JOURNAL OF APPLIED INTELLIGENT SYSTEMS & INFORMATION SCIENCES
VOL. 3, PP. 63-76, OCTOBER 2022.
Available at: www.journal.research.fanap.com
DOI: https://doi.org/10.22034/JAISIS.2022.365448.1049

A HYBRID METHOD OF COGNITIVE MAPPING AND FUZZY DATA ENVELOPMENT ANALYSIS TO ANALYZE THE DELAYS IN SOFTWARE PROJECTS

Serveh Kakaei[1,*]

[1]*Department of Industrial Engineering, University of Kurdistan, Sanandaj, IRAN*

ABSTRACT

Delay is one of the common events in software projects; that's why it is necessary to know the factors behind the delays in projects of this type and to determine their impact on achieving the set goals. The purpose of this research is to propose an intelligent method to analyze causal relationships between delay factors in software projects. In order to identify the delay factors, attempts are made to use the lessons learned, project documents, and the opinions of experts. A fuzzy cognitive map (FCM) is drawn to indicate the causal relationships between the delay factors and the evaluation factors. For the first time, a hybrid algorithm is used to identify the effect of each delay factor on the evaluation factors and to prioritize them by applying fuzzy data envelopment analysis (FDEA). For this purpose, the causal relationships between 16 delay factors and four evaluation factors are considered. The results of the research show that "Unrealistic deadlines", "Working on too many projects simultaneously" and "Unexpected risks" are the most important delay factors in software projects in Iran.

KEYWORDS: Fuzzy cognitive map, Software projects, Causal relationships, Fuzzy data envelopment analysis, Delay factors

1. INTRODUCTION

One of the most important items of global capital is undoubtedly technology. As for countries with less developed technology even with a strong defense industry or advanced agriculture and healthcare, there will be no prospect for growth. While companies with the biggest turnover in the world are developing as a result of informatics and software, economic policies are more affected by such new trends. The active role of countries with such advances in technology has led to the monopolization of business and the dominance of certain brands. However, software development, or A.k.A coding, has become one of the most valuable professions of the 21st century. Even though the word technology stands for a wide range of terms alone, the keyword in the background is software. The software industry, which is one of the leading sectors of the new economic system that focuses on technology and knowledge, is of great importance in the development of countries. For the last few decades, the information technology (IT) industry went through many evolutionary changes due to technological improvements. The IT industry is complex, and naturally, this leads to the complexity of projects performed in this field. Such complex projects need to be handled with great efficiency and activeness. These projects are prone to delays due to different challenging situations and issues. Despite extensive scientific work and improved management techniques, software projects are still unable to resolve multiple issues such as delays, conflicts, productivity loss, poor performance, coordination, and many more. Poor performance of software projects is a global issue. Developing software which is one of the subsets of the IT industry is a

[*] *Corresponding Author, Email:* s.kakaei@uok.ac.ir

RECEIVED: 12 OCTOBER 2022; ACCEPTED: 30 OCTOBER 2022; PUBLISHED ONLINE: 31 OCTOBER 2022

complicated process that is difficult to predict and estimate. Delays constitute a major problem in software projects. Therefore, these projects frustrate customers when they overrun the established schedule.
A project delay is an unplanned and unexpected deferment of a project because of some event or occurrence that impedes the project's commencement or continuation. It is the length of time that extends the project duration and causes a disruption in the delivery of project goals and objectives. Project delays can cause a slew of problems, some of which may not be visible at first. Unexpected delays have a significant impact on project finances. Every time a delay occurs, resources are wasted, and the business incurs additional expenses. Project delays can also negatively affect your company's reputation among your stakeholders and clients. Delay in one client project can cause late delivery of other projects by locking up resources needed elsewhere. Also, delays in projects can disturb the budget, cause the project to go off course, and worst of all result in missed deadlines.

In order to systematize project management, many project-oriented organizations have turned to establishing a project management office at the strategic levels of the organization. The effect of establishing such a unit is beyond the project environment and includes other aspects of the organization's business performance. This shows the necessity and importance of project management. By considering the rapid changes and developments in the development of software products, which, if neglected, will cause the product to be removed from the competition, as well as the optimal use of resources, the necessity of conducting research in this sector becomes obvious. According to a report by the Project Management Institute, employers will require nearly 87.7 million individuals to serve project management-oriented roles by 2027. Information technology is a high-profile industry where project management has acquired an active role in developing tested software solutions and projects for clients. IT companies big or small are looking to widen their scope of operations to attain high competence in challenging markets. Project managers are in high demand in the IT industry to help companies develop software solutions and execute multifarious projects for different clients in an effective and time-bound manner (Vinod, 2019).

Investigating project delays and how to reduce the occurrence of delays is one of the things that should be considered in every project, either case-by-case or comprehensive, regardless of the specific place or time, to bring preventive results for future projects. Identifying factors of stagnation in the progress of projects and their delays can improve the management of software development projects. Factors that, in addition to affecting management goals, also affect each other. For this reason, it is necessary to identify the relationships between these factors in the process of prioritizing them. Therefore, to measure the relationship between factors and identify the most effective factors affecting delay, it is useful to use methods such as cognitive mapping. The current research intends to prioritize the factors affecting the delay of software projects by examining the past research and documents of software projects that have been delayed, considering the cause-and-effect relationships between these factors. This prioritization is achieved using experts' opinions and mathematical methods such as fuzzy cognitive map (FCM) and fuzzy data envelopment analysis method without output.

The most important reason for using such a method is to help the decision-makers in the field of project management to understand complex systems accurately, comprehensively, and comprehensibly. In this way that at first, the factors affecting the delay of software projects are identified and the cause-and-effect relationships between these factors are determined. Then, by using the output results of the first phase of the research, the prioritization of these factors is achieved in such a way that the project management achieves the best possible result concerning the goal of reducing the project delay by allocating financial resources and limited time to eliminate the effect of the most effective delay factors. The purpose of this research is to use mathematical methods and consider the cause-and-effect relationships between the factors extracted from previous researches, documents of real projects, and opinions of experts active in software projects. The method used in this research is the fuzzy cognitive mapping method along with the hybrid learning algorithms and the fuzzy data envelopment analysis method based on slack variables, which is used to investigate and prioritize the delay factors in the projects. The structure of this research is as follows: in the second part, the background of research conducted in the field of software project delays is discussed. In the third part, the investigation of the research method including the theoretical foundations of fuzzy cognitive map methods and fuzzy data envelopment analysis is done along with the proposed approach. In the fourth part, the results obtained from the case study are analyzed and in the final part, the conclusion of the research is presented.

