Generation of Creative Game Scene Patterns by the Neutrosophic Genetic CoCoSo Method

Aurimas PETROVAS[1], Romualdas BAUSYS[1],
Edmundas Kazimieras ZAVADSKAS[2], Florentin SMARANDACHE[3]

[1] Vilnius Gediminas Technical University, Department of Graphical Systems,
11 Saulėtekio Avenue, Vilnius, 10223, Lithuania
aurimas.petrovas@vilniustech.lt, romualdas.bausys@vilniustech.lt

[2] Vilnius Gediminas Technical University, Institute of Sustainable Construction,
11 Saulėtekio Avenue, Vilnius, 10223, Lithuania
edmundas.zavadskas@vilniustech.lt (**Corresponding author*)

[3] University of New Mexico, Department of Mathematics, 705 Gurley Avenue, Gallup, NM 87301, USA
fsmarandache@gmail.com

Abstract: Creative procedural generation requires, by its nature, a nondeterministic result. Each set of outputs has to satisfy the required criteria and, at the same time, has to show unique results to be considered creative. One of the keys to increasing nondeterminism is to use fuzzy neutrosophic sets. A combination of a CoCoSo multi-criteria decision algorithm with a genetic algorithm to generate game scenes was proposed. The algorithm satisfies the required game rules and improves the aesthetics value of the visual game scene. The proposed approach seeks to produce a creative solution which satisfies the replayability principle, when the solution is not optimal in the optimization procedure sense. In this paper, improvements of the CoCoSo method were proposed in order to work with iterative genetic algorithm.

Keywords: MCDM, CoCoSo, Genetic algorithm, Procedural generation, Games, Neutrosophic sets.

1. Introduction

Automation improves the efficiency and quality of people's lives (Filip, 2021). And one of the tasks that is usually executed by humans is the creative/ intelligent problem solving. During the last decade, researchers paid a lot of attention to model creativity problems using strict mathematical approaches. The popularity of algorithmic solutions for creative tasks is increasing, and the traits that define creativity usually involve created value and novelty (Pichot et al., 2022). Creativity is a difficult task to achieve algorithmically, as there are some difficulties in constructing a mathematical model of this problem. It creates difficulties because traditional algorithms are not usually suitable for creative tasks. With the rise of computing power and new approaches meant to tackle the problem of creativity, many new methods are being developed to solve creativity tasks; however, it is still difficult to compete with human creativity in most cases (Franceschelli & Musolesi, 2021). Procedural Content Generation (PCG) is an algorithmic generation method that generates assets or asset composition. The results are usually represented by a computer-generated content that can be exploited at different levels of the game development process. This approach is commonly used to generate creative results and can modify a variety of different elements in video games. One of the common elements used in procedural game content generation is the layout of game objects. The main motivations for PCG are personalization, replayability, and costs (Togelius et al., 2011). In the research proposed in this paper, the focus is on the replayability side, by creating unique and varied game scenes with the same algorithm. Replayability represents a novelty in the context of creativity. This means that new results differ enough to be considered interesting after more than one instance of the same game level. Replayable content is important for video games if the intention is to use the same PCG algorithm for video game level generation.

There are numerous ways to procedurally generate content, and for the expansion of the variety of levels, the genetic algorithms (Herrmann, 1999) with an abstract fitness function can be used to increase the uniqueness aspect of the generated game level scene as shown in the related work for puzzle (Pereira et al., 2016; Thakkar et al., 2019) or arcade genre games (Safak et al., 2016). Unique content can be considered replayable because each instance of the generated level varies enough for it to still be interesting. One of the key aspects of the genetically created levels is the fitness function. The fitness functions determine the direction of evolution of the proposed task. The problem considered in this article is an optimization one, in which one tries to reach nondeterministic results, while keeping it functional. It needs to be vague to reach a sufficiently large numbers of satisfying results. At the same time, a criterion needs to be defined for the generated level. Usually, this criterion is divided into two categories: functional and aesthetic. Functional

