SELECTIVE MAINTENANCE OPTIMIZATION USING MACHINE LEARNING AND AGENT BASED APPROACH

M.Tech. Thesis

By

KUDALI SATISH TARUN

DEPARTMENT OF MECHANICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

JUNE 2021

SELECTIVE MAINTENANCE OPTIMIZATION USING MACHINE LEARNING AND AGENT BASED APPROACH

A THESIS

Submitted in partial fulfillment of the requirements for the award of the degree

of

Master of Technology

by

KUDALI SATISH TARUN

DEPARTMENT OF MECHANICAL ENGINEERING

INDIAN INSTITUTE OF TECHNOLOGY INDORE

JUNE 2021

INDIAN INSTITUTE OF TECHNOLOGY INDORE

CANDIDATE'S DECLARATION

I hereby certify that the work which is being presented in the thesis entitled **"SELECTIVE MAINTENANCE OPTIMIZATION USING MACHINE LEARNING AND AGENT BASED APPROACH"** in the partial fulfillment of the requirements for the award of the degree of **MASTER OF TECHNOLOGY** and submitted in the **DEPARTMENT OF MECHANICAL ENGINEERING, Indian Institute of Technology Indore**, is an authentic record of my own work carried out during the time period from May 2020 to June 2021 under the supervision of **Dr. Bhupesh Kumar Lad**, Associate Professor, Department of Mechanical Engineering, IIT Indore.

The matter presented in this thesis has not been submitted by me for the award of any other degree of this or any other institute.

Tarun
03-06-2021
Signature of the Student
(KUDALI SATISH TARUN)

This is to certify that the above statement made by the candidate is correct to the best of my knowledge.

4/6/21
Signature of the Supervisor

(Dr. BHUPESH KUMAR LAD)

KUDALI SATISH TARUN has successfully given his M.Tech. Oral Examination held on **4th June 2021.**

4/6/21
Signature of Supervisor of M.Tech. thesis
Date:

Satyanarayan
Convener, DPGC
Date: 07/06/2021

Signature of PSPC Member #1
Date: 4/6/2021

Girish
Signature of PSPC Member #2
Date: 07/06/21

ACKNOWLEDGEMENT

I take this opportunity to express my earnest respect and sincere gratitude to Dr. Bhupesh Kumar Lad for believing in me to carry out this research under his guidance. His vast experience and plethora of knowledge coupled with constant motivation, friendly interactions, and constructive support have enabled successful completion of this research work. His innovative perspective towards things and his continuous pursuit for perfection has had a profound effect on me and has transformed me majorly. I am forever grateful to him for providing invaluable career advice and suggestions extending beyond academic boundaries, whenever needed. I am honoured to be one of his students.

I am immensely grateful to Prof. Makarand Kulkarni, IIT Bombay for taking time from his busy schedule and providing valuable suggestions. Discussions with him have been extremely enriching and his advices have significantly shaped my research.

I am extremely thankful to Mr. Ram Mohril for being immeasurably helpful and always having my back whenever I ran into hurdles. He has been a constant presence over the entire duration of my research and has always rescued me at crucial junctures with his suggestions. Continuous moral support and countless discussions with him over topics ranging from research to career advice, have helped me have a smooth journey during my entire time in IIT Indore.

I am thankful to IIT Indore for giving me opportunity to carry out the research work and Industrial and Systems Engineering lab for providing all the required computational facilities.

I am immensely grateful to have spent most of my time at IIT Indore with M.Tech 2019 batch; especially Umesh Mali, Priyanshu Upadhyay, Piyush Chand, Deepak Kag and Vivek Kumar for being a family away from my family. I also express my gratitude to Mr. Manish Carpenter and Mr. Shriprasad for maintaining a very amicable atmosphere in the lab and helping me out whenever needed. I am also thankful to Mr. Aniket Adsule for his valuable suggestions during a very crucial moment in my research.

Finally, I am forever grateful to the two most important persons of my life. My father, K.R.Satish Babu and mother, K.Anuradha, for always being the pillars of my life and supporting me in every situation and every decision I take.

- *K.S.Tarun*

ABSTRACT

A Selective Maintenance (SM) policy is used for the maintenance of equipment that works in mission mode. Most of the approaches for Selective Maintenance Optimization (SMO) require higher computation time to determine the maintenance policy, which is undesirable. The advent of machine learning opens up new horizons for developing novel approaches that have the potential to substantially reduce computation time in SMO. The rapid rise in the use of sensors and computing infrastructure is transforming conventional industrial systems into smart machines. There is an opportunity to embrace this smartness in every aspect of industrial systems. Maintenance planning is one such inherent aspect. Technologies like multi-agent systems are making a move from centralized decision making to distributed realization of decision making.

This project proposes to develop a novel Reinforcement Learning (RL) based methodology and a distributed algorithm for intelligent maintenance planning to optimize the selective maintenance problem. As part of the RL based methodology, a temporal difference learning algorithm - *Q-Learning* is used to solve this optimization problem where the agent chooses a policy at the end of an epoch, based on the updated Q-Values. Different heuristics are embedded with the methodology to effectively determine the optimal policy, of which one smartly reduces the solution space and the other aids in increasing the agent's intelligence based on the reward policy. The objective of the maintenance optimization problem is to maximize the system reliability and it is formulated as a Semi-Markov decision process (SMDP). The reward function is also defined in a way that the agent will try to determine the best strategy with minimum consumption of maintenance resources.

The algorithm designed for agent based distributed maintenance planning fits into the Industrial Internet of Things (IIoT) paradigm and revolves around the idea of having individual agents to make maintenance decisions for respective subsystems and an overall coordinating agent that will decide the optimal policy from the preferences given by the subsystem level agent and decides what maintenance policy best for the enterprise

The efficiency of the developed algorithms is demonstrated by applying it to a benchmark multi-state industrial system for coal transportation. The results accentuate the supremacy of the developed RL based algorithm and agent based distributed approach over the commonly used methods like the enumeration approach and genetic algorithm based approach.

Table of Contents

LIST OF FIGURES

LIST OF TABLES

LIST OF GRAPHS

ABBREVIATIONS

SM	Selective Maintenance
SMO	Selective Maintenance Optimization
EM	Enumeration method
BF	Brute-Force algorithm
GA	Genetic Algorithm
ML	Machine Learning
RL	Reinforcement Learning
PM	Preventive Maintenance
DRL	Deep Reinforcement Learning
MSS	Multi State Systems
IoT	Internet of Things
IIoT	Industrial Internet of Things

Chapter 1 Introduction

1.1 Maintenance

The technical meaning of maintenance encompasses all tasks like functional checks, servicing, repairing or replacing of necessary components, equipment, machinery and supporting utilities in industrial, business and residential installations.

Maintenance is strictly connected to the utilization stage of the product or technical system, in which the concept of maintainability must be included. In this scenario, maintainability is considered as the ability of an item, under stated conditions of use, to be retained in or restored to a state in which it can perform its required functions, using prescribed procedures and resources.

In some domains like aircraft maintenance; terms like maintenance, repair and overhaul also include inspection, rebuilding, alteration and the supply of spare parts, accessories, raw materials, adhesives, sealants, coatings and consumables for aircraft maintenance at the utilization stage.

In international civil aviation maintenance means:

The performance of tasks required to ensure the continuing airworthiness of an aircraft, including any one or combination of overhaul, inspection, replacement, defect rectification, and the embodiment of a modification or a repair.

1.1.1 Major types of maintenance

- *Preventive Maintenance*

  Maintenance is carried out at regular fixed intervals or according to a given criteria to reduce the risk of failure or degradation of performance of the equipment.

- *Condition-based Maintenance*

  Maintenance is done on the equipment by continuously monitoring the performance and controlling the corrective actions being taken. Maintenance is performed when certain sensors show that the performance is deteriorating and probability of failure is increasing.
- *Risk-based Maintenance*

  This is a maintenance strategy which prioritizes maintenance resources towards assets that pose the most risk if they fail.
- *Corrective Maintenance*

  Maintenance strategy which involves carrying out maintenance activities to restore normal operating conditions only when an anomaly is detected.

1.2 Industry 4.0

The world, as we know it today, was greatly influenced by the three major technological revolutions, as shown in the figure. We are currently at the cusp of experiencing a paradigm shift in industries in the form of fourth industrial revolution (termed as Industry 4.0).

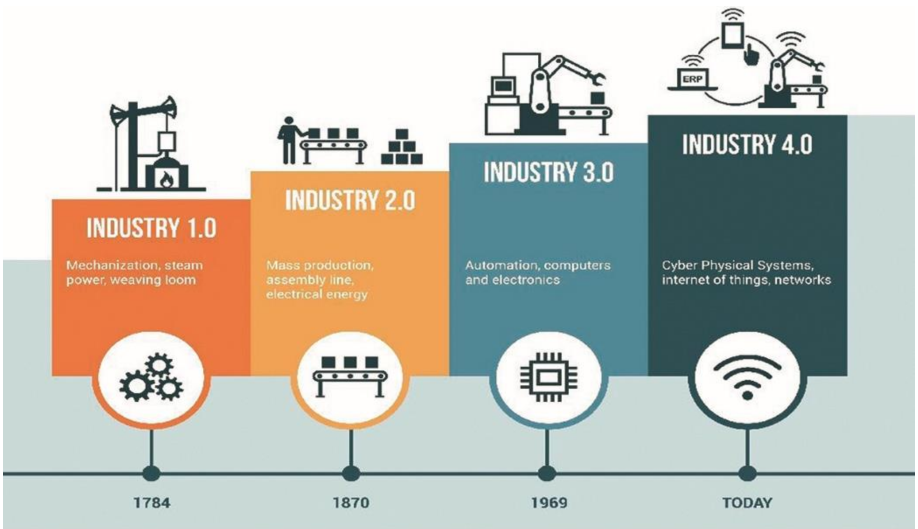


Figure 1.1: Chronology of Industrial Revolutions

Industry 4.0 is the automation of conventional manufacturing and industrial practices, using smart technologies like mobile devices, Internet of Things (IoT) platforms, smart sensors, augmented reality etc. Self-monitoring of systems and automation can be increased by integrating IoT and Machine-to-machine communication, which helps analyze and diagnose issues without human intervention.

1.2.1 Effect of Industry 4.0 on Maintenance

Maintenance planning is a strategic concern for industries, because maintenance activities carried out inappropriately leads to inefficient usage of assets. With the advent of Industry 4.0, Industrial Internet of Things (IIoT) provides a smarter approach by continuously analyzing data to achieve useful insights and predict failure of the system, increase system uptime and improve asset efficiency. The dawn of the fourth industrial revolution has created a new maintenance strategy known as predictive maintenance.

Predictive maintenance is using sensors and other technologies to collect and analyse data from the machines to accurately predict when maintenance work is needed. It uses an analytical approach by utilizing real time data and past data to accurately identify the fault in the machine so that it can be repaired ahead of time.

Benefits of Predictive maintenance:

- Continuous monitoring of the health of the machines.
- Time to intervene before the system undergoes catastrophic failure.
- Reduction in machine/system downtime.
- Early recognition of wear and tear of critical components.

1.3 Selective Maintenance

1.3.1 Introduction

Many industrial and military applications require systems to perform a sequence of missions with a maintenance break between two successive missions. However, performing maintenance on all components of the system isn't feasible due to the limited duration of the maintenance break. In situations like these, the decision-maker needs to decide on a subset of components to perform maintenance. This maintenance strategy is called Selective Maintenance (SM) and it is defined as a policy of determining which set of maintenance actions to perform when given a set of limited maintenance resources such as time, cost, spares and crew [1,2]. Selective maintenance is widely considered as a type of profit generating maintenance policy which plays a crucial role in balancing limited maintenance resources with system performance.

The Selective maintenance policy is widely used in the maintenance of military equipment and aircraft as they perform a sequence of missions. After a mission ends, the decision-maker needs to decide on the maintenance strategy to be followed to ensure the successful completion of the next mission. But as the maintenance break duration is a resource constraint, the decision-maker can't invest much time in deciding which maintenance policy to be adopted because it diminishes the maintenance break duration.

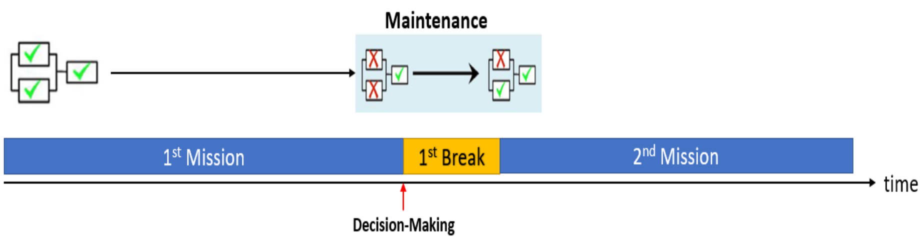


Figure 1.2: Illustration of Selective Maintenance

Selective Maintenance, a policy which aims to 'do more with minimum resources' comes under the paradigm of maintenance modeling and optimization [3]. SM has some important features which make it very practical to use in some scenarios. They are:

i. *SM is mission oriented.*

   SM is majorly used in those scenarios where a system is required to execute a sequence of missions and maintenance actions can be performed in the break between two successive missions.

ii. *SM is condition based.*

SM decisions are greatly influenced by systems' health status and mission profiles. In comparison with conventional condition-based maintenance policies, condition in SM has a wider array of factors to be considered.

1.3.2 Criteria for Selective Maintenance decision making

SM decision making to obtain optimal maintenance schedule considers various parameters such as system configuration, maintenance policies, maintenance degrees, optimization criteria, planning horizon etc. All these factors can be broadly categorized into 3 categories to provide a holistic structure. They are System Characteristics, Maintenance Characteristics and Mission Profile Characteristics.

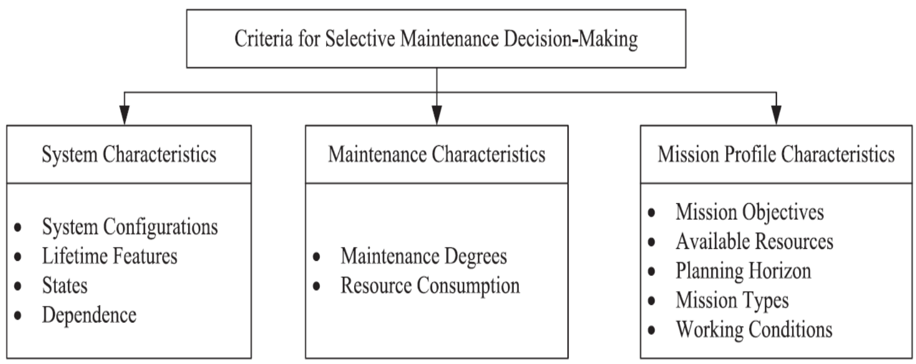


Figure 1.3: Criteria for SM decision making

I. System characteristics

System characteristics include machine's inherent features which are decided by its design and includes its configuration, lifetime features and states.

- *Systems Configurations:*

  The system configuration considered for selective maintenance optimization is designated as Series, Parallel and Complex (Series-Parallel) configurations. In a series configuration system, if a component fails; it is necessary to repair it at once. But in a system with parallel configuration, the system will be operational even if the failed component isn't repaired immediately.

- *Lifetime features:*

  All components of a system are associated with different lifetime features which can be explained by parametric or non-parametric information. The commonly used distributions for SM modeling are Exponential distribution and Weibull distribution. Exponential distribution is followed by those component's which have a constant failure rate, whereas Weibull distribution considers component's whose failure rate is time dependent.

- *States*

  States of the components of a system can be classified into binary or multiple. In case of binary states, a component at a specific instant can be in one of the two states, i.e., Working or Failed. But in many practical applications, the state of a system or component can vary between perfect functioning to complete failure.

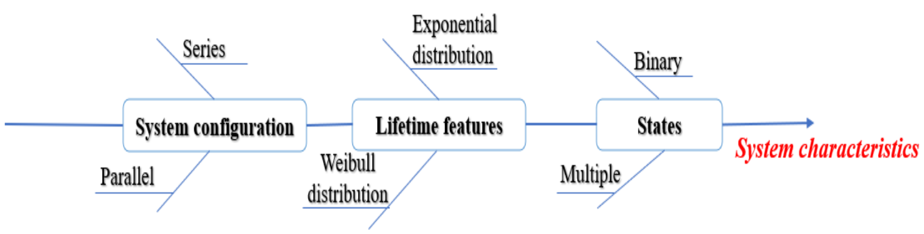


Figure 1.4: System Characteristics

II. Maintenance characteristics

Maintenance characteristics refer to the factors comprehending maintenance that can have a major effect on maintenance decisions like type of maintenance action, resource consumption etc.

- *Maintenance Degrees*

  According to the degree to which the component or system is restored, state of any component or system after maintenance will be somewhere between *as bad as old* and *as good as new.*

This is called Imperfect maintenance. Here, the condition of the system/component is improved. Imperfect maintenance ensures the modeling of SM problem is more realistic in nature.