2. Literature review

In the past three decades, the complexity of IT applications has greatly increased, and so has the managers' concern toward IT area management improvement. Different forms of software project delays exist which can undermine project success. Some delays have been reported as more influential than others. Producing software on time has become increasingly complex and difficult to manage due to the fast evolution of the software industry, the large application sizes, the unpredictable software activities, and the variety of software development processes and environments. Choetkiertikul et al. presented an approach to providing automated support for project managers in predicting whether a subset of software tasks in a software project has a risk of being delayed. They used collective classification to predict the degree of delay for a group of related tasks (Choetkiertikul et al., 2015). In another paper, Choetkiertikul et al. presented an approach to providing automated support for project managers and other decision-makers in predicting whether an issue is at risk of being delayed against its deadline. Issue-tracking systems have increasingly been used in many software projects. An issue could represent a software bug, a new requirement, a user story, or even a project task (Choetkiertikul et al., 2017).

Software projects tend to overrun costs and schedules. A study by McKinsey and the University of Oxford in 2012 of 5,400 large-scale IT projects found that on average 66 % of IT projects were over budget and 33 % went over the scheduled time (Bloch et al., 2012). In the well-known CHAOS report, Standish Group found that 82 % of software projects missed their schedules (Standish Group, 2004). To alleviate such problems, today's software development is trending toward agility and continuous delivery like DevOps. Even large software systems, e.g., Microsoft Windows, are moving from major releases to a stream of continuous updates (Bright, 2015). This is a shift from a model where all functionalities are shipped together in a single delivery to a model involving a series of incremental, continuous deliveries of a small number of resolved issues. The evaluation of project requirements also contributes to costs, time expenses, unfulfilled goals, or even cancellations of projects, becoming a natural, unwanted project danger of adverse effects on the reliability of software projects (Mahdi et al., 2021). The requirements for amending the specifications (in terms of multiple extensions, elimination, and modification) during the software development project are among the principal factors raising problems for the project (Ferreira et al., 2009; Tiwana, 2004; Sommerville, 2011 and Ali, 2019).

Improving the efficiency and maintaining the sustainability of a software project are obstacles that are faced by project managers. The probability of project failure is generally due to the lack of knowledge, skills, resources, and technology during project implementation (Oun et al., 2016; Maimon, 2019 and Saleem, 2019). The knowledge that is obtained from historical project data sets can be used for the development of predictive models by either utilizing a mathematical methodology, including linear regression and study of association, or machine learning approaches, such as artificial network networks and support vector machines. Predictive methods provide a method that is focused on present and historical project evidence to forecast the project's future (Mahdi et al., 2021). Mahdi et al. surveyed a comprehensive review of papers on the application of machine learning in software project management. Besides, they presented an extensive literature analysis of machine learning, software project management, and techniques from three main libraries, Web Science, Science Directs, and IEEE Explore. One-hundred and eleven papers are divided into four categories in these three repositories. They also offered a comprehensive perspective and a context that would be important for potential work in project risk management. In conclusion, they have shown that project risk assessment by machine learning is successful in minimizing the loss of the project, thereby increasing the likelihood of the project's success, providing an alternative way to efficiently reduce the project failure probabilities, and increasing the output ratio for growth, and it also facilitates analysis on software fault prediction based on the accuracy (Mahdi et al., 2021). A study conducted by Bloch et al. (2012) in collaboration with the University of Oxford suggests that half of all large IT projects massively blow their budget. On average, large IT projects run 45 percent over budget and 7 percent over time while delivering 56 percent less value than predicted. Software projects run the highest risk of cost and schedule overruns. Staggering as these findings are, most companies survive the pain of cost and schedule overruns. However, 17 percent of IT projects go so badly that they can threaten the very existence of the company (Bloch et al., 2012).

Biesialska et al. studied the decisions of large-scale agile teams regarding the identification of dependencies between user stories. Their goal is to explain the detection of dependencies through users' behavior in large-scale, distributed projects. The results showed that leveraging the agile lifecycle management monitoring data to automatically detect dependencies could help agile teams address work coordination needs and manage risks related to dependencies on time (Biesialska et al., 2021). Ma et al. (2000) conducted an industry survey to get current, sufficient, and precise information, and draw reliable conclusions about software project delays.

TaghiZadeh and Kashef presented the time and cost estimations for future software projects to assist Company's leaders and management team. They focused on analyzing the relationship between project complexity and cost/time overrun. The sample data from around 50 projects were collected from the Changepoint database and the two research hypotheses were defined and tested using statistical techniques (TaghiZadeh and Kashef, 2022).

By reviewing mentioned studies, it can be found that most previous studies have not been prioritized to pay more and more careful attention to the delay factors of the software projects. In contrast, this study uses a group decision-making process to screen the identified early factors based on previous researches. To achieve this, fuzzy cognitive maps and fuzzy data envelopment analysis methods are employed. The fundamental difference between this proposed group decision-making process and the existing statistical methods is in statistics, individuals are considered as masses, the population is uniform, sampling is random, and no one takes precedence over the others. Also, in statistics, if there is a biased sample and it has a large variance, that sample is deleted. But if the decision maker faces a type of decision that requires distinguishing between samples and individuals and there is no possibility to remove a sample, experts with different opinions should be considered and group decision-making should be used instead of statistics. In this approach, a questionnaire is distributed among the experts whose opinions are important. The FCM method used in this study can display what is happening in the system due to causal relationships between the concepts and the initial state. Also, it can examine complex systems with many concepts and causal relationships (Papageorgiou et al., 2004).

3. Methodology

This study aims to identify the most effective factors of delay in software projects. The causal relationships between delay factors are considered and then weighted according to experts. In the first phase, the systematic structure of these delay factors is investigated using FCM. Afterward, the impact of each factor on management goals or evaluation factors is determined using the hybrid learning algorithm. Then in the second phase, the FDEA method ranks the delay factors which were obtained in the previous phase.

3.1. Fuzzy cognitive map

A fuzzy cognitive map is an analysis tool that uses a graph structure to show the causal relationships of influence factors in a decision-making system. During the decision-making process, it is reasonable for experts to use linguistic variables to express their subjective opinions. Kosko proposed the fuzzy cognitive map to illustrate a system, which uses a graph of concepts and shows the cause-effect among concepts (Kosko, 1986). A simple FCM network can be shown where C_i is the status value of concept i. The status value can be represented as a number within [0, 1]. The weight W_{ij} indicates the relation degree between cause concept C_i to effect concept C_j, which can be represented as a number within [-1, 1]. If $W_{ij} > 0$, there is positive causality between concepts C_i and C_j. If $W_{ij} = 0$, there is no relationship between concepts C_i and C_j. If $W_{ij} < 0$, there is negative causality between concepts C_i and C_j (Kosko, 1986; Mei et al., 2013).

$$\boldsymbol{A_i^{(t+1)}} = \boldsymbol{f}\left(\sum_{\boldsymbol{j=1}}^{\boldsymbol{n}} \boldsymbol{A_j^{(t)}} \,.\, \boldsymbol{W_{ji}}\right) \tag{1}$$

where, A_i^t is the status value of concept C_i at period t; A_j^t is the status value of concept C_j at period t; A_i^{t+1} is the status value of concept C_i at period t+1; W_{ji} is the corresponding fuzzy relation degree between C_j and C_i, and f is a threshold function that transforms the computing result into the interval [0,1].