https://doi.org/10.24846/v31i4y202201

criteria define the rules and limitations of game design. Game design rules can be optimized, and limitations always have to be true. Aesthetic criteria define how appealing the generated levels or objects look (Statham et al., 2022). With this mathematical model, all the criteria have to be combined into a single fitness number and, for reaching this goal, multi-criteria decision-making algorithms can be used (MCDM) (Zavadskas et al., 2014). Nowadays, many modern MCDM algorithms use fuzzy sets, enabling nondeterministic results and increasing replayability of the generated game level. There are many examples of applications of fuzzy MCDM methods employed in the research industry, but they are mostly used with a static environment and a single iteration. Some examples of these method applications are solving communication (Peng & Li, 2021; Zhao et al., 2021), networking (Peng & Garg, 2021), risk and failure analysis (Yazdani et al., 2021; Yousefi et al., 2021), logistics (Svadlenka et al., 2020; Kieu et al., 2021), and pathfinding (Semenas et al., 2021) problems. By its nature, the genetic algorithm requires more than one subsequent iteration of the algorithm to converge properly. This research proposes a new strategy that enables MCDM for a set of subsequent iterations, explores the fuzzy limitations of MCDM, and updates the general algorithms to be suitable for the creative PCG task. Nondeterministic creativity problems can be modelled by different versions of modern fuzzy sets.

Fuzzy set theory was established almost 60 years ago (Zadeh, 1965), but successful applications of it started to emerge in the 1980s. Fuzzy set theory is evolving, and new ways to redefine and update fuzzy set theory are constantly evolving (Kahraman et al., 2016, Wu et al., 2021). Some examples of applications include software selection problems, intuitionistic linguistic aggregation, human resources management, smartphone selection, and others (Haque et al., 2020). There are different types and applications of fuzzy sets, the neutrosophic set being the one which can increase nondeterminism. They consist of three numbers which define three neutrosophic set values (truth, indeterminacy, falsity) (Broumi et al., 2018; Smarandache, 1999). All three of these members should be independent. The neutrosophy, by its nature, deals with neutralities and their interaction with the ideational spectra (Zavadskas et al., 2020). In this research, the neutrosophic set environment of the CoCoSo method (Yazdani et al., 2018) was applied to calculate the final fitness score for each chromosome in the genetic algorithm.

This paper proposes a new extension of the neutrosophic CoCoSo method that is suitable for the genetic scene layout generator in which the fitness function is calculated via this method. To solve this content generation problem, adaptation aspects of the CoCoSo method were investigated. Normalization is usually calculated in the local Min-Max range, which includes a single iteration of the algorithm, so, it needs to be updated with global Min-Max values, including all possible value ranges (Choi & Moon, 2003). For this reason, the range can be normalized in each criterion to fit into the normalized value range. Another aspect taken into consideration is the speed of the algorithm. The genetic algorithm calculates the fitness function by using the CoCoSo method, in each generation, for each chromosome, so it can be quite taxing to perform repeated calculations. Some values that are in a short range and have little impact on the result can be changed to constant values. A conversion of linear crisp number to neutrosophic set will also be needed, as the slightest nonproportional change can skew the results in unsuitable ways and be higher than mutation differences, which can numb the evolution (Herrmann, 1999).

This paper proposes methods that can solve the problems discussed. The structure of this paper is established as follows. Section 2 will explain the fitness calculation by using the CoCoSo method, while section 3 will explain the CoCoSo modifications and implementation in the genetic game-level scene generator. Section 4 will present a study case and the results obtained for the replayable PCG case. The last section provides the conclusion of the proposed research.

2. Classical CoCoSo Method

The following steps explain how the general CoCoSo method works. The alternatives correspond to the population of the genetic algorithm, and the individual chromosome criterion corresponds to the criterion of the CoCoSo method.