- *Resource Consumption*

  Maintenance budget and Maintenance time are the two major maintenance resources in SM modeling. Other resources include maintenance crew and spare parts etc. Considering resources as constraints in SM modeling will make the model more practical and closer to industrial scenario.

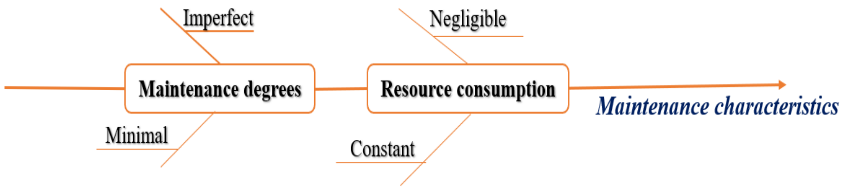


Figure 1.5: Maintenance Characteristics

III. Mission profile characteristics

Mission profile is the task which is to be executed by the system during the defined time under some specific conditions. Features of mission profile include objectives of the mission, planning horizon, type of mission and the working conditions.

- *Mission Objectives*

  To ensure the system completes a mission successfully, the objectives generally considered in SMO is to minimize maintenance time / maintenance cost or to maximize the system reliability.

- *Planning Horizon*

  It refers to the duration of mission considered for SMO. Any system can execute a single mission of a fixed time duration or multiple missions over a fixed time horizon.

1.4 Computational Complexity in SM

The problem of SMO is modeled as a non-linear programming problem with an objective to either maximize the system reliability or minimize the maintenance cost. It has constraints in the form of maintenance cost being less than the budget and maintenance time being within the maintenance break duration. So, the computational algorithm should choose the best set of maintenance actions to be performed on the system. For a system which has "*n*" components with "*m*" possible maintenance actions for each component, the total number of maintenance actions becomes m^n. So, the optimization problem becomes combinatorial in nature thereby increasing the computational complexity. Irrespective of the complexity involved, it is necessary to solve the SM problem in the minimum possible duration. As the limited available maintenance duration is itself a constraint in the problem, any approach which requires higher computation time is undesirable, since the decision-making process itself will consume most of the maintenance duration.

1.4.1 Necessity of efficient computational approaches

Initial research on SMO focused on using the enumeration approach. But it was found that this method was not suitable when applied to problems with larger solution space. This led to the development of the next generation of methods coupled with heuristics, like tabu search, genetic algorithm, particle swarm optimization, differential evolution approach etc. for optimizing the SM problem. They were effective in reducing the computation time significantly compared to the enumeration method, but the optimality of the solution was not guaranteed always. There is a need for a comprehensive approach which provides optimal or near-optimal result in significantly lesser computation time. So, the advent of Machine learning and Industry 4.0 paves way for development of novel and intelligent computational algorithms with the potential to be much more efficient than the existing approaches.

1.5 Research Objectives

I. Develop an Evolutionary algorithm-based approach for Selective maintenance optimization of a complex industrial system.

II. Develop a Machine learning based approach for Selective maintenance optimization of the complex industrial system. Reinforcement learning, a Machine learning paradigm is used to develop an algorithm which learns through experience.

III. Develop Multi agent based distributed approach for Selective maintenance optimization of the complex industrial system. To reap the benefits of interconnected assets which have more access to data and enable better decision making.

IV. Perform a comparative study of the above approaches with conventional approach of Enumeration method for the benchmark problem of coal transportation system.

1.6 Novelty of the project

Major innovations made in this project:

- Development of a neoteric Reinforcement Learning based algorithm for selective maintenance optimization problem.
- Development of a novel Agent based Distributed maintenance planning algorithm for selective maintenance optimization.

1.7 Thesis Organization

The thesis is broadly divided into seven chapters. The current chapter introduces the reader to the background of the work and the basics about maintenance and selective maintenance and also outlines the research objectives.

Chapter 2 presents a comprehensive literature review of the state-of-the-art research in selective maintenance. It also addresses the shortcomings of the previous research and provides a holistic view on the evolution of the selective maintenance optimization approaches till date.

Chapter 3 deals with the detailed problem formulation and also the aspects about Reliability estimation. This chapter introduces to the reader the benchmark problem considered in the subsequent chapters to solve the SMO problems with the developed algorithms. Finally, the methodology of implementing the Enumeration method along with the results obtained by using this algorithm is also discussed in this chapter.

Chapter 4 provides a brief introduction about the evolutionary algorithms. The methodology of its implementation for SMO problem is discussed in detail along with the various biological operators used. Then the algorithm is implemented on the benchmark problem defined in the previous chapter and the results are compared with those obtained using enumeration method.

Chapter 5 starts with the introduction to machine learning and its paradigm, reinforcement learning. It then discusses about how RL is applied for SM optimization problem and the modeling of the problem as a Semi-Markov decision process. The application of Q-Learning for solving this problem is also discussed along with the various novel heuristics defined. Finally, it is implemented on the benchmark problem and the results are compared with the earlier proposed methodologies.

Chapter 6 explains the development of a novel decentralized distributed maintenance planning algorithm. It starts with the introduction about distributed decision making and IIoT, followed by methodology of implementation of this approach. Reliability allocation is explained in detail and the algorithm is applied on the benchmark problem to and the results are compared with the proposed methodologies.

Chapter 7 discusses about the conclusions of the research work and also lays the foundation for future work to realize Industry 4.0.

Chapter 2 Literature Review

This section provides a comprehensive review of the state-of-the-art research in selective maintenance pertaining to the various system characteristics like system configuration, lifetime features, and various maintenance characteristics. The progression of research on selective maintenance can be broadly categorized into three generations. First generation research on selective maintenance, predominantly focused on modelling the problem by altering the system characteristics. In the second generation, researchers in a bid to reduce computation time, explored the use of various computational algorithms along with employing different kinds of heuristics. Presently in the presumed third generation, researchers are focused on further improving the computation efficiency with the aid of contemporary techniques like machine learning algorithms and incorporating Industry 4.0 technologies.

2.1 Evolution of SMO problem (I generation)

The introductory study on selective maintenance optimization was performed on a system with series-parallel configuration and constant component failure rates i.e., Exponential distribution [2]. The model could optimally decide a subset of failed components to be replaced before the next mission to maximize the system reliability of the next mission. Later, researchers improvised this by studying about a case in which the system configuration was complicated and also the components in each subsystem were not identical. Further, cost was also included as an additional resource constraint [4]. Then, the problem of selective maintenance optimization was illustrated on systems with partially redundant structures (components arranged in series and parallel) [5]. Following the earlier research, age of the component was introduced as a factor in reliability estimation and it was assumed that components' lifetimes follow Weibull distribution. Multiple maintenance actions were considered and repair action on the component could be minimal repair of failed components, replacement of failed components and replacement of functioning components (preventive maintenance) [6]. Then, imperfect maintenance was considered amongst the maintenance actions for selective maintenance optimization wherein the component was restored to the condition somewhere between as good as new and as bad as old and age of the component after maintenance was also affected by the maintenance action [7–9].

The prime focus of researchers lied in the development of computationally efficient approaches for solving this complex optimization problem. Enumeration method was only suitable for selective maintenance optimization problems with small solution space. The number of feasible solutions increased exponentially with increase in number of components and the method became computationally intractable. Heuristics were defined and a modified enumeration method with upper bounds was used initially [1]. The efficiency of enumeration method was improved by applying upper bounds and lower bounds on decision variables, applying objective function bounds based on branch and bound concepts and also by iterating through the values of the decision variables in descending order [10].

2.2 II generation methodologies for SMO

Construction heuristic and Tabu search were both implemented for solving this problem and Tabu search was found to be better as it gave a near optimal solution though the optimality was not guaranteed always [5]. Later use of Evolutionary algorithms started gaining prominence. Genetic Algorithm (GA) was employed to solve the selective maintenance optimization problem in which both multi-state systems and imperfect maintenance models were considered [7,11,12]. In a research to identify the maintenance actions before the start of maintenance break along with simulation to forecast the requirement of spares before and during a mission, GA was used to compute the optimum solution [13]. Simulated Annealing (SA) was applied on a selective maintenance optimization problem for multi-state system operating for more than one mission. Differential Evolution (DE) was applied to solve the selective maintenance optimization problem for selective maintenance scheduling over a finite planning horizon [8,14]. An exact method to solve selective maintenance optimization problem considering imperfect maintenance was proposed which reduced the solution space by 42.46 % compared to enumeration method [15]. To identify the optimal maintenance policy in a computationally efficient manner, use of Ant colony optimization (ACO) algorithm was proposed, where the algorithm can be tailored to search for the global optimum and tackle the constraints and infeasible solutions by constructing a tabu list [16].

Despite using various efficient computation algorithms for selective maintenance optimization of multi state systems there was no assurance that the optimum solution will be reached and there was still the possibility to reduce the computation time. In order to ensure accuracy of selective maintenance optimization results and tackle the underlying issue of large solution space which was affecting the computation time, researchers felt the need of using some heuristics. One such attempt was to use a heuristic to prioritize components for maintenance in this break based on the cost, time or reliability threshold. and reduce the solution space [17]. Generally, both break durations and mission durations are considered to be deterministic in nature. But researchers in a study considered the break duration and mission duration to be stochastic in nature and the problem of selective maintenance optimization in that case was formulated as a non-linear stochastic optimization problem [18]. Most of the research on selective maintenance optimization considered the assignment of repair task to repairpersons as a different problem. But integrating both the selective maintenance and repairperson assignment problem provided more effective solutions in terms of cost and system reliability. Later, the joint optimization of selective maintenance and repairperson assignment problem was done by considering using remanufactured parts for replacement in place of new components which solved both economic purpose as well as favored sustainable practices [19]. It was observed that components in a multi-state system exhibited multiple performance levels and the selective maintenance optimization problem in that case was also modelled as a nonlinear programming problem and evolutionary algorithms were used to obtain the optimal solution [20]. Selective maintenance scheduling was also applied to cases where the system was required to execute multiple consecutive missions over a determined time horizon. A maintenance scheduling over a time horizon model under imperfect maintenance was developed to determine the optimum schedule with an aim to minimize the total cost [14]. A customized simulated annealing-based GA was used to solve the max-min optimization problem modelled for determining the selective maintenance strategy for systems executing multiple consecutive missions with uncertainty [21].

2.3 III generation methodologies for SMO

In the presumed third generation, with increase in intelligence of machines due to incorporation of sensors and increase in sophistication of computation infrastructure, researchers are focused on using these to further improve the efficiency of selective maintenance optimization. The application domain of selective maintenance optimization does not allow for higher computation time, which may not be acceptable in nuclear, maritime and army applications. The selective maintenance optimization problem for multi-state system that can execute multiple consecutive missions over a finite horizon was solved using a Deep Reinforcement Learning approach based on the framework of actor-critic algorithms [22]. This customized DRL algorithm overcame the 'curse of dimensionality' and mitigated the uncountable state space when dealing with systems containing large number of components. Further, in a study to develop a selective maintenance optimization model for an intelligent multi state manufacturing system, the problem was formulated as a constrained combinatorial optimization problem and Particle Swarm Optimization (PSO) was used to solve this as it has higher convergence speed than other traditional optimization algorithms [23].

The distributed approach in operations planning is primarily centred around the notion of multi agent systems. Multi agent paradigm is characterized by decentralization and parallel execution of activities based on autonomous entities, called agents [24]. Agent systems, due to their inherent characteristic of decentralization along with autonomy, adaptability, coordination, cooperation, and robustness clubbed with reasoning ability, pose a promising platform for developing decision support systems [25]. In multi agent systems, problem solving is a group effort wherein agents of different types with partial access to system information collaborate to achieve a system goal. [24,26] have used the distributed approach for planning and decision making in several industrial arenas. Literature on distributed approaches in decision making for production planning suggests the supremacy of it over the conventional centralized approach. For example, [27] reported 50% reduction in computation time in operation planning when compared to centralized approach. In a nutshell, literature highlights the supremacy of employing distributed approach for decision making over the conventional centralized approaches. This enhances the potential of employing the multi agent based distributed approach for complex maintenance planning problems.

2.4 Summary

There is a profound need for further research in the direction of reducing computation time in solving selective maintenance optimization problem for the all-inclusive industrial scenarios. The present work is an attempt to contribute to the research on SMO in the third generation to make it more applicable to the real industrial scenarios.

Application of distributed approach in selective maintenance optimization could be one of the effective ways to reduce the computation time significantly. This project proposes to extensively research on applying distributed decision-making approaches on the selective maintenance planning problem of complex industrial systems.

Chapter 3 Enumeration Method

This chapter describes the selective maintenance optimization problem with its objectives and the associated constraints of budget and time. It also discusses the maintenance actions that can be performed and its corresponding restoration factors, along with determining the maintenance cost and time. This chapter also introduces the benchmark problem considered in this project for SMO and the various parameters associated with it. Finally, the methodology and results obtained through the traditional approach of Enumeration method to solve this SMO problem is discussed in detail.

3.1 Problem Description

3.1.1 Selective Maintenance Optimization

After the completion of a mission, due to the stochastic nature of the failure of components, each component has a different age. SMO is performed to choose the best maintenance action to be undertaken on the components of the system during the maintenance break to ensure successful completion of the next mission; while the break is coupled with several constraints.

3.1.2 Multi-State system

A multi-state system is comprised of various components connected in different configurations.

- There are ***s (i = 1,2…..,s)*** subsystems connected in series.
- Each subsystem ***i*** has ***p (j = 1,2,…p)*** components connected in parallel.
- There are ***N*** possible maintenance actions.

3.1.3 Maintenance Actions

Maintenance can be performed on a component (i, j) during a maintenance break to restore the age of the component. Maintenance action chosen for the component (i, j) is denoted as $l_{i,j}$. In this project, the maintenance action options available for the decision-maker are given in Table 1.

The option of maintenance action '*0*' means there is no requirement to perform maintenance on the component. The actions *1, 2* and *3* restore the component to a state somewhere between 'as bad as old' and 'as good as new'. The component is completely replaced when action '*4*' is chosen.

$$l_{i.j} \in [0,1,2,3,4] \tag{1}$$

Table 1: Maintenance Actions

Maintenance action ($l_{i,j}$)	**Action**	**Restoration factor**
0	Do nothing	0
1	Minimal repair	0.15
2	Intermediate repair	0.50
3	Major repair	0.90
4	Component replacement	1

3.1.4 Restoration Factor (RF)

The concept of restoration factor (RF) is used to determine the effective age of the component after the maintenance. In this paper Restoration factor based on the Kijima type II model is followed [28]. Because, type II restoration factor assumes that the maintenance performed in this break fixes all the wear and damage to the component accumulated up to the current time. An RF of 1 implies the component is "As good as new".

➢ Example: Component's age = 700 hrs, RF = 0.75

Effective age = (1-0.75) *700 = 175 hrs

3.1.5 Maintenance Cost

A component may or may not be considered for maintenance depending on its current condition. If maintenance action '0' i.e. *Do Nothing* is chosen for a component (i,j) , it incurs zero maintenance cost. Any action other than 'Doing Nothing' incurs a maintenance cost corresponding to the action $l_{i,j}$ performed on the component. It is assumed that performing *Minimal repair* ($l_{i,j}$ = 1) costs x% of the component's replacement cost. *Intermediate repair* ($l_{i,j}$ = 2) incurs y% of the component's replacement cost while *Major repair* ($l_{i,j}$ = 3) costs the company z% of it. The total maintenance cost C is computed as the sum of individual maintenance cost of all components. For illustration, the values of x, y and z have been taken as 5, 25 and 50 % respectively in this project. $C_{i,j}$ is the cost of performing maintenance on the component *(i, j)*.

$$C = \sum_{i=1}^{s} \sum_{j=1}^{p} C_{i,j}(l_{i,j}) \qquad (2)$$

3.1.6 Maintenance Time

The maintenance time to perform all the maintenance actions on the system has to be estimated as it plays a crucial role in determining the maintenance strategy. If a component isn't considered for maintenance, the time consumed for maintenance is zero. It is assumed that the maintenance time remains same for any maintenance action i.e., it takes the same amount of time to replace a component or to repair it. Because the time to replace or repair generally tends to vary depending on the size of the system being considered. If the system is a very large-scale industrial system and the repair time of a component may be higher due to the complexity involved in accessing it, while replacing it might be easier and quicker. The individual maintenance time of each component chosen for maintenance is denoted as $T_{i,j}$. The total maintenance time T is calculated as the sum of the individual component's maintenance time.

$$T = \sum_{i=1}^{s}\sum_{j=1}^{p} T_{i,j}(l_{i,j}) \quad (3)$$

3.1.7 Objective Function

Every maintenance event is always coupled with some predefined maintenance budget. It is always expected that the maintenance should be performed by utilizing minimum maintenance resources while ensuring achievement of minimum required probability of successfully performing the mission in terms of target mission reliability. Therefore, the objective function of the considered problem is set to be minimization of the maintenance cost (mathematically given as eqn. 4), while satisfying all the constraints as given below.

$$\boldsymbol{min\ C} = \sum_{i=1}^{s}\sum_{j=1}^{p} C_{i,j}(l_{i,j}) \quad (4)$$

Subject to:

$$\boldsymbol{R(t)} \geq \boldsymbol{R^{*}(t)}$$

$$\boldsymbol{T} \leq \boldsymbol{T_0}$$

$$\boldsymbol{0} \leq \boldsymbol{l_{i,j}} \leq (\boldsymbol{N-1})$$

Where,

R(t): System's Mission reliability

R(t):* System's Target Mission reliability

$$\boldsymbol{R(t)} = \prod_{i=1}^{s}\left(\boldsymbol{1} - \prod_{j=1}^{p}\left(\boldsymbol{1} - \boldsymbol{R_{i,j}}\right)\right)$$

$R_{i,j}$: Mission reliability of component

C_0: Total maintenance budget

T_0: Available maintenance break duration.

