CHAPTER 5

ESTIMATING SOFTWARE COST USING PSO BASED FUZZY SYSTEM

This chapter describes the software cost estimation using Particle Swarm Optimization (PSO) algorithm based on Fuzzy system.

5.1 INTRODUCTION

Effective cost estimation shells as the most challenging task in software cost estimation. In this chapter the proposed novel approach which is based on fuzzy logic coupled with the optimization task to estimate effort is studied. In effect, Fuzzy logic-based cost estimation models have showed its determination as the most competent model for the cost estimation in software engineering; this research tends to the inclusion of the optimization algorithm in fuzzy which further boosts the estimation process.

5.2 SOFTWARE COST ESTIMATION

Software cost estimation is the process of predicting the effort required to develop a software system. The estimation of effort in software is more challenging task and numerous researches where carried out in this field from last few decades. During the development process, the cost and
time estimates are useful for the initial rough validation and monitoring of the project's progress after completion, these estimates may be useful for project productivity assessment. Cost estimation can help in designing completely effective software.

In recent years, software has become the most expensive component of computer system projects. The majority cost of software development is due to the human effort, and most cost estimation methods focus on this aspect and give estimates in terms of person-months.

Accurate software cost estimates are critical to both developers and customers. They can be used for generating request for proposals, contract negotiations, scheduling, monitoring and control. Underestimating the costs may result in management approving proposed systems which then exceed their budgets, with underdeveloped functions and poor quality, and failure to complete on time. Overestimating may result in too many resources committed to the project, or, during contract bidding, result in not winning the contract, which can lead to loss of jobs.

Software cost estimation involves the determination of one or more of the following estimates:

1. Effort (usually in person-months)
2. project duration (in calendar time)
3. cost (in dollars)

In the proposed method the study uses fuzzy rule based software cost estimation with optimization algorithm incorporated to this fuzzy logic for optimizing the fuzzy rules. Figure 5.1 shows the general process of cost estimation in software.
5.3 SOFT COMPUTING BASED COST ESTIMATION

The basic input for the cost model is size measured in terms of Kilo Delivered Lines of Code (KDLOC) and set of Cost parameters. The advantage of cost estimation is Cost benefit analysis, proper resource utilization (software, hardware and people), staffing plans, functionality trade-offs, risks and modify budget. The software cost estimation problem deserves a special attention because of development of product is unique which undertakes results in uncertainty, with increased size of software projects estimation, mistakes could cost a lot in terms of resources allocated to the project. Early stage cost estimations can be defined as to make software development effort estimations at the initial stages more precisely and also during the design stage of SDLC. Carrying out cost estimations at the early stages is beneficial because the design stage prediction implies fewer overheads at the later stages of software development. The soft computing has become the major technique for estimating the cost values in software engineering in the recent years.
5.3.1 Soft Computing

Soft computing is a consortium of methodologies that works synergistically and provides, in one form or another, flexible information processing capability for handling real-life ambiguous situations. It aims to exploit the tolerance for imprecision, uncertainty, approximate reasoning, and partial truth in order to achieve tractability, robustness, low-cost solutions, and close resemblance to human-like decision making. The guiding principle is to devise methods of computation that lead to an acceptable solution at low cost by seeking an approximate solution to imprecisely/precisely formulated problem.

In real world, there are many problems which has no way to solve logically, or problems which could be solved theoretically, but actually impossible due to its requirement of huge resources and huge time for computation. For these problems, methods motivated by nature sometimes work very efficiently and effectively. Although the solutions obtained by these methods do not always equal to the mathematically strict solutions, a near optimal solution is sometimes enough in most practical purposes. These biologically inspired methods are called Soft Computing. Soft Computing is an umbrella term for a collection of computing techniques. This is based on natural as well as artificial ideas. It is referred as a computational intelligence.

Traditionally soft computing has been comprised by four technical disciplines. The first two, probabilistic reasoning (PR), and fuzzy logic (FL) reasoning systems, are based on knowledge-driven reasoning. The other two technical disciplines, Neuro Computing (NC) and Evolutionary Computing are data – driven search and optimization approaches.
Soft computing techniques have been recognized as attractive alternatives to the standard, well established hard computing paradigms. Soft computing is still in its initial stages of crystallization. Soft computing techniques, in comparison with hard computing employ different methods which are capable of representing imprecise, uncertain and vague concepts. Soft computing techniques are able to handle non-linearity and offers computational simplicity. The categories of soft computing are shown in the Figure 5.2.

