through the experiments, the efficiency of an improved method for reducing a PC matrix's inconsistency was checked in MATLAB. Additionally, as the followed result table shows, all the other values in the right columns increased when the matrix dimension increased, especially when the PC matrix is 32×32. The execution time is much larger than that of different dimension values.

Table 2: Program Execution Indicators Summary Table

The PC matrix dimension	Triad Number	Inconsistent Triad Number	Repetition Number	Execution Time
4 × 4	4	2	2	0.0009298
8 × 8	56	43	15	0.0074012
16 × 16	560	442	64	0.034138
32 × 32	4960	3962	241	6.4902

With the following two charts, the results will be illustrated more clearly and intuitively. When the order of the PC matrix doubled, the number of triads increased to about ten times, the tendency of the number of inconsistent triads rising is similar.

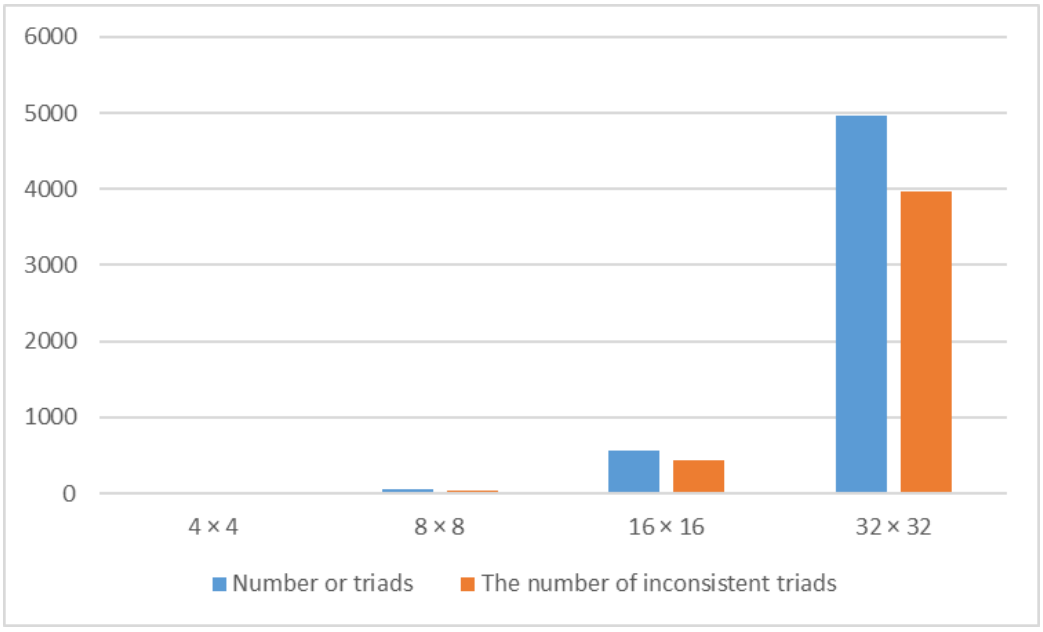

Figure 9: Program Execution Indicators-Traid

There are sharp rises in both the number of repetitions and the execution times when the PC matrix dimension is increased from 16×16 to 32×32. What needs to be noted is that if the dimension value of PC Matrix is 64×64, the execution time is so long that it cannot be measured so far.

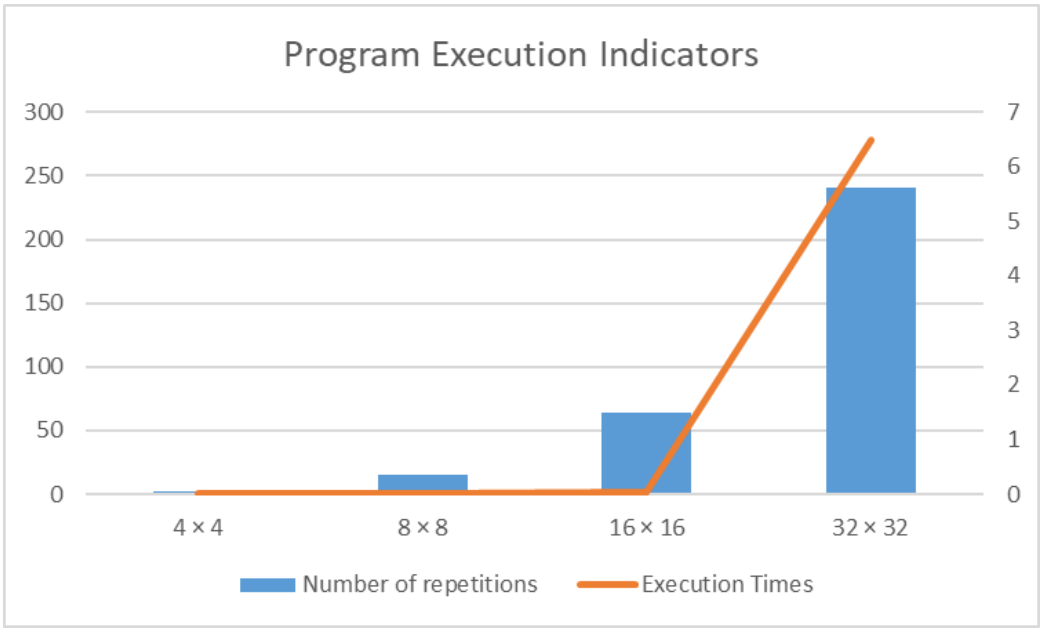

Figure 10: Program Execution Indicators-Time

The pairwise comparisons method is one of the universal approaches to solving severe problems. In particular, contributions to proving the superiority of the pairwise comparisons method in terms of higher precision of the solution are awaited.

Inconsistency analysis allows us to locate the most inconsistent triad. In practice, we change only one value in a triad. Depending on the application, it may take days or even weeks to call an expert panel, gather data, analyze

it, and decide which value should be altered. In our experimentation, we modify all three values: m_{ik}, m_{kj}, and m_{ij}. This is done by splitting the total modification into three triad elements by minimizing the change's effect on the initial PC matrix.

Since the PC matrix is mainly used to solve the problem, which contains the comparisons, for 32×32 dimensions, 32 entities need to be compared. In most cases, it may enough to solve the problem. But now and in the future, the more complex applications may exist, we need to find more efficient algorithms to solve the inconsistency improvement problems.

Hierarchical pairwise comparisons could be used for more complex MCDM, which has more than 32 entities to be compared. For weighting a group of complex alternatives, the users should be prompted to relocate these items to build a tree model. For a parent node, add not more than 32 children nodes(the comparison items) for a better user experience(according to the experiment result). There is no limitation of the number of tree levels since the comparisons only happen at the same group other than different levels.

4 The Design and Implementation of RConcluder application

4.1 Application Overview and User Interface

RConcluder is a Shiny web application based on the consistency-driven pairwise comparisons method for accelerating complex multi-criteria decision-making problems. The app could be reached via `https://lillian5120.shinyapps.io/RConcluder/`. This app's users could be the decision-maker or students who want to know how to use the pairwise comparisons method for solving multi-criteria decision-making problems.

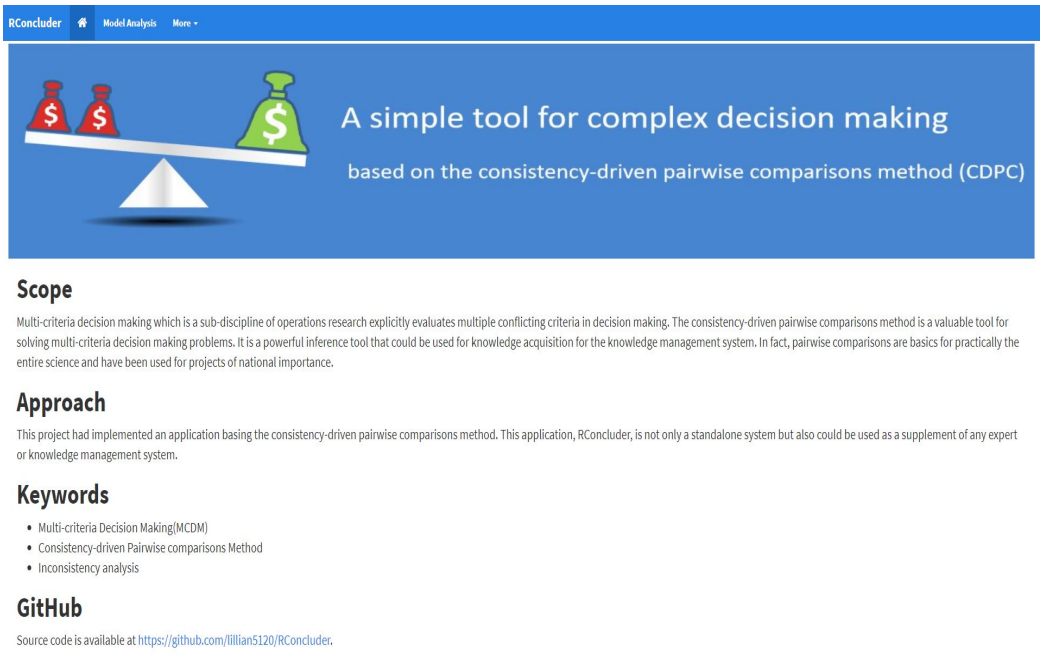

Figure 11: RConcluder Main User Interface

In the RConcluder app, the "Model Analysis" module contains core functions. The main operation steps could be described briefly as follows:

- Model Import/Create
- Model Edit
- Model Analysis
- Model Update
- Model Download

4.1.1 Model Import

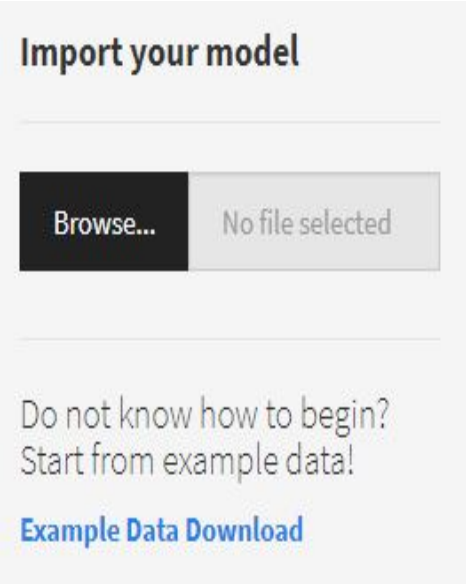

Figure 12: Model Import User Interface

The model could be imported by clicking the "Browse" button. For the first-time user, it recommends downloading the example file to glance at the model data structure and analyze the data. RConcluder has initialled a parent node, Root, with an initial weight of 100.

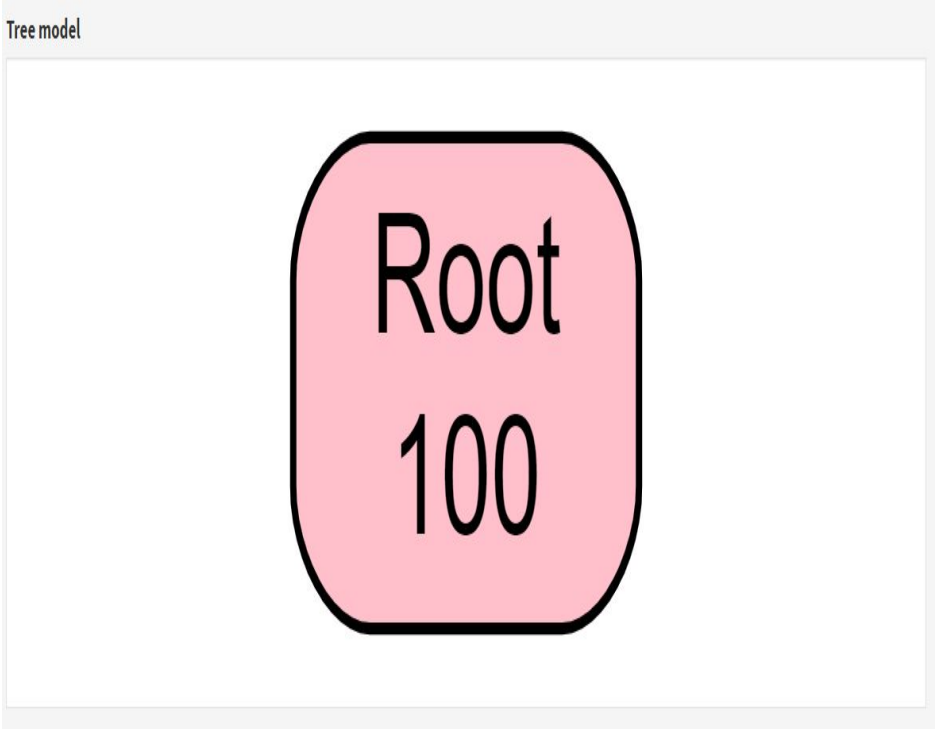

Figure 13: Initial Model Tree

4.1.2 Model Reset

Users who are not satisfied with the model they created could reset the model tree to the initial status using the "Reset Tree" button. Similarly, the "Reset Tree Weight" button is offered to reset the model weight to initial values.

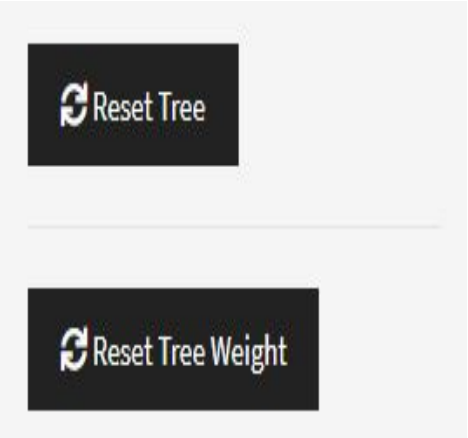

Figure 14: Model Reset Buttons

4.1.3 Add Node

To add a node for the model tree, a parent node name should be chosen from the "Parent name" drop-down box first, and then input the new node name. After clicking the "Add" button, the selected parent node will have a new child node.

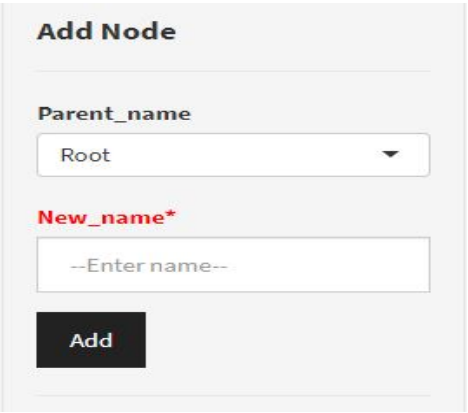

Figure 15: Add Node User Interface

There are two types of validating input checks for empty and duplicate node names, respectively. Error notification will be displayed at the bottom of the "Add Node" section once the input validates checking fails.

Figure 16: New Node Name is Empty Error

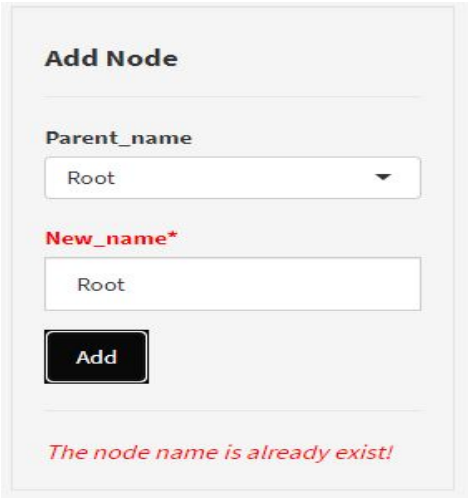

Figure 17: New Node Name is Duplicate Error

4.1.4 Change Node

In the "Change Node Name" section, it can alter the node name by choosing the target node and input the new node name.

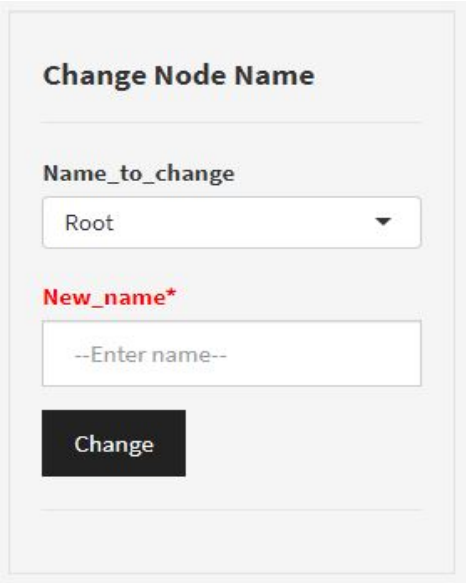

Figure 18: Change Node User Interface

Be Similar to the "Add Node" section, the "Change Node Name" section also has two kinds of validating input checks and error notifications for empty and duplicate node names, respectively.

Figure 19: New Node Name is Empty Error

Figure 20: New Node Name is Duplicate Error

4.1.5 Delete Node

The delete node function can be done by choosing a target node and clicking the "Remove" button. What needs to be aware is that the selected node and the children nodes of this node will both be deleted.

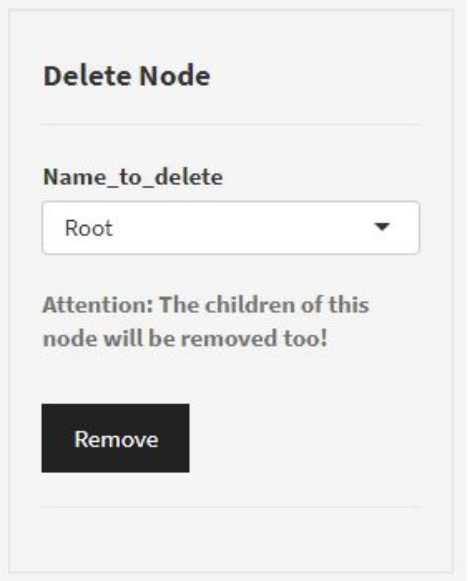

Figure 21: Delete Node User Interface

4.1.6 Analysis Model

After building the tree model, it could be easily analyzed by clicking the "Model Analysis" button. In most scenarios, The "Inconsistency Tolerance Level" is the default value, 0.33. It also can be changed under various conditions. The parent node name needs to be chosen. Only the nodes that have over two children nodes at the same level can be analyzed. Otherwise, an error notification will disappear at the bottom part.