T: Time required for performing selected maintenance activity

N: Number of maintenance actions

$l_{i,j}$: Maintenance action for component (i, j)

3.2 Reliability Estimation

Reliability is defined as the probability that a component/system will perform its intended function satisfactorily for a specified time interval 't' under the stated conditions. This relationship is mathematically expressed as a continuous random variable 'T' as the time to failure of the system (component). Thus, reliability is expressed as shown in eq. 5.

$$\boldsymbol{R(t) = Pr\{ T \geq t \}} \qquad (5)$$

3.2.1 The Weibull Distribution

All the systems considered in this paper are assumed to follow Weibull probability distribution. These distributions have hazard rate functions that are not constant over time. Weibull distribution can be used to model increasing, decreasing and constant failure rates and this versatile nature makes it the widely used failure distribution in reliability analysis. There are 2 crucial parameters in the Weibull distribution, the shape parameter β which is also known as the *slope* and the scale parameter η which is also known as the *characteristic life*. The hazard rate function is defined as:

$$\lambda(t) = \frac{\beta}{\eta}\left(\frac{t}{\eta}\right)^{\beta-1} \qquad (6)$$

The reliability function of a component (i, j) whose age is B and is following Weibull distribution is defined as:

$$R_{i,j}(t|B) = \frac{e^{-\left(\frac{t+B}{\eta}\right)^{\beta}}}{e^{-\left(\frac{B}{\eta}\right)^{\beta}}} \qquad (7)$$

3.3 Multi-State Industrial system

The developed algorithms are tested by applying them on the benchmark problem to comprehend the results better. The coal transportation system is most widely used industrial system in the literature [7,14,22] to analyze the efficacy of various approaches to solve the SMO problem. It has 5 subsystems comprising of a couple of conveyors and feeders and a stacker-reclaimer as shown in figure 3.1.

All the subsystems are connected in series while in each subsystem, the individual components are connected in parallel.

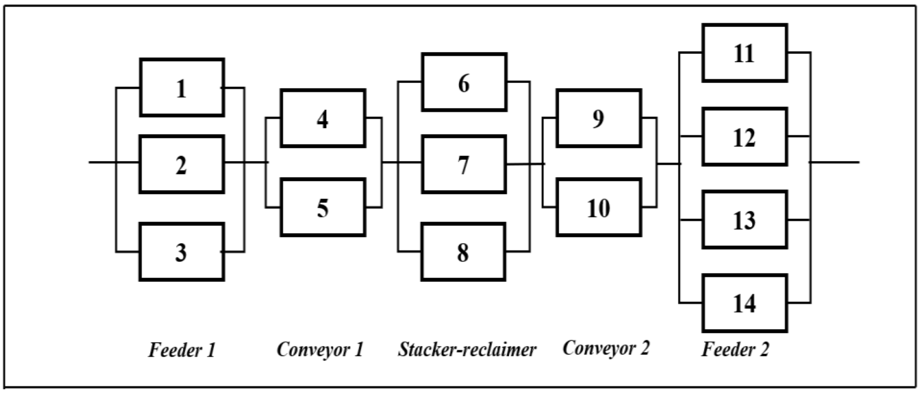


Figure 3.1: Block diagram of coal transportation system

Figure 3.2: Stacker-Reclaimer of a coal transportation system

The benchmark problem considered in this project consists of 14 components and 5 subsystems.

3.3.1 System Reliability Estimation

Subsystem Reliability:

The components in a subsystem are in parallel configuration to each other. Therefore,

$$R_i = 1 - \prod_{j=1}^{p} (1 - R_{i,j}) \qquad (8)$$

Table 2: Subsystem reliability estimation

Subsystem	Reliability of subsystem
Feeder 1	$R_1 = 1 - \prod_{j=1}^{3} (1 - R_{1,j})$
Conveyor 1	$R_2 = 1 - \prod_{j=1}^{2} (1 - R_{2,j})$
Stacker-Reclaimer	$R_3 = 1 - \prod_{j=1}^{3} (1 - R_{3,j})$
Conveyor 2	$R_4 = 1 - \prod_{j=1}^{2} (1 - R_{4,j})$
Feeder 2	$R_5 = 1 - \prod_{j=1}^{4} (1 - R_{5,j})$

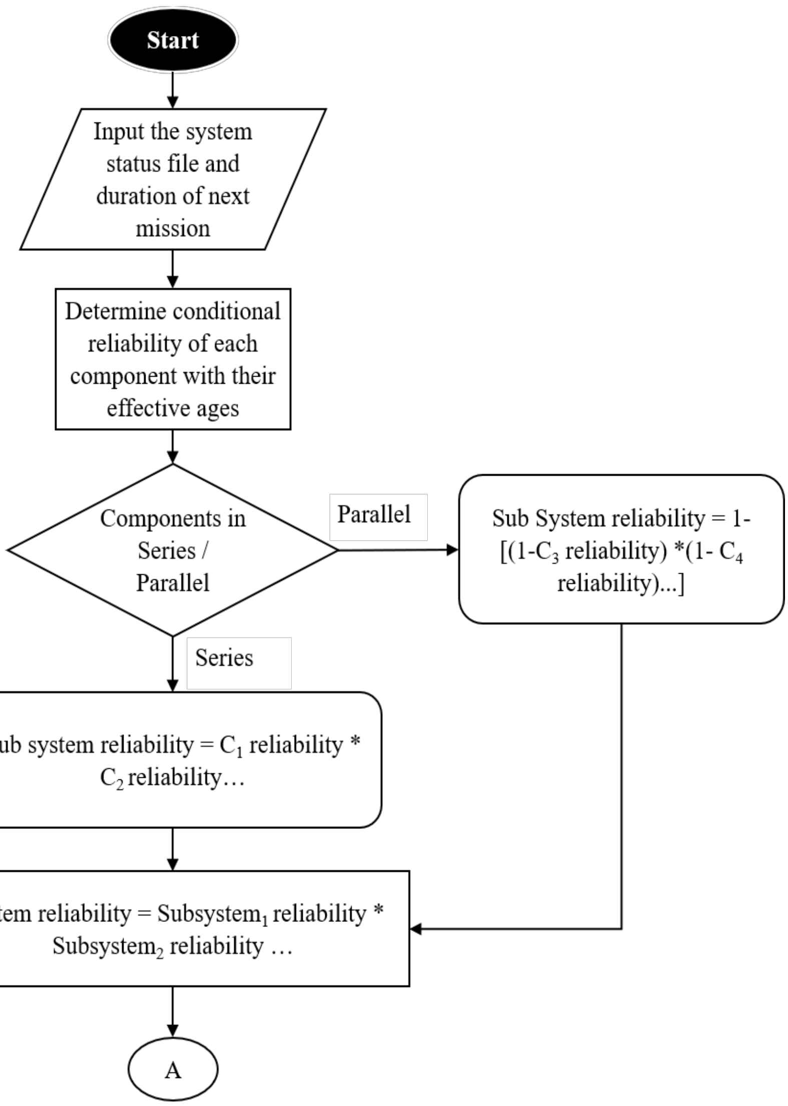
System Reliability:

All the 5 subsystems are in series to each other. So, the overall System reliability is estimated as given in eq. 9.

$$R(t) = \prod_{i=1}^{s} R_i \qquad (9)$$

3.4 Methodology of Enumeration method

Enumeration method or Brute-Force search is one of the conventional and most reliable algorithms used to solve the SM optimization problem. This method will search the complete solution space to identify the best maintenance action to perform. Enumeration method always guarantees an optimal solution, but this approach is inefficient for larger solution spaces.



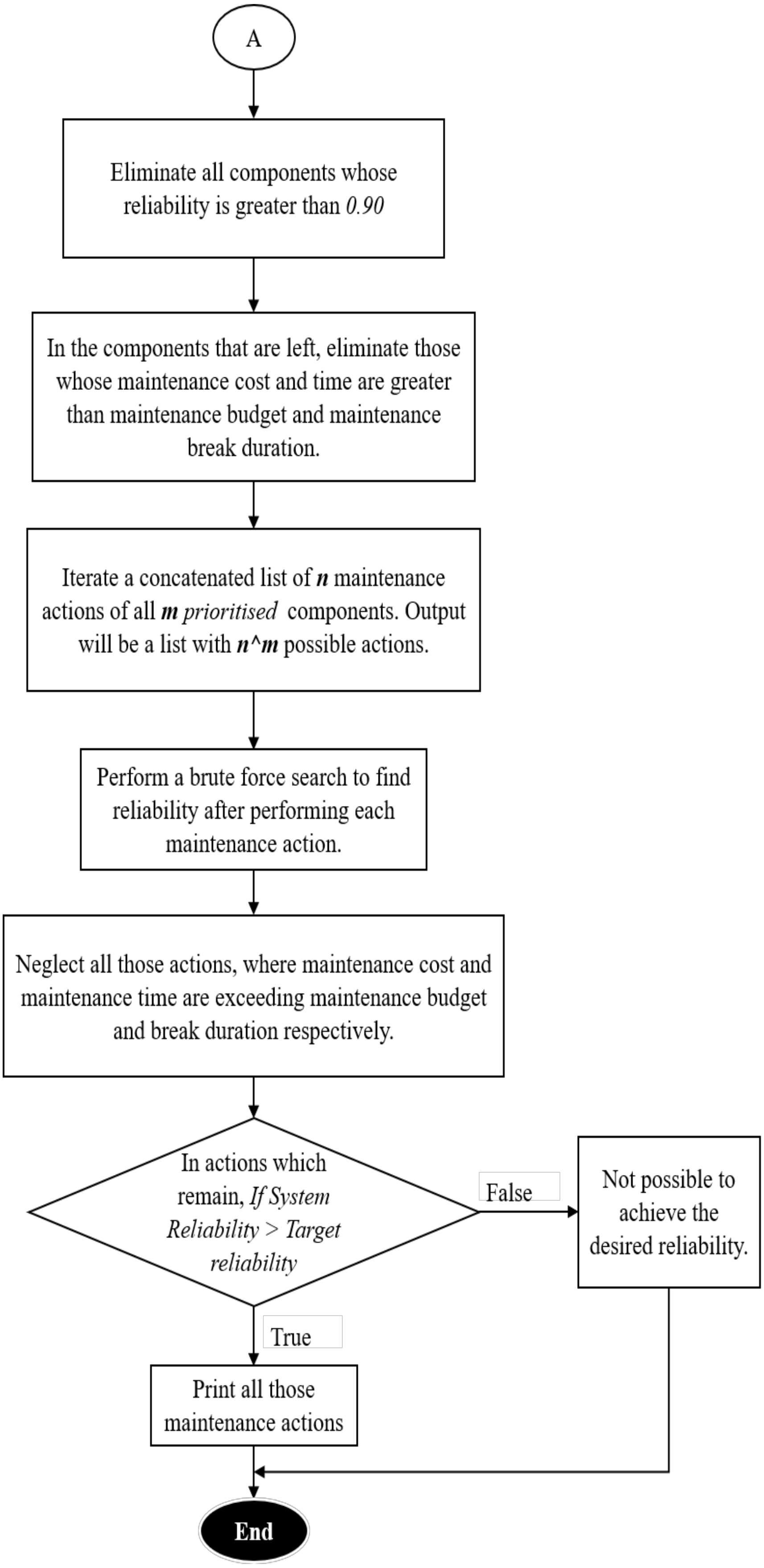


Figure 3.3: Flowchart of Enumeration method

The above figure shows the flowchart of the methodology followed to identify the optimal maintenance action by solving the non-linear programming problem of SMO.

3.5 Limitation of Enumeration method

The number of feasible solutions will increase exponentially with increase in number of components. So, the algorithm is suitable only for models with small solution space. For example, the feasible solution space for a system which contains 6 components and 5 maintenance actions can be performed is 5^6 i.e., 15,625 possible actions. But if any system containing 14 components is considered, the solution space becomes 5^{14} i.e., 6,103,515,625 possible actions. It is computationally not feasible to perform a brute force search over this exorbitantly large solution space.

3.6 Results and Discussion

The developed Enumeration method algorithm is applied for different cases which is based on varying ages of the components to simulate a practical industrial scenario where the system can be subject to maintenance in its earlier days or during a stage after considerable ageing of the components.

3.6.1 Case 1

The various parameters of the components i.e., its shape parameter (β) and scale parameter (η) are given in the table 3.

The duration of the upcoming mission *(t)* for the coal transportation system is continuous operation for the next 100 days i.e., 2400 hrs.

Table 3: Weibull parameters of the Coal transportation system

Subsystem	***Component***	***Scale parameter (hrs)***	***Shape parameter***
Feeder 1	1	7200	1.5
	2	7200	2.4
	3	6000	1.6
Conveyor 1	4	9600	2.6
	5	9600	1.8

Stacker-Reclaimer	6	9000	2.4
	7	9600	2.5
	8	9000	2
Feeder 2	9	9600	1.2
	10	9600	1.4
Conveyor 2	11	10800	2.8
	12	10800	1.5
	13	10200	2.4
	14	9600	2.2

The ages of the components and the maintenance cost and maintenance time of each component is mentioned in table 4. The reliability of each component before maintenance is also calculated and mentioned in the table.

Table 4: Reliability of the Components before maintenance

Component	***Age (hrs)***	***Maintenance cost (Rs)***	***Maintenance time (hrs)***	***Reliability before maintenance*** $(R_{i,j})$
1	2750	225000	6	**0.691500273**
2	2600	300000	6	**0.718896041**
3	2900	225000	6	**0.601991213**
4	2340	375000	7.2	**0.874450254**
5	4500	150000	7.2	**0.743641875**
6	2590	225000	3.6	**0.824905618**
7	2740	450000	7.2	**0.846840889**
8	2100	375000	2.4	**0.822377655**
9	4250	225000	9.6	**0.765265027**
10	3975	450000	4.8	**0.7612768**
11	2250	525000	3.6	**0.921186337**
12	2850	300000	6	**0.815980559**
13	2630	450000	8.4	**0.865347609**
14	2455	225000	8.4	**0.840822585**

Estimating reliability before maintenance of component 1:

$$R_{(1,1)}(\, t = 2400 \,|B = 2750) = \frac{e^{-\left(\frac{t+B}{\eta}\right)^{\beta}}}{e^{-\left(\frac{B}{\eta}\right)^{\beta}}} = \frac{e^{-\left(\frac{2400+2750}{7200}\right)^{1.5}}}{e^{-\left(\frac{2750}{7200}\right)^{1.5}}} = 0.6915$$

- System Reliability before maintenance, $R(2400) = 0.8775$
- Target Mission Reliability, $R^{*}(2400) = 0.90$

Since, mission reliability of the system is less than the target reliability for the next mission, Maintenance needs to be performed to ensure successful completion of the next mission.

Maintenance priority of components

All the components of the coal transportation system aren't selected for maintenance. Because, as we can see from table 4, most of the components are healthy and performing any sort of maintenance action is a futile exercise as it'll unnecessarily waste the useful life of the components. So, components are prioritized by setting a minimum threshold before considering for maintenance.

This is a novel heuristic developed as part of this project to simplify the SMO problem. In depth discussion about this heuristic is provided in Chapter 5.

- Minimum reliability threshold, $R_{(i,j)min}(2400) = 0.80$
- Components prioritized for maintenance in this case: [1,2,3,5,9,10]

These 6 components are prioritized for maintenance as they have less reliability than the defined threshold.

- *N = 5 maintenance actions*
- *M = 6 components*

Solution space = $N^M = 15{,}625\ actions$

Optimal actions are those which help in achieving the desired target reliability along with the maintenance cost being within the maintenance budget and maintenance time within the maintenance break duration.