The initial decision-making matrix is determined as follows (1), where $i = 1,2 \ldots, m; j = 1,2, \ldots n.$

$$X_{ij} = \begin{bmatrix} x_{11} & x_{12} & \cdots & x_{1n} \\ x_{21} & x_{22} & \cdots & x_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ x_{m1} & x_{m2} & \cdots & x_{mn} \end{bmatrix} \quad (1)$$

The normalization of the values for the benefit criterion (2) and for the cost criterion (3) are determined as follows:

$$r_{ij} = \frac{x_{ij} - \min\limits_{i} x_{ij}}{\max\limits_{i} x_{ij} - \min\limits_{i} x_{ij}} \tag{2}$$

$$r_{ij} = \frac{\max\limits_{i} x_{ij} - x_{ij}}{\max\limits_{i} x_{ij} - \min\limits_{i} x_{ij}} \tag{3}$$

An alternative sum of the weighted comparability (4) and the amount of the power weight of the comparability sequences (5) are calculated next.

$$R_i = \sum_{j=1}^{n} (w_j r_{ij}) \tag{4}$$

$$P_i = \sum_{j=1}^{n} (r_{ij})^{w_j} \tag{5}$$

Three appraisal score strategies are used to generate relative weights. k_{ia} expresses the arithmetic mean of the sums of the WSM and WPM scores (6). k_{ib} expresses the sum of the relative scores of WSM and WPM compared to the best (7), and k_{ic} releases the balanced compromise of the WSM and WPM model scores (8). λ is a manually selected constant which should be between 0 and 1, and defines the weight between R and P. λ is defined as 0.5.

$$k_{ia} = \frac{R_i + P_i}{\sum_{i=1}^{m} (R_i + P_i)} \tag{6}$$

$$K_{ib} = \frac{R_i}{\min\limits_{i} R_i} + \frac{P_i}{\min\limits_{i} P_i} \tag{7}$$

$$K_{ic} = \frac{\lambda(R_i) + (1-\lambda)(P_i)}{\left(\lambda \max\limits_{i} R_i + (1-\lambda) \max\limits_{i} P_i\right)} \tag{8}$$

Finally, the ranking of alternatives is determined (9):

$$k_i = (k_{ia} k_{ib} k_{ic})^{\frac{1}{3}} + \frac{1}{3}(k_{ia} + k_{ib} + k_{ic}) \tag{9}$$

3. Neutrosophic Genetic CoCoSo Procedural Game Scene Generator

CoCoSo is used to calculate the final fitness score for each chromosome in a genetic algorithm generation. For the decision matrix, the criterion is used of each chromosome.

```
------------------------------------------------------------
InitializeRandomPopulation:
DoFullEvolution:
  for amountOfEvolutionCycles
    CalculateAllCriteria
    FindUnderperformersAndPerformers
      for populationSize
        calculateFitness
          CoCoSo method
EvolveUnderperformersWithGeneticAlgorithm
DrawGrid(best fitness):
------------------------------------------------------------
```

The core of the procedural generator is a genetic algorithm. The original MCDM alternative (i) and the criteria matrix (j) are transformed into a three-dimensional grid. The alternatives in the MCDM algorithm correspond to the population and chromosomes (i) in the genetic algorithm and the third dimension adds generations (j) in the time layer (see Figure 1).

After the fixed number of generations, the algorithm constructs the layouts of the game levels. It initializes random-level layouts and then develops them. Criteria scores are calculated and normalized to fit in the range of 0.1 to 0.9.

