International Journal of Engineering Research & Management Technology

ISSN: 2348-4039

March-April-2023 Volume 10, Issue-2

Email: editor@ijermt.org www.ijermt.org

PSO-BASED FUZZY SYSTEM

Ambarish Kumar Patel

Research Scholar, Kalinga University

Dr Dev Ras Pandey

Department of Computer Science,Kalinga University

ABSTRACT

The process of developing software is constantly confronted with unforeseeable occurrences, one of which is the potential for schedule slippage and cost overruns. When it comes to project management, one of the most challenging challenges is arriving at an accurate estimation of the amount of work that is necessary to create software systems. In addition, the estimate of software effort has been identified as one of the three most significant difficulties that have emerged in the field of computer science over the course of more than half a century. Assessing the entire costs of developing a software product, which includes the amount of money, physical and technical resources, as well as the amount of time and effort spent on the creation of the product, is of the utmost significance and may bring significant financial and strategic advantages to software organisations. It is possible that having correct effort evaluations, particularly during the early stages of a software project, might greatly lessen the impact of the high risks that are implemented throughout the process of developing a software product. It is unfortunate that the high degree of uncertainty and volatility of project characteristics that impact development concerns may result in extremely inaccurate predictions of the amount of work that will be required to complete a project when it is first started. In addition, the inherent uncertainties that are present within project measures, which mostly depend on the subjective judgements of humans, have a significant impact on the accuracy of estimates of effort. In accordance with what Boehm argued for more than twenty years ago when he presented his "cone of uncertainty," the degree of uncertainty may cause estimates to be up to four times greater (or lower) than the actual cost, and this uncertainty reduces as the project moves closer and closer to completion.

INTRODUCTION

In addition, the dynamic nature of the software process often makes it difficult for conventional methods to provide a reliable estimation of the amount of effort required, and as a result, the SCE process continues to be extremely susceptible to human mistakes and biases. The unique internal social networks, the culture and environment inside each organisation, the quick change of technologies that are employed, the moving of people and staff, the activity of outsourcing, and the unique requirements and circumstances of the systems that are developed are all factors that contribute to the dynamic environment. The status, progress, productivity rates, resource allocation, management choices, management decisions, expenses, and timelines of projects are all subject to significant fluctuations as a consequence of these factors. However, one of the most significant challenges that must be overcome in order to create empirical cost estimate models is the form of the available software data samples that they depend on, which is mostly categorical and not well defined. This is one of the key hurdles that must be overcome in reality. In order to achieve accurate and reliable effort estimates, numerous methods in the relevant literature make use of a wealth of such historical project data in conjunction with numerical transformations. For instance, traditional parametric methods make use of project data in order to construct models that relate particular project attributes with effort by employing mathematical formulas. These data-driven models, on the other hand, are unable to include numerical and categorical variables into formulae, and because of this, they often result in equations that are quite complicated. In addition, they do not deal with the imprecision that is present in the data samples, and they do not provide a reasoning process that can understand the effort that is produced in a fashion that is comparable to the way that people think about and relate information. While it is highly necessary for the estimator (and typically the project manager) to be able to grasp the logic behind the estimate as well as the approach that was done, it is also very vital for them to be able to evaluate the outcome before they implement it. Although both the accuracy of the models and their explanatory value are regarded to be highly essential characteristics, the majority of the research that have been conducted on SCE have focused on the accuracy of the models up to now. It is therefore a highly desired research endeavour to conduct an investigation into methods that are capable of managing categorical data, to overcome limitations such as the uncertainty of the metrics that are used (including the estimate), which are characterised by imprecision and vagueness, and additionally, to generate an interpretable output that is easily understood by the end-users.