- Maintenance Budget = Rs. 2,00,000
- Maintenance Break duration = 24 hrs

Optimal maintenance action:

Component	1	2	3	5	9	10
Maintenance action	1	1	0	3	0	0

The maintenance policy is *[1,1,0,3,0,0]* which implies performing *Major repair* on component *5, Minimal repair* on components *1 & 2* and *Do Nothing* on all other components. This maintenance action costs Rs. 1,01,250 and the time required to perform this action is 19.2 hrs. Both of these are well within the constraints and the reliability of the system after maintenance is 0.9017.

The computation time for this problem using enumeration method is ***185 seconds.***

3.6.2 Case 2

The Weibull parameters of the components are same as mentioned in table 3. The ages of the components are slightly more as a more realistic case of older components are modeled in this scenario. The ages of the components and the reliability before maintenance is given in table 5.

Table 5: Ages of components (Case 2)

Component	*Age (hrs)*	*Reliability before maintenance* $(R_{i,j})$
1	3690	0.662978998
2	3810	0.616259864
3	4440	0.540343519
4	3880	0.789086325
5	4660	0.738711236
6	4130	0.734308793
7	4280	0.762485709
8	3720	0.747100507
9	4540	0.763107198
10	4800	0.748602772
11	3790	0.854563541
12	4390	0.787140509
13	4170	0.793672588
14	4510	0.744254548

The duration of the upcoming mission *(t)* for the coal transportation system is 100 days i.e., 2400 hrs.

- System Reliability before maintenance, $R(2400) = 0.8211$
- Target Mission Reliability, $R^{*}(2400) = 0.90$

Since, mission reliability of the system is less than the target reliability for the next mission, Maintenance needs to be performed to ensure successful completion of the next mission.

- Minimum reliability threshold, $R_{(i,j)min}(2400) = 0.75$
- Components prioritized for maintenance in this case: [1,2,3,5,6,8,10,14]

These 8 components are prioritized for maintenance as they have less reliability than the defined threshold.

- *N = 5 maintenance actions*
- *M = 8 components*

Solution space = $N^{M} = 3{,}90{,}625\ actions$

- Maintenance Budget = Rs. 4,00,000
- Maintenance Break duration = 24 hrs

Optimal maintenance action:

Component	**1**	**2**	**3**	**5**	**6**	**8**	**10**	**14**
Maintenance actions	**0**	**3**	**0**	**3**	**2**	**1**	**1**	**0**

The maintenance policy is *[0,3,0,3,2,1,1,0]* which implies performing *Minimal repair* on components *8 and 10*, *Intermediate repair* on component 6, *Major repair* on components *2 and 5* and *Do Nothing* on all other components. This maintenance action costs Rs. 3,22,500 and the time required to perform this action is 24 hrs. Both of these are well within the constraints and the reliability of the system after maintenance is 0.90012.

*The computation time for this problem using enumeration method is **72 minutes.***

As we can clearly see, the increase in solution space has caused a steep increase in the computation time of the algorithm from around 185 seconds in the previous case to more than one hour in this scenario.

3.6.3 Case 3

A hypothetical scenario with all 14 components prioritized for maintenance is simulated to check the potency of the Enumeration approach. Table 6 specifies the ages of the components and the reliability of each component before maintenance.

Table 6: Ages of components (Case 3)

Component	***Age (hrs)***	***Reliability before maintenance*** $(R_{i,j})$
1	7000	0.586773954
2	7000	0.382248979
3	7000	0.462427422
4	7000	0.602398158
5	7000	0.672698145
6	7000	0.569542624
7	7000	0.609744072
8	7000	0.615127371
9	7000	0.747872421
10	7000	0.720124979
11	7000	0.683208001
12	7000	0.748120373
13	7000	0.659113606
14	7000	0.634065707

- System Reliability before maintenance, $R(2400) = 0.6459$
- Target Mission Reliability, $R^*(2400) = 0.90$

Since, mission reliability of the system is less than the target reliability for the next mission, Maintenance needs to be performed to ensure successful completion of the next mission.

- Minimum reliability threshold, $R_{(i,j)min}(2400) = 0.75$
- Components prioritized for maintenance in this case: [1,2,3,4,5,6,7,8,9,10,11,12,13,14]

Maintenance needs to be performed on all the 14 components.

- *N = 5 maintenance actions*
- *M = 14 components*

Solution space = $N^M = 6{,}103{,}515{,}625\ actions$

The solution space is very large compared to earlier cases. The enumeration method becomes computationally intractable in this case. This is the major limitation of this algorithm as it becomes completely inefficient and takes huge time to compute for problems with very large solution space.

So, to overcome these computational inefficiencies, second generation of research on SMO primarily focused on using various heuristics and meta heuristics to solve the nonlinear programming problem of SM optimization.

Chapter 4 Evolutionary Algorithm approach

4.1 Introduction

An evolutionary algorithm (EA) is a generic population based metaheuristic optimization algorithm. An EA is based on natural evolution mechanisms like reproduction, crossover, mutation and selection. Possible solutions to the optimization problem, i.e., in this project, the candidate solutions are the maintenance actions; are the individuals in a population. Then fitness function determines the quality of the maintenance action, in other words this function evaluates the feasibility of the maintenance action and ensures *survival of the fittest.* Only the fittest maintenance actions are carried forward to continue the process of evolution after continuous application of the above evolution mechanisms.

4.1.1 Genetic Algorithm (GA)

Genetic algorithm is a metaheuristic which draws inspiration from biological process of natural selection and it is the most commonly used EA for solving optimization problems. GA depends on biological mechanisms like mutation, crossover and selection to identify the best solution.

The evolution process starts from a randomly generated set of chromosomes which constitute a population and the population in each iteration is called a *Generation.* It is an iterative process and in each generation after evaluating the fitness of every individual (fitness function is generally the objective function of the optimization problem), fitter chromosomes are stochastically selected from the present population. These chosen individuals are subjected to various genetic operators like crossover and mutation to form a new generation of individuals. The newer generation of population is then used during the next iteration of the GA and the algorithm is terminated when it reaches a set termination criterion like achieving the maximum number of generations or obtaining satisfactory fitness levels.

4.2 Methodology

4.2.1 Hyperparameters

The parameters which influence the evolution of the algorithm and in turn the final result have to be selected meticulously to ensure achieving optimal solution in a computationally efficient manner. These hyperparameters are listed below.

- Population Size
- Number of Generations
- Crossover Probability
- Mutation Probability

4.2.2 Fitness Function

The fitness function is the most crucial element in the GA methodology. Fitness function for the SMO problem of a coal transportation system is defined in such a way that if the individual chromosome (maintenance action) achieves the objective function and also satisfies the constraints, it is given a value of fitness based on its objective function i.e., cost. If it fails to fulfill the desired criteria, a very high value of fitness function is given. The GA is programmed in such a way that the evolution considers low fitness values as better individuals. Because, the objective function is to minimize the maintenance cost.

The flowchart of the fitness function is given in the figure 4.1.

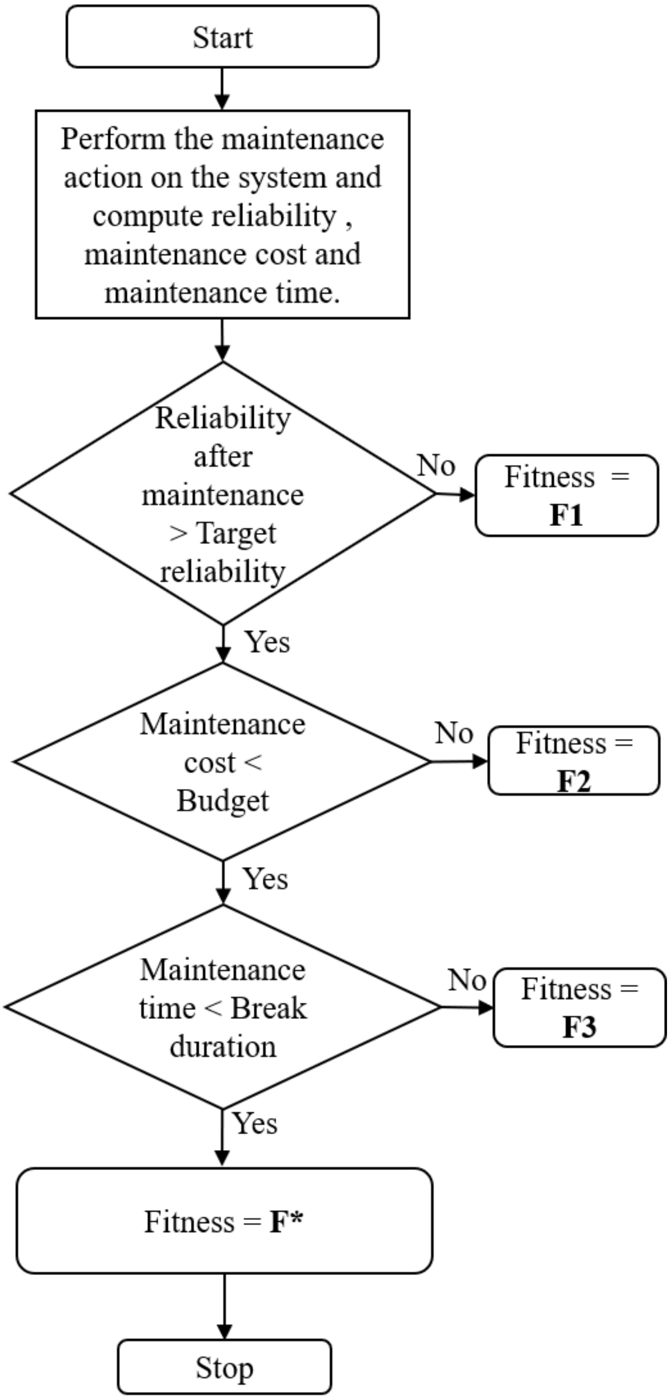


Figure 4.1: Fitness function of GA

4.2.3 Crossover Operator

The crossover genetic operator is analogous to biological crossover. Here more than one individual is selected, where each individual is considered as a parent. Off springs (More newer maintenance actions) are produced using the genetic material of the parents. Individuals which fulfill the fitness function criteria are selected for crossover.

A random crossover point is chosen and the tail end of each parent is swapped to obtain newer individuals. Crossover is generally applied with higher probability.

A schematic representation of this operation is shown in figure 4.2.

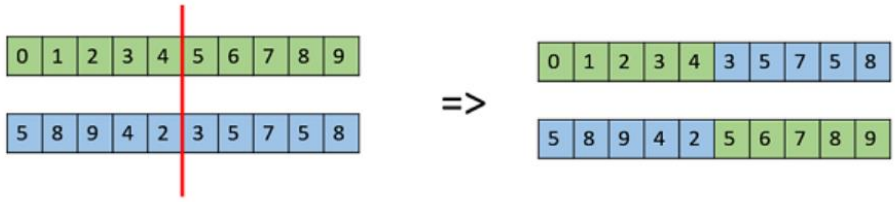


Figure 4.2: Crossover

The newer individuals are considered in the next generation's population.

4.2.4 Mutation

Mutation is a minute random tweak in the individual chromosome to obtain a new individual. Mutation is used to maintain and diversity in the genetic population. It is generally applied with a low probability. Mutation is the most crucial element in GA as it ensures *exploration* of the search space and is essential for the convergence of the algorithm.

The mutation operation is as shown in the figure 4.3.

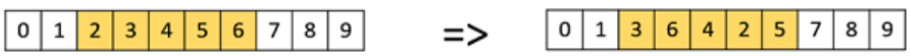


Figure 4.3: Mutation

4.2.5 Selection

Parent selection is the process of choosing maintenance actions which recombine to create off springs (newer maintenance actions) for the next generation. This is one of the most crucial steps as it is imperative for the convergence of GA and drives the individuals towards better and fitter solutions. It also ensures to maintain good diversity in the population.

Selection for this SM optimization problem is done based on Tournament selection. In this type of selection random individuals are chosen from the population at random and best out of these is chosen based on the minimum fitness values of competing individuals. This method is suitable for this case because we are working on minimizing fitness values. A generic schematic of tournament selection is depicted in the figure 4.4.

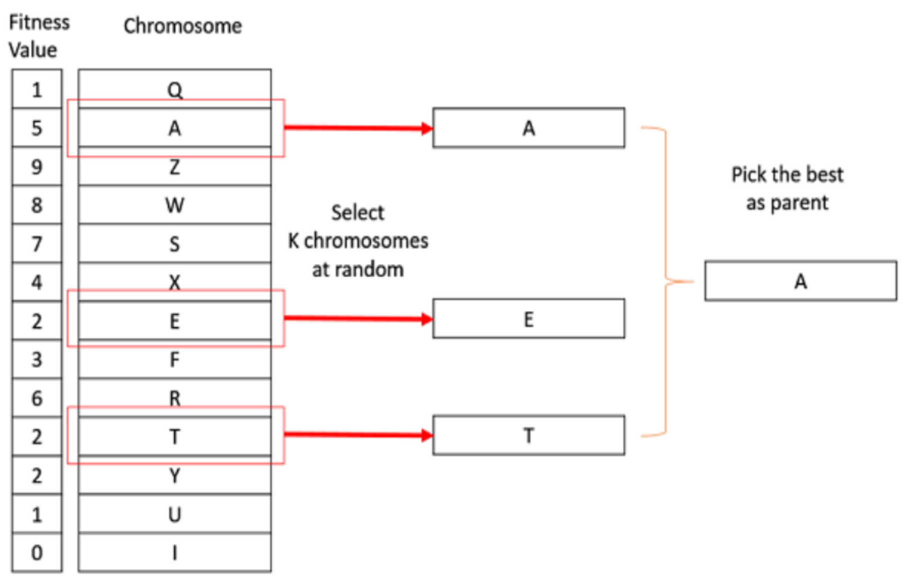


Figure 4.4: General Schematic of Selection

4.3 Results and discussion

The developed GA is applied on different cases based on varying ages of the components which are discussed in the previous chapter. The algorithm is applied on the same examples to compare the improvement in the solution obtained on using this metaheuristic and to get a fair comparison of the computation time on applying GA instead of Enumeration method.

4.3.1 Case 1

The various parameters of the components i.e., its shape parameter (β) and scale parameter (η) are given in table 3.

The duration of the upcoming mission *(t)* for the coal transportation system is continuous operation for the next 100 days i.e., 2400 hrs. The ages of the components and the reliability of each component is given in table 7.

Table 7: Ages and Reliability of the components (GA: Case 1)

Component	***Age (hrs)***	***Reliability before maintenance ($R_{i,j}$)***
1	2750	**0.691500273**
2	2600	**0.718896041**
3	2900	**0.601991213**
4	2340	**0.874450254**
5	4500	**0.743641875**
6	2590	**0.824905618**
7	2740	**0.846840889**
8	2100	**0.822377655**
9	4250	**0.765265027**
10	3975	**0.7612768**
11	2250	**0.921186337**
12	2850	**0.815980559**
13	2630	**0.865347609**
14	2455	**0.840822585**

Hyperparameters:

- Population Size = 2500
- Number of generations = 6
- Crossover probability = 0.90
- Mutation probability = 0.20

Optimal Maintenance action

Component	**1**	**2**	**3**	**5**	**9**	**10**
Maintenance action	**1**	**1**	**0**	**3**	**0**	**0**

The maintenance policy is *[1,1,0,3,0,0]* which implies performing *Major repair* on component *5, Minimal repair* on components *1 & 2* and *Do Nothing* on all other components. This maintenance action costs Rs. 1,01,250 and the time required to perform this action is 19.2 hrs. Both of these are well within the constraints and the reliability of the system after maintenance is 0.9017.

The computation time using GA is ***123 seconds.***

4.3.2 Case 2

In this case, which has a greater number of components prioritized for maintenance, the GA is applied by tuning the hyperparameters to accommodate the increased solution space.

Table 8: Ages of components (GA: Case 2)

Component	***Age (hrs)***	***Reliability before maintenance*** ($R_{i,j}$)
1	3690	0.662978998
2	3810	0.616259864
3	4440	0.540343519
4	3880	0.789086325
5	4660	0.738711236
6	4130	0.734308793
7	4280	0.762485709
8	3720	0.747100507
9	4540	0.763107198
10	4800	0.748602772
11	3790	0.854563541
12	4390	0.787140509
13	4170	0.793672588
14	4510	0.744254548

Hyperparameters:

- Population Size = 25000
- Number of generations = 20
- Crossover probability = 0.90
- Mutation probability = 0.20

Optimal maintenance action:

Component	**1**	**2**	**3**	**5**	**6**	**8**	**10**	**14**
Maintenance actions	**0**	**3**	**0**	**3**	**2**	**1**	**1**	**0**

The maintenance policy is *[0,3,0,3,2,1,1,0]* which implies performing *Minimal repair* on components *8 and 10*, *Intermediate repair* on component 6, *Major repair* on components *2 and 5* and *Do Nothing* on all other components. This maintenance action costs Rs. 3,22,500 and the time required to perform this action is 24 hrs. The system reliability after maintenance is 0.90012.