![Figure 5.2 Soft Computing Categories](image-url)
In the proposed method, Fuzzy logic has been combined with the evolutionary algorithm - Particle Swarm Optimization (PSO). The incorporation of the particle swarm optimization is done in the fuzzy rule selection phase so that optimized set of rules are generated at the fuzzy output. The fuzzy process performed here is similar to that of the normal fuzzy method except that the rules are being optimized in order to deliver better set of rules for processing. The fuzzification stage is followed by the rule creation stage where the study incorporates the PSO algorithm. The PSO algorithm optimizes the rules based on the fitness value and delivers optimized set of rules which are utilized in the further processing of the proposed method.

5.4 Fuzzy Logic

Fuzzy logic is a universal approximate of any multivariate function because it can be used for modeling highly non-linear, unknown or partially known controllers, plants or processes. Fuzzy logic helps an engineer for solving non-linear control problems in process control applications. It emulates human reasoning and provides an intuitive way to design functional block for an intelligent control system.

Fuzzy logic attempts to systematically and mathematically emulate human reasoning and decision making. It provides an intuitive way to implement control systems, decision making and diagnostic systems in various branches of industry. Fuzzy logic represents an excellent concept to close the gap between human reasoning and computational logic. Variables like intelligence, credibility, trustworthiness and reputation employ subjectivity as well as uncertainty. They cannot be represented as crisp values, however their estimation is highly desirable.
Fuzzy systems are emerging technologies targeting industrial applications and added a promising new dimension to the existing domain of conventional control systems. Fuzzy logic allows engineers to exploit their empirical knowledge and heuristics represented in the IF-THEN rules and transfer it to a functional block. Fuzzy logic systems can be used for advanced engineering applications such as intelligent control systems, process diagnostics, fault detection, decision making and expert systems. The basic fuzzy system process is shown in below Figure 5.3.

![Diagram of Fuzzy Logic System](image)

**Figure 5.3  Basic configuration of Fuzzy logic system**

### 5.5 PARTICLE SWARM OPTIMIZATION (PSO)

Optimization is the process of adjusting the inputs to find the minimum or maximum output or result. This mechanism is used in fields such as physics, chemistry, economics, and engineering where the goal is to
maximize efficiency, production or some other measure. Optimization can refer to either minimization or maximization; maximization of a function $f$ is equivalent to minimization of the opposite of this function. Optimization consists in trying variations on an initial concept and using the information gained to improve on the idea.

PSO is an algorithm that simulates the social behaviours of bird flocking or fish schooling and the methods by which they find roosting places, food sources or other suitable habitat. The algorithm maintains a population potential where each particle represents a potential solution to an optimization problem. The PSO algorithm works by simultaneously maintaining several candidate solutions in the search space. During each iteration of the algorithm, each candidate solution is evaluated by the objective function being optimized, determining the fitness of that solution. Each candidate solution can be thought as a particle “flying” through the fitness landscape finding the maximum or minimum of the objective function.

Initially, the PSO algorithm chooses candidate solutions randomly within the search space. The initial state of a six-particle PSO algorithm seeking the global maximum in a one-dimensional search space is shown in the Figure 5.4. The search space is composed of all the possible solutions along the x-axis. The curve denotes the objective function. PSO algorithm has no knowledge of the underlying objective function. Thus has no way of knowing if any of the candidate solutions are near to or far away from a local or global maximum. PSO algorithm uses the objective function to evaluate its candidate solutions and operates upon the resultant fitness values. Each particle maintains its position, composed of the candidate solution with its evaluated fitness and its velocity.
Additionally, it remembers the best fitness value it has achieved so far during the operation of the algorithm, referred to as the individual best fitness and the candidate solution that achieved this fitness, referred to as the individual best position or individual best candidate solution. Finally, PSO algorithm maintains the best fitness value achieved among all particles in the swarm. It is called the global best fitness. The candidate solution that achieved this fitness is called the global best position or global best candidate solution.

5.6 COST ESTIMATION USING FUZZY LOGIC

Fuzzy logic will enable the estimation process to handle the abstraction of the information acquired in early phases of a software
development process. It will help manage the insecurity about the precise meaning of semantic values used during the estimation process.

5.6.1 Fuzzy logic

Fuzzy logic can be conceptualized as a generalization of classical logic. Modern fuzzy logic developed to model problems in which imprecise data must be used or in which the rules of inference are formulated in a very general way making use of diffuse categories. In fuzzy logic, which is also sometimes called diffuse logic, there are not just two alternatives but a whole continuum of truth values for logical propositions. The difference between classical and fuzzy sets is established by introducing a membership function.

Fuzzy logic consists of the following three stages:

i) Fuzzification

ii) Inference

iii) Defuzzification

The fuzzifier transforms the input into linguistic terms using membership functions that represent how much a given numerical value of a particular variable fits the linguistic term being considered. The fuzzy inference engine performs the mapping between the input membership functions and the output membership functions using fuzzy rules that can be obtained from expert knowledge of the relationships being modeled. A defuzzifier carries out the Defuzzification process to combine the output into a single label or numerical value as required.
The cost estimation in the proposed method is depicted in the Figure 5.5 as a flow diagram.