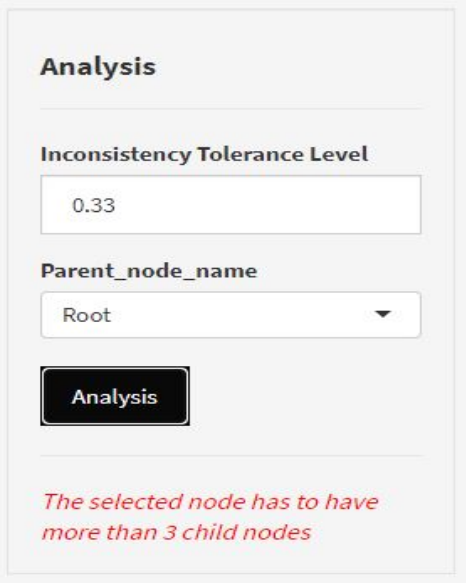

Figure 22: Analysis Model User Interface and The Error Notification

A hierarchical tree is the primary visualization form in RConcluder. It can help users to picture how a PC model is constructed.

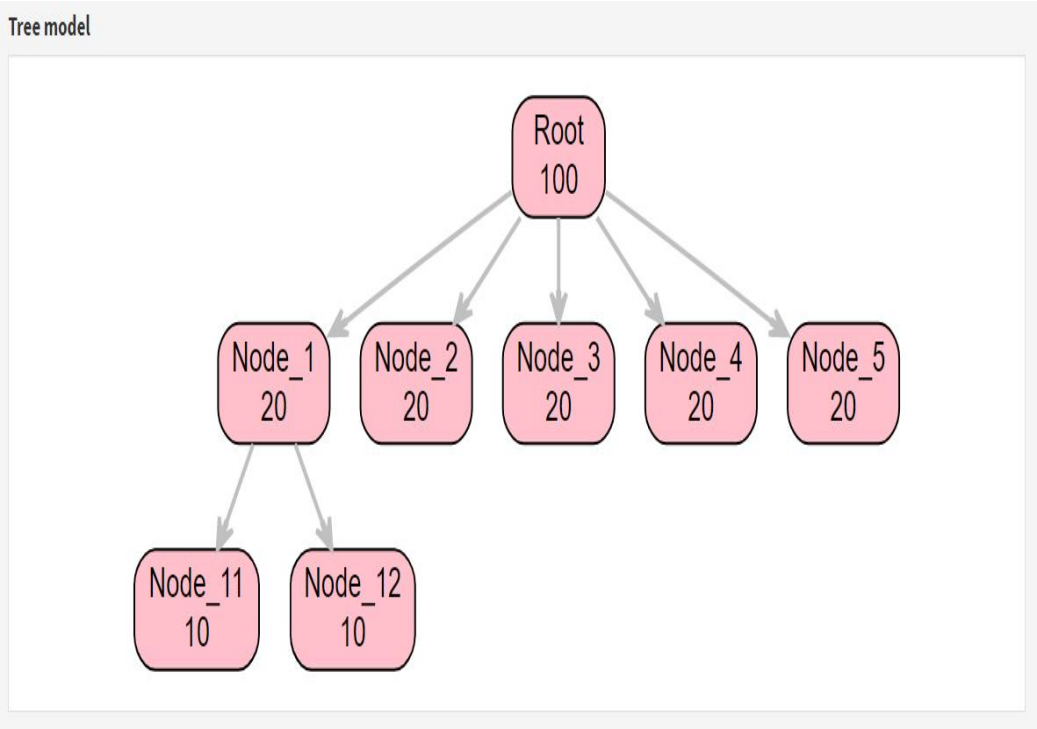

Figure 23: Generated Model Tree

4.1.7 Pairwise Comparisons Matrix

Pairwise Comparisons Matrix is the basic of the Pairwise Comparisons method, which represents the comparison of a set of items. Each cell in the matrix corresponds to a comparison of a pair of objects (hence the name "pairwise comparisons"). For example, The cell in row one and column two represent the comparison value of Node_1 and Node_2.

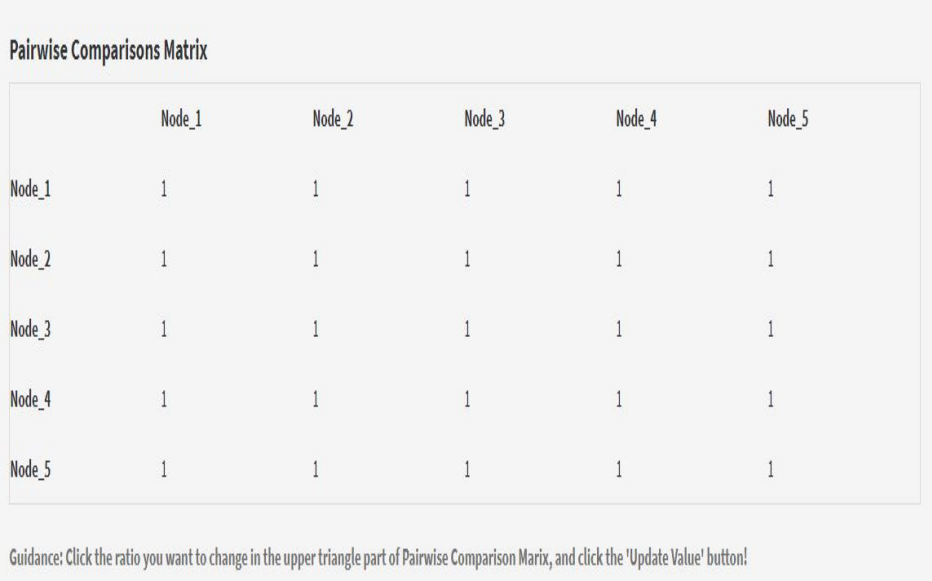

Figure 24: Pairwise Comparisons Matrix

Only the matrix's upper triangle is needed because comparing A to B is the same as comparing B to A. Since the rows and columns contain the same things in the same order, the matrix's lower triangle will have just a mirror image of the upper one.

Furthermore, the matrix's diagonal is irrelevant – it only does not make sense to consider how vital one criterion is to itself.

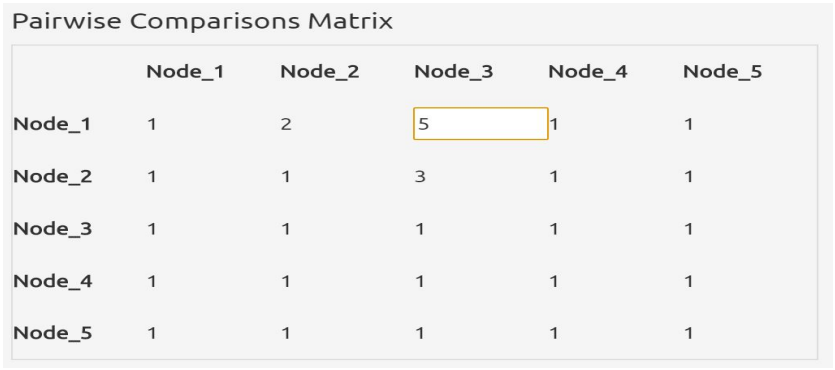

Figure 25: Edit Pairwise Comparisons Matrix

The values in the cell can be changed by double-clicking the cell. The updating data only will take effect after clicking the "Update Value" button.

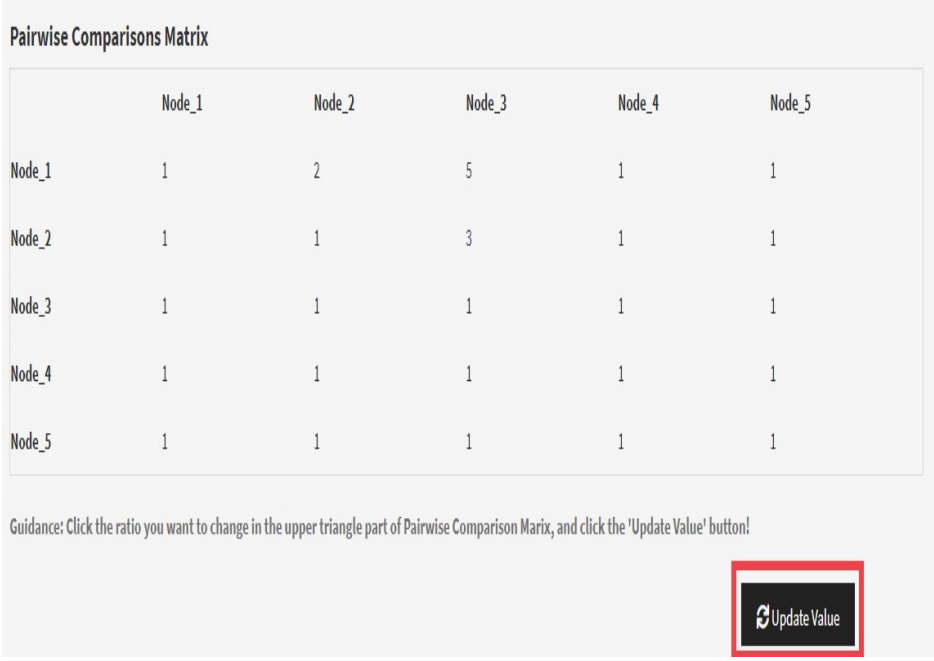

Figure 26: Update Pairwise Comparisons Matrix

The "Dashboard" section shows the analysis results of this Pairwise Comparisons Matrix, including whether the matrix is consistent under the specified tolerance level, locating the triad having the maximum Kii value and a chart. The chart shows the numbers of consistent triads and the inconsistent triads in a Bar diagrams style.