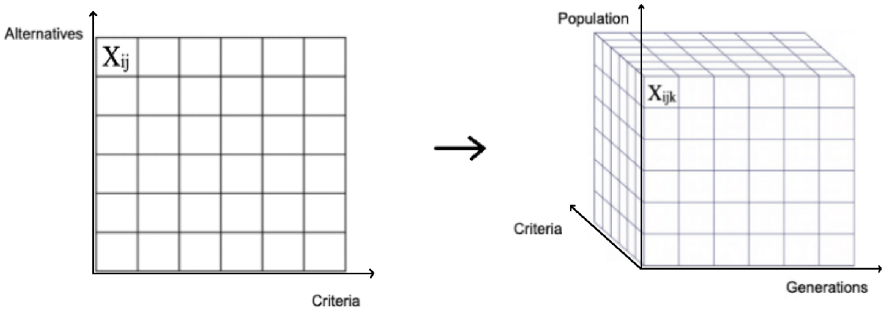

Figure 1. Grid transformation

In order to obtain a more reliable result for the MCDM algorithm, one avoids getting close to 0 and 1 values. The proposed set of criteria consists of 7 criteria. The first part of them are validation criteria, which check for the existence of a player, of an exit and of a path between them. The second part consists of optimizable criteria, which are symmetry, empty space balance, player exit distance, and safe zone calculation. After that, a final fitness score is calculated using the CoCoSo method and genetic operators are applied to the population. For each generation, the fitness calculation is repeated.

The linear conversion of crisp numbers to neutrosophic numbers is used, so the nonlinear differences would not compound over separate generations (10). It can skew the result as the modification can be higher than the evolved difference in a single generation.

$$N(t,i,f)=\begin{cases}C\\1-C\\1-C\end{cases} \tag{10}$$

Normalization is updated to use the smallest possible criterion values across all generations instead of the single ones generated across a single generation.

Beforehand, every criterion is manually normalised to fit into the 0-1 range, in order to achieve this goal, as the future generation values cannot be predicted and they have to be globally normalised for each generation (11).

$$0 < X_{ik} < 1 \tag{11}$$

The general normalization can also be skipped, as each criterion is normalized separately. This update also allows skipping the separate storage of data sets from neutrosophic sets in memory and incrementally adding and multiplying them to generate R_i (12) and P_i (13) values, without storing each of the single elements, where l is the number of generations for the genetic algorithm and n is the population size. Values r_{ij} are replaced by the values N_{ij} of the neutrosophic sets.

$$R_i=\sum_{j=1}^{nl}(w_j(N_{ij}) \tag{12}$$

$$P_i=\sum_{j=1}^{nl}(N_{ij})^{w_j} \tag{13}$$

It also allows the creation of an approximate constant A (14) for the divisor (15) of the *kia* (16) formula, rather than calculating a sum for each chromosome of the generation. It dramatically improves the performance of the iterative algorithm without visible loss of quality.

$$A=\frac{\sum_{i=1}^{nl}x_i}{nl} \tag{14}$$

$$d=nA \tag{15}$$

$$k_{ia}=\frac{S(R_{ik})+S(P_{ik})}{d} \tag{16}$$

It uses global min (r_1, r_2) and max (p_1, p_2) values of Ri and Pi, which are calculated before revealing future values (17, 18, 19, 20) based on the normalization of the criteria formula, so it would be consistent in interactions between generations (21, 22).

$$r_1=\min_{i\in nl}(R_i) \tag{17}$$

$$p_1=\max_{i\in nl}(R_i) \tag{18}$$

$$r_2=\min_{i\in nl}(P_i) \tag{19}$$

$$p_2=\max_{i\in nl}(P_i) \tag{20}$$

$$k_{ib}=\frac{S(R_{ik})}{r_1}+\frac{S(P_{ik})}{r_2} \tag{21}$$

$$k_{ic}=\frac{\lambda S(R_{ik})+(1-\lambda)S(P_{ik})}{\lambda p_1+(1-\lambda)p_2} \tag{22}$$

Finally, the original k_i formula is used to calculate the final fitness function for a single chromosome.

4. Results

The base used for the algorithm was taken from the original WASPAS procedural scene generator (Petrovas & Bausys, 2022). The CPU used for the calculations has 8-core processor with 2.4 GHz speed. Convergent fitness scores begin to stabilize after about 80–120 generations, and, compared to the WASPAS implementation, the convergence occurs at an early stage. Note that the fitness scores cannot be directly compared, because the CoCoSo method produces fitness values outside the 0-1 range. In the proposed demo, it usually ranges between 4 and 8, and it has a tighter relative value range. The comparison between the fitness curves of CoCoSo and WASPAS methods, in terms of calculation time, can be seen in Figure 2.