Figure 5.5  Proposed software cost estimation model.
5.6.2 Fuzzy triangular Membership function

The attributes having numerical values in the XML database is transformed into the fuzzy using the triangular membership function. Membership functions can either be chosen by the user randomly or be designed using machine learning methods like artificial neural networks, genetic algorithms and others. There are different shapes of membership functions; triangular, trapezoidal, piecewise-linear, Gaussian, bell-shaped, etc. This study chose the Triangular membership function in which \( p, q \) and \( r \) represents the \( x \) coordinates of the three vertices of \( f(x) \) in a fuzzy set where, \( p \) is the lower boundary and \( r \) is the upper boundary where membership degree is zero, \( q \) is the center where membership degree is 1. One of the key issues in all fuzzy sets is how to determine fuzzy membership functions,

- The membership function fully defines the fuzzy set.
- A membership function provides a measure of the degree of similarity of an element to a fuzzy set.
- Membership functions can take any form, but there are some common examples that appear in real applications.

The formula used to compute the membership values is depicted as below,

\[
f(x) = \begin{cases} 
0 & \text{if } x \leq p \\
\frac{(x - p)}{(q - p)} & \text{if } p \leq x \leq q \\
\frac{(r - x)}{(r - q)} & \text{if } q \leq x \leq r \\
0 & \text{if } x \geq r 
\end{cases}
\]  

(5.1)
where,

$p, q$ and $r$ - $x$ coordinates of the three vertices of $f(x)$ in a fuzzy set.

$p$ - Lower boundary

$r$ - Upper boundary where membership degree is zero

$q$ - center where membership degree is 1

The below Figure 5.6 shows a triangular membership function for a single fuzzy set. Here, it is understood that at $p$ and $r$ the value is zero, and it reaches steadily to a maximum of value 1 at the center point $q$ between the $p$ and $r$.

![Triangular membership function](image)

Figure 5.6 Triangular membership function

The below Figure 5.7 shows the plot considering all the three membership functions having overlapping values. In this figure, the curves for low, medium and high are shown for the software efforts.
Figure 5.7  Triangular membership function with defined parameters and their values

By using the fuzzy membership formula, the numerical attributes are transformed into the fuzzy.

5.7 FUZZY RULE OPTIMIZATION USING PSO

A set of rules have been defined for a fuzzy arbiter, where it outputs the software efforts in accordance with the attributes fuzzy value. The software efforts are grouped into different rules based on the parameters and in the proposed method the utilization of optimization procedure is done in order to optimize the fuzzy rules. The mighty Particle Swarm optimization is used for optimizing the rules.
The various parameters used for the fuzzy rule generation are,

- TeamExp
- Length
- Transactions
- Entities
- Effort

These parameters are utilized for the fuzzy rule generation. The optimization of these rules are carried out using the PSO.

The rules that are generated from the fuzzification are subjected to optimization technique in order to select the exact rule which suites the proposed model of cost estimation. The optimization is done with the help of PSO which is one of the better evolutionary algorithms. PSO owes its origin to the recreation of community conduct of birds in a herd. In this technique, every particle hurries in the pursuit space with a velocity adapted by its own flying memory and its companion’s flying practice. Every particle has its unbiased function value which is determined by a fitness function. PSO is an evolutionary computing method similar to that of the Genetic Algorithm in which a specific system is initialized by a population of arbitrary solutions.

Every particle goes along its coordinates in the dilemma space with reference to the superlative key. Moreover, the fitness value is also taken into account for the supplementary procedure. This fitness value is designed as $pbest$. The position of these solutions is deemed as $gbest$. In this ambitious method, an innovative procedure by means of a tailored edition of PSO is taken in to account. Here, the defective cases as well as the finest
cases are allocated in addition to the insertion of cross over process after the robustness choice; this is with an eye on enhancement of $e$ the probability of choosing the most excellent particle. The link between the PSO parameters and the software related attributes are, the attributes like fuzzy rules which is the input to the PSO, based on which the optimization is carried out.

It is evident that the PSO has come out with yielding a superior solution and the phases through which the process proceeds are very well illustrated in the following section. The usual stages of execution with regard to the Particle Swarm Optimization are exhibited in the figure 5.8.