Figure 27: Results Dashboard

4.1.8 Reduce Inconsistency

When the Pairwise Comparisons Matrix's consistency is not acceptable (Inconsistency is larger than the tolerance level.), we can choose to click the "Reduce Inconsistency" button. The consistency could be improved by using the Distance-based Inconsistency Reduction Algorithm mentioned earlier in this thesis.

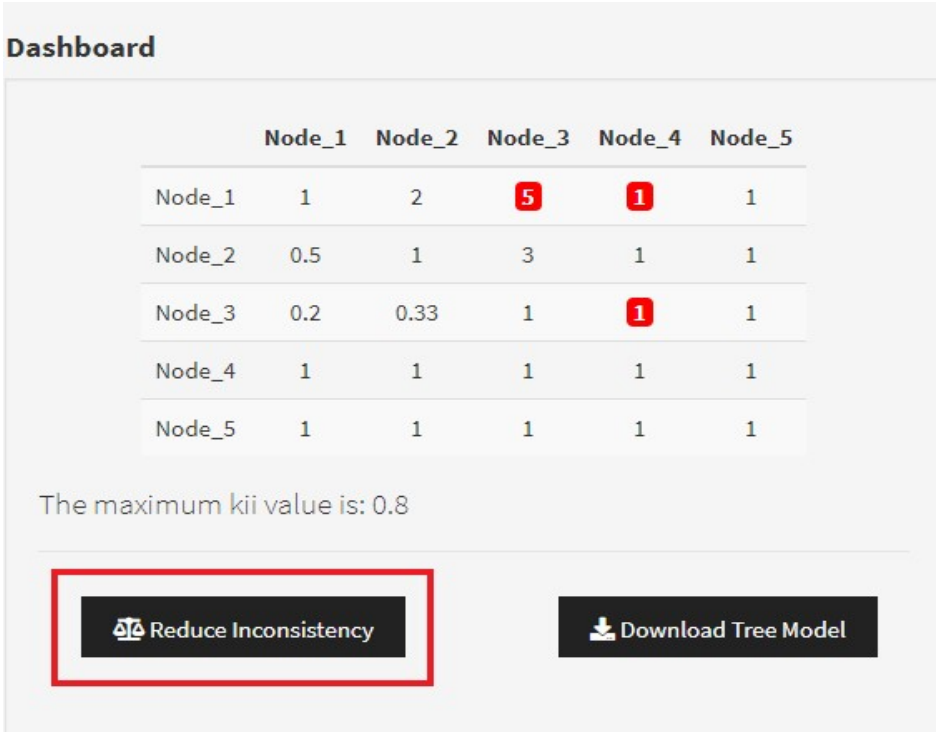

Figure 28: Reduce Inconsistency

After the inconsistency reducing computing, the "Dashboard" section will show the result of this inconsistency reducing operation. A notification of congrats and the maximum Kii value can be seen. The bar chart shows that

all the triads are consistent.

Figure 29: Results Dashboard

4.1.9 Model Download

Suppose the Pairwise Comparisons Matrix values need to be re-updated or re-analysis in the future. Then we only need to repeat the steps mentioned before. After we got a satisfactory result, we can download the analyzed tree model by clicking the "Download Tree Model" button. A CSV file will be saved for further use. This CSV file also could be imported into this app as

data input for further analysis.

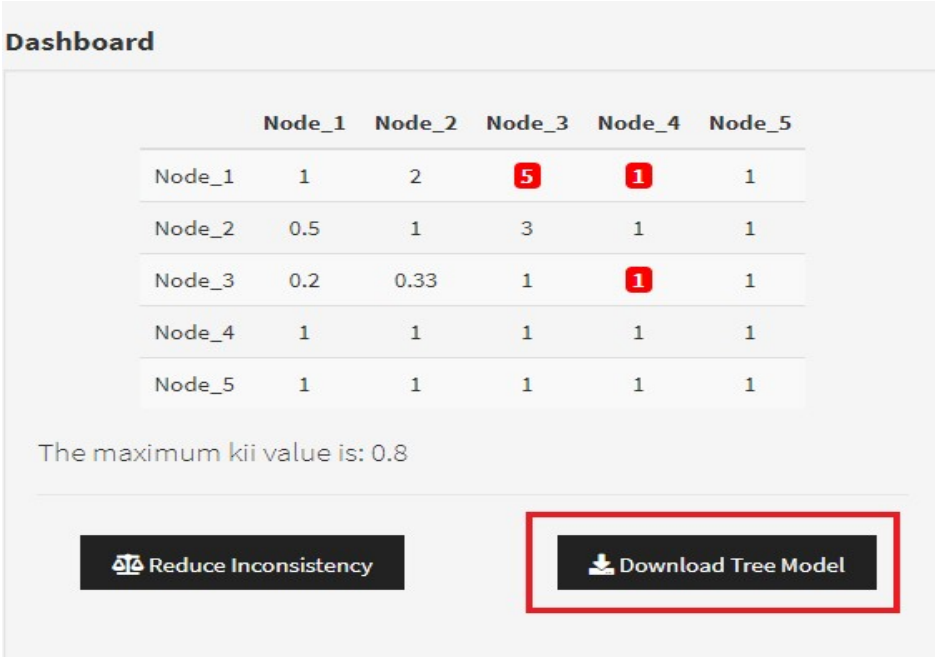

Figure 30: Model Download

4.1.10 Others

The RConcluder application's related knowledge, Pairwise Comparisons method's background information, and the RConcluder source code address are listed in the app.

4.2 Application Development Environment and Techniques

4.2.1 RStudio

RStudio is an open-source, and enterprise-ready professional software for the R community which offers an integrated development environment for R [13]. Data analysis scripts, interactive web applications, documents, reports, graphs, and quite more files could be created using RStudio. In [13], "It includes a console, syntax-highlighting editor that supports direct code execution and tools for plotting, history, debugging and workspace management".

RStudio is available in open source, and commercial editions. It runs on the desktop (Windows, Mac and Linux) or browsers that connect to the RStudio server or RStudio Server Pro.

Figure 31: RStudio [13]

In this thesis, open-source version RStudio is used for R code development.

4.2.2 R

R is an interpretive programming language and free software environment for statistical calculation and graphics supported by R Statistical Computing Foundation. Because of its powerful and flexible statistical programming language and computing environment, the R language is trendy among statisticians and data miners for operating statistical software and performing data analysis work.

R is comparable to popular commercial statistical packages such as SAS, SPSS, and Stata in recent years. It becomes an adequate standard among statisticians. It should be mentioned that interest in R among social scientists is increasing.

To make the R code more readable, proper commenting is essential. Other than that, comments also could be used for preventing execution when testing alternative code.

Generally speaking, comments starts with a #. When executing the R-code, R will ignore anything that starts with # [34]. For the comment of a single line, it could be at the before or end of a line of code.

```
# a comment before a line of code
print('Hello world') # a comment at the end of a line of code
```

Figure 32: Comments for a single line of code

Unlike other programming languages, such as Java, there is no syntax in R for multiline comments and documentation comments. However, we can insert a # at the beginning of each line to create multiline comments [34].

```
# This is a comment
# written in
# more than just one line

print('Hello world')
```

Figure 33: Multiline Comments

4.2.3 Shiny

Shiny is an R package that makes it easy to build interactive web applications using R code. It combines the analytical abilities of R with the interactivity of the modern web. Shiny Apps are reactive by design and ship with a potent and flexible layout engine, which provides the flexibility to lay apps out. According to [14], Shiny Apps are HyperText Markup Language(HTML) documents so that the output appearance could be customized with Cascading Style Sheets (CSS). Shiny uses the Bootstrap 3 CSS framework, the most popular HTML, CSS, and JS framework for developing responsive, mobile-first projects. The Shiny Apps could be enhanced with JavaScript and be hosted online or on an internal server.

Figure 34: Shiny [14]

Every Shiny app is maintained by a computer running R. It would be essential to use the basic shiny template for constructing apps. The template includes two main functions, ui and server. The ui function is responsible for defining user interface components, and the server function is used for creating reactivity by using Inputs to build rendered Outputs.

4.2.4 HTML, JavaScript and CSS

A collection of HTML, JavaScript and CSS files is used for the implementation of this web application. Cascading Style Sheets(CSS) is a framework

for customizing elements' appearance on a web page.

