Automating the Generation of Comparison Weights for Enhancing the AHP Decision-Making Process

Karim Zarour[1,2], Djamel Benmerzoug[1], Nawal Guermouche[2], and Khalil Drira[2]

[1] LIRE Laboratory, University of Constantine 2 - Abdelhamid Mehri, Constantine, Algeria
[2] LAAS-CNRS, University of Toulouse, Toulouse, France
{zarour.karim, djamel.benmerzoug}@univ-constantine2.dz
{nguermou, khalil}@laas.fr

Abstract. The Analytic Hierarchy Process (AHP) method is widely used to deal with multi-criteria decision-making problems thanks to its simplicity and flexibility. However, it is often criticized for subjectivity and inconsistency in assigning the comparison weights that are based on expert judgments. In order to remedy these shortcomings, we propose in this paper an algorithm that automatically generates the pairwise comparison weights of alternatives according to each considered criterion. In addition, we demonstrate through an example that the judgment matrices constructed by the algorithm are very consistent.

Keywords: Analytic Hierarchy Process, Multi-criteria decision-making, Automatic generation of comparison weights, Consistency ratio, Subjectivity.

1 Introduction

Real-world decision-making problems are becoming increasingly complex due to the large number of alternatives, heightened uncertainty, shorter deadlines, greater pressure, environment dynamicity, etc. [1]. Several multi-criteria decision-making (MCDM) methods have been proposed such as TOPSIS, and ELECTRE but the most used and popular one is the Analytic Hierarchy Process (AHP) method that attracts decision-makers by its simplicity and flexibility [4, 6, 9]. Despite its advantages, the AHP method is often criticized for subjectivity and inconsistency in assigning comparison weights. Indeed, the assigned weights must be readjusted until the Consistency Ratio (CR) defined by [7] is equal to or less than 10%. In addition, experts may be asked several times for pairwise comparisons, which is not practical, time-consuming, and very annoying when the number of criteria or alternative is high.

In this paper, we propose an algorithm that automatically generates the pairwise comparison weights of alternatives and fills the corresponding judgment matrices by taking as input the real values of alternatives according to a set of criteria. The main objective of the algorithm is to dispense with the tedious task of assigning weights, which is usually done manually and therefore the decision-making process time could be significantly reduced. In addition to the automation, we demonstrate through an example on hosting offer selection that the proposed algorithm constructs very consistent judgment matrices.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_53

The remainder of this paper is organized as follows. Section 2 overviews the AHP method and its limitations. Some related work are addressed in Section 3. Section 4 presents and explains the proposed algorithm. Section 5 demonstrates the usefulness of the algorithm through an illustrative example. Finally, Section 6 concludes this paper and provides directions for future work.

2 AHP limitations

The AHP is a method developed by T. Saaty [8] for resolving decision problem with multiple criteria through four major steps [7, 8]. Beyond its several advantages like the hierarchical structuring of decision problems, the AHP method relies on the experience and judgments of experts who give weights value directly. In this assessment way, the weights may be attributed with prejudices and the results may have subjectivity. Moreover, the influence of human factor in AHP method can lead to wrong decisions. Another issue may arise when experts need to be asked several times for pairwise comparisons. For instance, a decision-making problem of 4 criteria and 3 alternatives requires to ask experts 18 times. We find that this is not practical and may become very annoying when the number of criteria or alternative is high. Furthermore, the AHP method is supported by the Expert Choice tool but the comparison weights are filled manually.

3 Related work

There exists many work in the literature that improve the AHP method but most of them focus on the consistency of judgment matrices on the grounds that it is the main weakness of the AHP method. For example, Lin *et al.* [6] develop an adaptive AHP approach that uses a genetic algorithm to recover the relative importance weights of the considered criteria. Benitez *et al.* [1] provide an optimization method for improving consistency based on the minimization of the distance between each two judgment matrices. Khatwani and Kar [5] propose an algorithm that can adjust the entries of judgment matrices iteratively until reaching a desired level of consistency.

Certain researchers also aim to improve AHP in group decision-making. For instance, two consensus models have been defined in [2] for group decision-making by using the row geometric mean prioritization method. Huang *et al.* [4] demonstrate that the efficiency of AHP can be significantly enhanced via an optimal expert allocation. However, although consulting several experts for evaluating criteria and alternatives may reduce the bias of personal subjectivity, it requires further calculations and a longer decision process. Furthermore, few work have been realized for improving the AHP method in other aspects. For example, Xiulin and Dawei [9] aim to simplify the calculations needed to construct the judgment matrices by adopting a scale of only three values instead of nine values. In [3], the authors present and test a model based on Multi-layer Perceptron (MLP) neural networks that is capable of completing missing values in AHP judgment matrices. However, to the best of our knowledge, there is no work in the literature that automates the generation of pairwise comparison weights in the AHP method.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_53

4 An automatic comparison of alternatives

To overcome the AHP limitations tackled previously, we propose an algorithm (Algorithm 1) that automatically generates the pairwise comparison weights and fills the judgment matrix corresponding to each criterion. The algorithm takes as input a matrix 'TabVal' containing the real values of alternatives according to a set of criteria. The structure of input and output matrices is presented in Figure 1.