The computation time for solving the above case of SM optimization is ***44 minutes.***

4.3.3 Comparison with enumeration method

The GA is applied on two different scenarios and the results obtained are compared with those obtained by using Enumeration method. Because to overcome the drawbacks of enumeration method of computational inefficiency during brute force search, metaheuristics like GA was used.

- *Case 1*

Table 9: Comparison of GA with Enumeration method

	Enumeration method	***Genetic algorithm***
Components	[1,2,3,5,9,10]	[1,2,3,5,9,10]
Solution	[1,1,0,3,0,0]	[1,1,0,3,0,0]
Computation time	185 seconds	123 seconds

In this case, the solution space contains 15,625 possible actions. GA achieves the optimal solution with the selected hyperparameters in lesser time than enumeration method.

- *Case 2*

Table 10: Comparison of GA with Enumeration method

	Enumeration method	***Genetic algorithm***
Components	[1,2,3,5,6,8,10,14]	[1,2,3,5,6,8,10,14]
Solution	[0,3,0,3,2,1,1,0]	[0,3,0,3,2,1,1,0]
Computation time	**72 minutes**	**44 minutes**

In this case the solution space is increased to 3,90,625. The hyperparameters are also changed accordingly. We can see that to solve this SMO problem, enumeration method takes more than an hour but GA is computationally efficient and takes only 44 minutes to complete it.

4.3.4 Limitations of Genetic Algorithm

Though GA is beneficial over enumeration method, the optimality of the solution isn't always guaranteed. But enumeration method always results in an optimal solution, whereas sub optimal solutions are also possible with GA. And implementation of GA is tedious task which requires technically skilled people.

Chapter 5 Machine Learning based Approach

5.1 Introduction to Machine Learning (ML)

5.1.1 What is Artificial Intelligence (AI)?

Machines which exhibit intelligent behavior by perceiving its environment and taking decisions accordingly are said to possess artificial intelligence. By using AI, a machine can mimic cognitive human functions like learning and problem solving.

An AI system is created with Machine Learning (ML) and Deep Learning (DL) algorithms. We can infer from the schematic diagram shown below that ML and DL are subsets of AI.

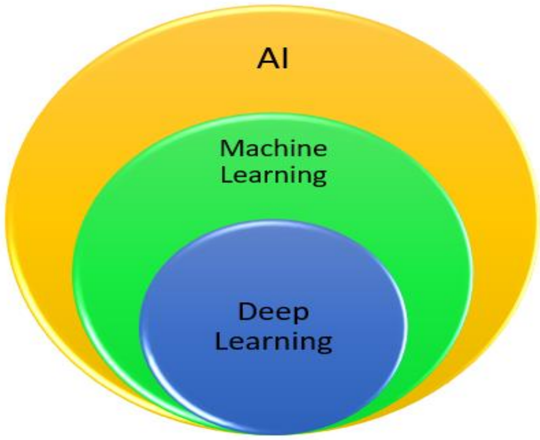


Figure 5.1: ML as subset of AI

5.1.2 Machine Learning (ML)

ML is an application of AI that provides systems the ability to automatically learn and improve from experience without being explicitly programmed. ML is a process where machines take data, analyze it to generate predictions and use those predictions to make decisions. The decisions generate results which are used to improve future predictions.

ML is a method of data analysis, that automates analytical model building. The process of feeding data to a software program and coming up with human like decisions is known as modeling.

5.1.3 Types of ML methods

Machine learning is comprised of three different paradigms. They are Supervised learning, Unsupervised learning and Reinforcement learning.

- *Supervised Learning*

Supervised learning algorithms apply already learned data to present data with labeled outputs to envisage future outcomes. A known training dataset is analyzed and the algorithm learns and a function is created to make anticipation about the future events. The learning model will compare its output with the exact output and modifies the model accordingly to make correct predictions.

The figure shown below clearly illustrates the learning process in supervised learning.

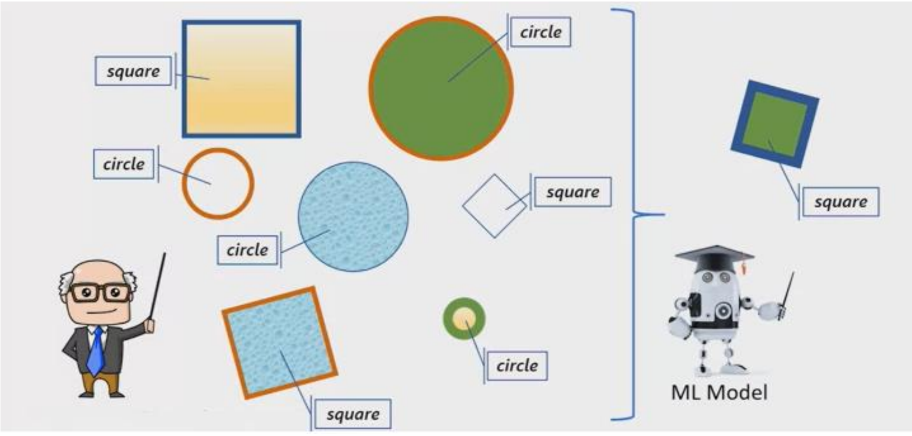


Figure 5.2: Illustration about Supervised Learning

- *Unsupervised Learning*

Unsupervised learning detects emerging properties in input data. These learning algorithms are useful when the information is not labeled. It is predominantly used to find out the hidden structure in unclassified data. Model constructs patterns or clusters as outputs because the system just explores the data to draw inferences.

- *Reinforcement Learning*

Reinforcement learning algorithms is a learning method that interacts with its environment by producing actions and discovers errors or rewards. This method allows machines and software agents to automatically determine the ideal behavior within a specific context in order to maximize its performance. Reinforcement learning is learning what to do and how to map situations to actions so as to maximize a numerical reward signal. The agent is not told which actions to take, but instead must discover which actions yield the most reward by trying them. The algorithm discovers through trial and error, which actions yield the best rewards [29].

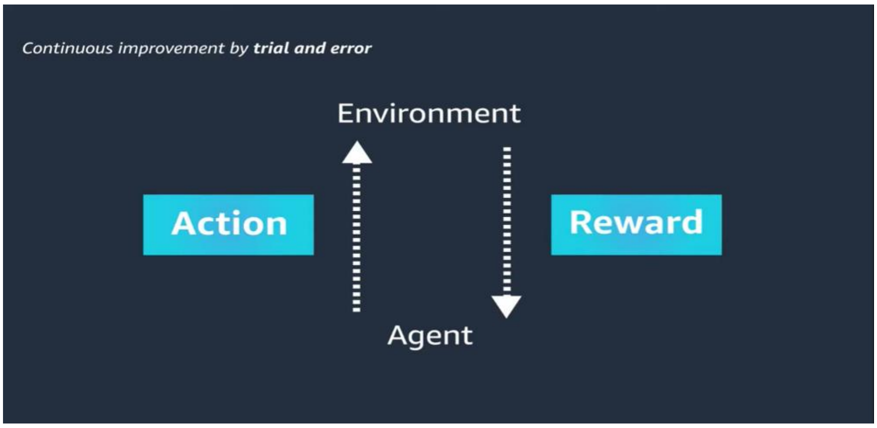


Figure 5.3: Learning in RL

"*Of all forms of Machine learning, Reinforcement learning is closest to the kind of learning that humans and animals do.*"

It is distinguished from other computational approaches by its emphasis on learning by an agent from direct interaction with its environment, without requiring exemplary supervision or complete models of the environment.

5.2 Reinforcement Learning (RL) for Selective Maintenance Optimization

Reinforcement learning for SMO requires the agent to determine an optimal policy. RL is distinguished from supervised and unsupervised learning as the former requires learning from a training set and the latter is typically about finding the hidden structure in unlabeled data, whereas RL is learning what to do, how to map states to actions to maximize a numerical reward signal. This learning methodology of RL makes it the most suitable ML paradigm for applying it to solve the computationally complex problem of SMO.

5.2.1 Steps involved in RL process:

[1] Observe the environment

[2] Perform an action based on the strategy

[3] Accept a reward or penalty

[4] Learn from the experience and modify the strategy

[5] Iterate until an optimal solution is found

5.2.2 Elements of RL

Any RL algorithm has four major components. They are:

i. *The Agent* – Learner or Decision maker

ii. *The Environment* – Industrial system for maintenance

iii. *Actions* – Maintenance actions performed on the system

iv. *Reward* – Feedback to the maintenance performed

Figure 5.4: Elements of RL

- *Return:* The Return function is the sum of the rewards in the simplest case. It is denoted as *G(t)*.

$$G(t) \doteq R_{t+1} + \gamma R_{t+2} + \gamma^2 R_{t+3} + \cdots = \sum_{k=0}^{\infty} \gamma^k R_{t+k+1} \qquad (10)$$

Where,

G(t): Return

R_t: Reward

γ: Discount rate

- *Action-Value function:* The action-value function is the expected return when starting from state *s* taking action *a* and following policy *π*.

$$\begin{aligned} q_\pi(s,a) &\doteq E_\pi[G_t |\, S_t = s,\ A_t = a] \\ &= E_\pi\left[\sum_{k=0}^{\infty} \gamma^k R_{t+k+1} |\, S_t = s\,, A_t = a\right] \end{aligned} \qquad (11)$$

Where,

$q_\pi(s, a)$: Action-value function for policy π

S_t: Current State

$E_\pi[.]$: Expected value given the agent follows policy π

R_t: Reward

5.3 Modeling of the SMO problem

Taking into account the mechanical nature of the industrial systems / components under consideration, it is assumed that the component lifetimes follow a 2-parameter Weibull distribution. This distribution has hazard rate function that is not constant over time, for shape parameter values ≠ 1. For preventive replacement, only those components qualify whose shape parameter > 1. The same is considered for the problem described in this project.

Markov Decision Processes (MDPs) are classical formalization of sequential decision making, where actions influence not just immediate rewards, but also subsequent situations and states which in turn affects future rewards.

Since the components in the Multi-State System (MSS) is assumed to follow the Weibull distribution, the SM problem is modelled as an SMDP [30]. The Action Space of the SMDP is a set of all possible maintenance actions, but only those actions which are feasible maintenance actions yield a positive reward. Feasible actions are those whose maintenance cost is within the maintenance budget and the maintenance time is less than the break duration as shown in eq. 12.

$$S^{act}_{feasible} = \{a|\Sigma C \leq Maintenance\ budget\ , \Sigma T \leq Maintenance\ break\ duration\} \tag{12}$$

Reward of the SMDP is based on the probability of the system successfully completing the next mission. The optimal selective maintenance strategy *(Π*)* is determined by:

$$\pi^* = argmax\ Q^*(a) \tag{13}$$

5.4 Q – Learning for SMO

Q - Learning is an off-policy temporal difference learning algorithm that learns from the reward obtained and chooses the best action to take at a given state. Q in Q-Learning stands for Quality. A higher Q-Value denotes a higher reward if that action is chosen in that state. The Q-Value is estimated using the Bellman equation.

$$Q(S_t, A_t) \leftarrow Q(S_t, A_t) + \alpha[R_{t+1} + \gamma \max Q(S_{t+1}, a) - Q(S_t, A_t)] \tag{14}$$

Where,

α: *Learning rate:* Extent to which Q – Values are updated in every epoch. α determines the weightage given to recent observations.

γ: *Discount rate:* This parameter helps determine the present value of a reward obtained in the future.

In this algorithm, the agent will learn from experience and choose the best action from the set of feasible maintenance actions. The Q – Values are updated in the Q - table which is maintained by the agent and the table will guide it to the best action for each component. The optimal Selective Maintenance strategy is determined by the set of maintenance actions that give the maximum Q – value for each component.

5.5 Exploration – Exploitation dilemma

The agent has two possibilities to choose from whenever it requires to make a decision. Whether to explore for better actions or exploit those actions which is already giving the agent a reward. There's a trade-off between choosing a random action to explore and choosing actions based on already learned Q-Values i.e., choosing greedy actions. If the agent sticks to an action which gives it good reliability of system at a feasible maintenance cost, there is also a possibility that on exploration the agent may find a much better action by consuming much lower maintenance resources.

A parameter **'ε'** known as exploration probability is defined, which denotes the probability that the agent will explore in a particular iteration. Reward maybe lower in the short run during exploration, but it can be higher in the long run because the agent may discover a better action and can exploit it.

5.6 Maintenance priority of components

At the end of a mission, when the system is ready for maintenance, if the number of components in the system is M and N maintenance actions are possible on the system, there are N^M possible maintenance action combinations. However, there are some components whose age is comparatively lesser resulting in their higher mission reliability. The decision makers are skeptical about replacing such components and considering these components in the solution space unnecessarily burdens the computation process. It is also observed that, even considering such components with higher reliabilities, serves no purpose as they do not get assigned with any maintenance action in the final solution. With this heuristic, the reliability of each component for the duration of the next mission gets estimated and the solution space gets smartly reduced by omitting the components with the reliability higher than the predefined reliability threshold. Omitting these components for maintenance serves two purposes:

i. Avoids maintenance on components that are already healthy and ensures the useful life of components isn't wasted.
ii. Reduces the solution space, thereby giving the agent a much narrow set of possibilities to choose from.

5.7 Reward Function

The reward function is a very important parameter in letting the agent learn and improve from experience. The agent is rewarded or penalized based on the chosen maintenance action satisfying the objective of the problem and the constraints. The flowchart of the reward function is given in Figure 5.5.

To make sure the agent will choose the maintenance action by consuming minimum maintenance resources, the reward given to the agent is further divided based on the system reliability after maintenance. As the difference between the reliability of the system after maintenance and target reliability increases, the cost and time required to perform maintenance increases and hence the agent will consume more resources. So, to help the agent choose optimal actions and refrain from choosing sub optimal solutions, harsh measures in the form of reducing rewards were adopted.

In this project, the penalties for the agent after performing a particular maintenance action is divided based on the constraints being satisfied.

If the agent fails to meet the reliability constraint, it is penalized heavily by a penalty P1. If the objective function of cost isn't met, then agent is penalized P2 and finally on failing to meet the maintenance time criteria, a penalty of P3 is imposed. For calculative purposes, empirical numerical values of 10000, 7000 and 4000 is chosen as P1, P2 and P3 respectively in this project.

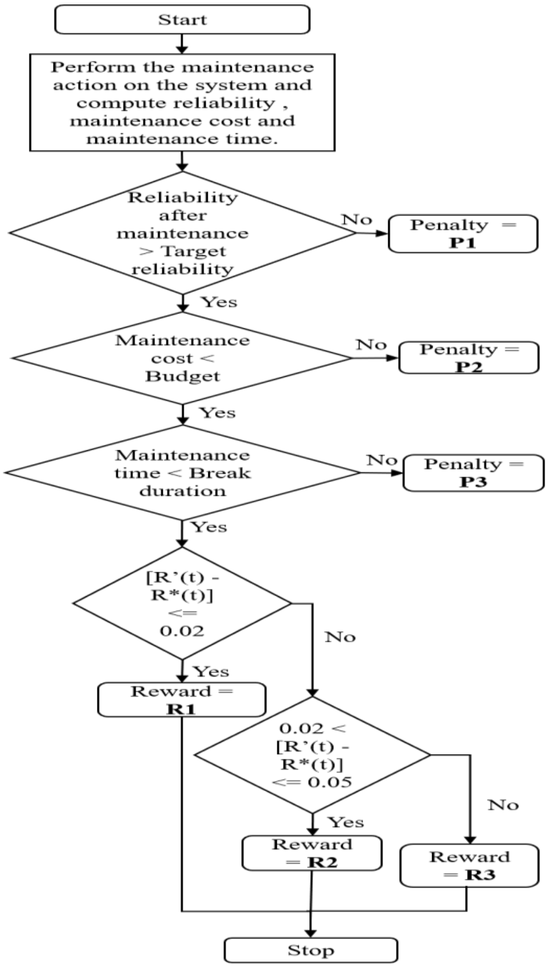


Figure 5.5: Flowchart of Reward function

5.8 Reward Factors

A novel heuristic is devised in the form of a reward factor which is additionally multiplied to the reward received by the agent, before using it in the Bellman equation (eqn. 14) to update the Q-Table. The use of this heuristic forces the agent to choose the action which consumes minimum maintenance resources. A higher reward factor is given if the agent chooses an action which doesn't involve replacing the component. The values of the reward factors are given in Table 11.

Table 11: Reward Factors

Maintenance action ($l_{i,j}$)	Action	Reward factor
0	Do nothing	10
1	Minimal repair	7.5
2	Intermediate repair	5
3	Major repair	2.5
4	Component replacement	1

5.9 Steps in Q-Learning

- *The initialization of Q – table:* The Q – table is initialized for M components where N actions are possible. The values are initialized at 0.

- *Choose an action to perform:* An action to perform on a component will be chosen depending on the agent's decision to explore or exploit. An immediate reward is given to the agent for choosing an action. After the agent chooses an action for each component, the set of maintenance actions chosen for each component is performed on the system and the reliability of the system, maintenance cost and maintenance time is determined.