4.3 Application Deploy

Shinyapps.io is a platform as a service (PaaS) for hosting Shiny web applications. However, the RConcluder application could be deployed to shinyapps.io or ran on a very own Shiny server. In this thesis, shinyapps.io is chosen for hosting the RConcluder application for the following reasons. It is free, easy to use, secure and scalable. It not only offers a self-service platform for sharing and managing your shiny applications on the web in an easy and handy way but also runs in the cloud on shared servers that are operated by RStudio, which is very convenient for application development.

In shinyapps.io, each application is self-contained and manipulates information extracted from a third-party data store, such as a database or Web service, by data or code uploaded with the application. Many users use shinyapps.io to prove some concepts, build out a prototype, or just run it for a short period for their purposes. Others are using it as a core component of their analytical offerings within a more extensive online property.

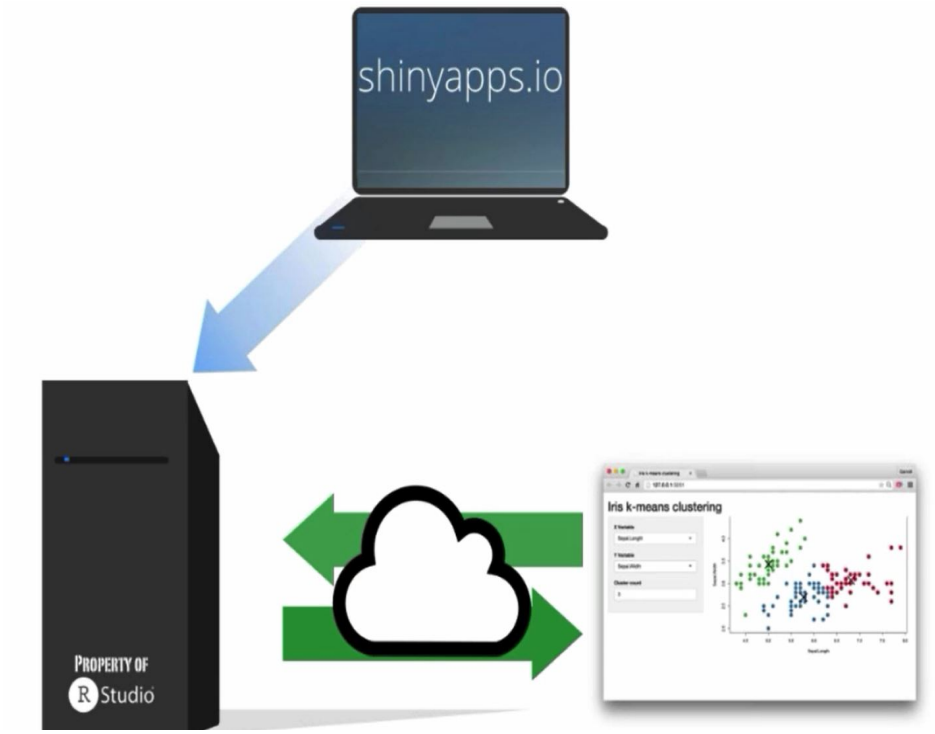

Figure 35: Shinyapps.io [14]

4.4 Deploy Guidance

Download the RConcluder app source codes from Github address: `https://github.com/lillian5120/RConcluder`. The folder named "www" is the designed storage place for the images, JavaScript and CSS files, and other documents that Shiny will share with the user's browser. The ui.R file is for the user interface of this app. The server.R file is for the backend control and function implementation codes.

After downloading and opening the ui.R and server.R files in your RStudio,

installing the required packages, the app can run directly by clicking the "Run App" button on the right top of the code editing area.
Host RConcluder apps at shinyapps.io by signing up for a free shinyapps.io account and installing the shiny app's package in RStudio IDE, under 0.99 version, while Building your server with Shiny Server or Shiny Server Pro is acceptable.

5 Conclusions

The method of Pairwise Comparisons creates a reliable ranking system with broad applications. Using the Koczkodaj inconsistency indicator to assess consensus and remove outliers represents a novel and unique approach to the problem. The RConluder web application offers a handy way to conduct complicated multi-criteria decision-making problem methods. The appendix codes can guide the prospective developers and fans of the pairwise comparisons method with their implementation in Shiny. The RConcluder app could be improved in different ways, such as integrating knowledge analysis and decision-making systems or just running on the developer's very own device for scientific study and research.

References

[1] Koczkodaj, W. W., Mikhailov, L., Redlarski, G., Soltys, M., Szybowski, J., Tamazian, G., . . . Yuen, K. K. (2016). Important facts and observations about pairwise comparisons (the special issue edition). Fundamenta Informaticae, 144(3-4), 291-307. doi:10.3233/fi-2016-1336

[2] Koczkodaj, W. W., Kułakowski, K., & Ligęza, A. (2014). On the quality evaluation of scientific entities in Poland supported by Consistency-driven pairwise comparisons method. Scientometrics, 99(3), 911-926. doi:10.1007/s11192-014-1258-y

[3] Koczkodaj, W., Urban, R. (2018). Axiomatization of inconsistency indicators for pairwise comparisons. International Journal of Approximate Reasoning, 94, 18-29. doi:10.1016/j.ijar.2017.12.001

[4] Fülöp, J., Koczkodaj, W. W., & Szarek, S. J. (2010). A different perspective on a scale for pairwise comparisons. Lecture Notes in Computer Science, 71-84. doi:10.1007/978-3-642-15034-0_5

[5] Koczkodaj, W. W., Kosiek, M., Szybowski, J., & Xu, D. (2015). Fast Convergence of Distance-based Inconsistency in Pairwise Comparisons. Fundamenta Informaticae, 137(3), 355-367. doi:10.3233/fi-2015-1184

[6] Saaty, T. L. (1977). A scaling method for priorities in hierarchical structures. Journal of Mathematical Psychology, 15(3), 234-281. doi:10.1016/0022-2496(77)90033-5

[7] Kakiashvili, T., Koczkodaj, W. W., Montgomery, P., Passi, K., & Tadeusiewicz, R. (2008). Assessing the properties of the world health organization's quality of life index. 2008 International Multiconference on Computer Science and Information Technology. doi:10.1109/imcsit.2008.4747232

[8] Gerard, H. B., & Shapiro, H. N. (1958). Determining the degree of inconsistency in a set of paired comparisons. Psychometrika, 23(1), 33-46. doi:10.1007/bf02288977

[9] Hill, R. J. (1953). A note on inconsistency in paired comparison judgments. American Sociological Review, 18(5), 564. doi:10.2307/2087442

[10] Moran, P. A. (1947). On the method of PAIRED COMPARISONS. Biometrika, 34(3/4), 363. doi:10.2307/2332449

[11] Saaty, T. L. (1977). A scaling method for priorities in hierarchical structures. Journal of Mathematical Psychology, 15(3), 234-281. doi:10.1016/0022-2496(77)90033-5

[12] Koczkodaj, W. W., & Szarek, S. J. (2010). On distance-based inconsistency reduction algorithms for pairwise comparisons. Logic Journal of IGPL, 18(6), 859-869. doi:10.1093/jigpal/jzp062

[13] Download the rstudio ide. (n.d.). Retrieved February 23, 2021, from https://rstudio.com/products/rstudio/download/

[14] Shiny. (n.d.). Retrieved February 23, 2021, from https://rstudio.com/products/shiny/

[15] Saaty, T. L., & Bennett, J. P. (1977). A theory of analytical hierarchies applied to political candidacy. Behavioral Science, 22(4), 237-245. doi:10.1002/bs.3830220402

[16] De Condorcet, N. (n.d.). ESSAI sur L'application DE L'ANALYSE à LA PROBABILITÉ des décisions Rendues à la Pluralité des voix. Essai Sur L'application De L'analyse à La Probabilité Des Décisions Rendues à La Pluralité Des Voix, 1-2. doi:10.1017/cbo9781139923972.002

[17] Koczkodaj, W., Liu, F., Marek, V., Mazurek, J., Mazurek, M., Mikhailov, L., . . . Yayli, Y. (2020). On the use of group theory to generalize elements of pairwise comparisons matrix: A cautionary note. International Journal of Approximate Reasoning, 124, 59-65. doi:10.1016/j.ijar.2020.05.008

[18] Sato, Y. (2009). Inconsistency of pair-wise comparison matrix based on geometric mean a comparison to principal eigenvector method. doi:10.13033/isahp.y2009.054

[19] Saaty, T.L. [1980]: The analytic hierarchy process,McGraw-Hill,NewYork.