Algorithm 1: Automatic filling of alternative comparison matrices.
Declaration
TabComp : a square matrix [(N+1) × (N+1)] of real numbers; /* *an empty comparison matrix, N is the number of alternatives to compare* */
AltComp : { TabComp }; /* *a set of comparison matrices (a TabComp for each criterion$_k$)* */
TabVal : a matrix [(M+1) × (N+1)] of real numbers; /* *TabVal is a matrix containing the values of alternatives according to each criterion* */
/* *TabComp and TabVal have the same number of columns N (the number of compared alternatives), M is the number of considered criteria.* */
i, j, k : integer;
Begin
Scalling (TabVal); /* *Scaling the real values of alternatives* */
AltComp ← { };
For (k=1 ; k ≤ M; k++) Do { /* *for each criterion$_k$* */
TabComp$_k$ = New TabCom (); /* *a new comparison matrix for each criterion$_k$* */
For (i=1 ; i ≤ N ; i++) Do {
TabComp$_k$(alt$_i$, alt$_i$) ← 1; /* *fill in the matrix diagonal with 1* */
}
i ← 1; j← 2;
If (Criterion$_k$ IS Criterion to be minimized) Then {
While (i ≤ N-1) Do { /* *browse TabVal* */
While (j ≤ N) Do { /* *browse TabVal* */
Compare (TabVal(crit$_k$, alt$_i$), TabVal(crit$_k$, alt$_j$)); /* *this procedure compares 2 alternatives and inserts the comparison weight into TabComp$_k$* */
j ← j +1;
}
i ← i +1; j ← i +1;
}
}
Else { /* *Criterion$_k$ is to be maximized* */
While (i ≤ N-1) Do { /* *browse TabVal* */
While (j ≤ N) Do { /* *browse TabVal* */
Compare (TabVal(crit$_k$, alt$_j$), TabVal(crit$_k$, alt$_i$));
j ← j +1;
}
i ← i +1; j ← i +1;
}
}
AltComp ← AltComp ∪ { TabComp$_k$ }; /* *add to the result the comparison matrix of alternatives for the criterion$_k$* */
}
Return (AltComp);
End ;

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_53

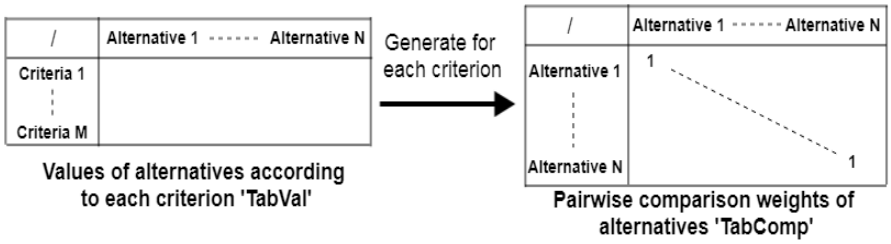

Fig. 1. Automatic generation of pairwise comparison matrices

Besides the automation, we will also show through the example of the next section that the proposed algorithm generates consistent judgment matrices (i.e. the consistency ratio is less than or equal to 10%). Indeed, judgment matrix consistency is one of the most challenging problems when using AHP [2, 9].

In order to respect the Saaty's scale [8], we have integrated in the beginning of the algorithm a procedure called 'Scalling', which allows to put the real values on a scale of numbers ranging from 0 to 8. After creating an alternative comparison table and filling its diagonal with '1', the 'Compare' procedure is iteratively called within two nested loops. This procedure takes as parameter two values characterizing two different alternatives according to a given criterion. Next, it generates the two comparison weights (one weight to compare Alternative i and Alternative j and another for the opposite) and inserts them in the 'TabCom' comparison matrix corresponding to the criterion in question. The instructions of 'Scaling' and 'Compare' procedures are described in the following:

Procedure Scaling (TabVal: Matrix [(M+1) × (N+1)] of real numbers)
Declaration MaxValue: float; i, k: integer;
Begin
For (k=1 ; k ≤ M; k++) Do { */* for each criterion$_k$ */*
MaxValue ← TabVal(crit$_k$, alt$_1$);
For (i=2; i ≤ N; i++){ */* search for the maximum value of alternatives according to criterion$_k$ */*
If (TabVal(crit$_k$, alt$_i$) > MaxValue) Then {
MaxValue ← TabVal(crit$_k$, alt$_i$);
}
}
For (i=1 ; i ≤ N; i++) {
TabVal(crit k, alt i) ← $\frac{\text{TabVal(crit k,alt i)} * 8}{MaxValue}$; */* scaling each value */*
}
}
End;

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_53

Procedure Compare (X: float, Y: float)
Declaration difference : float;
Begin
difference ← X - Y;
If (difference ≥ 0) Then {
TabComp(alt_i, alt_j) ← $\frac{1}{difference+1}$; TabComp(alt_j, alt_i) ← $difference + 1$;
}
Else {
TabComp(alt_i, alt_j) ← $|difference| + 1$; TabComp(alt_j, alt_i) ← $\frac{1}{|difference|+1}$;
}
End;

5 Illustrative example

In order to illustrate the application of our algorithm, we suppose a company that wants to host its online booking platform and it hesitates between six dedicated servers. Hence, the company decided to apply the AHP method in order to choose the hosting offer that best meets its requirements. The characteristics of the dedicated servers proposed by each online hosting offer that the company faces are represented in table 1 (values that are at the top of cells).

Table 1. Values of alternatives according to each criterion

Offer name / Criteria	Magic Epsilone	OVH EG-32	Magic Delta	Bluehost Prenium	eStructure Intel Xeon	Serverroom Intel E5-2630L	Hivelocity Skylake
CPU frequency (Ghz)	2.6	3.8	3.06	2.5	3.3	2.0	3.4
	5.5	8	6.4	5.2	6.9	4.2	7.2
	0.08	0.3	0.13	0.07	0.17	0.05	0.2
CPU cores	8	4	12	4	4	6	4
	(5.3)	(2.7)	(8)	(2.7)	(2.7)	4	(2.7)
	0.2	0.06	0.44	0.06	0.06	0.12	0.06
RAM (GB)	128	32	96	16	32	32	64
	8	2	6	(1)	2)	2)	(4)
	0.43	0.05	0.24	0.03	0.06	0.06	0.13
Storage (TB)	2	8	2	1	1	0.48	0.96
	(2)	(8)	(2)	(1)	(1)	(0.5)	(1)
	0.11	0.53	0.11	0.06	0.06	0.07	0.06
Bandwidth (Mbps)	150	500	150	200	100	300	1024
	(1.2)	(3.9)	(1.2)	(1.6)	(0.8)	(2.3)	(8)
	0.06	0.18	0.06	0.07	0.04	0.1	0.49
Price (€/month)	199	99.99	129	89	137	129	145
	(8)	(4)	(5.2)	(3.6)	(5.6)	(5.2)	(5.8)
	0.03	0.24	0.13	0.29	0.1	0.13	0.08

In what follows, we describe how our algorithm intervenes in the AHP method for solving this MCDM problem after structuring it hierarchically.

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_53

5.1 Comparison of alternatives

Before comparing alternatives, the algorithm scales their real values and the values of alternatives after scaling are represented in the middle in table 1. Once the real values are scaled, the algorithm generates a comparison matrix of alternatives according to each criterion. Table 2 represents the comparison matrices generated in our example for the criterion of CPU frequency (the absolute weights are represented without parentheses and the relative weights in parentheses) and it is the same principle for the other five criteria.

Table 2. The pairwise comparisons generated according to the CPU frequency

	Magic Epsilone	**OVH EG-32**	**Magic Delta**	**Bluehost**	**eStruxture**	**Serverroom**	**Hivelocity**	**Average**
Magic Epsilone	1 (0.08)	1/3.5 (0.09)	1/1.9 (0.06)	1.3 (0.09)	1/2.4 (0.07)	2.3 (0.11)	1/2.7 (0.07)	8%
OVH EG-32	3.5 (0.28)	1 (0.32)	2.6 (0.32)	3.8 (0.26)	2.1 (0.34)	4.8 (0.23)	1.8 (0.35)	30%
Magic Delta	1.9 (0.15)	1/2.6 (0.12)	1 (0.12)	2.2 (0.15)	1/1.5 (0.11)	3.2 (0.15)	1/1.8 (0.11)	13%
Bluehost	1/1.3 (0.06)	1/3.8 (0.08)	1/2.2 (0.06)	1 (0.07)	1/2.7 (0.06)	2 (0.10)	1/3 (0.07)	7%
eStruxture	2.4 (0.19)	1/2.1 (0.15)	1.5 (0.18)	2.7 (0.19)	1 (0.16)	3.7 (0.18)	1/1.3 (0.15)	17%
Serverroom	1/2.3 (0.03)	1/4.8 (0.07)	1/3.2 (0.04)	½ (0.03)	1/3.7 (0.04)	1 (0.05)	¼ (0.05)	5%
Hivelocity	2.7 (0.21)	1/1.8 (0.18)	1.8 (0.22)	3 (0.21)	1.3 (0.21)	4 (0.19)	1 (0.20)	20%