FUZZY ANALOGY

An example of a fuzzification of the classical analogy is the fuzzy analogy, which is utilised in situations where the data is in the form of numerical or categorical information. There is a similarity between fuzzy set theory and fuzzy logic, which is used in situations where decision making is difficult. Fuzzy logic is dependent on human actions and interpretation. The most important steps involved in estimating the amount of work required for a software project through analogy are the following: the identification of a candidate software project as a new case; the retrieval of comparable software projects from a database; and the recycling of information obtained from earlier software projects in order to produce an estimate of the amount of work required for the software project. There is a fuzzy set representation of these values used.

SOFTWARE PROJECT ESTIMATION

The first step in the process of project searching, accounting, and planning is the creation of software estimates. In the event that resources and plans are not invested with sufficient enthusiasm, company opportunities are lost, which in turn leads to significant losses. An estimate of the software project is comprised of three stages, which are the assessment of the size, the estimation of the work, and the calculation of the cost. The process of determining the most practical application of the work that is required for extending or supporting the development of the software project is referred to as the software application effort estimate. When evaluating the amount of work that needs to be done in terms of Person-Months (PM) for the Software enhancement task components of the Work Breakdown Structure (WBS), effort estimations are being employed.

Size Estimation

The software size estimate is the starting stage to develop successful project estimation. The customer's demands and system requirement establish a baseline for defining the size of the programme. Then, the system design document includes further information for projecting the complete size of the programme. Size estimate is done out using the following stages;

- Through the use of historical data from a system that has been deployed in the past, a technique known as analogous estimating is used to estimate the size of the project.
- The second method of estimate is calculated based on the characteristics and functionalities of the product. An analysis of the size of each subsystem is performed when the system is partitioned into a number of subsystems according to their respective features.

Effort Estimation

After the procedure of determining the size has been completed, the subsequent step is to ascertain the quantity of labour that will be necessary to complete the task. When determining the quantity of work that has to be done, the organizational particulars of the software development life cycle are taken into consideration. The efforts that are made by the project in respect to the specifications for deliverables are what determine whether or not any application software system is developed once the project has been completed. The phases that are stated below are the ones that are used to carry out the evaluation of the endeavor,

- Measuring prior projects of the expected size is the most effective method for determining the amount of work that was put into a project.
- Multiple models that are based on the algorithmic method are used in order to estimate the amount of work that will be required to complete the project in the event that it is of a varied nature and requires the organisations to implement a certain strategy.

Cost Estimation

When evaluating the cost of a software project, a variety of factors are taken into account. These factors include the cost of hardware, travel, operating expenditures, communications, and advice. In addition to the forecasts of the size and the quantity of work, these characteristics provide additional information. It is referred to as the software cost estimate technique, and it is a method that is used for the aim of generating software cost estimates. The collection of input data is provided to the process of cost estimate in order to accomplish the goal of identifying the levels of effort, duration, and loading that are connected with a software project. Numerous software cost estimating representations define the estimate process as a function that is evaluated based on a variety of distinct cost drivers. This is the case for the great majority of these representations. When it comes to the process of cost assessment, the aspects that are considered to be the most significant cost drivers are the specific software requirements. The process of estimating costs results in the production of two outputs, which are as follows:

- The process of manpower loading involves assigning a certain number of new recruits to the project in accordance with the length of time that has elapsed from the beginning of the load.
- The amount of time that is required to finish a project is referred to as the length of the project.