[20] Chu, A. T., Kalaba, R. E., & Spingarn, K. (1979). A comparison of two methods for determining the weights of belonging to fuzzy sets. Journal of Optimization Theory and Applications, 27(4), 531-538. doi:10.1007/bf00933438

[21] Crawford, G., & Williams, C. (1985). A note on the analysis of subjective judgment matrices. Journal of Mathematical Psychology, 29(4), 387-405. doi:10.1016/0022-2496(85)90002-1

[22] De Jong, P. (1984). A statistical approach to Saaty's scaling method for priorities. Journal of Mathematical Psychology, 28(4), 467-478. doi:10.1016/0022-2496(84)90013-0

[23] Barzilai, J., Cook, W., & Golany, B. (1987). Consistent weights for judgements matrices of the relative importance of alternatives. Operations Research Letters, 6(3), 131-134. doi:10.1016/0167-6377(87)90026-5

[24] Blankmeyer, E. (1987). Approaches to consistency adjustment. Journal of Optimization Theory and Applications, 54(3), 479–488. https://doi.org/10.1007/bf00940197

[25] Gass, S. I., & Rapcsák, T. (2004). Singular value decomposition in AHP. European Journal of Operational Research, 154(3), 573–584. https://doi.org/10.1016/s0377-2217(02)00755-5

[26] Koczkodaj, W. W. (1993). A new definition of consistency of pairwise comparisons. Mathematical and Computer Modelling, 18(7), 79–84. https://doi.org/10.1016/0895-7177(93)90059-8

[27] Koczkodaj, W. W., Herman, M. W., & Orłowski, M. (1997). Using consistency-driven pairwise comparisons in knowledge-based systems. Proceedings of the Sixth International Conference on Information and Knowledge Management - CIKM '97. doi:10.1145/266714.266867

[28] Luce, R. D., & Edwards, W. (1958). The derivation of subjective scales from just noticeable differences. Psychological Review, 65(4), 222–237. https://doi.org/10.1037/h0039821

[29] Llull, R.: Artifitium electionis personarum (before 1283)

[30] Thurstone, L. L. (1927). A law of comparative judgment. Psychological Review, 34(4), 273–286. https://doi.org/10.1037/h0070288

[31] Koczkodaj, W. W. (2016). Pairwise comparisons rating scale paradox. Lecture Notes in Computer Science, 1-9. doi:10.1007/978-3-662-49619-0_1

[32] Multiple-criteria decision analysis. (2021, January 24). Retrieved February 23, 2021, from https://en.wikipedia.org/wiki/Multiple-criteria_decision_analysis

[33] Bozóki, S. (2008). Solution of the least squares method problem of pairwise comparison matrices. Central European Journal of Operations Research, 16(4), 345–358. https://doi.org/10.1007/s10100-008-0063-1

[34] R Comments. (n.d.). Retrieved April 30, 2021, from https://www.w3schools.com/r/r_comments.asp

A R Source Code

A.1 ui.R

```
library(shiny)
library(shinythemes)
library(DiagrammeR)


fluidPage(
theme = shinytheme("cosmo"),
navbarPage(
title=strong("RConcluder"),
id = "navBar",
collapsible = TRUE,
inverse = TRUE,
position = "fixed-top",
header = tags$style(
".navbar-right {
float: right !important;
}",
"body {padding-top: 55px;}"),


tabPanel(icon("home"),
fluidRow(
tags$img(src="BalanceScale.jpg",width="100%",height="300px")
```

```
),
# General info.
tabPanel(
"Overview",
tags$h1(strong("Scope")),
tags$p(style = "font-size:20px;","Multi-criteria decision making
    which is a sub-discipline of operations research explicitly
    evaluates multiple conflicting criteria in decision making. The
    consistency-driven pairwise comparisons method is a valuable tool
     for solving multi-criteria decision making problems. It is a
    powerful inference tool that could be used for knowledge
    acquisition for the knowledge management system. In fact,
    pairwise comparisons are basics for practically the entire
    science and have been used for projects of national importance.")
    ,
tags$h1(strong("Approach")),
tags$p(style = "font-size:20px;","This project had implemented an
    application basing the consistency-driven pairwise comparisons
    method. This application, RConcluder, is not only a standalone
    system but also could be used as a supplement of any expert or
    knowledge management system."),
tags$h1(strong("Keywords")),
tags$ul(
```

```
tags$li(style = "font-size:20px;",HTML("Multi-criteria Decision
    Making(MCDM)")),
tags$li(style = "font-size:20px;",HTML("Consistency-driven Pairwise
    comparisons Method")),
tags$li(style = "font-size:20px;",HTML("Inconsistency analysis"))
),
tags$h1(strong("GitHub")),
tags$p(style = "font-size:20px;",HTML("Source code is available at <
    a href=\"https://github.com/lillian5120/RConcluder\">https://
    github.com/lillian5120/RConcluder</a>."))
)
),
tabPanel(strong("Model Analysis"),
sidebarPanel(
tags$h4(tags$b("Import your model")),
tags$hr(),
uiOutput("file1"),
tags$hr(),
tags$h4("Do not know how to begin?Start from example data!"),
downloadLink("downloadData", tags$b("Example Data Download")),
# Horizontal line ----
tags$hr(),
actionButton("reset_tree","Reset Tree", icon("refresh"),style="
    margin-right:40px;"),
```

```
tags$hr(),
actionButton("reset_tree_weight","Reset Tree Weight", icon("refresh"
    ),style="margin-right:40px;"),
width = 2
),
mainPanel(
wellPanel(
uiOutput("add_child_ui"),
titlePanel(h4(strong("Tree model"))),
wellPanel(
style = "background: white",
grVizOutput("xx")
)
,uiOutput("show_matrix")
)
)
),
navbarMenu(title=strong("More"),
tabPanel(strong("About"),
tags$h1(strong("Background Info")),
tags$p(style = "font-size:20px;",HTML(" Multi-criteria decision
    making (MCDM) is a handy method to apply to many complex
    decisions. Especially for solving problems that are characterized
     as a choice among alternatives. It has all the characteristics
```

```
    of a helpful decision support tool which are helping us focus on
    what is important, logical and consistent, and easy to use.")),
tags$p(style = "font-size:20px;",HTML(" Pairwise comparisons(PC) is
    widely recognized as a useful and effective method for supporting
     the MCDM process based on subjective judgments. When the
    relative importance of these items can not be rated directly, the
     benefit of using it for building a global ranking from binary
    comparisons become apparent. The consistency-driven pairwise
    comparisons(CDPC) method is a primary technique for MCDM. It
    provides an elegant means by comparing, ranking, quantizing and
    offering methods for the identification and resolution of
    inconsistencies. It is based on an inconsistency index, Koczkodaj
     inconsistency indicator, which is proposed and named after prof.
     W.W. Koczkodaj (in 1993) and its use as a validation technique."
    )),
tags$p(style = "font-size:20px;",HTML(" Hierarchical pairwise
    comparisons belong to the MCDM. A hierarchical structure reduces
    complexity (in terms of a number of pairwise comparisons) from $O
    (n^2)$ to close to $n*log(n)$, which is a major step forward.
    Users should be prompted to enter entity names and asked to group
     them in not more than 8.")),
),
tabPanel(strong("Version"),style = "font-size:20px;bold","Current
    Version is V 0.2.")
```

```
)
)
)
```