Fuzzy cognitive maps have been applied in many different fields (Mei et al., 2013; Jetter and Kok, 2014; Dias et al., 2015; Papageorgiou et al., 2015; Ahmed et al., 2018; Poczeta et al., 2020). In recent years, few learning algorithms are used for improving the accuracy and convergence of weights in FCM. Learning rules for a connectionist system include algorithms or methodologies that determine changes in connections' weights in a network. FCM learning algorithms can be classified into three groups: Hebbian-based, population-based, and hybrid learning algorithms (Papageorgiou and Kannappan, 2012). The Hebbian learning rule is one of the simplest and most well-known learning algorithms (Hebb, 1949). Based on the data considered in this research work, the hybrid learning algorithm is the best option. This hybrid algorithm is a combination of meta-heuristic and Hebbian algorithms and is appropriate for correcting weights that are combined with time-series data or experts' opinions. Therefore, this paper proposed a non-linear Hebbian learning-differential evolution (NHL-DE) algorithm, which can update non-zero weights in different iterations and maintains the relationships between defined concepts in the initial map.

The proposed NHL-DE algorithm includes two procedures. The first procedure is the main learning algorithm based on a non-linear Hebbian learning algorithm. In this procedure, the weights and concepts value are calculated and updated according to this algorithm until the termination conditions happen. For this purpose, two different functions are considered. The second procedure is an algorithm based on a population-based meta-heuristic algorithm. The calculated weights in the first algorithm are transmitted to the second algorithm. This procedure updates more accurate values of weights and concepts. In the presented pseudocode, $A^{(0)}$ is the initial state matrix of the system, $W^{(0)}$ is the initial weight matrix, $A^{(t)}$ and $A^{(t+1)}$ are the values of concept C_i and C_j at periods t and t + 1, $W_{ji}^{(t)}$ and $W_{ji}^{(t+1)}$ are updated weight between concepts C_i and C_j at periods t and t + 1: η and γ are learning rates, $W_{NHL}^{(t+1)}$ represents the final weight matrix between concepts in the first stage, "sgn" is the sign function, "NP" is the number of population, and f is a transformation function. In this paper, according to the concept values, which are between zero and one, the most appropriate transformation function is the sigmoid function. Also, the termination conditions of FCM calculations include: a stable state, that is, until $A_i^{(t+1)}$is equal to $A_i^{(t)}$ or they have a little difference, a loop of numerical values under a specific period and a variety of numerical values in a non-deterministic, random way (Papageorgiou et al., 2006). Figure 1 shows the pseudocode of the non-linear Hebbian learning-differential evolution algorithm.

```
Procedure 1: Non-linear Hebbian learning algorithm:
Step 1-1: Read the input concept state $A^0$ and initial weight matrix $W^0$.
Step 1-2: Repeat for each period t.
Step 1-3: Calculate $A^{(t)}$ according to equation (1):
    $$A_i^{(t+1)} = f(\sum_{i=1}^{n} A_j^{(t)} . W_{ji})$$
Step 1-4: Update the weights:
    $$W_{ji}^{(t)} = \gamma W_{ji}^{(t-1)} + \eta A_i^{(t-1)} (A_j^{(t-1)} - sgn(W_{ji}^{(t+1)} A_i^{(t-1)}))$$
Step 1-5: Calculate the termination functions (two functions).
Step 1-6: The calculations continue until the termination conditions happen.
Steps 1-7: Send the final weights to procedure 2.

Procedure 2: Differential evolution algorithm:

Step 2-1: Initialize the DE population in the neighborhood of $W_{NHL}^{(t+1)}$ and weight constraints.
Step 2-2: Repeat for each period (t).
Step 2-3: For i=1 to NP (number of populations) do steps 2-4 to 2-6.
Step 2-4: Mutation $W_i^{(t)}$ → Mutant_Vector.
Step 2-5: Crossover (Mutant_Vector) → Trial_Vector
Step 2-6: If F (Trial_Vector) ≤ F($W_i^{(t)}$), accept Trial_Vector for the next generation.
Step 2-7: The calculations continue until the termination conditions happen.
```

Fig. 1. Pseudocode of the non-linear Hebbian learning-differential evolution algorithm

3.2. *Fuzzy data envelopment analysis (FDEA)*

Data envelopment analysis (DEA) is a prominent technique for evaluating the relative efficiency of a set of entities called decision-making units (DMUs). Fuzzy DEA is a powerful tool for evaluating the performance of DMUs with imprecise data. When the input or output variables are fuzzy, the FDEA method should be used to assess DMUs (Chen et al., 2013). Thus, in this study, two fuzzy slack-based models (fuzzy SBM) are used to measure efficiency, super efficiency, and prioritization before providing mathematical models (Tone, 2001). In this model, $\tilde{X}_{ji}$, $\tilde{Y}_{jr}$ respectively represent non-deterministic inputs (i = 1, …, m) and outputs (r = 1, …, s) for DMUs (j = 1, …, n) and they can be indicated $\mu_{\tilde{X}_{ji}}$, $\mu_{\tilde{Y}_{ji}}$by the membership function in a convex fuzzy set. In a fuzzy environment, the efficiency of the kth DMU($\tilde{\delta}_k$) is calculated using Equation (2):

$$\begin{aligned}
&Min\ \tilde{\delta}_k = q - \frac{1}{m}\sum_{i=1}^{m} S_i^- / \tilde{X}_{ik} \\
&s.t.\quad 1 = q - \frac{1}{s}\sum_{r=1}^{s} S_r^+ / \tilde{Y}_{rk} \\
&\qquad q\tilde{X}_{ik} = \sum_{j=1}^{n} \tilde{X}_{ik}\lambda_j' + S_i^- \quad i = 1,\dots,\mathrm{m} \\
&\qquad q\tilde{Y}_{rk} = \sum_{j=1}^{n} \tilde{Y}_{rk}\lambda_j' + S_r^+ \quad r = 1,\dots,\mathrm{s} \\
&\qquad \sum_{j=1}^{n} \lambda_j' = q \\
&\qquad \lambda_j' \ge 0, j = 1,\dots,n,\ \ S_i^- \ge 0, i = 1,\dots,\mathrm{m},\ S_r^+ \ge 0, r = 1,\dots,\mathrm{s},\ \ q > 0.
\end{aligned} \tag{2}$$

In Equation (2), all inputs and outputs are considered fuzzy data. The next step, defuzzification, is to convert the fuzzy set of inputs and outputs into exact values by defining the threshold of membership and the α-cut method. S($\tilde{X}_{ji}$), S($\tilde{Y}_{jr}$) has been created to support $\tilde{X}_{ji}$, $\tilde{Y}_{jr}$. The support is the set of elements with membership functions larger than 0. Finally, $\tilde{X}_{ji}$, $\tilde{Y}_{jr}$ are defined by the α-cut as follows:

$$\begin{aligned}
(X_{ji})_\alpha &= \left\{x_{ji} \in S(\tilde{X}_{ji}) \mid \mu_{\tilde{X}_{ji}}(x_{ji}) \ge \alpha\right\},\ \forall j,i \\
(Y_{jr})_\alpha &= \left\{y_{jr} \in S(\tilde{Y}_{jr}) \mid \mu_{\tilde{Y}_{jr}}(y_{jr}) \ge \alpha\right\},\ \forall j,r
\end{aligned} \tag{3}$$