BUILDING THE FUZZY COST IMPLICATION SYSTEM

In this section, the most important principles and terminology of fuzzy logic (FL) that were used in the process of developing the recommended cost estimation model are explained in a way that is clear and straightforward throughout. Taking into consideration the following comprehension, fuzzy set theory provides a definition for, Sets that have uncertain limits may be generated, and these sets are known as fuzzy sets. It is feasible to construct fuzzy sets. The extent to which components belong to a set may be declared with the use of services known as membership functions. By providing a gradual function that characterises the participation of an element rather than a clear demarcation (belongs or does not belong), fuzzy sets, in contrast to crisp sets, result in more flexible descriptions. Crisp sets are characterised by their ability to give a distinct demarcation. When compared to crisp sets, this is the fundamental advantage that fuzzy sets provide. When dealing with empirical datasets, the process of translating numerical values into fuzzy words helps to lessen the risks that are linked with the uncertainty of the project measurements. This is because fuzzy words are more difficult to understand than numerical values. Because of this, the information is presented in a manner that is simple to comprehend, and this is made possible by the Fuzzy Decision Trees (FDT) that were constructed by making use of the linguistic terms. An explanation of the technique that was used to portray the information that was obtained from the FDTs is provided in each of the three stages that are listed below: First, a fuzzification process is conducted, which is based on the suggestions provided by prior work and which eventually leads in descriptions of variables and values via the utilisation of fuzzy sets and certain types of membership functions. This process is applied in order to get the desired outcome. In the next stage, the fuzzy implication of the rules is carried out, and in the third and last step, the defuzzification process is carried out.

MODELING SOFTWARE EFFORT ESTIMATION USING HYBRID PSO-ANFIS

As a result of the fast expansion of the software business in recent years, one of the subjects that has garnered a lot of attention is the pricing of software. In the context of a software project, one of the measures of success is the estimation of the cost. For the purpose of managing financial difficulties and monitoring the activities of development and on-time delivery, it is extremely necessary to arrive at an accurate estimate for the amount of work, cost, and schedule involved in the programme. According to the findings of the Standish Group's CHAOS Report 2012, EXTREME, there were 31 percent of software projects that were unsuccessful, 43 percent of projects that encountered challenges, and just 18 percent of initiatives that were successful. Not only did the firm suffer financial losses, but the failure of the information technology project also resulted in a fall in the company's reputation. A great number of contributions have developed and proven estimating strategies in order to decrease and remove mistakes in estimation. This was done in order to obtain an accurate estimate. The highlights of this study include the creation of software assessment efforts utilising the Adaptive NeuroFuzzy Inference System (ANFIS) on the historical dataset COCOMO 81, as well as the demonstration of the relevance of this method in comparison to other machine learning techniques.

SOFTWARE COST ESTIMATION TECHNIQUES

When it comes to software firms, cost estimating is regarded to be one of the most difficult and tough activities since it involves the process or techniques that assist us in anticipating the real and complete cost that will be required for our software. They want to create software that is not only affordable but also of high quality, and this is their objective. An assessment of the cost of software is mostly used by system analysts in order to get an approximate estimate of the vital resources that are required by a certain software project as well as their schedules. Size, time, effort, and other factors are important elements to consider when determining costs. The estimating process for software is primarily comprised of four stages. First, we make an estimate of the size of the programme, then we determine the amount of work that will be required, then we derive the timetable, and last, we compute the cost of the software. During the process of estimating the cost of software, a number of different strategies are used. These techniques may be generally classified into two groups, namely algorithmic techniques and nonalgorithmic techniques. It is possible to assess the cost of software using algorithmic approaches, which depend on mathematical formulae. An algorithmic collection of models known as the Constructive Cost Model (COCOMO) is one of the most well-known and regularly used.

STRUCTURE IDENTIFICATION OF FUZZY SYSTEMS

The selection of the membership functions, the kind of inference, and the design of the rule base are the primary components used in the process of structure identification. Within the body of research, the membership functions that have been used may be divided into two primary categories.Triangular and trapezoidal membership functions are further examples of simple, piecewise linear membership functions. The derivatives of these membership functions are not continuous. The fact that they are straightforward and straightforward to implement with low-end hardware is one of their advantages. They are not amenable to optimisation via the use of systematic gradient optimisation techniques. Based on Rao's research from 1998, the offline design technique becomes more time consuming and less efficient when random optimisation methods are used.