### 5.7.1 Steps in Particle Swarm Optimization

The diverse stages through which the execution of PSO flows are shown below.

i. At first, initialize a population of particles (solutions) with position and velocity selected arbitrarily for n-variable in the dilemma space.

ii. For every one of these particles, estimate the optimization robustness traits in n-variables.

iii. Thereafter, analyze this fitness value with the particles $p_{best}$ value. If the current fitness value is superior to the $p_{best}$ then select it as the $p_{best}$ for the supplementary dispensation.

iv. These fitness values are contrasted with the whole best prior values and if the current value is superior, then revise the $g_{best}$ for the current particles array index and treat it as the new $g_{best}$. 
Modify the velocity and the position of the particle and then replicate the steps till the standard of superior fitness is achieved. The velocity and the position of the particle are amended by using the following equations.

\[ v_j(n+1) = v_j(n) + a_1 d_1 (g_j(n) - h_j(n)) + a_2 d_2 (g_j(n) - h_j(n)) \]  
(5.2)

\[ h_j(n+1) = h_j(n) + v_j(n+1) \]  
(5.3)

where,

- \( v_j(n) \) - Velocity of the \( n^{th} \) particle of \( j^{th} \) iteration.
- \( h_j(n) \) - Position of the \( n^{th} \) particle of \( j^{th} \) iteration.
- \( g_j(n) \) - Global best value of the \( n^{th} \) particle of \( j^{th} \) iteration.
- \( a_1, a_2 \) - Acceleration constants.
- \( d_1, d_2 \) - Distance measure
- \( v_j(n+1) \) - Velocity of \( (n+1)^{th} \) particle of \( j^{th} \) iteration
- \( h_j(n+1) \) - Position of \( (n+1)^{th} \) particle of \( j^{th} \) iteration.

v. The procedure is executed again and again, till the key with superior fitness value is attained.

In the above equations, \( a_1 \) and \( a_2 \) symbolize the acceleration constants essential for blending every particle with the \( pbest \) and \( gbest \). Revise the best position of the particle as per the subsequent equation.
\[
g_j(n+1) = \begin{cases} 
g_j(n), & k(h_j(n+1)) \geq k(g_j(n)) \\
h_j(n+1), & k(h_j(n+1)) < k(g_j(n)) 
\end{cases}
\tag{5.4}
\]

where,

\(g_j(n+1)\) - Global best value of \((n+1)^{th}\) particle of \(j^{th}\) iteration.

\(h_j(n+1)\) - Position of \((n+1)^{th}\) particle of \(j^{th}\) iteration.

\(g_j(n)\) - Global best value of the \(n^{th}\) particle of \(j^{th}\) iteration.

The particle velocity at every dimension is restricted to the interval \([-V_{\text{max}} V_{\text{max}}]\) and then calculated and contrasted with \(V_{\text{max}}\). The \(V_{\text{max}}\) is a vital constraint. The \(V_{\text{max}}\) has function of ascertaining the resolution with which the region between current position and the target position are hunted. The \(V_{\text{max}}\) values provide a basis to assess whether the particles turn out a superior key or not. With the use of the equations given above, the fitness of the solution can be assessed and the optimal solutions according to these fitness values are chosen. Global maxima/minima is the final solution from the PSO algorithm. The global value is the best value obtained in the certain iteration and this will be considered as the final solution, that is, the rule for processing cost estimation.

The general flow diagram for Particle Swarm Optimization is presented in the below Figure 5.8.
Figure 5.8  General Flow Diagram for Particle Swarm Optimization.

This makes it clear that PSO are capable of being employed to find solutions to the optimization dilemmas such as other evolutionary algorithms. It can be extensively exploited in diverse domains such as signal processing, robotics, simulations applications and the like. In the proposed scheme, the study has made use of PSO for the purpose of choosing the fuzzy rule for estimation of software cost. An assortment of trial cases are executed in the algorithm with an eye on snatching the superior outcome related to software structural design. Once the rules are optimized and selected, the final step in the fuzzy process called defuzzification is carried out.
5.8 DEFUZZIFICATION

Prior to the defuzzification, aggregation process is done in which fuzzy sets that represent the outputs of each rule are combined into a single fuzzy set. The output of the aggregation process is one fuzzy set for each output variable. The input for the defuzzification process is a fuzzy set (the aggregate output fuzzy set) and the output is a single number. As much as fuzziness helps the rule evaluation during the intermediate steps, the final desired output for each variable is generally a single number. However, the aggregate of a fuzzy set encompasses a range of output values, and must be defuzzified in order to resolve a single output value from the set. The Defuzzification method used is the weighted average method. The cost estimated through the proposed method is lesser when compared with other methods and it proved to be an efficient technique in software cost estimation.

5.9 SUMMARY

The research has launched an advanced scheme to estimate the software project effort, which is mainly based on fuzzy logic and optimization process. In the fuzzy analogy approach, both categorical and numerical data are considered by fuzzy sets and hence, while producing rules the study deploys the optimization by means of PSO with an eye on improving the rules for superior functioning. The experimental results discussed in the next chapter prove the superiority of this method when compared with existing techniques.