The $(X_{ji})_\alpha$ and $(Y_{ji})_\alpha$ are crisp sets. Therefore, when using α-cut, input and output can both be expressed as the crisp intervals of various α-level. A set of standard α levels defined in equation (3) can be expressed as follows:

$$\begin{aligned}
(X_{ji})_\alpha &= \left\{x_{ji} \in S(\tilde{X}_{ji}) \mid \mu_{\tilde{X}_{ji}}(x_{ji}) \ge \alpha\right\} = \left[(X_{ji})_\alpha^L, (X_{ji})_\alpha^U\right],\ \forall j,i \\
&= \left[\min_{x_{ji}} \left\{x_{ji} \in S(\tilde{X}_{ji}) \mid \mu_{\tilde{X}_{ji}}(x_{ji}) \ge \alpha\right\}, \max_{x_{ji}} \left\{x_{ji} \in S(\tilde{X}_{ji}) \mid \mu_{\tilde{X}_{ji}}(x_{ji}) \ge \alpha\right\}\right] \\
(Y_{jr})_\alpha &= \left\{y_{jr} \in S(\tilde{Y}_{jr}) \mid \mu_{\tilde{Y}_{jr}}(y_{jr}) \ge \alpha\right\},\ \forall j,r \\
&= \left[\min_{y_{jr}} \left\{y_{jr} \in S(\tilde{Y}_{jr}) \mid \mu_{\tilde{Y}_{jr}}(y_{jr}) \ge \alpha\right\}, \max_{y_{jr}} \left\{y_{jr} \in S(\tilde{Y}_{jr}) \mid \mu_{\tilde{Y}_{jr}}(y_{jr}) \ge \alpha\right\}\right].
\end{aligned} \tag{4}$$

In the situation of various α levels for $\left\{\left(X_{ji}\right)_\alpha \mid 0 < \alpha \le 1\right\}$ and $\left\{\left(Y_{jr}\right)_\alpha \mid 0 < \alpha \le 1\right\}$, the fuzzy-DEA model can be converted into the crisp-DEA model. According to Zimmermann (Zimmermann, 1975) and Yager (Yager, 1981), the efficiency membership function for the *j*th DMU can be expressed in the form of Equation (5):

$$\mu_{\tilde{E}_k}(z) = \sup_{x,y}\ \min\left\{\mu_{\tilde{X}_{ji}}(\mathrm{x}_{ji}), \mu_{\tilde{Y}_{jr}}(y_{jr}), \forall\ \mathrm{j,r,i} \mid \mathrm{z} = \mathrm{E}_k(x,y)\right\} \tag{5}$$

In Equation (5), $E_k(x,y)$ is the efficiency value calculated using the traditional SBM model under a set of inputs and outputs. According to Equation (5), for any efficiency value with the combination x_{ji}, y_{jr} of z, its minimum degree of membership equals the membership of $\tilde{\ } E_k$ on point z. Based on the definitions in Equations (4) and (5), the upper and lower bounds of α-cut under $\mu_{\sim E_k}$ can be determined. The two-step mathematical programming model can be transformed into a traditional one-step programming model using the Pareto-optimal solution. Therefore, the Fuzzy SBM model (Equation (2)) can be transformed into:

$$
\begin{aligned}
& Min(\delta_k)_\alpha^U = q - \frac{1}{m}\sum_{i=1}^{m}(S_i^-)^L / (\mathrm{x}_{ik})_\alpha^L \\
& s.t. \quad 1 = q - \frac{1}{s}\sum_{r=1}^{s}(S_r^+)^U / (\mathrm{y}_{rk})_\alpha^U \\
& \qquad q(\mathrm{x}_{ik})_\alpha^L = \sum_{j=1,\neq k}^{n}(\mathrm{x}_{ij})_\alpha^U \lambda_j^/ + (\mathrm{x}_{ik})_\alpha^L \lambda_j^/ + (S_i^-)^L \quad i = 1,\dots,\mathrm{m} \\
& \qquad q(y_{rk})_\alpha^U = \sum_{j=1,\neq k}^{n}(y_{rj})_\alpha^L \lambda_j^/ + (y_{rk})_\alpha^U \lambda_j^/ - (S_r^+)^U \quad r = 1,\dots,\mathrm{s} \\
& \qquad \sum_{j=1}^{n}\lambda_j^/ = q \\
& \qquad \lambda_j^/ \geq 0, j = 1,\dots,n, \ (S_i^-)^L \geq 0, i = 1,\dots,\mathrm{m}, \ (S_r^+)^U \geq 0, r = 1,\dots,\mathrm{s}, \ \ q > 0.
\end{aligned}
\tag{6a}
$$

$$
\begin{aligned}
& Min(\delta_k)_\alpha^L = q - \frac{1}{m}\sum_{i=1}^{m}(S_i^-)^U / (\mathrm{x}_{ik})_\alpha^U \\
& s.t. \quad 1 = q - \frac{1}{s}\sum_{r=1}^{s}(S_r^+)^L / (\mathrm{y}_{rk})_\alpha^L \\
& \qquad q(\mathrm{x}_{ik})_\alpha^U = \sum_{j=1,\neq k}^{n}(\mathrm{x}_{ij})_\alpha^L \lambda_j^/ + (\mathrm{x}_{ik})_\alpha^U \lambda_j^/ + (S_i^-)^U \quad i = 1,\dots,\mathrm{m} \\
& \qquad q(y_{rk})_\alpha^L = \sum_{j=1,\neq k}^{n}(y_{rj})_\alpha^U \lambda_j^/ + (y_{rk})_\alpha^L \lambda_j^/ - (S_r^+)^L \quad r = 1,\dots,\mathrm{s} \\
& \qquad \sum_{j=1}^{n}\lambda_j^/ = q \\
& \qquad \lambda_j^/ \geq 0, j = 1,\dots,n, \ (S_i^-)^U \geq 0, i = 1,\dots,\mathrm{m}, \ (S_r^+)^L \geq 0, r = 1,\dots,\mathrm{s}, \ \ q > 0.
\end{aligned}
\tag{6b}
$$