HARDWARE IMPLEMENTATION OF FUZZY SYSTEMS

When it comes to hardware implementation, fuzzy systems present a variety of challenges that are practical in nature. Obtaining the membership value is the most challenging aspect of the situation. It is necessary to make a choice between speed, accuracy, complexity, and cost when it comes to implementation. If there is a need for a high level of system accuracy, then the computational complexity will grow. When the computing power per unit of time is increased, the cost of the system also rises. There are many different approaches that have been proposed in order to enhance the speed of calculation.

- Choice of a simple fuzzy system structure.
- Use of look-up table to obtain the membership value.
- Placing of constraints on the nature and shape of membership functions. Only simple piecewise linear membership functions should be used.
- Use of parallel processing to accelerate the computation process.
- Design of analog VLSI for fuzzy implementation.

FUZZY SYSTEMS

The processing of data is carried out using a rule-based system known as a fuzzy system, which employs fuzzy logic rather than Boolean logic. Fuzzy logic is a paradigm in computer science that offers a framework for describing and processing information in a manner that is analogous to the way in which humans communicate and reason.Fuzzy systems are structures that are directed towards information processing and are based on fuzzy approaches. These structures are described as fuzzy systems. In circumstances in which the use of conventional sets theory and binary logic is either impossible or problematic, these systems are utilised as a solution. In this context, the term "fuzzy" refers to things that are not clear or that are confusing because of their lack of clarity. In the actual world, we regularly find ourselves in predicaments in which we are unable to tell whether the condition is an accurate representation of the truth or an illusion. One of the many benefits that their fuzzy logic gives is the flexibility that it enables for thinking, which is a significant advantage. We are able to take into consideration the errors and uncertainties that are present in every particular setting as a result of this. Fuzzy logic is a sort of many-valued logic that differs from traditional logic in that it allows the truth values of variables to be any real number between 0 and 1. This is in contrast to the conventional logic, which relies entirely on the values of true or false. It is also possible to refer to fuzzy logic as fuzzy logic.

Fuzzy Sets

There is a generalization of crisp or classical sets that is known as fuzzy sets. A set having a clear border is referred to as a classical set. For instance, a classical set A in a universe of discourse, or simply universe U, may be described by naming all of its members or by setting requirements to identify the components. Both of these methods are valid.

$$x \in \boldsymbol{A} \text{ i.e.}$$

$$\boldsymbol{A}=\{x \mid x \text{ satify some condition}\}$$

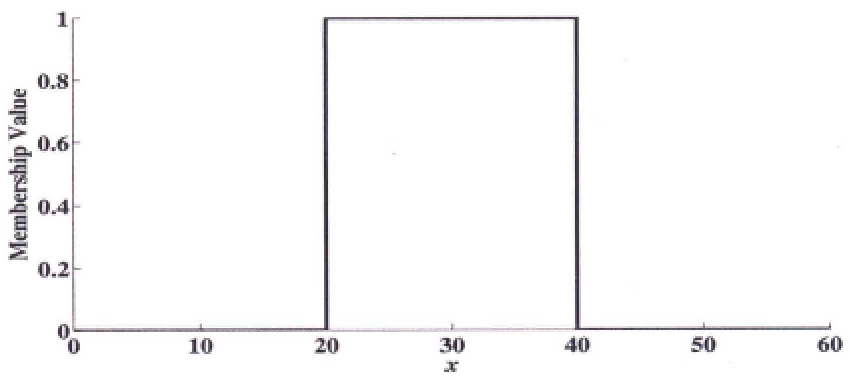
In the event that the criterion is not met, then the value z does not belong to this classification. The characteristic function, which is also often referred to as the membership function, is a mapping that is connected with A.

