Optimal Testing Effort Allocation for a Software Reliability Growth Model in Fuzzy Environment via Optimal Control Theoretic Approach

Deepika Gaur[1], Pradeep Kumar[1], Kuldeep Chaudhary[1,*], Shivani Bali[2] and Vijay Kumar[1]

[1]*Department of Mathematics, Amity Institute of Applied Sciences, Amity University Uttar Pradesh, Noida, India*
[2]*Jaipuria Institute of Management, Noida, Uttar Pradesh, India*
E-mail: chaudharyiitr33@gmail.com
[*]*Corresponding Author*

Received 18 April 2025; Accepted 11 July 2025

Abstract

The available testing resources are usually restricted during the software testing process. It's usual to presume that these limitations are deterministic when discussing optimal control models. This isn't true in practice. For instance, a budget for the use of testing resources could be the first step in the testing process. However, it is possible that fault detection is accelerated during the fault testing process to meet unexpectedly high fault counts, which calls for additional funding for testing resources. These enhanced numbers are obviously unknown in nature. In this case, fuzzy set theory is a plausible model in sense of degree of uncertainty and hence the testing resource expenditure constraints i.e. total budget becomes imprecise in nature. By integrating fuzzy logic into the budgetary framework, this approach offers flexibility in decision-making, enabling more realistic and adaptable strategies. The aim

Journal of Reliability and Statistical Studies, Vol. 18, Issue 2 (2025), 313–342.
doi: 10.13052/jrss0974-8024.1823

of this research is to look into an optimal way to allocate testing resources in order to minimize software costs during the testing and operation phases while taking independent and dependent faults into account. The model is formulated as an optimal control problem for where the budget constraint is expressed as necessity and /or possibility type. The proposed problem is solved for an optimal testing effort policy employing Pontryagin's Maximum Principle. A numerical example is presented to support the theoretical optimal control model. The values of the optimal testing effort expenditure function are displayed in both tabular and graphic forms.

Keywords: Fuzzy constraint, maximum principle, optimal control theory, testing effort allocation, necessity/possibility.

1 Introduction

Nowadays, every industry is becoming computerized, with many complex tasks being managed by software systems. From the era when computers occupied entire rooms to the modern day, where they fit in the palm of our hands, our reliance on these machines has grown exponentially. Today, we depend on computers for even the simplest calculations, and their role in organizations has become indispensable. As many businesses now rely exclusively on computer systems, any failure can result in significant losses of resources, time, and money. This heightened dependency has also exposed situations where software failures have led to substantial financial and temporal setbacks. Consequently, software companies now invest as much in testing their products as they do in developing them. Testing involves a series of predefined activities designed to ensure the final product meets the customer's expectations. Studies suggest that software testing and removing fault costs can range from 30% to 80% of the cost of producing a working version of the software [1, 2]. An inadequate infrastructure for software testing is predicted to cost $59.5 billion annually (US), according to NIST (2002). Improvements to the infrastructure that are practicable might save $22.2 billion. This amounts to around 0.6 and 0.2 percent of the $10 trillion GDP of the United States, respectively. About 40% of the total impacts were attributable to software creators, and 60% were attributable to software consumers.

The software development industry is a cornerstone of the modern economy, driving innovation and transformation across various sectors. With rapid

technological advancements and evolving market demands, the industry continues to grow and adapt, presenting both exciting opportunities and complex challenges for developers and businesses alike. The goal is to create software that meets user requirements, function reliably, and provides positive user experience. The cost, fault identification, removal of fault, and testing efforts are integral components of the software development process that directly impact the quality, reliability, and success of software products. Each aspect plays a vital role in ensuring that the software meets user requirements, operates reliably, and provides positive user experience. By investing in robust testing practices, continuous integration, efficient fault identification and removal processes, organizations can minimize costs, enhance software quality, and deliver reliable software solutions to the market.

The cost of software development can vary widely depending on several factors, including the complexity of the software, the technologies used, the size of the development team, and the location of the development resources. The cost components in software development typically include fault identification, also known as bug detection, which is a critical process in software development. It involves identifying errors or faults in the software that could cause it to behave unexpectedly or fail to meet specified requirements. Effective fault identification is crucial for maintaining software quality and reliability. Once faults are identified, the next step is fault removal, which involves debugging and fixing the fault in the code. Fault removal is a crucial part of the software development process, as unresolved fault can lead to software failures, security vulnerabilities, and poor user experience. One can use debugging, Regression testing, refactoring, continuous integration.

Testing effort refers to the time and resources required to thoroughly test the software to ensure it is free from fault and meets the specified requirements. It is a critical part of the software development lifecycle and plays a significant role in ensuring software quality as software engineers not only identify faults during testing but also resolve them, thereby enhancing the reliability of the software. Testing process demands significant resources, including CPU hours and human effort, which substantially impacts the cost of software development. Therefore, efficiently allocating these resources is a critical responsibility for a project manager. In the last thirty years, we have seen the proposal of numerous software reliability growth models (SRGMs) to reduce overall testing effort expenditures [3–7]. Huang et al. [5] and Kapur et al. [7] developed S-shaped testing effort-dependent SRGMs based on Yamada et al. [8], delayed S-shaped model, which describes the

learning phenomenon observed during the software testing phase. These models highlight the importance of balancing testing efforts and costs to ensure optimal fault detection. Peng et al. [9] proposed a modelling framework that incorporated testing effort and imperfect debugging into both fault detection and fault correction processes. Kapur and Bardhan [6], investigated the connection between testing effort, duration and the number of faults eliminated. However, these models are primarily predicated on the idea that the relationship between testing effort consumption and testing time (the calendar time) follows an exponential and Rayleigh distribution.

Modular software system plays a significant role in enhancing software reliability and improving the effectiveness of the software reliability growth model [10–14]. Modular software analyzes and predicts the reliability of software system during testing by identifying and removing faults. It allows for parallel testing of individual modules and increasing the rate of fault detection. By targeting specific modules for fault detection and removal, resources are optimized, reducing cost associated with testing and debugging. Kapur et al. [11] examined the optimization problem of allocating testing resources in software with a modular structure. They proposed that the allocation of testing effort should be based on the size and severity of faults. Additionally, they suggested that for various resource constraints, a trade-off can be developed between the maximum number of faults to be removed in each module and the effort required. Huang et al. [12] describes an optimal resource allocation problem in modular software systems during the testing phase. The primary goal is to reduce the cost of software development when a certain amount of testing effort and a desired level of reliability are specified. According to Kapur et al. [13], the testing resource allocation problem among modules aims to maximize the overall fault removal from software that is composed of multiple independent modules. Khan et al. [14] study addresses the issue of allocating testing resources for a modular software system in the best possible way to maximize the number of defects fixed, provided that the testing effort is fixed, a specific percentage of defects are to be eliminated, and a desired level of reliability is to be attained.

Optimal control theory is widely used to derive optimal policies in many research fields including economics, marketing, inventory-production and engineering [15–27]. It is a mathematical framework that provides systematic approaches for optimizing the performance of dynamic systems. In software reliability growth models, it plays a critical role in allocation of testing resource expenditure under dynamic conditions. Kapur et al. [21] used optimal control theoretic technique to know the behaviour of dynamic model

in testing resource allocation for detection and removal of faults. Kumar and Sahni [22] conducted additional research in optimal control modeling by developing an effort allocation model that takes into account various budgetary constraints on the fault detection and correction processes. Kumar et al. [23] developed an optimal control model for resource allocation during the testing phase that incorporates a flexible software reliability growth model and testing effort in a dynamic environment. Pradhan et al. [25] use the optimal control theoretic approach in software systems to solve profit maximization problems by merging software enhancement and customer growth concepts. Pradhan et al. [26] created a software reliability growth model (SRGM) that incorporates a Weibull testing effort function (TEF) into the software fault detection and repair processes (FDP and FCP), successfully reflecting the complexities of software testing.