A.2 server.R

```
library(shiny)
library(data.tree)
library(DiagrammeR)
library(shinyMatrix)
library(dplyr)
library(kableExtra)
library(readr)
library(ggplot2)


function(input, output,session){
#Create reative value to app
vv=reactiveValues(org=NULL,names=NULL,weight=NULL)


########## ExampleData downloading ##########
exampleData <- data.frame(
from = c(
"WHOQOL", "WHOQOL", "WHOQOL", "WHOQOL", "WHOQOL", "SR", "SR", "SR"
),
to = c(
"PH", "SR", "ENV1", "ENV2", "PSYCH", "p_rel", "social","sex"
),
weight = c(
31.57, 18.8, 19.87, 19.87, 9.93, 10.96, 4.35, 3.45
```

```
)
)

output$downloadData <- downloadHandler(
filename = function() {
paste("ExampleData-", Sys.Date(), ".csv", sep="")
},
content = function(file) {
write_csv(exampleData, file)
}
)

########## initial hierarchical tree ##########
observe({

output$file1 <- renderUI({
input$reset ## Create a dependency with the reset button
fileInput('file1', label = NULL)
})

inFile <- input$file1
if (is.null(inFile)){
#set initial weight for root 100
vv$org <- Node$new("Root",weight=100)
```

```
# get node names of hierarchical tree
vv$names=vv$org$Get('name')
#initial hierarchical tree
output$xx=renderGrViz({
# set the style of tree graph
SetGraphStyle(vv$org, rankdir = "TB")
SetEdgeStyle(vv$org, arrowhead = "vee", color = "grey", penwidth =
    2)
SetNodeStyle(vv$org, style = "filled,rounded", shape = "polygon",
    fillcolor = "pink", fontname = "helvetica", tooltip = getNodeTip,
     label = getNodeLabel)
grViz(DiagrammeR::generate_dot(ToDiagrammeRGraph(vv$org)),engine = "
    dot")
})

}else{
# read csv file
useRdf <- read.csv(inFile$datapath, stringsAsFactors = FALSE)

#convert data frame to Node
useRtree <- FromDataFrameNetwork(useRdf)

#initial the weight of root node to 100
rootNode <- FindNode(useRtree,name=useRtree$root$name)
```

```
rootNode$weight = 100


#generate hierarchical tree
output$xx=renderGrViz({
# set the style of tree graph
SetGraphStyle(useRtree, rankdir = "TB")
SetEdgeStyle(useRtree, arrowhead = "vee", color = "grey", penwidth =
    2)
SetNodeStyle(useRtree, style = "filled,rounded", shape = "polygon",
   fillcolor = "pink", fontname = "helvetica", tooltip = getNodeTip,
    label = getNodeLabel)
grViz(generate_dot(ToDiagrammeRGraph(useRtree)),engine = "dot")
})


vv$org <- useRtree
vv$names <- vv$org$Get('name') # get names of main tree


# Clear analysis result, need re-analysis
output$show_matrix <- NULL
}
})


########## initial input UI ##########
output$add_child_ui=renderUI({
```

```
fluidRow(
style = "position:relative",
column(3,
style = "position:relative",
wellPanel(
tags$h4(tags$b("Add Node")),
tags$hr(),
selectInput("Parent_name","Parent_name",vv$names),
div(id='my_textinput', textInput ('new_node_name', label = "New_name
    *", value = "",placeholder = "--Enter name--")),
actionButton("add_child","Add"),
tags$hr(),
textOutput("add_error_msg"),
tags$head(tags$style("#add_error_msg{color: red;
font-size: 16px;
font-style: italic;
}"))
)),

column(
3,
style = "position:relative",
wellPanel(
tags$h4(tags$b("Change Node Name")),
```

```
tags$hr(),
selectInput("Name_to_change","Name_to_change",vv$names),
div(id='my_textinput', textInput ('new_name', label = "New_name*",
    value = "",placeholder = "--Enter name--")),
tags$head(tags$style(type="text/css", "#my_textinput {color: red}"))
    ,
actionButton("Change_name","Change"),
tags$hr(),
textOutput("update_error_msg"),
tags$head(tags$style("#update_error_msg{color: red;
font-size: 16px;
font-style: italic;
}"))
)),


column(3,
style = "position:relative",
wellPanel(
tags$h4(tags$b("Delete Node")),
tags$hr(),
selectInput("Name_to_delete","Name_to_delete",vv$names),
helpText(strong("Attention: The children of this node will be
    removed too!")),
tags$br(),
```

```
actionButton("remove_node","Remove"),
tags$hr()
)),

column(3,
style = "position:relative",
wellPanel(
tags$h4(tags$b("Analysis")),
tags$hr(),
numericInput("eps","Inconsistency Tolerance Level",0.33, min = 0,
    max = 1, step = 0.1),
selectInput("Parent_node_name","Parent_node_name",vv$names),
actionButton("analysis","Analysis"),
tags$hr(),
textOutput("error_msg"),
tags$head(tags$style("#error_msg{color: red;
font-size: 16px;
font-style: italic;
}"))
))
)
})
```

```
########## update hierarchical tree weight function ##########
updateTreeWeights <- function(Tree){
height <- Tree$height
nodeName <- Tree$Get('name',filterFun= function(x) x$level == 1)
rootNode <- FindNode(Tree,name=nodeName)
rootNode$weight <- 100

for(i in 1:height){
nodeName <- Tree$Get('name',filterFun= function(x) x$level == i)
for(j in nodeName){
nodeObj <- FindNode(Tree,name = j)
childrenNum <- length(nodeObj$children)
for(k in nodeObj$children){
k$weight = round(nodeObj$weight/childrenNum,2)
}
}
}
}

########## Get node label function ##########
getNodeLabel <- function(node){
label = paste0(node$name, "\n",node$weight)
return(label)
}
```

```
########## Get node tip function ##########
getNodeTip<- function(node){
tip = node$weight
return(tip)
}

########## Oberserve event for reset hierarchical tree button
    ##########
observeEvent(input$reset_tree,{
vv$org <- Node$new("Root",weight=100) #set initial weight for root
    100
vv$names = vv$org$Get('name') # get names of main tree

#initial hierarchical tree
output$xx=renderGrViz({
grViz(generate_dot(ToDiagrammeRGraph(vv$org)),engine = "dot")
})

# need re-analysis
output$show_matrix <- NULL
})

########## Oberserve event for reset tree weight button ##########
```

```
observeEvent(input$reset_tree_weight,{
updateTreeWeights(vv$org)

#refresh hierarchical tree
output$xx=renderGrViz({
grViz(generate_dot(ToDiagrammeRGraph(vv$org)),engine = "dot")
})

# need re-analysis
output$show_matrix <- NULL

})

########## add child node ##########
observeEvent(input$add_child,{
# valid check for node name
if (input$new_node_name ==""){
output$add_error_msg <- renderText('New node name is empty!')
return(FALSE)
}else if(!is.null(FindNode(vv$org,name=input$new_node_name))){
output$add_error_msg <- renderText('The node name is already exist!'
    )
return(FALSE)
}else{
```

```
output$add_error_msg <- renderText("")
}

FindNode(node=vv$org,name = input$Parent_name)$AddChildNode(Node$new
    (input$new_node_name, weight=input$new_node_weight)) # add child

vv$names=vv$org$Get('name')# get names of new tree

updateTreeWeights(vv$org)

#re-generate hierarchical tree
output$xx=renderGrViz({
grViz(DiagrammeR::generate_dot(ToDiagrammeRGraph(vv$org)),engine = "
    dot")
})
})



########## change hierarchical tree node name ##########
observeEvent(input$Change_name,{
#valid check for name
if (input$new_name ==""){
output$update_error_msg <- renderText('New node name is empty!')
return(FALSE)
```

```
}else if(!is.null(FindNode(vv$org,name=input$new_name))){
output$update_error_msg <- renderText('The node name is already
    exist!')
return(FALSE)
}else{
output$update_error_msg <- renderText("")
}

aa=FindNode(node=vv$org,name = input$Name_to_change)
aa$name=input$new_name # Change name

vv$names=vv$org$Get('name')# get names of new tree

#re-generate hierarchical tree
output$xx=renderGrViz({
grViz(generate_dot(ToDiagrammeRGraph(vv$org)),engine = "dot")
})
})

########## remove hierarchical tree node ##########
observeEvent(input$remove_node,{
#Prune tree
Prune(vv$org, function(x) x$name != input$Name_to_delete)
vv$names=vv$org$Get('name') # get names of new tree
```

```
updateTreeWeights(vv$org)

#re-generate hierarchical tree
output$xx=renderGrViz({
grViz(DiagrammeR::generate_dot(ToDiagrammeRGraph(vv$org)),engine = "
    dot")
})

# need re-analysis
output$show_matrix <- NULL
})

########## Oberserve event for Analysis button ##########
observeEvent(input$analysis,{

eps <- input$eps

aa = FindNode(node=vv$org,name = input$Parent_node_name)

#get the number of children of current node
n =length(aa$children)

if(n < 3){
```

```
errorMsg = "The selected node has to have more than 3 child nodes"
output$error_msg <- renderText(errorMsg)
output$show_matrix <- NULL
return(FALSE)
}else{
errorMsg = ""
output$error_msg <- renderText(errorMsg)
}

childrenLevel <- aa$level+1

nodeList <- aa$Get('weight',filterFun= function(x) x$level ==
    childrenLevel)

#generate reciprecial matrix
genMatrix <- generateMatrix(nodeList)
#show Pairwise Comparisons Matrix section
output$show_matrix=renderUI({
wellPanel(
titlePanel(h4(strong("Pairwise Comparisons Matrix"))),
fluidRow(
style = "position:relative",
column(
12,
```

```
style = "position:relative",
matrixInput(
inputId = "matrix",
value = genMatrix,
class = "numeric",
cols = list(