$$\mu_A : U \rightarrow \{0,1\} \text{ such that } \forall x \in \boldsymbol{U},$$

$$\mu_A(x) = 1, \text{ if } x \in \boldsymbol{A}$$

$$\mu_A(x) = 0, \text{ if } x \ni \boldsymbol{A}$$

illustrates the membership function (MF) that is used to express the crisp set A ={ x| 20 <= x <= 40}



crisp sets defined as A ={ x| 20 <= x <= 40}

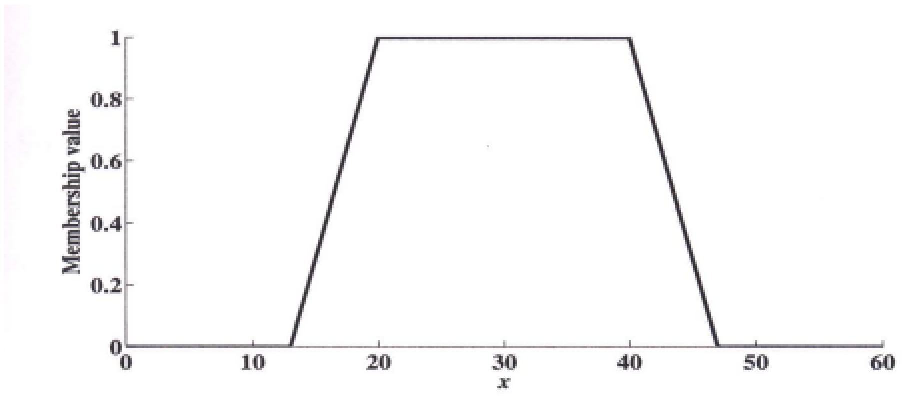
A fuzzy set, on the other hand, is a set that does not have a clear border and in which the transition from belonging to a set to not belonging to a set is slow and seamless. A membership function that maps the elements of a domain, space, or universe of discourse U to the unit interval is what distinguishes a fuzzy set F from other sets [0,1]

$$\mu_F : U \rightarrow [0,1]$$

It is possible to express a fuzzy set Fin U as a collection of ordered pairs consisting of a generic element ZE U and the grade of membership it has:

$$\boldsymbol{F} = \{(\mu_F(x)/x | x \in \boldsymbol{U}\}$$

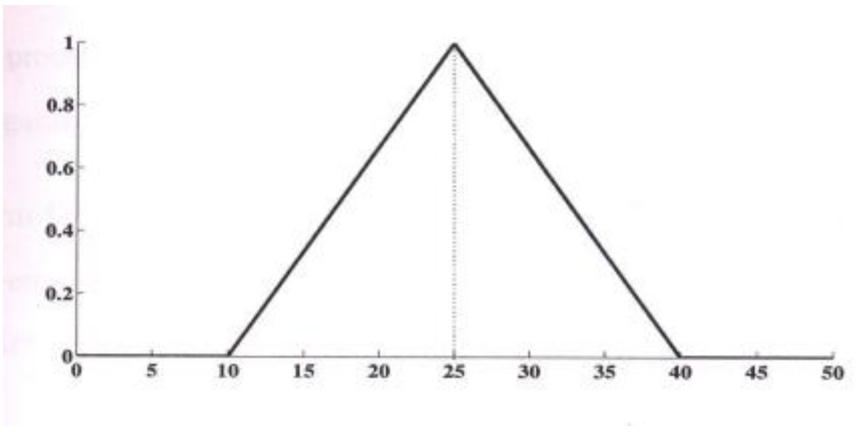
The process of assigning a membership value µF (2) to a given value is referred to as fuzzification since it is an operation. In Figure 1.2, the membership function of the fuzzy array is shown set F = {x| x is between 20 and 40}.



fuzzy set defined as F = {x| x is between 20 and 40}.

It is possible for membership functions to assume any shape that is arbitrary. Membership functions describe element-wise membership criteria. It is possible to represent a membership using a mathematical formulation, which is a useful approach to explain a membership. The definition of membership functions is often accomplished via the use of parameterized functions. This is due to the fact that these functions play a significant part in adaptive fuzzy systems, where the parameters are set by some learning process.