1.1 Contribution

The testing resource expenditure and other restrictions that are applied in optimal control models were thought to be deterministic until recently. However, in practice, these limitations must be imprecise, meaning they must be enforced in a non-stochastic way. In reality, this is not true. For example, the testing process may begin with a budget for testing resource expenditure. However, throughout the fault testing process, it is possible that in order to meet unexpectedly increasing faults, fault detection is accelerated, demanding an allocation of some extra budget for testing resources. These enhanced numbers are obviously unknown in nature. Fuzzy set theory provides an effective approach to handling such uncertainty, as it allows resource constraints – such as total testing expenditure and budget capacity – to be represented in an imprecise manner. Since a fuzzy constraint corresponds to a fuzzy event, it must be satisfied within a specified level of possibility or necessity [27, 28]. Similar to chance-constrained programming with stochastic parameters [29, 30], a fuzzy environment assumes that certain constraints must be met with at least a given level of possibility, denoted as η_1, where 'possibility' serves as the fuzzy counterpart to 'chance'. Maiti and Maiti [31] proposed a fuzzy inventory model for a two-warehouse system under possibility constraints.

Recently, Samal et al. [32] used fuzzy logic in software reliability assessment. This chapter [32] also covers studies on determining a software system's trustworthiness and developing release policies in fuzzy environments. However, no research so far has explored a dynamic allocation of

testing resource model for fault removal process in software reliability with imprecise testing resource constraints using optimal control theory. Our focus is on determining the optimal allocation of resources across independent software modules while adhering to an imprecise budget for testing. To minimize the cost of testing in a modular software system, we formulate the problem as an optimal control theory problem. The objective is to derive the optimal rate of testing effort allocation for each module to reduce the overall testing cost while adhering to the budget constraints.

The entire available budget for testing and debugging is handled as fuzzy in our work because it is unpredictable in many real-world settings. The aim of this study is to examine the impact of fuzzy budget constraint on the fault removal process in modular software system and fill the gap by determining the optimal testing effort allocation strategies in a modular software system environment. To address this uncertainty, we use fuzzy set theory to express the budget as a triangular fuzzy number. After clarifying or defuzzing the imprecise financial constraint, which could be based on necessity, we employ Pontryagin's maximum principle to solve the clear model, seeking to improve the development process while minimizing costs and optimizing software performance.

The related work for development of proposed optimal control problem is established in Section 2. Section 3 provides assumption, development of the proposed model. The model is switch to the corresponding Crisp Optimal Control Problem in Section 4. In Section 5 we have discuss the particular cases. Section 6 illustrates the numerical examples and discussion and conclusion in Section 7.

2 Related Work

2.1 Nonhomogeneous Poisson Process – A General Description

The NHPP models [33–35] are based on the assumption that 'Software failure occurs at random times during testing caused by faults lying dormant in the software'. Hence NHPP can be used to describe the failure phenomenon during testing as well as operational phases of software development. The counting process $\{N(t), t \geq 0\}$ of an NHPP process with mean $m(t)$ is given as follows.

$$Pr\{N(t) = k\} = \frac{\{m(t)\}^k}{k!} e^{-m(t)},\ k = 0, 1, 2, \ldots \tag{1}$$

The expected number of faults detected up to time t is represented by the mean value function $m(t)$, which can be expressed as follows:

$$m(t) = \int_0^t \lambda(x)dx \tag{2}$$

The intensity function of NHPP, $\lambda(t)$, represents the instantaneous rate of fault detection. Define $a(= m(\infty))$ as the expected cumulative number of faults to be eventually detected. From (1), we can see that

$$\lim_{t\to\infty} Pr\{N(t) = k\} = \frac{a^k}{k!}e^{-a},\ k = 0, 1, 2, \ldots$$

This implies that over infinitely long duration testing, $N(t)$ will follow a Poisson distribution with mean a.

The estimated number of software faults remaining in the software system at testing time t is $a - m(t)$. The fault detection rate per fault (per unit time) during testing time t can be used as a valuable software reliability growth index and expressed as

$$b(t) = \frac{\lambda(t)}{a - m(t)}$$

The relationship between $b(t)$ & $m(t)$ can be expressed as follows:

$$m(t) = a(1 - e^{-\int_0^t b(u)du})$$

The intensity function $\lambda(x)$ (or the mean value function $m(t)$) is the basic building block of all the NHPP models existing in the software reliability engineering literature.

2.2 Kapur and Garg SRGM for Removing Error (1992)

This model is predicated on the idea that the debugging can eliminate some errors without creating any failures, while simultaneously eliminating some new faults. Independent faults are those that are found during a failure, whereas dependent faults are those that are also eliminated. Above assumptions can be described mathematically for fault removal process with the following differential equation

$$\frac{d\Omega_r(t)}{dt} = p(a - \Omega_r(t)) + q\frac{\Omega_r(t)}{a}(a - \Omega_r(t)) \tag{3}$$

$\Omega_r(t)$ represents the predicted number of removals, while p and q represent fault detection rates for independent and dependent faults, respectively. Under the initial condition $m_r(0) = 0$, the mean value function obtained as

$$\Omega_{\mathrm{r}}(t) = \frac{a(1 - e^{-(p+q)t})}{1 + \left(\frac{q}{p}\right) e^{-(a+b)t}} \tag{4}$$

The failure phenomenon is the same as removal when $p = 0$, which reduces the model to the G-O model. Otherwise, the phenomenon of failure is expressed as

$$\Omega_f(t) = \int_0^t p(a - \Omega_r(x))dx = \frac{ap}{q} ln \left[\frac{p+q}{p + qe^{-(p+q)t}}\right] \tag{5}$$

An s-shaped flexible growth curve is described by the model. If we assume logistic learning fault detection rate, which is provided by $b(t) = (\frac{b}{1+\beta e^{-bt}})$ with $b = (p + q)$ and $\beta = (q/p)$, we can alternatively construct the K-G model [3]. Obha's [37] and Bittanti et al.'s [38] models are comparable to the K-G model and may be obtained using the formulas $b(t) = b(r + (\frac{(1-r)\Omega(t)}{a}))$ and $b(t) = k_i + (k_f - k_i)(\frac{\Omega(t)}{a})$, respectively, where k_f and k_i represent the final and initial fault exposure coefficients, respectively. While the Bittanti Model offers a range of dependability trends, the Obha model [37] describes a growing error detection rate.

2.3 Optimal Allocation of Testing Effort in SRGM: A Control Theoretic Approach (Kapur, P. K., et al., 2013)

The main objective of this paper is to examine the best possible resource allocation strategy to reduce software costs during the testing and operation stages in dynamic environments. The optimal control formulation for the testing resource allocation can be given by

$$\left.\begin{array}{c} min \int_0^T [c_1(t)x(t) + c_2 w_1(t)]dt \\ s.t. \\ x(t) = \dfrac{d\Omega(t)}{dt} = x(b, w_1(t)),\ \Omega(0) = 0 \\ \dfrac{\Omega(t)}{a} \geq \Omega_T \Rightarrow \Omega(T) \geq \Omega_d = a\Omega_T) \\ (w_1(t); w_2(t) \geq 0 \ and \ c_1(t) = c_1(\Omega(t), w_2(t)) \end{array}\right\} \tag{6}$$

Where, T is planning period; $w_1(t)$ denotes the effort consumption related to testing at time t; $w_2(t)$ is the effort consumption due to debugging the fault at time t; $c_1(t)$ denotes the cost per unit at time t for cumulative fault removal and debugging effort; c_2 is the cost of testing per unit testing efforts. The planning problem is finding the allocation of resources that minimizes the total expenditure. The article investigates how resources can be allocated efficiently during the software testing phase to minimize the overall cost.