5.2 Verification of the comparison consistency

We have used the Expert Choice tool to check the consistency of the pairwise comparisons and we observe in figure 2 that excepted the storage criterion, the consistency ratio (CR) is fairly stable and very far from the acceptable limit of 10%. This proves that our algorithm generates very consistent comparison matrices. Furthermore, we have verified through tests that the algorithm generates consistent comparison matrices (the CR varies between 1% and 3%) whatever the number of alternatives.

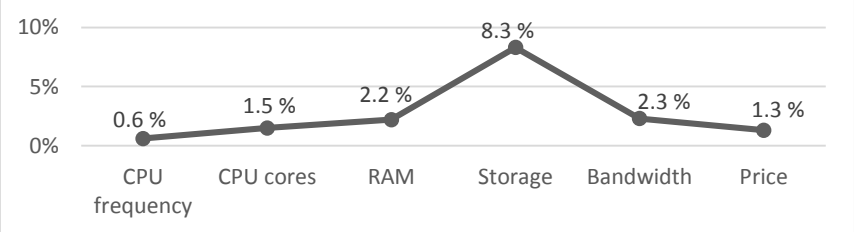

Fig. 2. The consistency ratio corresponding to each alternative comparison matrix

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_53

The last step consists calculating the final score of each alternative by multiplying the final values of alternatives (shown at the bottom of cells in table 1) by the final weights of criteria that depend company's requirements.

6 Conclusion and future work

In order to enhance the AHP method, we have proposed in this paper an algorithm that automatically generates the pairwise comparison weights of alternatives and construct a judgment matrix for each considered criterion. The algorithm takes as input the real values of alternatives according to a set of criteria and puts them on a scale of eight values so that the generated weights respect the Saaty's scale. The applicability and usefulness of our algorithm has been demonstrated through an example on hosting offer selection. In addition to the automation, the illustrative example has also shown that the judgment matrices generated by our algorithm are very consistent.
For future work, we envisage to integrate into the proposed algorithm a procedure that quantify qualitative criteria. We also want to incorporate new methods or reuse existing ones that allow readjusting the comparison weights of judgment matrices in order to further reduce the consistency ratio. Moreover, it would also be interesting to compare the obtained consistency ratios with those of similar work. Finally, we intend to extend the Expert Choice software with the proposed algorithm through a plugin.

References

1. Benitez, J., Delgado-Galvan, X., Izquierdo, J., Pérez-Garcia, R.: Improving consistency in AHP decision-making processes. Applied Mathematics and Computation 219(5), 2432–2441 (2012).
2. Dong, Y., Zhang, G., Hong, W., Xu, Y.: Consensus models for AHP group decision making under row geometric mean prioritization method. Decision support systems 49(3), 281–289 (2010).
3. Gomez-Ruiz, J.A., Karanik, M., Pelaez, J.I.: Improving the Consistency of AHP Matrices Using a Multi-layer Perceptron-Based Model. In: Proceedings of the 9th International Work-Conference on Artificial Neural Networks, pp. 41–48. Springer (2009).
4. Huang, E., Zhang, S., Lee, L., Chew, E., Chen, C.: Improving Analytic Hierarchy Process Expert Allocation Using Optimal Computing Budget Allocation. IEEE Transactions on Systems, Man, and Cybernetics: Systems 46(8), 1140–1147 (2016).
5. Khatwani, G., Kar, A.K.: Improving the Cosine Consistency Index for the analytic hierarchy process for solving multi-criteria decision-making problems. Applied Computing and Informatics 13(2), 118–129 (2017).
6. Lin, C., Wang, W., Yu, W.: Improving AHP for construction with an adaptive AHP approach (A3). Automation in Construction 17(2), 180–187 (2008).
7. Saaty, T.L.: Decision making with the analytic hierarchy process. International Journal of Services Sciences 1(1), 83–98 (2008).
8. Saaty, T.L.: The Analytic Hierarchy Process. McGraw-Hill. New York (1980).
9. Xiulin, S., Dawei, L.: An Improvement Analytic Hierarchy Process and its Application in Teacher Evaluation. In: Proceedings of the 5th International Conference on Intelligent Systems Design and Engineering Applications, pp. 169–172. IEEE (2014).

ICCS Camera Ready Version 2019
To cite this paper please use the final published version:
DOI: 10.1007/978-3-030-22750-0_53