names = TRUE,
editableNames = TRUE
),
rows = list(
names = TRUE,
editableNames = TRUE
)
),
helpText(strong("Guidance: Click the ratio you want to change in the
    upper triangle part of Pairwise Comparison Marix, and click the
   'Update Value' button!")),
actionButton("update","Update Value",style="margin-top: 1.5%; margin
   -left: 80%; height: 40px;", icon = icon("refresh")),
tags$hr()
)
),
titlePanel(h4(strong("Dashboard"))),
wellPanel(
```

```
fluidRow(
style = "position:relative",
column(
6,
style = "position:relative",
tableOutput("highlight_table"),
h4(textOutput("nTraids")),
tags$hr(),
actionButton("reduce"," Reduce Inconsistency",style="margin-top:
    1.5%; margin-left: 5%; height: 40px;", icon = icon("balance-scale
    ")),
downloadButton("save", "Download Tree Model",style= "margin-top:
    1.5%; margin-left: 17%; height: 40px;")
),
column(
6,
style = "position:relative",
plotOutput("pieChartPlot")
)
)
)
)
})
```

```
df <- findAllTrials(genMatrix)
print(df)
max_kii <- df[1,7]


output$nTraids <- renderText({
eps <- input$eps
if(max_kii > eps){
paste("The max kii value is:", max_kii)
}else{
paste("Congras! you get a consistent PC Matrix with kii value:",max_
    kii)
}
})


output$highlight_table <- function() {
# highlight sign
if(max_kii > eps){
i <- df[1,1]
j <- df[1,2]
k <- df[1,3]
r <- nrow(genMatrix)
genMatrix[(j-1)*r+i] <- cell_spec(genMatrix[i,j], color = "white",
    bold = T,background = "red")
```

```
genMatrix[(k-1)*r+j] <- cell_spec(genMatrix[j,k], color = "white",
    bold = T,background = "red")
genMatrix[(k-1)*r+i] <- cell_spec(genMatrix[i,k], color = "white",
    bold = T,background = "red")
}

genMatrix %>%
kbl(escape = F, align = "c") %>%
kable_styling("striped", full_width = F)
}

#Generate Pie chart

# get the triads number
nTriads <- nrow(df)

# get the triads number which kii larger than threshold
nInconsisTraids <- nrow(subset(df,X7 > eps))
print(nTriads)
print(nInconsisTraids)
output$pieChartPlot <- renderPlot({
df <- data.frame(
group = c("Inconsistent Triads", "Consistent Triads"),
value = c( nInconsisTraids, nTriads-nInconsisTraids)
```

```
)
ggplot(df, aes(x="", y=value, fill=group))+
geom_bar(width = 1, stat = "identity")+
scale_y_continuous(
breaks = function(x) seq(ceiling(x[1]), floor(x[2]), by = 1),
expand = expand_scale(mult = c(0, 0.05))
) +
theme_minimal()
})


})


########## generate PC Matrix for the model ##########
generateMatrix<-function(nodeList){


dim <- 0
weight_elements <- c()
index_names <-names(nodeList)


#generate reciprecial matrix
for(i in index_names){
dim <- dim + 1
weight_elements[dim] <- nodeList[i]
}
```

```
matrix_elements <- c()
index <- 1

for(i in weight_elements){
for(j in weight_elements){
matrix_elements[index] <- round(j/i,2)
index <- index +1
}
}

pc_matrix <- matrix(data = matrix_elements, nrow = dim, ncol = dim)

rownames(pc_matrix) <- index_names
colnames(pc_matrix) <-index_names

return(pc_matrix)
}

########## Oberserve event for Update matrix button ##########
observeEvent(input$update,{
new_matrix <- input$matrix
eps <- input$eps
```

```
#Update reciprocal pairwise comparison matrix
dim <- nrow(new_matrix)
for(r in 1:dim){
for( c in 1:dim){
if(c > r){
new_matrix[c,r] <- round(1/new_matrix[r,c],2)
}
if(c == r){
new_matrix[c,r] <- 1
}
}
}

updateMatrixInput(session,"matrix",new_matrix)

df <- findAllTrials(new_matrix)

max_kii <- df[1,7]

output$nTraids <- renderText({
if(max_kii > eps){
paste("The maximum kii value is:", max_kii)
}else{
```

```
paste("Congras! you get a consistent PC Matrix with kii value:", df
    [1,7])
}
})

output$highlight_table <- function(){
# highlight sign
if(max_kii > eps){
i <- df[1,1]
j <- df[1,2]
k <- df[1,3]
r <- nrow(new_matrix)
new_matrix[(j-1)*r+i] <- cell_spec(new_matrix[i,j], color = "white",
     bold = T,background = "red")
new_matrix[(k-1)*r+j] <- cell_spec(new_matrix[j,k], color = "white",
     bold = T,background = "red")
new_matrix[(k-1)*r+i] <- cell_spec(new_matrix[i,k], color = "white",
     bold = T,background = "red")
}

new_matrix %>%
kbl(escape = F, align = "c") %>%
kable_styling("striped", full_width = F)
}
```

```
#Update Pie chart part


# get the triads number
nTriads <- nrow(df)


# get the triads number which kii larger than threshold
nInconsisTraids <- nrow(subset(df,X7 > eps))
print(nTriads)
print(nInconsisTraids)
output$pieChartPlot <- renderPlot({
df <- data.frame(
group = c("Inconsistent Triads", "Consistent Triads"),
value = c( nInconsisTraids, nTriads-nInconsisTraids)
)
ggplot(df, aes(x="", y=value, fill=group))+
geom_bar(width = 1, stat = "identity")+
scale_y_continuous(
breaks = function(x) seq(ceiling(x[1]), floor(x[2]), by = 1),
expand = expand_scale(mult = c(0, 0.05))
) +
theme_minimal()
})
```

```
})


########## Oberserve event for Reduce inconsistency button
    ##########
observeEvent(input$reduce,{
matrix <- input$matrix
df <- findAllTrials(matrix)
eps <- input$eps

# get the triads which kii larger than threshold
traids <- subset(df,X7 > eps)

N <- nrow(traids)

while(N !=0){

i <- df[1,1]
j <- df[1,2]
k <- df[1,3]

X <- df[1,4] #a_ij
Y <- df[1,5] #a_ik
Z <- df[1,6] #a_jk
```

```
#improve the value
new_x <- X^(2/3)*Z^(-1/3)*Y^(1/3)
new_y <- X^(1/3)*Z^(1/3)*Y^(2/3)
new_z <- X^(-1/3)*Z^(2/3)*Y^(1/3)



#update matrix using new_x,y,z
matrix[i,j] <- round(new_x,2)
matrix[i,k] <- round(new_y,2)
matrix[j,k] <- round(new_z,2)

matrix[j,i] <- round(1/new_x,2)
matrix[k,i] <- round(1/new_y,2)
matrix[k,j] <- round(1/new_z,2)



#recalculate the kii of all traids and sort by kii desc
df <- findAllTrials(matrix)
#get the number of kii value is larger than eps
subFrame <- subset(df,X7 > eps)
N <- nrow(subFrame)
paste("new N value is: ",N)
}
```

```
updateMatrixInput(session,"matrix",matrix)


output$nTraids <- renderText({
paste("Congras! you get a consistent PC Matrix with kii value:", df
    [1,7])
})


output$highlight_table <- function() {
matrix %>%
kbl(escape = F, align = "c") %>%
kable_styling("striped", full_width = F)
}



#update the tree
colIDs = rownames(matrix)
#get node by first element name
node = FindNode(node=vv$org,name = colIDs[1])
#pNodeName = node$parent$name
pNodeWeight = node$parent$weight


#initial element ratio array
eRatio <- c()
```

```
# element ratio array index
r <- 1


#initial element sum number
sum <- 0


#Get the elements in the first row of PC matrix
for(j in matrix[1,]){
eRatio[r] <- 1/j
r <- r+1
sum <- sum + 1/j
}


newWeightArrary <- round((pNodeWeight/sum)*eRatio,2)


output$pieChartPlot2 <- renderPlot({
df <- data.frame(
group = colIDs,
value = newWeightArrary
)


ggplot(df, aes(x="", y=value, fill=group))+
geom_bar(width = 1, stat = "identity")+
coord_polar("y", start=0) +
```

```
theme_minimal()
})

#Calculate the weight for each node
m=1
for(i in colIDs){
aa = FindNode(node=vv$org,name = i)
oldWeight = aa$weight
aa$weight = newWeightArrary[m]

#if current node has children
childrenNum <- length(aa$children)

if(childrenNum != 0){
#assign weight by children's ratio
for(k in aa$children ){
k$weight = round(aa$weight/oldWeight*k$weight,2)
}
}
m=m+1
}

#re-generate tree
output$xx=renderGrViz({
```

```
grViz(DiagrammeR::generate_dot(ToDiagrammeRGraph(vv$org)),engine = "
    dot")
})

#update Pie chartpart
#get the triads number
nTriads <- nrow(df)
#get the triads number which kii larger than threshold
nInconsisTraids <- nrow(subset(df,X7 > eps))

output$pieChartPlot <- renderPlot({
df <- data.frame(
group = c("Inconsistent Triads", "Consistent Triads"),
value = c( nInconsisTraids, nTriads-nInconsisTraids)
)
ggplot(df, aes(x="", y=value, fill=group))+
geom_bar(width = 1, stat = "identity")+
scale_y_continuous(
breaks = function(x) seq(ceiling(x[1]), floor(x[2]), by = 1),
expand = expand_scale(mult = c(0, 0.05))
) +
theme_minimal()
})
```

```
})


########## Find all trials in a PC matrix function ##########
findAllTrials <- function(matrix){
#get the number of rows in matrixs
row <- nrow(matrix)
templist <- list()
for(i in 1:(row-1)){
for(j in (i+1):row){
X<- matrix[i,j]  # a_ij
if(j < row){
for (k in (j+1):row){
Z <- matrix[j,k] # a_jk
Y <- matrix[i,k] # a_ik
kii <- round(1 - min(Y/(X*Z),(X*Z)/Y),4)
templist <- c(templist,list(c(i,j,k,X,Y,Z,kii)))
}
}
}
}
tempmatr <- data.frame(do.call(rbind,templist))
orderFrame <- tempmatr[order(-tempmatr$X7),]
}
```

```
########## Save model to CSV file ##########
output$save <- downloadHandler(
filename = function() {
paste(getwd(), Sys.Date(), '.csv',sep = '')
},
content = function(con) {
data3 <- ToDataFrameNetwork(vv$org, "weight")
write.csv(data3, file = file.path(con),row.names=FALSE)
}
)
}
```