Models (6a) and (6b) have a limit on the maximum value of relative efficiency (The limit is one). Therefore, the prioritization process will be difficult due to the interval of the calculated efficiency values. In other words, based on the usual data coverage analysis models, it is not possible to prioritize decision-making units whose efficiency is equal to one. For this reason, the fuzzy data envelopment analysis model based on slack was developed to measure super efficiency. The super efficiency model of fuzzy data envelopment analysis based on slack was presented as models (7a) and (7b). Chen et al. have developed a fuzzy SBM model for measuring super-efficiency (Chen et al., 2013). According to the process that was used in the development of the fuzzy SBM model, they developed the model of Andersen and Petersen (Andersen and Petersen, 1993). The new model (fuzzy super SBM) is explained as follows:

$$
\begin{aligned}
&(\tau_k)_\alpha^U = Min \frac{1}{m}\sum_{i=1}^{m} (\bar{x}_i')^L / (X_{ik})_\alpha^L \\
&s.t. \quad 1 = \frac{1}{s}\sum_{r=1}^{s} (\bar{y}_r')^U / (Y_{rk})_\alpha^U \\
&\qquad (\bar{x}_i')^L \geq \sum_{j=1,\neq k}^{n} (X_{ik})_\alpha^L \lambda_j' \quad i = 1,\ldots,m \\
&\qquad (\bar{y}_r')^U = \sum_{j=1,\neq k}^{n} (Y_{rk})_\alpha^U \lambda_j' \quad r = 1,\ldots,s \\
&\qquad \sum_{j=1,\neq k}^{n} \lambda_j' = q \\
&\qquad \lambda_j' \geq 0, j = 1,\ldots,n, \neq k,\ (\bar{x}_i')^L \geq q(X_{ik})_\alpha^L, \\
&\qquad i = 1,\ldots,m, (\bar{y}_r')^U \leq q(Y_{rk})_\alpha^U, (\bar{y}_r')^U \geq 0, r = 1,\ldots,s,\ \ q > 0.
\end{aligned} \tag{7a}
$$

$$
\begin{aligned}
&(\tau_k)_\alpha^L = Min \frac{1}{m}\sum_{i=1}^{m} (\bar{x}_i')^U / (X_{ik})_\alpha^U \\
&s.t. \quad 1 = \frac{1}{s}\sum_{r=1}^{s} (\bar{y}_r')^L / (Y_{rk})_\alpha^L \\
&\qquad (\bar{x}_i')^U \geq \sum_{j=1,\neq k}^{n} (X_{ik})_\alpha^U \lambda_j' \quad i = 1,\ldots,m \\
&\qquad (\bar{y}_r')^L = \sum_{j=1,\neq k}^{n} (Y_{rk})_\alpha^L \lambda_j' \quad r = 1,\ldots,s \\
&\qquad \sum_{j=1,\neq k}^{n} \lambda_j' = q \\
&\qquad \lambda_j' \geq 0, j = 1,\ldots,n, \neq k,\ (\bar{x}_i')^U \geq q(X_{ik})_\alpha^U, \\
&\qquad i = 1,\ldots,m, (\bar{y}_r')^L \leq q(Y_{rk})_\alpha^L, (\bar{y}_r')^L \geq 0, r = 1,\ldots,s,\ \ q > 0.
\end{aligned} \tag{7b}
$$

Equations (7a) and (7b) calculate the upper and lower bound of supper efficiency under various α levels. The relative efficiency values calculated by Equations (7a) and (7b) differ from the crisp values calculated by the traditional DEA model in that they are fuzzy numbers. Therefore, it is difficult to rank the DMUs according to their efficiency scores. Also, the membership functions of the efficiency scores are unknown because the efficiency values in this study are the upper and lower bounds of the relative efficiency scores calculated under various α levels. To solve this problem, Chen and Klein (1997) presented the following Equation (8) that was used in this study:

$$
I(\tilde{}\,E_k^{\sim}, R) = \frac{\sum_{i=0}^{m}\left[(E_k)_{\alpha_i}^U - c\right]}{\sum_{i=0}^{m}\left[(E_k)_{\alpha_i}^U - c\right] - \sum_{i=0}^{m}\left[(E_k)_{\alpha_i}^L - d\right]}, \ m \to \infty \tag{8}
$$

In the equation above, it is assumed that k (k = 1, …, n) and i ($\alpha_i = ih / m, i = 0, \ldots, m$) are respectively the counter number of DMUs and counter of α levels. Also, the amount of c and d are $min_{i,k}\{(E_k)_{\alpha_i}\}$ and $max_{i,k}\{(E_k)_{\alpha_i}\}$. According to Equation (8), the bigger the fuzzy ranking index $I(\tilde{}\,E_k^{\sim}, R)$, the higher the priority for the DMU.

3.3. Proposed approach

The purpose of this study is to present an approach in two phases to investigate the delay factors in software projects. Employing this proposed approach leads to identifying effective delay factors in these projects and prioritizing them in terms of impacts on the management goals such as delay. Delay factors have been identified according to the review of the previous researches, lessons learned, and software project documents. After identifying these factors, since the purpose of this research was not to use a statistical questionnaire, an opinion form was used to receive the opinions of experts who were product managers, product owners, and scrum masters who are responsible for planning and scheduling software project development and professors from universities in IT project management to identify the most important factors among the primary factors. Then, by summarizing the opinions of experts using the voting decision method, the primary factors are screened, then the final list of factors affecting the delay of software projects is extracted. These factors are used as the initial input for the fuzzy cognitive mapping method. Primary factors have been screened in two stages, according to the polling technique. So, some of the factors have been removed from the initial list due to the low score from the poll. Finally, according to the opinions of the experts, the factors which are close to each other in terms of the nature of the impact on delay have been integrated. The polling methods are among the fundamental procedures of problem-solving (Hwang and Lin, 2012). These methods are used either in anonymous isolation or in a group setting. These methods may be used when the participants are separated physically and are cheaper than many other groups setting methods (Hwang and Lin, 2012). The experts referred to in this research include active experts in software project management, such as product managers, product owners, and scrum masters, as well as professors from universities in the IT project management field. The views of these experts have been used in two sections including determining the final list of delay factors and identifying the causal relationships between these factors. In addition, after determining the existence of the relationship between the delay factors, weights are assigned to these relationships by each expert. Finally, the weights of the causal relationships between the factors based on the agreement between the experts are determined. In the first phase, FCM is drawn to identify factors with goals that management wants to measure. Experts determine causal relationships between concepts in the FCM and weights of identified relationships. At this stage, according to the agreement between the senior experts, the triangular fuzzy numbers are determined for weighting. After that in the second phase, the FDEA method is used to prioritize the identified factors, because the evaluation criteria include fuzzy data. The values are obtained by running the hybrid learning algorithm and updating weights and concepts until convergence takes place. These values represent the amount of effect of each factor on management goals.