- *Calculate the reward and update the Q – Table:* Based on the reliability of the system after maintenance and the cost and time incurred for maintenance, an accumulated reward is estimated and based on this reward, the Bellman optimality equation is used to determine the Q – Value and update the Q – Table at every epoch.

The flowchart of the complete Q-Learning process for an epoch is depicted in Figure 5.6.

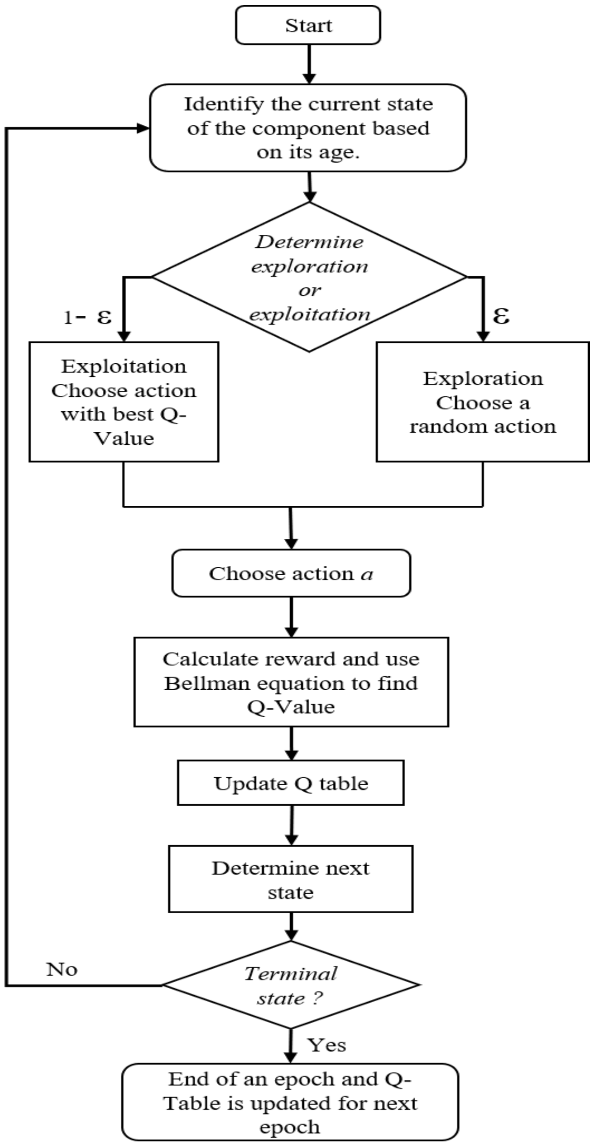


Figure 5.6: Flowchart of Q-Learning process

5.10 Results and Discussion

The proposed RL based methodology is illustrated on two different problems. Firstly, the methodology is applied on a small-scale problem and later, it is illustrated on the benchmark industrial system for coal transportation.

5.10.1 Illustrative Example I – Small scale system

To discuss the application of methodology in detail, firstly, it is applied on a small-scale system consisting of 2 subsystems connected in series configuration. Both sub systems are configured with two components connected in parallel. The block diagram for the considered small-scale system is shown in Figure 5.7.

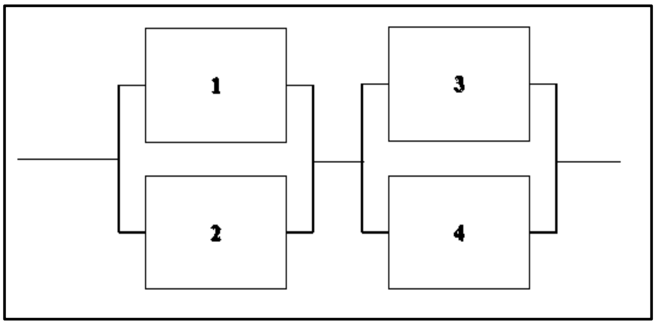


Figure 5.7: Series-Parallel system

The status of the system along with details regarding probability distribution parameters, maintenance cost, maintenance duration, etc. are provided in Table 12.

Table 12: Parameters of the small-scale system

Component	η (hrs)	β	Age (hrs)	Maintenance cost	Maintenance time (hrs)
1	3500	1.5	1000	15000	7
2	2600	2.4	850	12500	6
3	2800	2.4	900	20000	3
4	3800	1.8	1000	10000	5

The next mission for the system is of 1000 hrs. of operation. Required probability of successfully completing the mission in terms of target mission reliability is 0.90. Based on the status of the system, the mission reliability before maintenance is 0.872, which is less than the target mission reliability, therefore, the maintenance needs to be performed. Total available maintenance budget is limited to Rs. 10000 and duration of maintenance break is limited to 12 hrs.

This selective maintenance optimization problem is solved using the proposed RL methodology. There are 4 components (M = 4) considered for maintenance and with 5 maintenance actions (N = 5) possible on each component, the solution space is $5^4 = 625$ possible actions. So, the RL agent has to choose the optimal maintenance action which maximizes the system reliability while satisfying maintenance constraints. The hyperparameters for the algorithm are selected as following – Table 13.

Table 13: Hyperparameters

Learning rate (α)	0.1
Discount rate (γ)	0.95
ε (greedy policy)	0.25

The Q-Learning is initiated with all the values set to 0. Each row in the Q-table has 5 elements which corresponds to each action for a component. The present value of 0 shows the initial Q-Value for every action over every component.

```
[[ 0   0   0   0   0]
 [ 0   0   0   0   0]
 [ 0   0   0   0   0]
 [ 0   0   0   0   0]]
```

After initialization, the agent chooses an action and updates the Q-Values according to the Bellman optimality equation (eqn.14).

After 1000 iterations, the RL agent has now learnt the best policy to be chosen. The optimal strategy to be performed for each component is determined by the highest Q-Value for each component. The updated Q-Values after the 1000 iterations is given below. We can see in the Q-table given below that for component 1, the highest Q-Value is when action 0 is performed. Similarly, the argument for maximum Q-Value for each of the components 2, 3 and 4 is 2, 0, and 1 respectively. So, the optimal maintenance policy chosen by the RL agent for this system is [0,2,0,1] which denotes performing *Intermediate repair* on component 2, *Minimal repair* on component 4 and *do nothing* on all the other components.

```
[[127376.759291  60286.31974115  50861.55050376  47805.36016135  48639.39413355]
 [ 53024.191904   38575.67700382  88174.92520108  27872.24432152  34554.85849217]
 [ 74446.000637   9877.09255053  14527.68093753  18850.56215037  14838.4208591]
 [ 2325.3370561   34416.26387366  -3451.30225156  -2507.87586735  -3683.50695618] ]
```

The maintenance cost incurred is 3625 units and the time required to perform this maintenance action is 11 hrs. The reliability after maintenance is 0.901. As can be seen, the agent consumes minimum resources and also succeeds in fulfilling the objective of the selective maintenance optimization problem.

5.10.2 Illustrative example II - Coal transportation system:

5.10.2.1 Case 1

To further evaluate the proposed methodology, it is applied on the coal transportation system introduced by [7,14,22]. It has 5 subsystems comprising of a couple of conveyors and feeders and a stacker-reclaimer as shown in figure 3.1. Various parameters of the system including its current status (randomly selected), maintenance cost, maintenance time are listed in table 14.

Table 14: Parameters of Coal transportation system

Component	***Age (hrs)***	***Maintenance cost (Rs)***	***Maintenance time (hrs)***	***Reliability before maintenance*** **($R_{i,j}$)**
1	2750	225000	6	**0.691500273**
2	2600	300000	6	**0.718896041**
3	2900	225000	6	**0.601991213**
4	2340	375000	7.2	**0.874450254**
5	4500	150000	7.2	**0.743641875**
6	2590	225000	3.6	**0.824905618**
7	2740	450000	7.2	**0.846840889**
8	2100	375000	2.4	**0.822377655**
9	4250	225000	9.6	**0.765265027**
10	3975	450000	4.8	**0.7612768**
11	2250	525000	3.6	**0.921186337**
12	2850	300000	6	**0.815980559**
13	2630	450000	8.4	**0.865347609**
14	2455	225000	8.4	**0.840822585**

The next mission for the system is of 100 days of continuous operation. Required probability of successfully completing the mission in terms of target mission reliability is 0.90. Based on the status of the system, the present mission reliability is 0.8775 which is less than the target mission reliability, therefore, the maintenance needs to be performed. Total available maintenance budget is limited to Rs. 200000 and duration of maintenance break is limited to 24 hrs.

As per the reward factors given in Table 11, and reliability threshold ($R_{(i,j)\ min}$) of 0.80 is set for a component to be prioritized for maintenance as per the devised heuristic, the hyperparameters for the algorithm are selected as used in the case of illustrative example I.

The Q-Learning is initiated with all the values set to 0.

```
[[0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]
 [0. 0. 0. 0. 0.]]
```

After choosing an action randomly in the first epoch, the Q - table is updated. The action chosen initially is *[1,1,1,0,1,0]*. The updated Q – Table is given below. Based on the defined reward function, the agent is either penalized or rewarded.

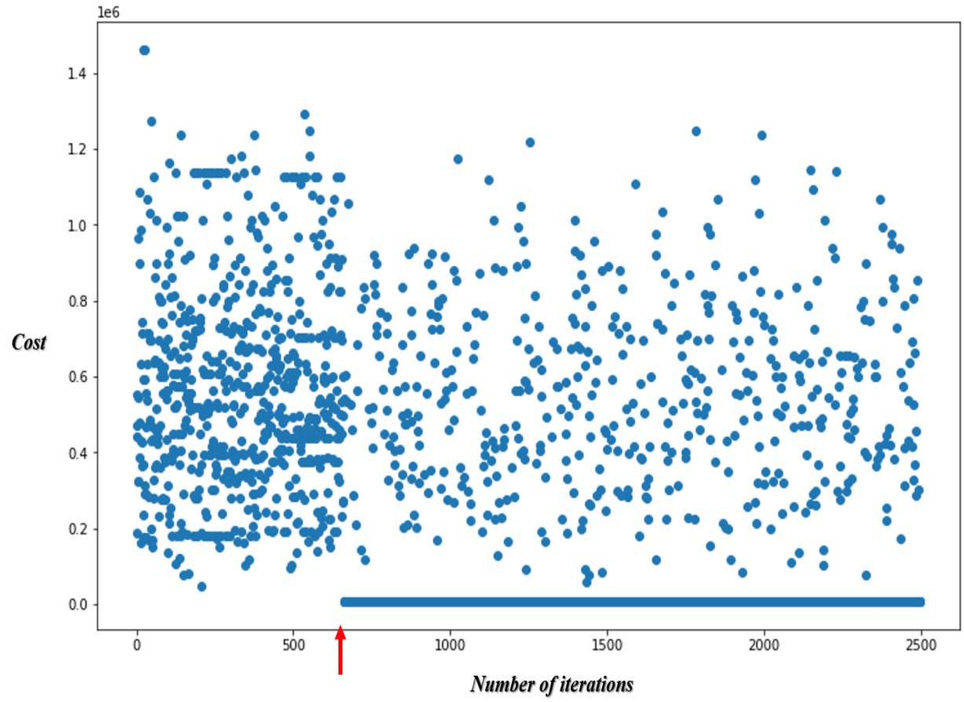
```
[[   0.      -699.9005    0.         0.         0.     ]
 [   0.      -699.9005    0.         0.         0.     ]
 [   0.      -699.9005    0.         0.         0.     ]
 [-699.9005     0.        0.         0.         0.     ]
 [   0.      -699.9005    0.         0.         0.     ]
 [-699.91       0.        0.         0.         0.     ]]
```

After continuously learning up to 2500 epochs, the agent arrives at the optimal solution and the optimal maintenance policy is determined by the maximum Q-Value at each state which corresponds to the action which gives the best return in long run. From the final Q-Table shown below, the maintenance policy is [1,1,0,3,0,0] which implies performing *Major repair* on component *5, Minimal repair* on components *1 & 2* and *Do Nothing* on all other components.

```
[[ 1.45408434e+07  1.76774934e+07  1.49388402e+07  1.43498335e+07  1.45288856e+07]
 [ 1.22460601e+07  1.54788268e+07  1.24124031e+07  1.22489665e+07  1.20687034e+07]
 [ 1.33047972e+07  8.54998709e+06  8.45163600e+06  8.40121311e+06  8.46796036e+06]
 [ 8.23673524e+06  8.00913679e+06  8.21954400e+06  9.16673326e+06  7.99708855e+06]
 [ 8.84607922e+06  4.06858163e+06  4.22241569e+06  4.16702764e+06  3.75386898e+06]
 [ 4.74438096e+06 -3.89680518e+03 -3.75236723e+03 -3.68373684e+03 -3.68373684e+03]]
```

This maintenance action costs Rs. 101250 and the time required to perform this action is 19.2 hrs. Both of these are well within the constraints and the reliability of the system after maintenance is 0.9017.

The computation time using RL for this case is ***64 seconds.***



Graph 1: Learning process of the agent

The graph 1 shows the learning process of the agent. Initially until around 650 iterations, the agent tends to explore to identify the optimal solution. But once the agent on finding an optimal solution, tends to exploit it to get maximum reward and at the same time also explore in search of any other better solutions. This clearly depicts the intelligence of the agent and we can infer that the agent is learning with experience.

5.10.2.2 Case 2

This scenario considers higher ages of the components of the MSS of Coal transportation system. The ages of the components and the values of other parameters are listed in the table below.

Table 15: Ages of the components

Component	***Age (hrs)***	***Reliability before maintenance*** $(R_{i,j})$
1	3690	0.662978998
2	3810	0.616259864
3	4440	0.540343519
4	3880	0.789086325
5	4660	0.738711236
6	4130	0.734308793
7	4280	0.762485709
8	3720	0.747100507
9	4540	0.763107198
10	4800	0.748602772
11	3790	0.854563541
12	4390	0.787140509
13	4170	0.793672588
14	4510	0.744254548

The duration of next mission is 2400 hrs and Reliability of the system before maintenance, R (2400) is 0.821 while R*(2400) is 0.90. So, there is a need for the system to undergo maintenance before the start of next mission to successfully complete it. The selective maintenance optimization problem is formulated with the objective to maximize the system reliability while consuming minimum maintenance resources. The reliability threshold $R_{(i,j)min}$ is set at 0.75 and the components prioritized for maintenance based on the reliability of the components before maintenance are 1,2,3,5,6,8,10 and 14. The number of actions available to be performed on each component is 5 and since there are 8 components for maintenance, the action space for the agent to choose from is 5^8 = 3,90,625. This action space is considerably greater than the one discussed previously.

Hyperparameter α is set at 0.1, while γ is 0.95 and the parameter of ε-greedy policy is 0.25. The constraints of maintenance budget considered during this break is Rs. 4,00,000 while the maintenance break duration is 48 hrs. The Q-table is initialized as a MxN matrix where M is 8 and N is 5.

The agent starts learning by exploring initially. After 1,00,000 iterations, the agent arrives at a feasible maintenance action which after performing on the system, achieves the desired reliability and also the resources like maintenance cost and time consumed is also well within the constraints. The feasible maintenance action proposed by the agent in this case is to perform *Major repair* on components 2,5 and 6 while perform *Minimal repair* on components 3,8 and 10 and *do nothing* on all other components.

Component	1	2	3	5	6	8	10	14
Maintenance actions	0	3	1	3	3	1	1	0

Rs. 3,90,000 has to be spent to perform the chosen maintenance action and it takes 30 hrs. The reliability of the system after maintenance $R'(2400)$ is 0.90511. The computation time for solving this SM optimization problem is ***38 minutes***.

5.10.3 Comparison with Enumeration method and GA

- *Case 1*

For the Scenario 1, the maintenance action suggested by all the three approaches is same i.e., to perform *Major repair* on component *5, Minimal repair* on components *1,2* and *Do Nothing* on all other components as shown in the *table 16.*

Table 16: Comparison with GA and BF (Case 1)

	Enumeration Method (EM)	Genetic Algorithm (GA)	Reinforcement Learning (RL)
Components	[1,2,3,5,9,10]	[1,2,3,5,9,10]	[1,2,3,5,9,10]
Solution	[1,1,0,3,0,0]	[1,1,0,3,0,0]	[1,1,0,3,0,0]
Computation time	**185 seconds**	**123 seconds**	**64 seconds**

Though all the approaches are suggesting the optimal action, there is stark contrast in computation time required for each of the approaches to arrive at the optimal solution. The solution space in this case is $5^6 =$ 15,625 actions. Enumeration method takes 185 seconds to compute this problem, while GA takes only 123 seconds to identify the solution. The proposed RL based approach takes only 64 seconds to compute the selective maintenance optimization problem and return the optimal solution.

There is a 34 % reduction in computation time on using GA when compared to enumeration approach, and RL based methodology takes 48% lesser computation time when compared to GA based approach, and 65% lesser computation time when compared to enumeration approach.