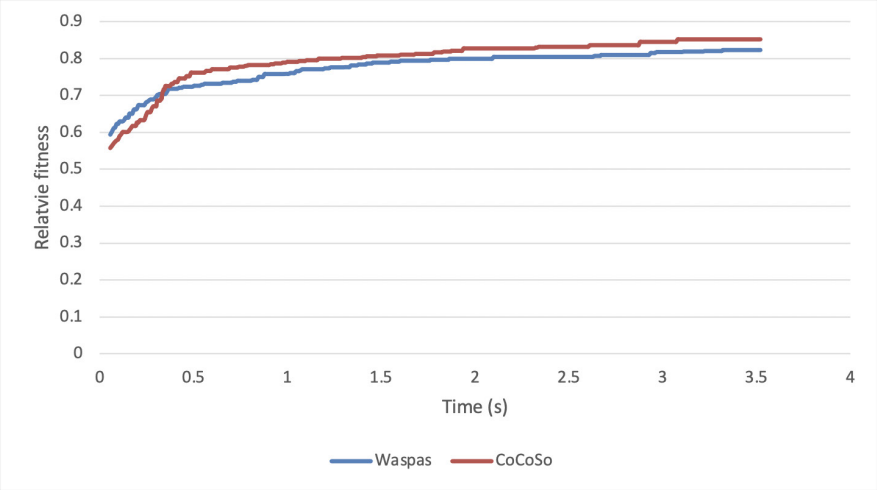

Figure 2. Fitness curve

It takes around 23 seconds for 2000 generations. But around 350 generations are usually enough to have satisfying results with the CoCoSo method, so the increased calculation time can be saved with a reduced number of generations. The visual results generated by the algorithm can be observed in Figures 3-6.