4. CASE STUDY

To extract data about the factors of delays in software projects in Iran and where they are more prominent during software development, semi-structured interviews were first conducted with ten experienced software practitioners. A list of the most probable factors that make software projects fall behind schedule was identified. Discussions were also held about the stages where most of the time delays were encountered during development. Following this, a set of questionnaires were developed which were administered to a set of 10 software developers involving small, medium, and large size software projects in Iran. Hence, 50 affecting delay factors are identified. These factors are classified into five categories as follows: 1) technology-related causes, 2) product-related causes, 3) personnel-related causes, 4) managerial-related causes, and 5) organization-related causes. Therefore, 40 primary factors due to the method of polling have been reduced. In this study, a poll form has been used to identify the most important opinions of ten experts. Then, the identified factors have been screened with the integration of experts' opinions and group decision-making methods. The factors which are the same in terms of nature and have the same effects on the delay of the project's implementation are integrated and the final list has been adjusted to 16 factors. Table 1 shows significant factors affecting delays in software projects.

Four evaluation factors including changes in the time of the predicted value (F17), changes in the cost of the predicted value (F18), changes in the quality of the predicted value (F19), and the reduction in economic value (F20) are added then the map is improved with new factors. These four criteria are called evaluation factors. Then, the causal relationships matrix creates between four new factors and other factors. Finally, the new 20 × 20 casual relationships matrix is used to draw FCM. For clarity in the FCM model, only 16 factors and causal relationships between these factors with fuzzy weights are shown in Figure 2 (The medium values of fuzzy numbers are demonstrated in Fig.1). Also, the casual relationships matrix for evaluation effective factors on delay in construction projects are presented in Table 2.

Table 1. Affecting factors of delay in software projects

	Factors
F1	Unrealistic deadlines
F2	Improper planning and estimation
F3	Lack of communication and teamwork
F4	Requirements and not priorities challenges
F5	Technological challenges
F6	Uncommitted clients
F7	Unexpected risks
F8	Lack of resources
F9	Inability to track the progress
F10	Employee turnover
F11	Lack of testing
F12	Working on too many projects simultaneously
F13	Bottlenecks with third-party integrations
F14	Negligence in Quality Assessment
F15	Non-involvement of project sponsors and stakeholders
F16	Change in the scope of the project

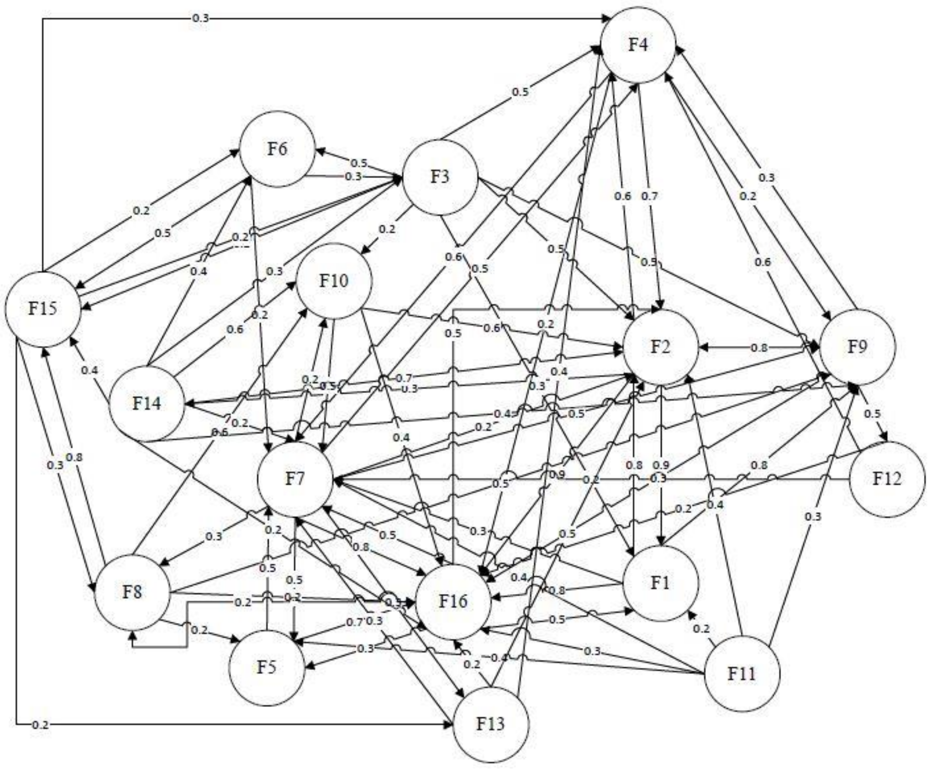

Fig. 2. FCM Model of the effective delay factors in software projects

Table 2. Casual relationships matrix for evaluation factors

Factors	F17	F18	F19	F20
F1	(0.8,0.9,1)	(0.8,0.9,1)	0	(0.4,0.5,0.6)
F2	(0.8,0.9,1)	(0.8,0.9,1)	0	(0.4,0.5,0.6)
F3	0	0	0	0
F4	(0.1,0.2,0.3)	0	0	(0.3,0.4,0.5)
F5	(0.4,0.5,0.6)	(0.1,0.2,0.3)	0	(0.1,0.2,0.3)
F6	0	0	0	0
F7	(0.6,0.7,0.8)	(0.6,0.7,0.8)	(0.2,0.3,0.4)	(0.2,0.3,0.4)
F8	(0.6,0.7,0.8)	(0.6,0.7,0.8)	0	(0.4,0.5,0.6)
F9	(0.4,0.5,0.6)	0	0	(0.4,0.5,0.6)
F10	(0.2,0.3,0.4)	(0.4,0.5,0.6)	0	(0.2,0.3,0.4)
F11	(0.1,0.2,0.3)	(0.1,0.2,0.3)	0	(0.6,0.7,0.8)
F12	(0.1,0.2,0.3)	(0.1,0.2,0.3)	0	(0.1,0.2,0.3)
F13	(0.1,0.2,0.3)	(0.1,0.2,0.3)	0	(0.2,0.3,0.4)
F14	(0.1,0.2,0.3)	(0.1,0.2,0.3)	0	(0.2,0.3,0.4)
F15	0	0	0	(0.2,0.3,0.4)
F16	(0.4,0.5,0.6)	(0.4,0.5,0.6)	0	(0.4,0.5,0.6)

Then, the hybrid learning algorithm (NHL-DE algorithm) is used to evaluate the impact of 16 delay factors on the four evaluation factors which are selected by the decision makers or project managers. The value of concepts is updated and the FCM structure becomes stable using this algorithm, in addition to reducing the dependency on expert opinions. In this regard, the scenarios are defined, and each scenario assumes that only one of 16 factors is effective in delay in construction projects. This leads to activating delay factors in software projects. Then, the hybrid learning algorithm is implemented by MATLAB for considering the weight of the causal relationships in the FCM. According to three weight matrices including the upper, middle, and lower weights, three scenarios are used for each delay factor based on a fuzzy weight matrix. In each scenario, the values of the four evaluation factors 17, 18, 19, and 20 are used as criteria for the prioritization of 16 delay factors. The calculated values for the four evaluation factors for each scenario are fuzzy numbers and are presented in Table 3.