- *Case 2*

In this scenario, the solution space is $5^8 =$ 3,90,625 possible maintenance actions. The maintenance action obtained from Enumeration method is the optimal policy which suggests performing *Minimal repair* on components 8 and 10, while *Intermediate repair* on component 6, *Major repair* on components 2,5 and *do nothing* on all other components. The maintenance cost for performing this maintenance action is Rs. 3,22,500 while the maintenance time required is 24 hrs.

The result obtained from GA is also the optimal maintenance action as suggested by Enumeration method. The RL methodology gives a near optimal solution which fulfils the objectives and abides by the constraints. Though the solution is near optimal, RL based methodology is computationally superior than the traditional method and the evolutionary algorithm-based method. As shown in the table below, computation time for Q-Learning agent to learn is 38 minutes while the Enumeration method takes 72 minutes to compute the selective maintenance optimization problem and GA too consumes 44 minutes.

Table 17: Comparison with GA and BF (Case 2)

	Enumeration method	**Genetic algorithm**	**Reinforcement Learning**
Components	[1,2,3,5,6,8,10,14]	[1,2,3,5,6,8,10,14]	[1,2,3,5,6,8,10,14]
Solution	[0,3,0,3,2,1,1,0]	[0,3,0,3,2,1,1,0]	[0,3,1,3,3,1,1,0]
Computation time	**72 minutes**	**44 minutes**	**38 minutes**

5.10.4 Limitations of RL based methodology

The major drawback of traditional approach of using enumeration method is that the computation time becomes intractable when the solution space is large. These problems can be overcome by using the evolutionary algorithm based approach and the proposed novel RL based methodology.

The proposed RL based Q-Learning algorithm is far more efficient than the GA approach. Though sometimes due to the stochastic nature of learning of the agent, it may not arrive at the optimal solution, but the agent always learns and returns a feasible solution well within the budget and time constraints.

It has been observed that, selection of number of epochs for the proposed RL based methodology for solving SMO, has significant effect on the quality of the solution as well as the computation time required. Although the proposed methodology does not guarantee the optimality of the solution all the time, it is guaranteed that the methodology will result in near optimal solutions when comparatively lesser number of epochs are chosen in an attempt to solve the problem in less computation time. In order to reach up to the optimality, selection of number of epochs need to be on higher side at the expense of slightly higher computation time. Even in such case with near optimal solution, the RL based methodology resulted into better performance in terms of computation time when compared to traditional approaches. Therefore, for the industrial scenarios where the attention is more on the computation time while compromising on optimality of the solution, this RL based methodology will prove to be more effective.

The implementation of RL based methodology also requires sophisticated computation infrastructure and technically skilled personnel to implement the algorithm properly. This adds to the financial burden of industries. With the advancement in technology and rapid modernization of industries, application of ML based algorithms for maintenance planning in industries isn’t a distant reality anymore.

Chapter 6 Agent Based Distributed Approach

6.1 Introduction

The combination of sensors and computing infrastructure is becoming increasingly popular in the industries. Developments like these are making way for the automation of various industrial practices and are motivating the need to replace conventional maintenance planning techniques with methods that can employ the capabilities of Cyber-physical systems and IIoT. Maintenance planning is a crucial decision-making task which significantly reduces unplanned downtime and improves system efficiency.

Till now, we have seen the evolution of methodologies used for SM optimization from the introduction of the problem. Initially, it was traditional methods like brute-force search and in the next generation, metaheuristics like GA were used. Now, the rapid rise in ML's popularity opened up new avenues for using machine learning based methods for SM optimization. In the last chapter, implementation of reinforcement learning based methodology was discussed.

In this chapter, a neoteric agent-based distributed maintenance planning algorithm for SMO is discussed in detail. This is a first of its kind attempt to use this decentralized approach for SMO. This algorithm will is developed to be compatible with the present trend of IIoT and smart manufacturing and pave way for Industry 4.0. Rapid rise of Industry 4.0 in present day manufacturing processes is resulting in a move from centralized decision making to distributed decision making and supervisory control across industries.

6.2 Machine-Level Agents and Coordinating Agent

- *Machine-Level agents:*

Distributed systems are characterized as being more robust and agile, which are some of the highly relevant qualities in today's competitive world. Decentralized approaches generally have different agents at system level to represent each system or subsystem. They choose what is best for the respective subsystem with respect to the maintenance which is to be performed. They just ensure that the maintenance policy they are preferring for their respective subsystem will not violate the constraint of maintenance budget and time.

The reliability required for each subsystem to ensure the complete system achieves the desired target mission reliability is allocated to each subsystem and the task of the agent at subsystem level is to achieve the allocated subsystem reliability. The agent at subsystem level (*machine level agent*) will give its preferred choices of maintenance for its respective subsystem. The preferences can vary by requirement. For example, in a particular case, all the machine level agents can give the top 5 preferences based on the least maintenance cost. This quantity of preferences can be 7,10 or more depending on the requirement.

- *Coordinating agent*

The coordinating agent will take in all the preferences given by each machine level agent and now its task is to decide which maintenance policy to adopt based on the choices, so that the target mission reliability of the entire system is achieved. This agent will think over the broader perspective of the enterprise as a whole. So, the maintenance preferences obtained from the agents will be used to decide the optimal maintenance policy whose maintenance cost will be less than the maintenance budget and time to perform the maintenance is within the break duration. The coordinating agent uses any computation algorithm to arrive at the optimal solution considering all the constraints. In this project, enumeration method is used for optimizing the SM problem. This decentralized planning ensures the solution space is reduced significantly to an extent that only potential maintenance actions are present. It filters out all unnecessary actions which helps in reducing the computational complexity.

The diagrammatic representation of the distributed maintenance planning algorithm for the benchmark problem of the coal transportation system considered in this work is depicted in figure 6.1.

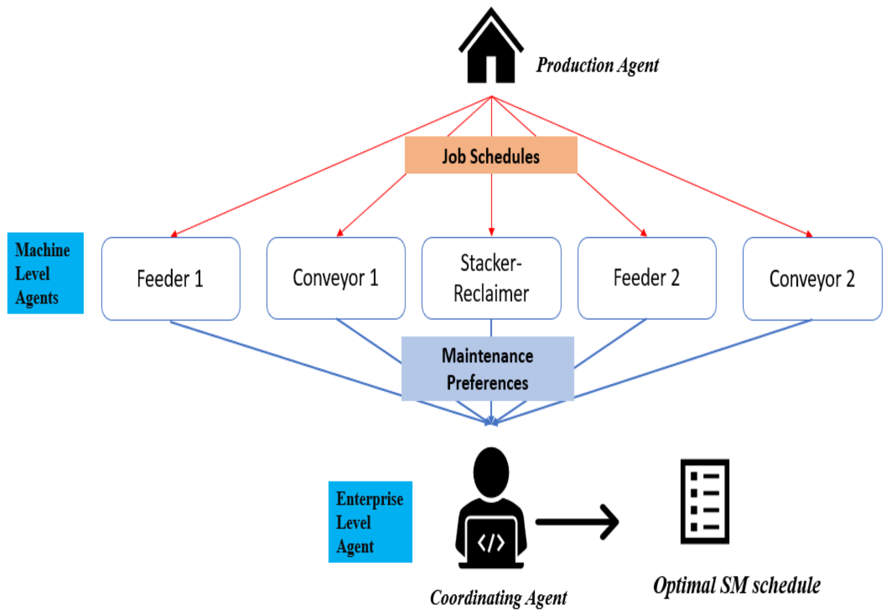


Figure 6.1: Agent based Distributed Maintenance planning

The coal transportation system has 5 different subsystems connected in series. Each subsystem has an agent which will determine what is best for the respective subsystem. So, for the problem considered in this project, there are *5 machine level agents* and *1 coordinating agent.*

6.3 Reliability Allocation

Ages of the components are known and the reliabilities are estimated. Then based on the target reliability, reliability allocation takes place, where each subsystem is allocated a target component reliability to achieve after maintenance. This ensures the target system reliability is reached after maintenance. The reliability allocation to components and subsystems will support the enterprise goals of achieving the target reliability.

There are various methods to allocate reliability. In this project, I have used two different reliability allocation techniques to identify the best approach suitable for this situation. They are:

- *Equal reliability allocation*
- *Minimum effort method*

6.3.1 Equal Reliability Allocation

In this method, reliability is equally allocated to all the components/subsystems. If an industrial system contains *M subsystems,* the reliability is equally allocated to each subsystem irrespective of the number of components present in the subsystem.

$R^*(t)$ – Target system reliability

M = 5 [Number of subsystems in the coal transportation system

$$R_i(t) = [R^*(t)]^{1/5} \qquad (15)$$

6.3.2 Minimum effort method

In this method, the reliabilities of subsystem are arranged in increasing order. This method is called minimum effort method, because now the target for the least reliability subsystem is increased up to the next highest reliability subsystem. So, the aim is to spend minimum effort to achieve the target system reliability ($R^*(t)$). After increasing the subsystem reliability by a level, then the present system reliability is estimated. If it is greater than the target reliability, then the reliability of the subsystem has to be increased to a level between the current reliability and the next subsequent reliability. If it is less than the target reliability, then the reliability of the first two subsystems in increasing order is increased up to the reliability of the third subsystem in line. This process is repeated until the target system reliability is achieved.

6.4 Results and Discussion

The developed distributed maintenance planning algorithm is applied on the benchmark problem of coal transportation system and the efficiency of the algorithm is compared with the various methodologies developed in this project like the enumeration method, GA and RL based Q-Learning algorithm.

The system parameters and ages of the components along with the reliabilities of the components are mentioned in tables 18 and 19. The duration of the upcoming mission *(t)* for the coal transportation system is continuous operation for the next 100 days i.e., 2400 hrs.

Table 18: Weibull parameters of the components

Subsystem	***Component***	***Scale parameter (hrs)***	***Shape parameter***
Feeder 1	1	7200	1.5
	2	7200	2.4
	3	6000	1.6
Conveyor 1	4	9600	2.6
	5	9600	1.8
Stacker-Reclaimer	6	9000	2.4
	7	9600	2.5
	8	9000	2
Feeder 2	9	9600	1.2
	10	9600	1.4
Conveyor 2	11	10800	2.8
	12	10800	1.5
	13	10200	2.4
	14	9600	2.2

Table 19: Reliabilities of the components

Component	Age (hrs)	Maintenance cost (Rs)	Maintenance time (hrs)	Reliability before maintenance ($R_{i,j}$)
1	2750	225000	6	**0.691500273**
2	2600	300000	6	**0.718896041**
3	2900	225000	6	**0.601991213**
4	2340	375000	7.2	**0.874450254**
5	4500	150000	7.2	**0.743641875**
6	2590	225000	3.6	**0.824905618**
7	2740	450000	7.2	**0.846840889**
8	2100	375000	2.4	**0.822377655**
9	4250	225000	9.6	**0.765265027**
10	3975	450000	4.8	**0.7612768**
11	2250	525000	3.6	**0.921186337**
12	2850	300000	6	**0.815980559**
13	2630	450000	8.4	**0.865347609**
14	2455	225000	8.4	**0.840822585**

6.4.1 Estimating Subsystem Reliability

[1] Reliability of Feeder 1:

Feeder 1 consists of 3 components which are connected in parallel configuration.

$$R_1 = 1 - \prod_{j=1}^{3}(1 - R_{1,j})$$

$$R_1 = 0.965484481$$

[2] Reliability of Conveyor 1:

Conveyor 1 has 2 parallelly connected components.

$$R_2 = 1 - \prod_{j=1}^{2}(1 - R_{2,j})$$

$$R_2 = 0.967814303$$

[3] Reliability of Stacker-Reclaimer:

Stacker-reclaimer consists of 3 parallelly connected components.

$$\boldsymbol{R_3 = 1 - \prod_{j=1}^{3}(1 - R_{3,j})}$$

$$\boldsymbol{R_3 = 0.995236648}$$

[4] Reliability of Feeder 2:

Feeder 2 has 2 parallelly connected components.

$$\boldsymbol{R_4 = 1 - \prod_{j=1}^{2}(1 - R_{4,j})}$$

$$\boldsymbol{R_4 = 0.943963316}$$

[5] Reliability of Conveyor 2:

Conveyor 2 has 4 components which are parallel to each other.

$$\boldsymbol{R_5 = 1 - \prod_{j=1}^{4}(1 - R_{5,j})}$$

$$\boldsymbol{R_5 = 0.999689143}$$

6.4.2 Equal Reliability Allocation

- System Reliability before maintenance, $R(2400) = 0.8775$
- Target Mission Reliability, $R^*(2400) = 0.90$

Since, the system reliability before maintenance is less than the target mission reliability, maintenance has to be performed on the system. The reliability is allocated for each subsystem equally.

$$R_i(t) = [0.90]^{\frac{1}{5}} = 0.9791$$

Based on reliabilities allocated for each subsystem, the machine-level agent identifies the best possible maintenance actions for the respective subsystem and the top 5 preferences are forwarded to the coordinating agent to make the final decision regarding the maintenance action to be performed.

The maintenance budget is Rs. 2,00,000 and the break duration is 24 hrs. The chosen policy has to abide by these constraints and at the same time achieve the target reliability.

The top preferences given by each machine-level agent for its respective subsystem is listed below. Every preference set will have the action of *Doing nothing* on all components as a preference to ensure the optimality of the maintenance policy by not overdoing maintenance on redundant components.

I. Preferences of Agent 1 for Feeder 1:

	Actions	Cost	Time	Reliability
0	(0, 0, 0)	0.0	0	0.965484
1	(0, 2, 0)	75000.0	6	0.979170
2	(0, 2, 1)	86250.0	12	0.980242
3	(1, 2, 0)	86250.0	12	0.980121
4	(1, 2, 1)	97500.0	18	0.981144

II. Preferences of Agent 2 for Conveyor 1:

	Actions	Cost	Time	Reliability
0	(0, 0)	0.0	0.0	0.967814
1	(1, 2)	56250.0	14.4	0.980664
2	(0, 3)	75000.0	7.2	0.987118
3	(1, 3)	93750.0	14.4	0.988945
4	(2, 0)	93750.0	7.2	0.982146

III. Preferences of Agent 3 for Stacker-reclaimer:

	Actions	Cost	Time	Reliability
0	(0, 0, 0)	0.0	0.0	0.995237
1	(1, 0, 0)	11250.0	3.6	0.995844
2	(0, 0, 1)	18750.0	2.4	0.995652
3	(0, 1, 0)	22500.0	7.2	0.995901
4	(1, 0, 1)	30000.0	6.0	0.996207

IV. Preferences of Agent 4 for Feeder 2:

There is only option here because the maintenance actions which can improve the reliability of this subsystem up to the allocated reliability are more expensive to perform as they cost more than the maintenance budget. So, these actions are omitted and only *doing nothing* is preferred as the only feasible option.

	Actions	Cost	Time	Reliability
0	(0, 0)	0	0	0.943963

V. Preferences of Agent 5 for Conveyor 2:

	Actions	Cost	Time	Reliability
0	(0, 0, 0, 0)	0.0	0.0	0.999689
1	(0, 0, 0, 1)	11250.0	8.4	0.999723
2	(0, 1, 0, 0)	15000.0	6.0	0.999705
3	(0, 0, 1, 0)	22500.0	8.4	0.999730
4	(0, 1, 0, 1)	26250.0	14.4	0.999737

Now that all the agents have given their preferences to the coordinating agent at the enterprise level, it is now the task of that agent to perform a brute force search over all the preferences and identify the best maintenance policy which can fulfill the criterions of the constraints and achieve the objective function.

After performing the brute force search over the solution space, the top 5 maintenance actions with least maintenance cost are listed below.

	Actions	Costs	Time	Reliability after maintenance
0	[0, 2, 0, 1, 2, 0, 0, 0, 0, 0, 0, 0, 0]	131250.0	20.4	0.901831
1	[0, 2, 0, 1, 2, 1, 0, 0, 0, 0, 0, 0, 0]	142500.0	24.0	0.902381
2	[0, 2, 0, 1, 2, 0, 0, 1, 0, 0, 0, 0, 0]	150000.0	22.8	0.902207
3	[0, 2, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0]	150000.0	13.2	0.907765
4	[0, 2, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 1]	161250.0	21.6	0.907796

The maintenance action highlighted above suggests performing *Intermediate repair* on components 2 and 5, *Minimal repair* on component 4 and *do nothing* on all other components. This maintenance action costs Rs. 1,31,250 and the maintenance time is 20.4 hrs. The reliability of the system after maintenance is 0.901831.

As we can see, equal reliability allocation is a very naive way of allocating reliability. This allocation technique tends to overfit the model and it doesn't ensure the optimality of the results. In this case, we have obtained a sub optimal solution which costs Rs. 30000 more than the optimal solution obtained from enumeration method and GA.