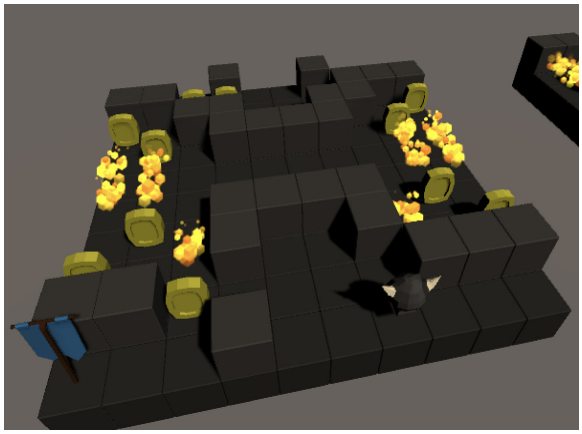

Figure 3. CoCoSo generated example 1

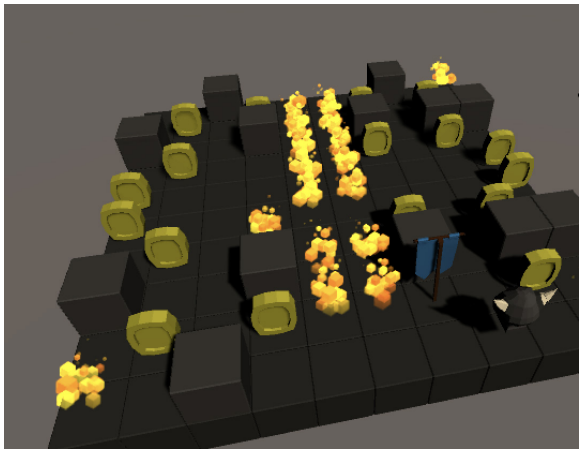

Figure 4. CoCoSo generated example 2

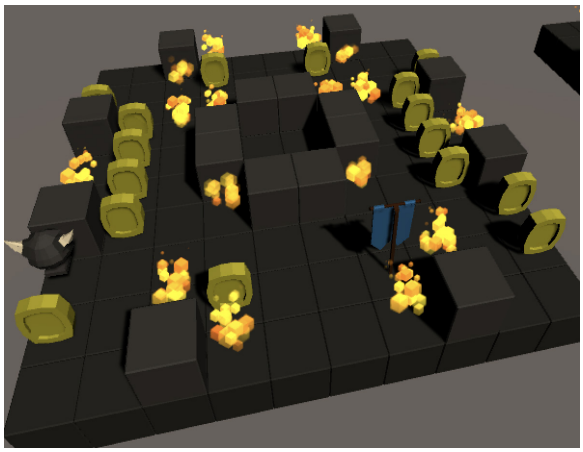

Figure 5. CoCoSo generated example 3

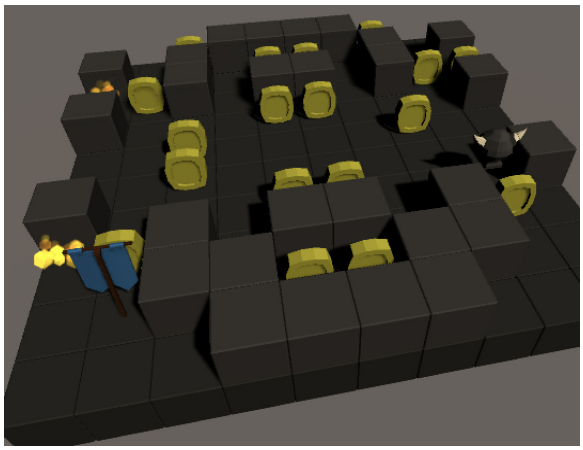

Figure 6. CoCoSo generated example 4

It can be seen that the generated game scene-level layouts satisfy both the aesthetics and the functional criteria and that the patterns between these two simultaneous results are different. It can also produce interesting patterns faster than the original algorithm, but calculation time

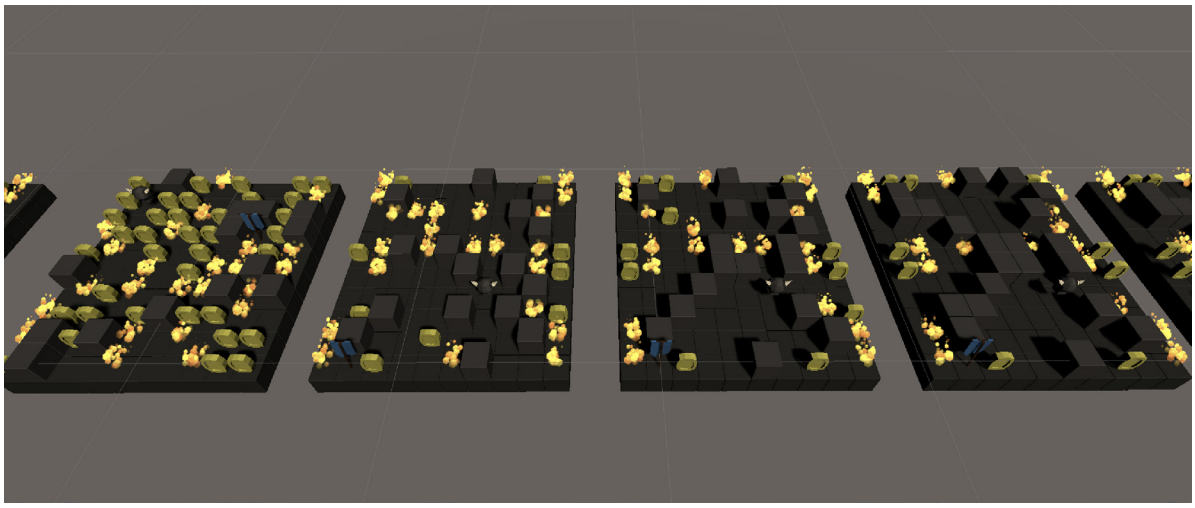

Figure 7. CoCoSo pattern progression over generations

is increased by 10% for the same number of generations and for the same population size. It can be noticed that, even in the initial steps of the algorithm and with a lower number of generations, the desired patterns begin to form (see Figure 7).