These factors are prioritized using the FDEA model without outputs and according to criteria obtained by analyzing the causal relationships. The values of evaluation factors are considered DEA inputs because they are the managers' target for reduction. Moreover, since the amount of evaluation factors or DEA inputs is fuzzy in this research, the models with constant output mentioned in Section 3 are used. As a result, after running Models (6a) and (6b), and (7a) and (7b) for different α levels and using Equation (8), prioritization of effective factors on delay in software projects is achieved. Prioritization of combining FCM and FDEA is closer to reality because of considering the causal relationships between factors. The results of prioritization leading to the identification of effective factors for delay in software projects are presented in Table 4. As shown in Table 4, the upper and lower bound of efficiency scores for different α levels are prioritized using Equation (8). According to Table 4, whenever the DMU or studied factor has a higher score, it means that it has a lower rank. In other words, the higher impact of a factor on the evaluation factors increases the value of inputs, decreases the efficiency score, and increases the priority investigation level. Therefore, the factors, which have the lowest efficiency point, have the highest rank among the 16 factors. According to the results obtained in Table 4, "Unrealistic deadlines" (F1) have the most impact on delays in software projects in Iran. In addition, "Working on too many projects simultaneously" (F12), "Unexpected risks" (F7), "Improper planning and estimation" (F2), and "Bottlenecks with third-party integrations" (F13) are ranked second to fifth, respectively. In Fact, as experts in the field of software development say, "Unrealistic deadlines" and "Working on too many projects simultaneously" have a great impact on postponing software projects, which is completely consistent with the mathematical model used.

5. Conclusions

This study focuses on identifying and prioritizing the factors of delay in software projects and helps project managers to know the most effective delay factors on software projects and plan for them earlier. Unlike most of the previous studies, this study has used mathematical methods and intelligent algorithms. After reviewing previous lessons learned on several projects and the opinions of product owners and scrum masters, factors of delay in construction projects are identified. Then, the FCM method is used to identify the most effective factors in management goals. The hybrid learning algorithm is also used to achieve the effect of delay factors on management goals. Finally, short-listed factors have been prioritized according to the results of that learning algorithm and FDEA. The results show that the "Unrealistic deadline" factor was the most effective in the delay of software projects.

Table 3. Calculated values for the four evaluation factors in each scenario

	F17			**F18**			**F19**			**F20**		
Factors	Lower	Medium	Upper	Lower	Medium	Upper	Lower	Medium	Upper	Lower	Medium	Upper
F1	0.984	0.993	0.998	0.972	0.989	0.996	0.691	0.679	0.741	0.975	0.990	0.997
F2	0.986	0.989	0.997	0.968	0.974	0.979	0.624	0.792	0.874	0.893	0.981	0.991
F3	0.976	0.981	0.988	0.974	0.981	0.988	0.505	0.552	0.850	0.931	0.977	0.989
F4	0.988	0.990	0.996	0.980	0.988	0.993	0.771	0.826	0.883	0.718	0.792	0.912
F5	0.928	0.978	0.992	0.963	0.976	0.989	0.730	0.865	0.873	0.940	0.973	0.989
F6	0.981	0.985	0.990	0.947	0.976	0.990	0.560	0.704	0.866	0.949	0.983	0.990
F7	0.987	0.990	0.998	0.961	0.980		0.672	0.734	0.860	0.939	0.980	0.992
F8	0.970	0.978	0.984	0.955	0.976	0.991	0.676	0.725	0.811	0.911	0.964	0.979
F9	0.986	0.990	0.996	0.973	0.984	0.988	0.510	0.574	0.810	0.906	0.963	0.981
F10	0.981	0.988	0.991	0.961	0.975	0.984	0.703	0.778	0.835	0.928	0.972	0.995
F11	0.953	0.984	0.990	0.967	0.980	0.988	0.529	0.726	0.839	0.906	0.972	0.993
F12	0.978	0.983	0.987	0.972	0.988	0.992	0.622	0.754	0.824	0.862	0.901	0.995
F13	0.982	0.988	0.990	0.953	0.985	0.989	0.778	0.792	0.918	0.937	0.971	0.988
F14	0.974	0.981	0.988	0.977	0.983	0.989	0.559	0.634	0.882	0.887	0.906	0.966
F15	0.931	0.979	0.989	0.981	0.980	0.991	0.790	0.871	0.946	0.819	0.908	0.945
F16	0.940	0.973	0.998	0.986	0.988	0.992	0.702	0.779	0.822	0.901	0.962	0.978

Table 4. Prioritization of effective delay factors with FDEA

	Efficiency (α = 0)		**Efficiency (α = 0.2)**		**Efficiency (α = 0.4)**		**Efficiency (α = 0.6)**		**Efficiency (α = 0.8)**		**Efficiency (α = 1)**			
Factors	Lower	Upper	Lower	Upper	Lower	Upper	Lower	Upper	Lower	Upper	Lower	Upper	Score	Rank
F1	0.769	1	0.804	1	0.838	1	0.799	1	0.882	1	0.912	0.912	0.642815	1
F2	0.76	1.037	0.794	1.02	0.822	1.002	0.853	1.001	0.889	1.0006	0.931	0.931	0.659467	4
F3	0.761	1.0047	0.796	1	0.83	1	0.853	1	0.879	1	1.0001	1.0001	0.67444	6
F4	0.751	1	0.79	1	0.828	1.0005	0.855	1.0011	0.896	1.0017	1.0022	1.0022	0.675125	7
F5	0.789	1	0.826	1	0.86	1	0.879	1	0.9	1	0.957	0.957	0.688732	8
F6	0.768	1	0.81	1.0002	0.851	1.0026	0.882	1.0014	0.917	1.0013	1.0006	1.0006	0.697088	10
F7	0.756	1.0033	0.793	1.0031	0.829	1.0024	0.856	1.0014	0.887	1.0004	0.9313	0.9313	0.655119	3
F8	0.811	1	0.846	1	0.876	1	0.89	1	0.911	1	0.9434	0.9434	0.701831	12
F9	0.776	1	0.816	1.0003	0.853	1.0003	0.88	1	0.909	1	0.979	0.979	0.691305	9
F10	0.83	1	0.864	1	0.894	1	0.912	1	0.931	1	0.9609	0.9609	0.729697	15
F11	0.815	1.0053	0.851	1.0036	0.881	1.0019	0.899	1.0004	0.92	1	0.9506	0.9506	0.712316	14
F12	0.756	1.0017	0.793	1	0.828	1.0002	0.852	1	0.879	1	0.9259	0.9259	0.650137	2
F13	0.81	1.0015	0.845	1.0096	0.876	1.0079	0.893	1.0054	0.62	1.0031	1.0008	1.0008	0.662258	5
F14	0.83	1	0.864	1	0.894	1	0.912	1	0.931	1	0.9609	0.9609	0.729697	16
F15	0.772	1	0.813	1.0004	0.854	1.0024	0.882	1.0011	0.914	1.0013	1.0009	1.0009	0.698643	11
F16	0.81	1.0014	0.845	1	0.875	1	0.89	1	0.915	1	0.9508	0.9508	0.70452	13

REFERENCES

Ahmed, A., Woulds, C., Drake, F., & Nawaz, R. (2018). Beyond the tradition: Using Fuzzy Cognitive Maps to elicit expert views on coastal susceptibility to erosion in Bangladesh. Catena, 170, 36-50.