3 Proposed Methodology

Testing plays a crucial role in the SDLC (Software Development Life Cycle) as it helps to evaluate reliability of software. Since most software systems are modular, it is essential for management to optimally allocate limited testing resources across various modules [11]. Although several S-shaped models for software reliability exist in the literature, we focus on the Kapur and Garg model [3] due to its simplicity and ability to analyze the failure growth rate of software. During testing, some faults can be eliminated without causing software failures, though this may require extra effort. A fault that is removed after causing a failure is termed as a leading fault. In the process of eliminating leading faults, other faults, which could have led to failures, may also be detected and addressed. These are referred to as dependent faults.

3.1 Assumption and Notation

For proposed testing effort allocation optimal control model considering fuzzy budget constraint, following assumptions and notations are used.

Notations

$N(t)$	Counting process represents the cumulative number of software errors found in interval $[0, t]$.
$m(t), \Omega_i(t)$	Mean value function/cumulative number of faults (or module *i*) detected by time 't'
a, a_i	Initial fault content in software (or module *i*)
p, p_i	Fault detection rate for independent faults (or module *i*)
q, q_i	Fault detection rate for dependent faults (or module *i*)
$b, b_i(t)$	Fault detection rate per remaining fault
$u_i(t)$	Current testing effort expenditure for testing and debugging for module *i*

$C_{1i}(u_i)/C_{1i}$	Cost of testing effort used to correct an error/fault during testing for module *i*
$C_{2i}(u_i)/C_{2i}$	Cost of testing effort used to correct an error/fault during the operational phase for module *i*
$C_{3i}(u_i)/C_{3i}$	Cost of testing per unit of testing effort for module *i*
W	Available total testing expenditure budgetary

Assumptions

The following assumptions form the basis of our model:

1. The fault detection process follows a Non-Homogeneous Poisson approach (NHPP).
2. The software system contains M different modules, each of which is tested independently.
3. A fuzzy variable represents the overall budget for module testing resources, which is fixed.
4. The system manager aims to allocate resources among software modules to reduce testing costs while maintaining quality.
5. The entire software system fails if any module includes a fault.
6. When a failure-causing fault is found, other errors in the same module are also found, stopping those mistakes from causing more failures.

The goal is to minimize overall testing costs while debugging as many faults as possible. When the budget is uncertain, testing resources must be distributed optimally across software modules.

3.2 Model Development

A testing effort function describes the amount of testing resources used during the testing period. In the development of our optimal control theoretic model, we assume that the firm controls its resource expenditures for testing and debugging during a finite planning time, and that the number of faults detected per unit testing effort is proportional to the remaining faults, both dependent and independent. Under the above assumptions 1–6, the fault detection procedure with testing effort in modular software systems can be described by differential equation as:

$$\frac{d\Omega_i(t)}{dt} = u_i(t)\left(p_i(a_i - \Omega_i(t)) + q_i\frac{\Omega_i(t)}{a_i}(a_i - \Omega_i(t))\right)$$
$$\forall\, i = 1 \text{ to } n, \Omega_i(0) = 0 \qquad (7)$$

Where, $u_i(t)$ is current testing effort and time dependent function; p_i is the detection rate at which residual faults cause failure in the software and q_i is the detection rate at which additional faults cause failure in the software.

In this paper, we focus on optimizing testing resources allocation in modular software systems to minimize testing costs where the total budget for testing resources is fixed and presents an optimal control framework. Each software module is tested independently, and the system manager must allocate resources in a way that minimizes the overall cost while ensuring reliable performance. To determine the entire cost of software testing using cost criteria, including the cost of testing-effort expenditures during the software development and testing phases, as well as the cost of rectifying errors before and after release. Jha et al. [36] state the overall software cost for each module as follows:

$$C(t) = \sum_{i=1}^{M} \left\{ C_{1i}(u_i)\Omega_i(T) + C_{2i}(u_i)(\Omega_i(\infty) - \Omega_i(T)) + C_{3i} \int_0^T u_i(t)dt \right\} \tag{8}$$

C_{1i} represents the cost of testing effort used to correct an error during testing, C_{2i} represents the cost of testing effort used to correct a fault during the operational phase, and C_{3i} represents the cost of testing per unit of testing effort spent. Keep in mind that addressing a bug after release often costs more than repairing a bug when testing.

If we assume that $\Omega_i(\infty) = a_i$, then the above functional can be written as

$$C(t) = \sum_{i=1}^{M} \left\{ C_{1i}(u_i)\Omega_i(T) + C_{2i}(u_i)(a_i - \Omega_i(T)) + C_{3i} \int_0^T u_i(t)dt \right\} \tag{9}$$

To derive optimal strategies for testing resource allocation under fixed budget constraints, aiming to reduce the overall testing cost without compromising software quality. To minimize the overall cost of software testing, we develop an optimal control problem to assign an optimal amount of testing effort to each module. With the condition that the system has M modules, the objective is to select the optimal control variable $u_i(t)$ in a way that minimizes the overall testing cost in planning horizon $[0, T]$. Let T be the software product's release time. The objective function can now be expressed as

$$\min_{u_i(t)} J = \int_0^T \left(\sum_{i=1}^{M} \{ C_{1i}(u_i)\dot{\Omega}_i(t) + C_{2i}(u_i)(a_i - \dot{\Omega}_i(t)) + C_{3i}u_i \} \right) dt \tag{10}$$

and the constraint on the testing effort expenditures for every module

$$\int_0^T \left(\sum_{i=1}^{M} u_i \right) dt \leq \tilde{W} \tag{11}$$

The fuzzy variable $\tilde{W}$ represents the entire budget for testing resources.

3.3 Fuzzy Optimal control Formulation

The total amount available of testing-resource expenditure is taken to be imprecise. To obtain the optimal dynamic testing effort allocation strategy while minimizing the total testing cost, which can be described by the following dynamic optimization problem as an optimal control problem:

$$\left.\begin{aligned} \max_{u_i(t)} J = \int_0^T \left(\sum_{i=1}^{M} \{(C_{2i}(u_i) - C_{1i}(u_i))\dot{\Omega}_i(t) - C_{2i}(u_i)a_i - C_{3i}u_i\} \right) dt \\ \frac{d\Omega_i(t)}{dt} = (u_i(t)) \left(p_i(a_i - \Omega_i(t)) + q_i \frac{\Omega_i(t)}{a_i}(a_i - \Omega_i(t)) \right) \\ \Omega_i(0) = 0;\ u_i^l \leq u_i(t) \leq u_i^u \ \forall\, i = 1 \ldots M; \\ \int_0^T \left(\sum_{i=1}^{M} u_i \right) dt \leq \tilde{W} \end{aligned}\right\} \tag{12}$$

The joint allocation of testing effort resources among all modules is referred to as the constraint $\int_0^T (\sum_{i=1}^{M} u_i) dt \leq \tilde{W}$. This is the only constraint that links the modules together and restricts us from solving a single module problem. Using the Theorem 1 stated in Maity and Maiti [32], we get from Equation (11) as

$$\sum_{i=1}^{M} u_i \leq \frac{\tilde{W}}{T} \tag{13}$$

Equation (14) provides a fuzzy constraint that can be viewed in terms of necessity. It is possible to depict fuzzy relations in a variety of ways. Fuzzy numbers are used in necessity/possibility theory to understand these relationships, indicating the level of uncertainty. A variety of fuzzy constraint combinations are used to decrease constraint (13) [31, 32].