5. Conclusion

Applying the proposed approach, the genetic procedural scene layout generator was extended with the CoCoSo method. The CoCoSo method generates satisfying results with a smaller number of generations and takes into account more various aspects of the problem compared to the WASPAS method. Neutrosophic sets are a great tool to work with uncertain information due to modelling of truth, indeterminacy, and falsity membership functions. The implementation of neutrosophic set environment in the CoCoSo approach allows improving the variety of the generated levels. Within the framework of the genetic algorithm, the proposed neutrosophic CoCoSo method is improved compared to the original CoCoSo approach. This improvement can be stated as: if one has very close values of the divisor in equations (21) and (22), one sets them to constant values. The conversions from the crisp numbers to the neutrosophic ones were updated to suit the iterative evolution. The normalization procedures were updated to work with global min and max values, by adapting them to the set of generations, in which the values depend on the previous generations. The generated patterns satisfy the replayability principle for both the aesthetics and the functional criteria.

REFERENCES

Broumi, S., Bakali, A., Talea, M., Smarandache, F., Uluçay, V., Sahin, M, Dey, A., Dhar, M, Tan, R. P., Bahnasse, A. & Pramanik, S. (2018). Neutrosophic Sets: An Overview. In Smarandache, F. & Pramani, S. (Eds.), *New Trends in Neutrosophic Theory and Applications, vol. 2*, 403-434. Brussels: Pons Edition.

Choi, S. S. & Moon, B. R. (2003). Normalization in genetic algorithms. In *Genetic and Evolutionary Computation Conference* (pp. 862-873).

Filip, F. G. (2021). Automation and Computers and Their Contribution to Human Well-being and Resilience, *Studies in Informatics and Control*, *30*(4), 5-18. DOI: 10.24846/v30i4y202101

Franceschelli, G. & Musolesi, M. (2021). Creativity and machine learning: A survey, *eprint arXiv:2104.02726*. DOI: 10.48550/arXiv.2104.02726

Haque, T. S., Chakraborty, A., Mondal, S. P. & Alam, S. (2020). Approach to solve multi-criteria group decision-making problems using exponential operational law in a generalised spherical fuzzy environment, *CAAI Transactions on Intelligence Technology*, *5*(2), 106-114.

Herrmann, J. W. (1999). A genetic algorithm for minimax optimization problems. In *Proceedings of the 1999 Congress on Evolutionary Computation-CEC99, Vol. 2* (pp. 1099-1103).

Kahraman, C., Öztayşi, B. & Çevik Onar, S. (2016). A comprehensive literature review of 50 years of fuzzy set theory, *International Journal of Computational Intelligence Systems*, *9*(1), 3-24.

Kieu, P. T., Nguyen, V. T., Nguyen, V. T. & Ho, T. P. (2021). A Spherical Fuzzy Analytic Hierarchy Process (SF-AHP) and Combined Compromise Solution (CoCoSo) Algorithm in Distribution Center Location Selection: A Case Study in Agricultural Supply Chain, *Axioms*, *10*(2), article id: 53.

Peng, X. & Garg, H. (2021). Intuitionistic fuzzy soft decision making method based on CoCoSo and CRITIC for CCN cache placement strategy selection, *Artificial Intelligence Review*, *55*, 1567–1604.

Peng, X. & Li, W. (2021). Spherical fuzzy decision making method based on combined compromise solution for IIoT industry evaluation, *Artificial Intelligence Review*, *55*, 1857-1886.

Pereira, L. T., Toledo, C., Ferreira, L. N. & Lelis, L. H. (2016). Learning to speed up evolutionary content generation in physics-based puzzle games. In *2016 IEEE 28th International Conference on Tools with Artificial Intelligence (ICTAI)*, (pp. 901-907).