Ali, N., Hwang, S., & Hong, J. E. (2019). Your opinions let us know: mining social network sites to evolve software product lines. KSII Transactions on Internet and Information Systems (TIIS), 13(8), 4191-4211.

Andersen, P., & Petersen, N. C. (1993). A procedure for ranking efficient units in data envelopment analysis. Management science, 39(10), 1261-1264.

Biesialska, K., Franch, X., & Muntés-Mulero, V. (2021). Mining dependencies in large-scale agile software development projects: A quantitative industry study. In Evaluation and Assessment in Software Engineering (pp. 20-29).

Bloch, M., Blumberg, S., & Laartz, J. (2012). Delivering large-scale IT projects on time, on budget, and on value. Harvard Business Review, 5(1), 2-7.

Bright, P. (2015). What windows as a service and a 'free upgrade' mean at home and at work. Available from: https://goo.gl/ Fzwflg.

Chen, C. B., & Klein, C. M. (1997). A simple approach to ranking a group of aggregated fuzzy utilities. IEEE transactions on systems, man, and cybernetics, Part B (Cybernetics), 27(1), 26-35.

Chen, Y. C., Chiu, Y. H., Huang, C. W., & Tu, C. H. (2013). The analysis of bank business performance and market risk—Applying Fuzzy DEA. Economic modelling, 32, 225-232.

Choetkiertikul, M., Dam, H. K., Tran, T., & Ghose, A. (2015, November). Predicting delays in software projects using networked classification (t). In 2015 30th IEEE/ACM international conference on automated software engineering (ASE) (pp. 353-364). IEEE.

Choetkiertikul, M., Dam, H. K., Tran, T., & Ghose, A. (2017). Predicting the delay of issues with due dates in software projects. Empirical Software Engineering, 22(3), 1223-1263.

Dias, S. B., Hadjileontiadou, S. J., Hadjileontiadis, L. J., & Diniz, J. A. (2015). Fuzzy cognitive mapping of LMS users' quality of interaction within higher education blended-learning environment. Expert systems with Applications, 42(21), 7399-7423.

Ferreira, S., Collofello, J., Shunk, D., & Mackulak, G. (2009). Understanding the effects of requirements volatility in software engineering by using analytical modeling and software process simulation. Journal of Systems and Software, 82(10), 1568-1577.

Hebb, D. O. (1949). The organization of behavior: a neuropsychological theory. Science editions.

Hwang, C. L., & Lin, M. J. (2012). Group decision making under multiple criteria: methods and applications (Vol. 281). Springer Science & Business Media.

Jetter, A. J., & Kok, K. (2014). Fuzzy Cognitive Maps for futures studies—A methodological assessment of concepts and methods. Futures, 61, 45-57.

Kosko, B. (1986). Fuzzy cognitive maps. International journal of man-machine studies, 24(1), 65-75.

Ma, Z., Collofello, J. S., & Smith-Daniels, D. E. (2000, March). Improving software on-time delivery: an investigation of project delays. In 2000 IEEE Aerospace Conference. Proceedings (Cat. No. 00TH8484) (Vol. 4, pp. 421-434). IEEE.

Mahdi, M. N., Mohamed Zabil, M. H., Ahmad, A. R., Ismail, R., Yusoff, Y., Cheng, L. K., ... & Happala Naidu, H. (2021). Software project management using machine learning technique—A Review. Applied Sciences, 11(11), 5183.

Maimone, C. (2019). Good Enough Project Management Practices for Researcher Support Projects. In Proceedings of the Practice and Experience in Advanced Research Computing on Rise of the Machines (learning) (pp. 1-8).

Mei, S., Zhu, Y., Qiu, X., Zhou, X., Zu, Z., Boukhanovsky, A. V., & Sloot, P. M. (2013). Individual decision making can drive epidemics: a fuzzy cognitive map study. IEEE Transactions on Fuzzy Systems, 22(2), 264-273.

Oun, T. A., Blackburn, T. D., Olson, B. A., & Blessner, P. (2016). An enterprise-wide knowledge management approach to project management. Engineering Management Journal, 28(3), 179-192.

Papageorgiou, E. I., Stylios, C. D., & Groumpos, P. P. (2004). Active Hebbian learning algorithm to train fuzzy cognitive maps. International journal of approximate reasoning, 37(3), 219-249.

Papageorgiou, E. I., Stylios, C., & Groumpos, P. P. (2006). Unsupervised learning techniques for fine-tuning fuzzy cognitive map causal links. International Journal of Human-Computer Studies, 64(8), 727-743.

Papageorgiou, E. I., & Kannappan, A. (2012). Fuzzy cognitive map ensemble learning paradigm to solve classification problems: Application to autism identification. *Applied Soft Computing*, *12*(12), 3798-3809.

Papageorgiou, E. I., Subramanian, J., Karmegam, A., & Papandrianos, N. (2015). A risk management model for familial breast cancer: A new application using Fuzzy Cognitive Map method. *Computer methods and programs in biomedicine*, *122*(2), 123-135.

Poczeta, K., Papageorgiou, E. I., & Gerogiannis, V. C. (2020). Fuzzy cognitive maps optimization for decision making and prediction. *Mathematics*, *8*(11), 2059.

S. Group, "Chaos report," West Yarmouth, Massachusetts: Standish Group, Tech. Rep., 2004.

Saleem, N. Empirical analysis of critical success factors for project management in global software development. *In Proceedings of the 2019 ACM/IEEE 14th International Conference on Global Software Engineering (ICGSE), Montreal, QC, Canada,* 25–26 May 2019; pp. 68–71.

Sommerville, I. Software Engineering, 9th ed.; Pearson: London, UK, 2011; ISBN 0137035152.

Tiwana, A., & Keil, M. (2004). The one-minute risk assessment tool. *Communications of the ACM*, *47*(11), 73-77.

Tone, K. (2001). A slacks-based measure of efficiency in data envelopment analysis. *European journal of operational research*, *130*(3), 498-509.

VINOD, S. (2019, January 23). How Significant is Project Management in Software Development? https://project-management.com/how-significant-is-project-management-in-software-development/

Yager, R. R. (1981). Concepts, theory, and techniques a new methodology for ordinal multiobjective decisions based on fuzzy sets. *Decision Sciences*, *12*(4), 589-600.

Zadeh, M. T., & Kashef, R. (2022). The Impact of IT Projects Complexity on Cost Overruns and Schedule Delays. *arXiv preprint arXiv:2203.00430*.

Zimmermann, H. J. (1975). Description and optimization of fuzzy systems. *International journal of general System*, *2*(1), 209-215.