$$Nes \left\{ \left(\sum_{i=1}^{N} u_i \right) < \frac{\tilde{W}}{T} \right\} \geq \eta_1$$

Another way to write this is as

$$Pos\left\{\left(\sum_{i=1}^{N} u_i\right) \geq \frac{\tilde{W}}{T}\right\} \leq 1-\eta_1 \tag{14}$$

3.4 Equivalent Crisp Optimal Control Problem

The above model implies that the total testing resource budget will be represented by a triangular fuzzy number. Let $\tilde{W} = (W_1, W_2, W_3)$, then $\frac{\tilde{W}}{T} = (\frac{W_1}{T}, \frac{W_2}{T}, \frac{W_3}{T}) = (W_1', W_2', W_3'$. The constraint is converted into a crisp one using the concept of necessity or possibility, and therefore the problem represented by (12) reduces to its equivalent crip version as follows:

$$\left.\begin{aligned} \max_{u_i(t)} J = \int_0^T \left(\sum_{i=1}^{M}\{(C_{2i}(u_i) - C_{1i}(u_i))\dot{\Omega}_i(t) - C_{2i}(u_i)a_i - C_{3i}u_i\}\right) dt \\ \frac{d\Omega_i(t)}{dt} = (u_i(t))\left(p_i(a_i - \Omega_i(t)) + q_i\frac{\Omega_i(t)}{a_i}(a_i - \Omega_i(t))\right) \\ \Omega_i(0) = 0;\ u_i^l \leq u_i(t) \leq u_i^u\ \forall\, i = 1 \ldots M; \end{aligned}\right\} \tag{15}$$

subjected to the constraint (13) for all scenarios and

$$\frac{\sum_{i=1}^{M} u_i - W_1'}{W_2' - W_1'} \leq 1-\eta_1 \tag{16}$$

4 Mathematical Approach to Solve the Crisp Optimal Control Problem

Optimal control theory is a significant field of dynamic optimization that uses mathematical optimization to determine the best course of action for controlling a dynamical system. For the aforementioned optimum control problems (15) and (16), the Hamiltonian function is

$$H = \sum_{i=1}^{M}\{(C_{2i}(u_i) - C_{1i}(u_i))\dot{\Omega}_i(t) - C_{2i}(u_i)a_i - C_{3i}u_i\} + \sum_{i=1}^{M}\mu_i\dot{\Omega}_i(t) \tag{17}$$

With respect to their state equations, $\mu_i(t)$ are adjoint variables. To put it simply, the Hamiltonian is the sum of the current $(C_{2i}(u_i) - C_{1i}(u_i))\dot{\Omega}_i(t)$ and future cost $(\mu_i\dot{\Omega}_i(t))$. The needed optimality criteria are provided by the

maximum principle [36], assuming that there is an optimal control solution. The function $\mu_i(t)$ is piecewise continuously differentiable for any $t \in [0, t]$. It explains how dual variables behave similarly in optimal control theory to how they do in nonlinear programming. The corresponding Lagrangian function with constraint (17)

$$L = H + \lambda \left[(1 - \eta_1) W_2' + \eta_1 W_1' - \sum_{i=1}^{M} u_i \right] \tag{18}$$

Where, λ is Lagrangian multiplier. Then K-T conditions:

$$\lambda \left[(1 - \eta_1) W_2' + \eta_1 W_1' - \sum_{i=1}^{M} u_i \right] = 0 \tag{19}$$

The adjoint functions $\mu_i(t)$ are obtained by using the Maximum Principle as follows:

$$\frac{d}{dt}\mu_i(t) = -\frac{\partial L}{\partial \Omega_i(t)},\ \mu_i(T) = 0 \tag{20}$$

From Equation (21) with transversality conditions $\mu_i(T) = 0$, we have

$$\frac{d}{dt}\mu_i(t) = -[(C_{2i}(u_i) - C_{1i}(u_i)) + \mu_i(t)] \frac{\partial \dot{\Omega}_i(t)}{\partial \Omega_i(t)} \tag{21}$$

On simplification, we get

$$\begin{aligned} \frac{d\mu_i}{dt} &= [(C_{2i}(u_i) - C_{1i}(u_i)) + \mu_i(t)](u_i(t)) \\ &\quad \times \left[(p_i - q_i) + \frac{2q_i}{a_i} \Omega_i(t) \right] \end{aligned} \tag{22}$$

According to the Pontryagin's maximum principle, the Lagrangian function should be maximized in relation to admissible controllable testing effort function $u_i(t) \forall\ i = 1, 2, \ldots M$. From Equation (18), we have

$$\begin{aligned} \frac{\partial L}{\partial u_i(\mathrm{t})} &= \left(p_i + q_i \frac{\Omega_i(t)}{a_i} \right) (a_i - \Omega_i(t)) \\ &\quad \times \left[\left(\frac{\partial C_{2i}(u_i)}{\partial u_i} - \frac{\partial C_{1i}(u_i)}{\partial u_i} \right) u_i(t) \right. \end{aligned}$$

$$+ (C_{2i}(u_i) - C_{1i}(u_i)) + \mu_i(t)\Big]$$

$$- \frac{\partial C_{2i}(u_i)}{\partial u_i} a_i - C_{3i} - \lambda \tag{23}$$

and the necessary optimality conditions for Equation (23) are given by

$$\frac{\partial L}{\partial u_i(\mathrm{t})} = 0 \tag{24}$$

From the optimality conditions (24), we get the testing effort rate $u_i^*(t)$ for each module. It is given by

$$u_i^*(t) = \frac{1}{\left(\frac{\partial C_{2i}(u_i)}{\partial u_i} - \frac{\partial C_{1i}(u_i)}{\partial u_i}\right)} \times \left(\frac{\frac{\partial C_{2i}(u_i)}{\partial u_i} a_i + C_{3i} + \lambda}{\left(p_i + q_i \frac{\Omega_i(t)}{a_i}\right)(a_i - \Omega_i(t))} - ((C_{2i}(u_i) - C_{1i}(u_i)) + \mu_i(t)) \right) \tag{25}$$

From the Equations (7), (21) and (26), with terminal conditions $\mu_i(T) = 0$, and initial conditions $\Omega_i(0) = 0$, we obtain the optimal value expected number of fault removal at time t as

$$\Omega_i(t) = a_i \left(\frac{1 - exp(-(p_i + q_i) \int_0^t u_i^*(s)ds)}{1 + \frac{q_i}{p_i}(exp(-(p_i + q_i) \int_0^t u_i^*(s)ds))} \right) \tag{26}$$

Define the cumulative testing efforts for the ith module as $U_i^*(t) = \int_0^t u_i^*(s)ds$. Then Equation (27) can be written as

$$\Omega_i(t) = a_i \left(\frac{1 - exp(-(p_i + q_i)U_i^*(t))}{1 + \frac{q_i}{p_i}(exp(-(p_i + q_i)U_i^*(t)))} \right) \tag{27}$$

The above expression is similar to model with testing effort. Equation (27) gives the optimal trajectory of the number of faults removed for ith module. If $u_i^*(t) = \alpha$, we can deduce the following three scenarios.