Taking a triangular membership function as an example, which is one of the most often used functions, the parameters a, b, and c are utilised to specify the function. A triangle membership that is defined by 10, 25, and 40 is seen in Figure 1.3. The breadth of the membership function may be adjusted by varying the values of the parameters a and c, while the centre of the membership function can be chosen by adjusting the value of the parameter b. Trapezoidal, Gaussian, and bell-shaped membership functions are some of the other well-known membership functions.



triangular membership function reprentation

Fuzzy Operations

In the same way as the conventional set operations of union, intersection, and complement are defined in classical set theory, fuzzy sets also have operations that are comparable to those operations. The traditional set operations of intersection and union are analogous to the logic operations of conjuction and disjunction, which are correspondingly applicable to fuzzy sets accordingly. Both the fuzzy conjunction and the fuzzy disjunction are available in a wide variety of options. The use of the minimum for conjunction and the maximum for disjunction is the most usual option.When it comes to operators, fuzzy disjunction and fuzzy conjunction are also known as t-conorm and t-norm operators. These terms are used interchangeably in the literature.When it comes to fuzzy systems, the fuzzy complement (negation) operator is generally defined as described below:

$$\mu\lambda(2) = 1 - \mu A(2)$$

Fuzzy Inference

On the other hand, the process of drawing conclusions based on facts and logic is referred to as fuzzy inference. The fuzzy reasoning mechanism is comprised of three sub-processes: implication, aggregation, and defuzzification. Both of these processes are active participants.Observation (Fuzzy If-Then Rules): An Implication The subjects and verbs of fuzzy logic are fuzzy sets and fuzzy operators, and together they convey knowledge in the form of If-Then rules. Fuzzy logic is a subfield of logic that focuses on fuzzy logic.Following is an example of a fuzzy rule:If (input fuzzy condition) then (output fuzzy conclusion) The antecedent (or premise) of the input condition and the output conclusion are referred to as the antecedent and consequent, respectively.

The following is a definition of a fuzzy rule:

If (x is A) then (z is C)

Specifically, the letter ae is used to represent the antecedent of this rule, while the letter z is used to represent the consequent of this rule. Language values A and C are both given as fuzzy sets across their respective universes of conversation. Both of these values are linguistic characteristics.

A mapping or an implication operation is provided by each fuzzy rule in its most basic form A→ C for a given input value z to fuzzy set C" =μA→C(x) The most commonly used implication operators are: Mamdani, Larsen, Lukasiewicz, drastic product, Zadeh, bounded product and Godelian.

A rule does not fire if its antecedent returns an empty implication i.e. μ A (x) = 0. Consequently, the inference process is not affected by such rules since they do not contribute to it.

The antecedents of a fuzzy model are defined by the fuzzy partition of input space, and the number of partitions in the model determines the number of fuzzy rules that are included inside the model. When it comes to fuzzy partitioning, the grid partitioning method is the one that is used the most often. A uniform grid partition of a two-dimensional input space is seen in Figure 1.4(a). This partition is produced by three symmetric membership functions that are evenly spaced apart, which results in the definition of nine rules that are (3x3). A non-uniform grid partition is generated by non-uniformly positioned asymmetric membership functions, This partition is not uniform. An adaptive fuzzy partition is often used, in which the initial partition is achieved via the utilisation of a uniform grid that is optimised through the utilisation of a learning method such as gradient-descent. Although grid partition is fairly simple to use, it is susceptible to the rule-explosion issue when there are a high number of variables that are introduced into the system. The term "curse-of-dimensionality" refers to this rule-explosion concern that has arisen.

Take into consideration a multi-input single-output system that has n number of inputs and the number of fuzzy sets for the inputs to, and consider the following be m_1 , m_2 , m_3 ,m_4 , m_5,...,$m_n$respectively.