Petrovas, A. & Bausys, R. (2022). Procedural Video Game Scene Generation by Genetic and Neutrosophic WASPAS Algorithms, *Applied Sciences*, *12*(2), 772.

Pichot, N., Bonetto, E., Pavani, J. B., Arciszewski, T., Bonnardel, N. & Weisberg, R. W. (2022). The construct validity of creativity: empirical arguments in favor of novelty as the basis for creativity, *Creativity Research Journal*, *34*(1), 2-13.

Safak, A. B., Bostanci, E. & Soylucicek, A. E. (2016). Automated Maze Generation for Ms. Pac-Man Using Genetic Algorithms, *International Journal of Machine Learning and Computing*, *6*(4), 226-240.

Semenas, R., Bausys, R. & Zavadskas, E. K. (2021). A Novel Environment Exploration Strategy by m-generalised q-neutrosophic WASPAS, *Studies in Informatics and Control*, *30*(3), 19-28. DOI: 10.24846/v30i3y202102

Smarandache, F. A. (1999). *A Unifying Field in Logics. Neutrosophy: Neutrosophic Probability, Set, and Logic*. Rehoboth: American Research Press.

Statham, N., Jacob, J. & Fridenfalk, M. (2022). Game environment art with modular architecture, *Entertainment Computing*, *41*, article id: 100476.

Svadlenka, L., Simic, V., Dobrodolac, M., Lazarevic, D. & Todorovic, G. (2020). Picture Fuzzy Decision-Making Approach for Sustainable Last-Mile Delivery, *IEEE Access*, *8*, 209393-209414.

Thakkar, S., Cao, C., Wang, L., Choi, T. J. & Togelius, J. (2019). Autoencoder and evolutionary algorithm for level generation in lode runner. In *2019 IEEE Conference on Games (CoG)*, (pp. 1-4).

Togelius, J., Kastbjerg, E., Schedl, D. & Yannakakis, G. N. (2011). What is procedural content generation? Mario on the borderline. In *Proceedings of the 2nd International Workshop on Procedural Content Generation in Games* (pp. 1-6).

Wu, Z., Liao, H., Lu, K. & Zavadskas, E. K. (2021). Soft Computing Techniques and Their Applications in Intelligent Industrial Control Systems: A Survey, *International Journal of Computers, Communications & Control*, *16*(1), 1-28.

Yazdani, M., Mohammed, A., Bai, C. & Labib, A. (2021). A novel hesitant-fuzzy-based group decision approach for outsourcing risk, *Expert Systems with Applications*, *184*, article id: 115517.

Yazdani, M., Zarate, P., Zavadskas, E. K. & Turskis, Z. (2018). A Combined Compromise Solution (CoCoSo) method for multi-criteria decision-making problems, *Management Decision*, *57*(9), 2501-2519.

Yousefi, S., Valipour, M. & Gul, M. (2021). Systems failure analysis using Z-number theory-based combined compromise solution and full consistency method, *Applied Soft Computing*, *113*(Part A), article id: 107902.

Zadeh, L. A. (1965). Fuzzy sets, *Information and Control*, *8*(3), 338-353.

Zavadskas, E. K., Bausys, R., Lesciauskiene, I., & Omran, J. (2020). M-generalised q-neutrosophic MULTIMOORA for Decision Making, *Studies in Informatics and Control*, *29*(4), 389-398. DOI: 10.24846/v29i4y202001

Zavadskas, E. K., Turskis, Z. & Kildienė, S. (2014). State of art surveys of overviews on MCDM/MADM methods, *Technological and Economic Development of Economy*, *20*(1), 165-179.

Zhao, H., Zhang, R., Zhang, A. & Zhu, X. (2021). Multi-attribute Group Decision Making Method with Unknown Attribute Weights Based on the Q-rung Orthopair Uncertain Linguistic Power Muirhead Mean Operators, *International Journal of Computers, Communications & Control*, *16*(3), 1-20.