Case 1: If $\frac{\partial L}{\partial u_i(t)} > 0$ then the Lagrangian function L is an increasing function of $u_i^*(t)$ and is given by minimum$\{\alpha, u_i^u\}$

Case 2: If $\frac{\partial L}{\partial u_i(t)} = 0$ then Lagrangian function L is maximised for the value of $u_i(t)$ as $u_i^*(t) = \alpha(for\ \lambda \geq 0)$

Case 3: If $\frac{\partial L}{\partial u_i(t)} < 0$ then Lagrangian function L is a decreasing function of $u_i(t)$ and $u_i(t) = maximum\{\alpha, u_i^l\}$

5 Particular Cases Owing Sections, We Alter the Correcting Costs to Investigate How the Suggested Optimal Control Problem Behaves

5.1 When the Costs are Linear Function of Testing Effort

The costs in this section are assumed to be linearly related to the testing effort, that is, $C_{1i}(u_i) = C_{1i}u_i$ and $C_{2i}(u_i) = C_{2i}u_i$. Where C_{1i} and C_{2i} are constant values, respectively. The Hamiltonian can now be expressed as

$$H = \sum_{i=1}^{M}\{(C_{2i} - C_{1i})u_i\ \dot{\Omega}_i(t) - C_{2i}\ u_i a_i - C_{3i}u_i\} + \sum_{i=1}^{M}\mu_i\dot{\Omega}_i(t) \quad (28)$$

The corresponding Lagrangian

$$L = H + \lambda\left[(1-\eta_1)W_2' + \eta_1 W_1' - \sum_{i=1}^{M} u_i\right] \quad (29)$$

From Equation (29) with transversality conditions $\mu_i(T) = 0$, we have

$$\frac{d}{dt}\mu_i(t) = -[(C_{2i} - C_{1i})u_i + \mu_i(t)]\frac{\partial\dot{\Omega}_i(t)}{\partial\Omega_i(t)} \quad (30)$$

On simplification, we get

$$\frac{d\mu_i}{dt} = ((C_{2i} - C_{1i})u_i + \mu_i(t))\left[(p_i - q_i) + \frac{2q_i}{a_i}\Omega_i(t)\right] \quad (31)$$

According to Pontryagin's maximum principle, the Lagrangian function should be maximized in relation to the control variables $u_i(t)\forall\ i = 1, 2, \ldots M$ and $\frac{\partial L}{\partial u} = 0$ are required optimality criteria. So, now for $u_i(t)$:

$$\frac{\partial L}{\partial u_i(\mathrm{t})} = \left(p_i + q_i\frac{\Omega_i(t)}{a_i}\right)(a_i - \Omega_i(t))[2(C_{2i} - C_{1i})u_i(\mathrm{t}) + \mu_i(t)] - C_{2i}a_i - C_{3i} - \lambda \quad (32)$$

and

From the optimality conditions (32), we get the testing effort rate $u_i^*(t)$ for each module. It is given by

$$u_i^*(t) = \frac{1}{2(C_{2i} - C_{1i})} \left(\frac{C_{2i}a_i + C_{3i} + \lambda}{\left(p_i + q_i \frac{\Omega_i(t)}{a_i}\right)(a_i - \Omega_i(t))} - \mu_i \right) \tag{33}$$

The Lagrange's multiplier λ must satisfy the complementary slackness conditions $\lambda \geq 0$.

$$\lambda \left[(1-\eta_1)W_2' + \eta_1 W_1' - \sum_{i=1}^{M} u_i \right] = 0$$

When $\lambda = 0$, then we have

$$u_i^*(t) = \frac{1}{2(C_{2i} - C_{1i})} \left(\frac{C_{2i}a_i + C_{3i}}{\left(p_i + q_i \frac{\Omega_i(t)}{a_i}\right)(a_i - \Omega_i(t))} - \mu_i \right) \tag{34}$$

If $\lambda > 0$, it implies for $((1-\eta_1)W_2' + \eta_1 W_1' - \sum_{i=1}^{M} u_i) = 0$. We have the optimal testing effort as

$$u_i^*(t) = \frac{1}{2(C_{2i} - C_{1i})} \left(\frac{C_{2i}a_i + C_{3i} + \lambda}{\left(p_i + q_i \frac{\Omega_i(t)}{a_i}\right)(a_i - \Omega_i(t))} - \mu_i \right) \tag{35}$$

Where,

$$\lambda = \frac{\left((1-\eta_1)W_2' + \eta_1 W_1' - \sum_{i=1}^{M} \frac{1}{2(C_{2i} - C_{1i})} \left(\frac{C_{2i}a_i + C_{3i}}{\left(p_i + q_i \frac{\Omega_i(t)}{a_i}\right)(a_i - \Omega_i(t))} - \mu_i \right) \right)}{\sum_{i=1}^{M} \frac{1}{2(C_{2i} - C_{1i})\left(p_i + q_i \frac{\Omega_i(t)}{a_i}\right)(a_i - \Omega_i(t))}}.$$

The optimal policies for allocating testing effort can be interpreted as initially increasing $\Omega_i(t)$ due to the S-shaped character of the fault detection function, followed by a decrease in testing effort expenditure. When $\Omega_i(t)$ slows down due to S-shaped fault detection rate, the allocation path for testing effort consumption begins to increase.

5.2 When the Costs are Constant

The costs C_{2i} and C_{1i} are assumed to be constant values in this section, meaning that they are not directly influenced by the testing effort rate, i.e., $C_{1i}(u_i) = C_{1i}$ and $C_{2i}(u_i) = C_{2i}$. Now, we can re-write the Hamiltonian function as

$$H = \sum_{i=1}^{M} \{(C_{2i} - C_{1i} + \mu_i)\, \dot{\Omega}_i(t) - C_{2i}\, a_i - C_{3i} u_i\} \tag{36}$$

The corresponding Lagrangian

$$L = H + \lambda \left[(1 - \eta_1) W_2' + \eta_1 W_1' - \sum_{i=1}^{M} u_i \right] \tag{37}$$

From Equation (37) with transversality conditions $\mu_i(T) = 0$, we have adjoint variables equation:

$$\frac{d}{dt}\mu_i(t) = -(C_{2i} - C_{1i} + \mu_i) \frac{\partial \dot{\Omega}_i(t)}{\partial \Omega_i(t)} \tag{38}$$

On simplification, we get

$$\frac{d\mu_i}{dt} = -(C_{2i} - C_{1i} + \mu_i) \left((q_i - p_i) - \frac{2q_i}{a_i} \Omega_i(t) \right) \tag{39}$$

In the Lagrangian function, the term $\sum_{i=1}^{N} C_{2i} B_i$ remains constant. Furthermore, the fixed value $\sum_{i=1}^{N} C_{2i} B_i$ can be removed from the equation. Because the Lagrangian function is linear in control variable, the maximum principle for the best testing effort is given by

$$u_i^*(t) = \begin{cases} u_i^l, & if\ S_i(t) < 0 \\ Singular, & if\ S_{\mathrm{i}}(t) = 0 \\ u_i^u, & if\ S_i(t) > 0 \end{cases} \tag{40}$$

In this case, the coefficient of u_i in Lagrangian function is $S_i(t) = (C_{2i} - C_{1i} + \mu_i) \frac{\partial \dot{\Omega}_i(t)}{\partial u_i} - C_{3i} - \lambda$, which is also known as the "switching function". In the theoretical framework of optimal control theory, this kind of optimum control is referred to as "bang bang" [36]. However, on an arc along which