For example, for n = 5 and $m_1 = m_2 = m_3 = m_4 = m_5 = 3$

Number of fuzzy rules = $\prod_{i=1}^{n} m_i = 3^5 = 243$

In the event that an additional input including three sets of fuzzy sets is supplied, the total number of fuzzy rules would increase to 729 (3^6).

the clustering strategy divides the input and output spaces into many subspaces. Each of these subspaces may be represented by a rule, as indicated in the figure. This strategy, in contrast to the grid-type division, results in a reduction in the number of fuzzy rules that are created.

Aggregation: Due to the fact that fuzzy systems are essentially parallel systems, it follows that all of the fuzzy rules are assessed at the same time. As a result of the overlap between membership functions, many rules would fire concurrently with varying degrees of intensity, and the output of each rule would result in a fuzzy collection of outputs. A single fuzzy set is created by the process of aggregation, which involves collecting all of the fuzzy sets that reflect the output of each rule and combining them into a single cloud. The maximum operator is the aggregation operator that is used the most often.

Defuzzification: An interpretation of the qualitative nature is included in a single output fuzzy set that is generated by aggregating rule executions. A crisp value must be extracted due to the fact that the majority of applications need it. The term "defuzzification" refers to the process that takes a fuzzy set and generates a value that is clear and distinct. The Centre of Area/Gravity, the Centre of Sums, the Centre of Largest Area, and Weighted Average are some of the defuzzification procedures that are used the great majority of the time. The defuzzification of the middle of the maximum and the height.

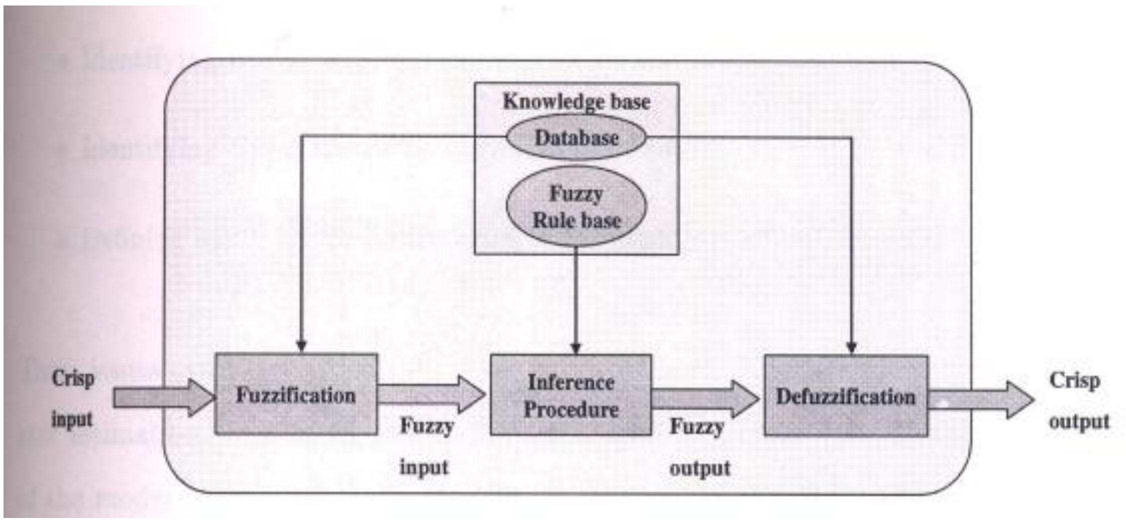
FUZZY INFERENCE SYSTEM

The actual process of mapping from a given set of input space(s) to the output space(s) via the use of fuzzy logic is referred to as fuzzy inference. A fuzzy inference system is any system whose operation is based on the principles of fuzzy set theory, fuzzy if-then rules, and fuzzy reasoning. The phrase "fuzzy inference system" is used to describe any such system. Other names for the fuzzy inference system include fuzzy rule-based system, fuzzy expert system, fuzzy associative memory, and fuzzy logic controller. These are only some of the names that are used to refer to this methodology.