$S_i(t) = 0$, interior control is feasible. "Singular arc" is the term used to describe such an arc. The Equation (40) can now be expressed as

$$u_i^*(t) = \begin{cases} u_i^l, & if\ \mu_i(t) < \dfrac{(C_{3i}+\lambda)}{\left(p_i + q_i \frac{\Omega_i(t)}{a_i}\right)(a_i - \Omega_i(t))} - (C_{2i} - C_{1i}) \\ Singular, & if\ \mu_i(t) = \dfrac{(C_{3i}+\lambda)}{\left(p_i + q_i \frac{\Omega_i(t)}{a_i}\right)(a_i - \Omega_i(t))} - (C_{2i} - C_{1i}) \\ u_i^u, & if\ \mu_i(t) > \dfrac{(C_{3i}+\lambda)}{\left(p_i + q_i \frac{\Omega_i(t)}{a_i}\right)(a_i - \Omega_i(t))} - (C_{2i} - C_{1i}) \end{cases} \tag{41}$$

Equation (41) provides optimal policies, which can be interpreted as follows: The minimum testing effort expenditure rate should be applied to all modules if the entire cost of fixing a fault is less than the cost of unit testing. However, all testing effort expenditure rates should be at their maximum for all modules if the entire cost of fixing an error exceeds the cost of unit testing. Using optimal testing efforts, the optimal values of state trajectory (optimal fault removal) with initial condition $\Omega_i(0) = 0$ is

$$\Omega_i(t) = a_i \left(\frac{1 - exp\left(-(p_i + q_i)\int_0^t u_i^*(s)ds\right)}{1 + \frac{q_i}{p_i}\left(exp\left(-(p_i + q_i)\int_0^t u_i^*(s)ds\right)\right)} \right) \tag{42}$$

Using Equation (39), we observed that $\mu_i(t)$ is the decreasing and negative function. The $\mu_i(t)$ represents the marginal value of faults at time t, which should be negative as the quantity of faults increases the correcting cost. Equation (42) illustrates how the switching function affects the ideal expected mean number of faults found in time $(0, t)$ with optimal testing effort $u_i^*(= u_i^l$ or $u_i^u)$.

6 Numerical Illustration

Here, a numerical example is provided to illustrate how to use the proposed model mentioned above. Using the method suggested by Rosen [40], the corresponding crisp control theory issue given by Equation (15) with constraint (14) is converted to a discrete form. For its numerical application, the discrete

Table 1 Values of parameters

Parameters	Module-1	Module-2	Module-3
a_i	274	275	276
p_i	0.035	0.037	0.038
q_i	0.136	0.138	0.140
C_{1i}	10	11	12
C_{2i}	50	51	52
C_{3i}	300	310	320
η_1	0.5	0.5	0.5
u_i^l	0.40	0.40	0.40
u_i^u	1	1	1

formulation of the problem as follows is taken into consideration:

$$\max_{u_i(t)} J = \sum_{k=1}^{T} \left(\sum_{i=1}^{M} \{(C_{2i}(\mathrm{u_i(k)}) - C_{1i}(u_i(k)))(\Omega_i(k+1) - \Omega_i(k)) - C_{2i}(u_i(k))a_i - C_{3i}u_i(k)\} \right) \tag{43}$$

$$\Omega_i(k+1) = \Omega_i(k) + u_i(k) \left(p_i(a_i - \Omega_i(k)) + q_i \frac{\Omega_i(k)}{a_i}(a_i - \Omega_i(k)) \right) \tag{44}$$

$$\Omega_i(0) = 0;\ u_i^l \le u_i(k) \le u_i^u;\ \forall\, i = 1 \ldots M,\ k = 1, 2 \ldots T; \tag{45}$$

With necessity constraint

$$\frac{\sum_{i=1}^{N} u_i(k) - W_1'}{W_2' - W_1'} \le 1 - \eta_1\ \forall\, k = 1, 2 \ldots T \tag{46}$$

Lingo19 solves the discrete formulation of the proposed model mentioned above. The number of modules, indicated by the value of M, is assumed to be 3. Twelve equal-length time periods are considered to make up the total time interval. In Table 1, the values for the various parameters are given. These values are often provided by the developer based on prior experience, but they are assumed for numerical examples [3].

The total available budget which is assumed to be fuzzy in nature, is represented by the triangular fuzzy number and the value of (W_1', W_2', W_3') is taken as (240, 440, 540).

Table 2 Optimal value of Fault detected by time t

Time Periods	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12
$\Omega_1(t)$	10	14	19	23	28	33	37	43	48	53	59	64
$\Omega_2(t)$	10	14	19	24	29	34	39	44	50	55	61	67
$\Omega_3(t)$	10	15	19	24	29	35	40	45	51	57	63	69

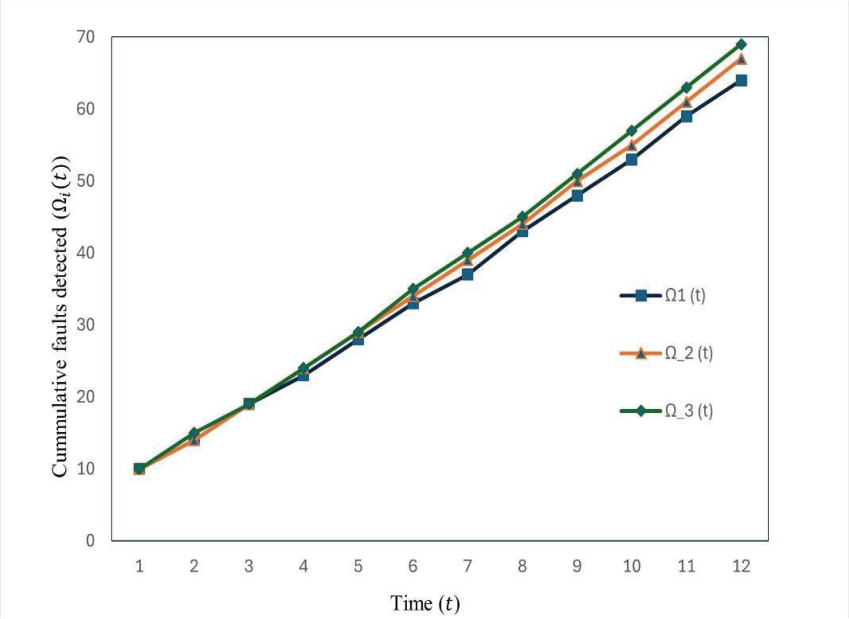

Figure 1 Optimal values of faults detected when cost are linear function of testing effort.

Table 3 Optimal value of Testing Effort by time t

Time Periods	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12
$u_1^*(t)$	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	1
$u_2^*(t)$	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	1
$u_3^*(t)$	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	0.40	1

When considering cost as a linear function of testing efforts for correcting during the testing and operational phases, the total minimum optimal cost is 186518.6. Table 3 shows the optimal testing effort rates for the three modules. Table 2 displays the values of optimal faults obtained in each module, which are depicted in Figure 1 for three separate modules during the planning period.

Table 4 Optimal value of Fault detected by time t

Time Periods	T1	T2	T3	T4	T5	T6	T7	T8	T9	T10	T11	T12
$\Omega_1(t)$	10	21	32	44	57	71	85	100	114	129	143	157
$\Omega_2(t)$	10	21	33	46	59	74	89	104	120	135	149	164
$\Omega_3(t)$	10	21	34	47	61	76	92	107	123	138	153	167

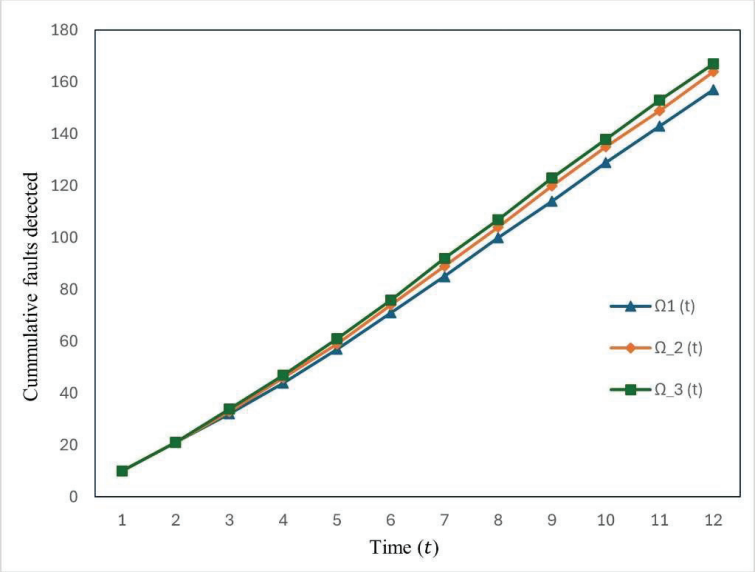

Figure 2 Optimal values of faults detected when cost are constant.

The optimal testing effort for each module has upper values $u_i^u = 1$ and the values of total optimal cost obtained is 454741.2, when we consider testing costs are constant for correcting during testing and operational phase. The values of optimal faults obtained in each module depicted in Table 4 and plotted in Figure 2 for three different modules in planning period.

Taking into account necessity and/or possibility limitations for both cost scenarios, we find that when testing costs are constant and independent to testing effort, the necessity/possibility testing effort constraints require a greater degree of testing effort. Therefore, a higher degree of testing effort leads to a higher defect detection rate. However, when testing cost is a linear function of testing effort, we find that the necessity/possibility constraints for the second cost scenarios require a lower level of testing effort. Therefore, a decision maker will have the freedom to allocate testing effort resources at a lower level if they wish to strictly adhere to the limited resource limitations.

Additionally, the overall optimal testing cost and cumulative detected faults are greater in the case of constant cost with budgetary limits than in the case of testing effort dependent cost.

7 Conclusion

The testing effort based SRGM can be used to characterize the link between the testing resources used during module testing and the software defects found. How much testing resources should be allotted to each module has been the focus of the testing resource allocation problem. Although each software module has a distinct function, they might not all be equally significant. While some may just provide a few features, others might offer the software significant capability. While faults in certain modules may be as simple as possible, those in others may be more complex. It necessitates accurate monitoring of testing progress as well as careful allocation of testing resources among modules. To reduce testing costs, software project managers should continuously monitor the testing process and allocate resources effectively.

The testing resources that are available are typically limited during the software testing process. In the context of an optimal control problem, these constraints are often thought to be deterministic. This isn't actually the case. For instance, a budget for the use of testing resources may be the first step in the testing process. However, it's possible that fault detection is accelerated during the fault testing procedure to meet unexpectedly increasing faults, necessitating the allocation of some more funding for testing resources. It is clear that nature is unaware of these increased numbers. Given the degree of uncertainty in this situation, fuzzy set theory makes sense as a model; as a result, the testing of resource expenditure limits, such as the overall budget, becomes imprecise. According to the proposed model, a triangle fuzzy number represents the entire testing resource budget that is available. In such circumstances, you can replicate the decision maker's subjective imaginations as accurately as they can be expressed by modeling the problem in a fuzzy environment. The optimal control theory technique in a fuzzy environment is becoming increasingly essential, as some additional sources contribute to uncertainty in issue descriptions.

In this paper, we present an alternative explanation for optimal testing resource allocation utilizing fuzzy constraints in a dynamic setting. This allows us to easily manage the consumption rate of testing-effort expenditures and identify more problems in a given time interval. This means that developers and testers may focus their time and resources to completing testing jobs

while keeping costs under control. This study defines necessity/possibility constraints in the context of the allocation of testing effort resources for modular software systems, which are define and defuzzified. We use fuzzy set theory to deal with this uncertainty and use a triangular fuzzy number to describe the budget. Using necessity constraints, the fuzzy optimal control model is developed and solved using optimal control theory approach. The model is further discretized and solved using the LINGO19 Software. In the future, the proposed model can be extended to various testing resource allocation models that use fuzzy parameters to describe the budget and fault detection rate with different types of testing effort functions.

Declarations

Disclosure Statement

The authors declare no conflicts of interest or disclosures to report.

Funding

The authors declare that this research was conducted without any specific grant or funding support.

Data Availability Statement

The data that support the findings of this study have been included in the manuscript.

References

[1] Hailpern, B. and Santhanam, P., 2002. Software debugging, testing, and verification. *IBM Systems Journal*, *41*(1), pp. 4–12.

[2] Xie, M. and Yang, B., 2003. A study of the effect of imperfect debugging on software development cost. *IEEE Transactions on Software Engineering*, *29*(5), pp. 471-473.

[3] Kapur, P.K. and Garg, R.B., 1990. Cost reliability optimum release policies for a software system with testing effort. *Operations Research*, *27*(2), pp. 109–116.

[4] Yamada, S., Hishitani, J. and Osaki, S., 1993. Software-reliability growth with a Weibull test-effort: a model and application. *IEEE Transactions on Reliability*, *42*(1), pp. 100–106.

[5] Huang, C.Y., Kuo, S.Y. and Chen, Y., 1997, November. Analysis of a software reliability growth model with logistic testing-effort function. In *Proceedings The Eighth International Symposium on Software Reliability Engineering* (pp. 378–388). IEEE.

[6] Kapur, P.K. and Bardhan, A.K., 2002. Testing effort control through software reliability growth modelling. *International Journal of Modelling and Simulation*, *22*(2), pp. 90–96.

[7] Kapur, P.K., Gupta, A., Shatnawi, O. and Yadavalli, V.S.S., 2006. Testing effort control using flexible software reliability growth model with change point. *International Journal of Performability Engineering*, *2*(3), p. 245.

[8] Yamada, S., Ohba, M. and Osaki, S., 2009. S-shaped reliability growth modeling for software error detection. *IEEE Transactions on reliability*, *32*(5), pp. 475–484.

[9] Peng, R., Li, Y.F., Zhang, W.J. and Hu, Q.P., 2014. Testing effort dependent software reliability model for imperfect debugging process considering both detection and correction. *Reliability Engineering & System Safety*, *126*, pp. 37–43.

[10] Kapur, P. K., Bardhan, A. K., and Jha, P. C. (2003). Optimal reliability allocation problem for a modular software system. *Opsearch*, *40*, 138–148.

[11] Kapur, P.K., Jha, P.C. and Bardhan, A.K., 2004. Optimal allocation of testing resource for a modular software. *Asia-Pacific Journal of Operational Research*, *21*(03), pp. 333–354.

[12] Huang, C.Y. and Lo, J.H., 2006. Optimal resource allocation for cost and reliability of modular software systems in the testing phase. *Journal of Systems and Software*, *79*(5), pp. 653–664.

[13] Kapur, P.K., Bardhan, A.K. and Yadavalli, V.S., 2007. On allocation of resources during testing phase of a modular software. *International Journal of Systems Science*, *38*(6), pp. 493–499.

[14] Khan, M.G., Ahmad, N. and Rafi, L.S., 2016. Determining the optimal allocation of testing resource for modular software system using dynamic programming. *Communications in Statistics-Theory and Methods*, *45*(3), pp. 670–694.

[15] Singh, Y., Manik, P. and Chaudhary, K., 2013. Optimal production policy for multi-product with inventory-level-dependent demand in segmented market. *YUJOR*, *23*(2), pp. 237–247.