A fuzzy inference system may be described as shown in Figure 1.5, which depicts the fundamental architecture of the system. The following are the primary conceptual components that make up this component: The first component is a knowledge base, which includes fuzzy rules and a database or dictionary that defines the membership functions. The second component is an inference mechanism, which uses a fuzzy reasoning process to create a fuzzy output. The third component is a DE fuzzifier, which converts the fuzzy output into a crisp value.

FUZZY MODELING PROBLEM

Generally speaking, there are two ways to modelling: the expert or knowledge-driven approach, and the data-driven approach. The expert-driven modelling technique, also known as the heuristic-based modelling strategy, is a modelling approach that continues Zadeh's theory by attempting to construct a fuzzy model that directly utilises the domain knowledge of experts. The unfortunate reality is that there is no universal technique that can be used for the execution of this modelling approach,



besic architecture of a fuzzy inference system

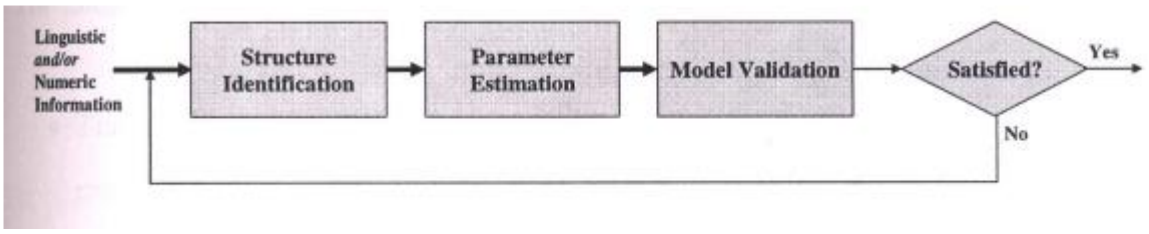
This is more of an art that is dependent on experience and intuition. When the available information is insufficient or when the issue space is too big, this modelling strategy becomes more challenging. Despite the fact that this design technique has resulted in a significant number of applications that have been successful, it is time-consuming and has been exposed to criticism due to the fact that it does not include any principles or systematic procedures. On the other hand, when it comes to the data-driven modelling technique, there is no previous knowledge of the system that is being considered in order to develop the rules. The numerical information that is received from input-output measurements is used in this modelling technique. In addition to this, it is feasible to combine the two different modelling methodologies.

Some of the challenges that are included in the problem of fuzzy model identification are as follows:

- In the process of choosing the fuzzy model type
- The process of choosing the variables that will be used in the model
- Determining the organizational framework of membership functions
- Finding out how many hazy rules there are overall
- The process of determining the parameters of the antecedent and consequent membership functions
- Identifying the parameters of regulations that are therefore imposed
- In order to evaluate fuzzy models, it is necessary to define certain performance criteria.

In accordance with the information shown in Figure 1.6, these challenges may be broken down into three distinct subproblems: structure identification, parameter estimates, and model validation. In the event that the performance of the model that was produced is not sufficient, the structure of the model is altered, and the parameters under consideration are re-estimated until the performance is satisfactory.

When there are an excessive number of inputs, the rule base explodes, which in turn increases the amount of computational complexity and memory that this process requires. Therefore, it is essential to include the majority of the relevant inputs.



fuzzy model identification process

CONCLUSION

In the process of identifying fuzzy models, one of the most essential challenges is to discover a reasonable trade-off between the training data and to keep the model simple by employing fewer rules. This is one of the major difficulties. The capacity of a model to approximate testing data from the same process that was not utilised in training the model is defined by the generalization capability, which is provided by a simple model. A simple model is more clear and more controllable, and it also offers the generalization capability. On the other hand, maintaining the model's simplicity could lead to a lack of precision. On the other hand, a fuzzy model that has a significant number of rules is likely to run into the danger of overfitting, which indicates that the model is unable to make suitable judgements when it is presented with test data.