[16] Mehta, S., Chaudhary, K. and Kumar, V., 2020. Optimal promotional effort policy in innovation diffusion model incorporating dynamic

market size in segment specific market. *International journal of mathematical, engineering and management sciences*, *5*(4), pp. 682–696.

[17] Kumar, P., Chaudhary, K., Kumar, V. and Singh, V.B., 2022. Advertising and pricing policies for a diffusion model incorporating price sensitive potential market in segment specific environment. *International Journal of Mathematical, Engineering and Management Sciences*, *7*(4), pp. 547–557.

[18] Kumar, P., Chaudhary, K., Kumar, V. and Chauhan, S., 2023. Impact of goodwill on consumer buying through advertising in a segmented market: an optimal control theoretic approach. *Axioms*, *12*(2), p. 223.

[19] Chaudhary, K., Kumar, P., Chauhan, S. and Kumar, V., 2022. Optimal promotional policy of an innovation diffusion model incorporating the brand image in a segment-specific market. *Journal of Management Analytics*, *9*(1), pp. 120–136.

[20] Chaudhary, K., Singh, Y. and Jha, P.C., 2011, January. Optimal control policy of a production and inventory system for deteriorating items in segmented market. In *Proceedings of the 2011 International Conference on Industrial Engineering and Operations Management* (pp. 1009–1015).

[21] Kapur, P.K., Pham, H., Chanda, U. and Kumar, V., 2013. Optimal allocation of testing effort during testing and debugging phases: a control theoretic approach. *International Journal of Systems Science*, *44*(9), pp. 1639–1650.

[22] Kumar, V. and Sahni, R., 2016. An effort allocation model considering different budgetary constraint on fault detection process and fault correction process. *Decision Science Letters*, *5*(1), pp. 143–156.

[23] Kumar, V., Kapur, P.K., Taneja, N. and Sahni, R., 2017. On allocation of resources during testing phase incorporating flexible software reliability growth model with testing effort under dynamic environment. *International Journal of Operational Research*, *30*(4), pp. 523–539.

[24] Kumar, V. and Sahni, R., 2020. Dynamic testing resource allocation modeling for multi-release software using optimal control theory and genetic algorithm. *International Journal of Quality & Reliability Management*, *37*(6/7), pp. 1049–1069.

[25] Pradhan, S.K., Kumar, A. and Kumar, V., 2024. An optimal software enhancement and customer growth model: a control-theoretic approach. *International Journal of Quality & Reliability Management*, *41*(9), pp. 2333–2350.

[26] Pradhan, S.K., Kumar, A. and Kumar, V., 2025. Modeling reliability-driven software release strategy considering testing effort with fault detection and correction processes: A control theoretic approach. *International Journal of Reliability, Quality and Safety Engineering*, *32*(02), p. 2440002.

[27] Dubois, D. and Prade, H., 1983. Ranking fuzzy numbers in the setting of possibility theory. *Information sciences*, *30*(3), pp. 183–224.

[28] Dubois, D. and Prade, H., 1987. The mean value of a fuzzy number. *Fuzzy sets and systems*, *24*(3), pp. 279–300.

[29] Chakraborty, D., 2002. Redefining chance-constrained programming in fuzzy environment. *Fuzzy Sets and Systems*, *125*(3), pp. 327–333.

[30] Liu, B. and Iwamura, K., 1998. Chance constrained programming with fuzzy parameters. *Fuzzy sets and systems*, *94*(2), pp. 227–237.

[31] Maiti, M.K. and Maiti, M., 2006. Fuzzy inventory model with two warehouses under possibility constraints. *Fuzzy Sets and systems*, *157*(1), pp. 52–73.

[32] Samal, U., Kushwaha, S., Nain, G., Singh, S., Usmani, S. and Kumar, A., 2025. Necessity of fuzzy logic: Trends in software reliability assessment. In *Reliability Assessment and Optimization of Complex Systems* (pp. 275–285). Elsevier.

[33] Kapur, P.K., Pham, H., Gupta, A. and Jha, P.C., 2011. *Software reliability assessment with OR applications* (Vol. 364). London: Springer.

[34] Pham, H., 2007. *System software reliability*. Springer Science & Business Media.

[35] Musa, J.D., 1987. *Software reliability: Measurement, Prediction, Applications*. New York: Mc Graw Hill.

[36] Seierstad, A. and Sydsaeter, K., 1986. *Optimal control theory with economic applications*. Elsevier North-Holland, Inc.

[37] Ohba, M., 1984. Software reliability analysis models. *IBM Journal of research and Development*, *28*(4), pp. 428–443.

[38] Bittanti, S., Bolzern, P., Pedrotti, E., Pozzi, M. and Scattolini, R., 1988. A flexible modelling approach for software reliability growth. *Software reliability modelling and identification*, pp. 101–140.

[39] Jha, P.C., Gupta, D., Yang, B. and Kapur, P.K., 2009. Optimal testing resource allocation during module testing considering cost, testing effort and reliability. *Computers & Industrial Engineering*, *57*(3), pp. 1122–1130.

[40] Rosen, J.B., 1968. Numerical Solution of Optimal Control Problems. *Mathematics of the Decision Sciences*, *2*, pp. 37–45.

Biographies

Deepika Gaur is a promising young researcher serving as a Research Scholar at the Department of Mathematics within the Amity Institute of Applied Sciences at Amity University Uttar Pradesh, Noida, India. She received her MSc degree in Mathematics from Maharishi Dayanand University, Rohtak, India. Her research interest includes software reliability and mathematical modelling.

Pradeep Kumar received his MSc in Mathematics from CSJMU Kanpur India in 2016. He has completed his PhD from the Department of Mathematics, Amity University Noida, Uttar Pradesh, India. He has published more than 7 research papers in the areas of mathematical modelling and optimisation, marketing and software reliability in international journals and conferences.

Kuldeep Chaudhary is an accomplished academic professional serving as an Associate Professor at the Department of Mathematics, Amity Institute of Applied Sciences, Amity University Uttar Pradesh, Noida, India. He has published more than 30 research articles in the International Journals/book chapters/conferences. He is an editorial board member of IJSA, Springer. He is a life member of Society for Reliability Engineering, Quality and Operations Management (SREQOM).

Shivani Bali is a Professor and Area Chair (Business Analytics) at Jaipuria Institute of Management, Noida. She holds a Master's and Ph.D. from the Department of Operational Research, University of Delhi, India, and brings over two decades of expertise in teaching, research, corporate training, and consultancy in Artificial Intelligence (AI), Machine Learning (ML), Data Science (DS), and Business Analytics. She has published research articles in many reputed journals indexed in SCI/ WoS/Scopus and has authored two academic and three edited books. She also has three Patents granted in her name by the Government of India.

Vijay Kumar is a Professor at Department of Applied Mathematics, Amity University, Noida. He received his MSc in Applied Mathematics and MPhil in Mathematics from Indian Institute of Technology (IIT), Roorkee, India in 1998 and 2000, respectively. He has completed his PhD from the Department of Operational Research, University of Delhi. Currently, he is a Professor in the Department of Mathematics, Amity Institute of Applied Sciences, Amity University, Noida, India. He is co-editor of two book and has published more than 70 research papers in the areas of software reliability, mathematical

modelling and optimisation in international journals and conferences of high repute. His current research interests include software reliability growth modelling, optimal control theory and marketing models in the context of innovation diffusion theory. He has edited special issues of IJAMS and RIO journal. He is an editorial board member of IJSA, Springer. He is a life member of Society for Reliability Engineering, Quality and Operations Management (SREQOM).