To overcome this issue, Reliability allocation using Minimum effort method is performed and the results are explored.

6.4.3 Minimum Effort Method

- System Reliability before maintenance, $R(2400) = 0.8775$
- Target Mission Reliability, $R^*(2400) = 0.90$

Since, the system reliability before maintenance is less than the target mission reliability, maintenance has to be performed on the system. The reliability is allocated to each subsystem based on the minimum effort method.

Step 1:

Arranging the reliabilities in increasing order.

Table 20: Subsystem Reliability

Subsystem	Reliability
4	0.943963
1	0.965484
2	0.967814
3	0.995237
5	0.999689

→ Reliability of the system [$R(2400)$] < $R*$

∴ $R_4 = R_1 = 0.965484$

After first allocation, $R(2400) = 0.89824$

System reliability is still less than target reliability.

Step 2:

$R_4 = R_1 = R_2 = 0.967814$

$R(2400) = 0.901918$

System reliability > Target reliability

So, the R_4 and R_1 should be between the previous reliability value and R_2.

$$\therefore R_4 = R_1 = \left[\left(\frac{R^*}{(R_2 \times R_3 \times R_5)}\right)^{\left(\frac{1}{2}\right)}\right] = 0.966784$$

$$R(2400) = 0.966784 \times 0.966784 \times 0.967814 \times 0.995237 \times 0.999689$$

$$= 0.90$$

The allocation process is complete as $R(2400) = R*$.

The final reliability allocation is as mentioned in table 21.

Table 21: Reliability allocation by Minimum effort method

Subsystem	Reliability
4	0.966784
1	0.966784
2	0.967814
3	0.995237
5	0.999689

The top 10 preferences given by each machine-level agent for its respective subsystem is listed below.

I. *Preferences of Agent 1 for Feeder 1:*

	Actions	Cost	Time	Reliability
0	(0, 0, 0)	0.0	0	0.965484
1	(1, 0, 0)	11250.0	6	0.967060
2	(0, 0, 1)	11250.0	6	0.967260
3	(0, 1, 0)	15000.0	6	0.969620
4	(1, 0, 1)	22500.0	12	0.968755
5	(1, 1, 0)	26250.0	12	0.971007
6	(0, 1, 1)	26250.0	12	0.971183
7	(1, 1, 1)	37500.0	18	0.972499
8	(2, 0, 0)	56250.0	6	0.971319
9	(0, 0, 2)	56250.0	6	0.972075

II. *Preferences of Agent 2 for Conveyor 1:*

	Actions	Cost	Time	Reliability
0	(0, 0)	0.0	0.0	0.967814
1	(0, 1)	7500.0	7.2	0.970514
2	(1, 0)	18750.0	7.2	0.972380
3	(1, 1)	26250.0	14.4	0.974696
4	(0, 2)	37500.0	7.2	0.977468
5	(1, 2)	56250.0	14.4	0.980664
6	(0, 3)	75000.0	7.2	0.987118
7	(1, 3)	93750.0	14.4	0.988945
8	(2, 0)	93750.0	7.2	0.982146
9	(2, 1)	101250.0	14.4	0.983644

III. *Preferences of Agent 3 for Stacker-reclaimer:*

	Actions	Cost	Time	Reliability
0	(0, 0, 0)	0.0	0.0	0.995237
1	(1, 0, 0)	11250.0	3.6	0.995844
2	(0, 0, 1)	18750.0	2.4	0.995652
3	(0, 1, 0)	22500.0	7.2	0.995901
4	(1, 0, 1)	30000.0	6.0	0.996207
5	(1, 1, 0)	33750.0	10.8	0.996424
6	(0, 1, 1)	41250.0	9.6	0.996259
7	(1, 1, 1)	52500.0	13.2	0.996736
8	(2, 0, 0)	56250.0	3.6	0.997202
9	(2, 0, 1)	75000.0	6.0	0.997446

IV. *Preferences of Agent 4 for Feeder 2:*

There is only option here because the maintenance actions which can improve the reliability of this subsystem up to the allocated reliability are more expensive to perform as they cost more than the maintenance budget. So, these actions are omitted and only *doing nothing* is preferred as the only feasible option.

	Actions	Cost	Time	Reliability
0	(0, 0)	0	0	0.943963

V. *Preferences of Agent 5 for Conveyor 2:*

	Actions	Cost	Time	Reliability
0	(0, 0, 0, 0)	0.0	0.0	0.999689
1	(0, 0, 0, 1)	11250.0	8.4	0.999723
2	(0, 1, 0, 0)	15000.0	6.0	0.999705
3	(0, 0, 1, 0)	22500.0	8.4	0.999730
4	(0, 1, 0, 1)	26250.0	14.4	0.999737
5	(1, 0, 0, 0)	26250.0	3.6	0.999738
6	(0, 0, 1, 1)	33750.0	16.8	0.999760
7	(1, 0, 0, 1)	37500.0	12.0	0.999767
8	(0, 1, 1, 0)	37500.0	14.4	0.999743
9	(1, 1, 0, 0)	41250.0	9.6	0.999751

The maintenance policy is performing *Major repair* on component *5, Minimal repair* on components *1 & 2* and *Do Nothing* on all other components. This maintenance action costs Rs. 1,01,250 and the time required to perform this action is 19.2 hrs. Both of these are well within the constraints and the reliability of the system after maintenance is 0.90019.

Actions	Costs	Time	Reliability after maintenance
[1, 1, 0, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0]	101250.0	19.2	0.900198
[0, 1, 1, 0, 3, 0, 0, 0, 0, 0, 0, 0, 0, 0]	101250.0	19.2	0.900361
[1, 1, 0, 0, 3, 1, 0, 0, 0, 0, 0, 0, 0, 0]	112500.0	22.8	0.900747
[0, 1, 1, 0, 3, 1, 0, 0, 0, 0, 0, 0, 0, 0]	112500.0	22.8	0.900911
[1, 1, 0, 0, 3, 0, 0, 1, 0, 0, 0, 0, 0, 0]	120000.0	21.6	0.900574

The optimal policy is obtained after using the minimum effort method and the coordinating agent obtains this solution on prioritizing top 7 preferences given by the machine-level agents. The computation time required is 88 seconds for solving this SM optimization problem.

6.5 Comparison of the proposed methodologies

To establish the supremacy of the developed Agent based distributed approach and RL based methodology over the commonly used methods, the selective maintenance optimization problem is solved using Brute-Force search and an Evolutionary algorithm based methodology using Genetic Algorithm. The results obtained from these approaches are compared with those obtained from proposed Distributed maintenance planning algorithm and RL methodology and the computation time required for these methods is also compared to identify the most efficient approach.

In the Enumeration method, the complete solution space of N^M actions is explored and the maintenance action which achieves the desired target reliability by consuming minimum maintenance resources is chosen. Enumeration method always returns the optimal solution but is very tedious to compute for larger solution space. The same problem is then solved using the genetic algorithm to choose the optimal action.

For the considered SM Problem, all the four approaches resulted into the optimal maintenance activities as solution. i.e., to perform *Major repair* on component *5, Minimal repair* on components *1 & 2* and *Do Nothing* on all other components.

Table 22: Comparison of developed methodologies

	Enumeration Method (EM)	**Genetic Algorithm (GA)**	**Reinforcement Learning (RL)**	**Agent based Distributed approach**
Components	[1,2,3,5,9,10]	[1,2,3,5,9,10]	[1,2,3,5,9,10]	[1,2,3,4,5,6,7,8,9, 10,11,12,13,14]
Solution	[1,1,0,3,0,0]	[1,1,0,3,0,0]	[1,1,0,3,0,0]	[1,1,0,0,3,0,0,0,0, 0,0,0,0,0]
Computation time	**185 seconds**	**123 seconds**	**64 seconds**	**88 seconds**

Though all the approaches are suggesting the optimal action, there is stark contrast in computation time required for each of the approaches to arrive at the optimal solution. The proposed RL based approach took 64 seconds to compute the optimal solution for the SM problem, which is significantly lesser than the Enumeration method and GA approaches. Table 22 shows the optimal solution resulted from all the three approaches along with the computation time required to reach up to the optimal solution.

Though the computation time taken using agent based distributed method is higher than computation time taken with RL based methodology, the approach is equally valuable as the machine learning based method because, it will be computationally efficient and even better than the raw centralized decision making RL based Q – learning algorithm, if we incorporate the machine learning pedagogy into this distributed decision making. This decentralized distributed maintenance planning is currently being used with brute force search at the coordinating agent level. The brute force search or enumeration method is inferior in terms of computational efficiency when used for distributed maintenance planning. So, the use of reinforcement learning based methodology at the agent level will make this distributed approach a force to reckon with.

Chapter 7 Conclusion

In this project, a novel attempt has been made to solve the selective maintenance optimization problem using a Reinforcement Learning based methodology and a decentralized distributed maintenance planning approach. The proposed RL based methodology is more effective in the real scenario where the computation time is of importance. And the optimization problem needs to be solved in minimal time, so that the completion of the system's maintenance is ensured within the overall limited maintenance break in between two missions. Q-learning – a temporal difference control algorithm is used here to solve the non-linear programming problem which is modelled as a Semi Markov decision process. This choice proves to be correct considering the nature of the formulated problem. To efficiently solve the optimization problem, two novel heuristics are clubbed within the methodology. The first one refines the solution space in the problem by prioritizing the components for maintenance, based on their reliability against the specified mission. The other heuristic instructs the agent to identify the best policy by consuming minimum maintenance resources. The proposed RL based methodology outperforms the conventional approaches from the viewpoint of the computational efficiency, when demonstrated on the problem of SMO for the benchmark industrial system for coal transportation.

A novel approach is developed to decentralize the decision-making process to various nodes, thereby reducing computational complexity and making the planning algorithm's runtime scalable with increase in problem size. The approach is illustrated for the problem of maintenance scheduling for a multi-state industrial system of coal transportation. The algorithm tackles large problem sizes without compromising on solution accuracy, in significantly lesser runtime when compared to conventional centralized approaches like Brute Force Search and Genetic algorithm. This approach makes the scheme a perfect fit for an Industrial IoT architecture consisting of Cyber-Physical Systems, i.e., Industry 4.0. The distributed approach will give new dimension to conventionally done centralized operations planning in industries.

The distributed maintenance planning approach is beneficial over the RL based methodology when scalability becomes an issue. When dealing with industrial systems with very high number of components or subsystems, the RL based methodology tends to provide sub optimal results due to the stochasticity involved in the learning process. But the agent based distributed approach can deal this issue with ease because of initial filtering process occurring at the subsystem agent level. This significantly reduces the solution space for the coordinating agent to identify the optimal action. This process of two-layer decentralised planning tends to be advantageous over the centralised algorithms.

7.1 SCOPE FOR FUTURE WORK

The developed RL based methodology can be further refined and made robust by ironing out the current limitations. In the present study, the mission duration and maintenance break duration are considered deterministic. However, the stochastic component in these durations can be considered in further work, as suggested by [16,31]. Considering such variances in many other factors would lead to further research in the developed methodology, and will make the methodology more comprehensive. Although the proposed methodology's effectiveness is found to be impressive, it requires the high-end computation infrastructure to train the RL agent for complex real industrial systems. With the advancement in computational technologies, this problem will get resolved, and it will open newer avenues to look into the SMO problems from the standpoint of contemporary machine learning techniques.

The benefits of the proposed agent based distributed approach can be further enhanced by the addition of one more layer of decentralization. This can be done by implementing agents at the individual machine level rather than at the subsystem level. Newer performance indexes can be defined at the machine level to signify the effort required to achieve the reliability at component level. With this agent at individual machine level in addition to the subsystem agent and coordinating agent will improve the accuracy of the approach significantly. This layered distribution will also improve scalability of the problem and expedite the decision making capability for larger industrial systems.

With the omnipresence of sensors and portable computation devices, an agent-based distributed approach for solving SMO problem more expeditiously using proposed RL based methodologies deserves consideration from the research fraternity. Real time monitoring of health of various machine components with the help of advanced sensors will enhance the data available to train and develop more accurate machine learning based methodologies.

REFERENCES

[1] W.F. Rice, C.R. Cassady JN. Optimal maintenance plans under limited maintenance time. Proc. Ind. Eng. Conf., Banff, BC, Canada: 1998.

[2] Rice WF. Optimal Selective Maintenance Decisions for Series Systems. Mississippi State Universsity, 1999.

[3] Cao W, Jia X, Hu Q, Zhao J, Wu Y. A literature review on selective maintenance for multi-unit systems. Qual Reliab Eng Int 2018;34:824–45. https://doi.org/10.1002/qre.2293.

[4] Cassady, C.R., Pohl, E.A. and Paul Murdock W. Selective maintenance modeling for industrial systems. J Qual Maint Eng 2001;7:104–17.

[5] T. Lust, O. Roux FR. Exact and heuristic methods for the selective maintenance problem. Eur J Oper Res J Oper Res 2009;197:1166–77. https://doi.org/https://doi.org/10.1016/j.ejor.2008.03.047.

[6] Cassady CR, Murdock Jr WP PE. Selective maintenance for support equipment involving multiple maintenance actions. Eur J Oper Res 2001;129:252–8. https://doi.org/https://doi.org/10.1016/S0377-2217(00)00222-8.

[7] Liu Y, Huang HZ. Optimal selective maintenance strategy for multi-state systems under imperfect maintenance. IEEE Trans Reliab 2010;59:356–67. https://doi.org/10.1109/TR.2010.2046798.

[8] Pandey M, Zuo MJ, Moghaddass R, Tiwari MK. Selective maintenance for binary systems under imperfect repair. Reliab Eng Syst Saf 2013;113:42–51. https://doi.org/10.1016/j.ress.2012.12.009.

[9] Pham H, Wang H. Imperfect maintenance. Eur J Oper Res 1996;94:425–38. https://doi.org/10.1016/S0377-2217(96)00099-9.

[10] Rajagopalan, R. ; Cassady CR. An improved selective maintenance solution approach. J Qual Maint Eng 2006;12:172–85. https://doi.org/https://doi.org/10.1108/13552510610667183.

[11] Jia X, Cao W, Hu Q. Selective maintenance optimization for random phased-mission systems subject to random common cause failures. Proc Inst Mech Eng Part O J Risk Reliab 2019;233:379–400. https://doi.org/10.1177/1748006X18791724.

[12] Yuo-Tern Tsai ; Kuo-Shong Wang ; Hwei-Yuan Teng.

Optimizing preventive maintenance for mechanical components using genetic algorithms. Reliab Eng Syst Saf 2001;74:89–97. https://doi.org/https://doi.org/10.1016/S0951-8320(01)00065-5.

[13] Sharma P, Kulkarni MS, Yadav V. A simulation based optimization approach for spare parts forecasting and selective maintenance. Reliab Eng Syst Saf 2017;168:274–89. https://doi.org/10.1016/j.ress.2017.05.013.

[14] Pandey M, Zuo MJ, Moghaddass R. Selective maintenance scheduling over a finite planning horizon. Proc Inst Mech Eng Part O J Risk Reliab 2016;230:162–77. https://doi.org/10.1177/1748006X15598914.

[15] Cao W, Song W, Hu Q, Du Y. An exact method for solving selective maintenance problems considering imperfect maintenance. Proc - 2016 Int Conf Intell Netw Collab Syst IEEE INCoS 2016 2016:522–6. https://doi.org/10.1109/INCoS.2016.78.

[16] Liu Y, Chen Y, Jiang T. On sequence planning for selective maintenance of multi-state systems under stochastic maintenance durations. Eur J Oper Res 2018;268:113–27. https://doi.org/10.1016/j.ejor.2017.12.036.

[17] Zhou H, Gao S, Qi F, Luo X, Qian Q. Selective Maintenance Policy for a Series-Parallel System Considering Maintenance Priority of Components. IEEE Access 2020;8:23221–31. https://doi.org/10.1109/ACCESS.2020.2969279.

[18] Khatab A, Aghezzaf EH, Diallo C, Djelloul I. Selective maintenance optimisation for series-parallel systems alternating missions and scheduled breaks with stochastic durations. Int J Prod Res 2017;55:3008–24. https://doi.org/10.1080/00207543.2017.1290295.

[19] Khatab A, Diallo C, Venkatadri U, Liu Z, Aghezzaf EH. Optimization of the joint selective maintenance and repairperson assignment problem under imperfect maintenance. Comput Ind Eng 2018;125:413–22. https://doi.org/10.1016/j.cie.2018.09.012.

[20] Pandey M, Zuo MJ, Moghaddass R. Selective maintenance modeling for a multistate system with multistate components under imperfect maintenance. IIE Trans (Institute Ind Eng 2013;45:1221–34. https://doi.org/10.1080/0740817X.2012.761371.

[21] Jiang T, Liu Y. Selective maintenance strategy for systems executing multiple consecutive missions with uncertainty. Reliab Eng Syst Saf 2020;193:106632. https://doi.org/10.1016/j.ress.2019.106632.

[22] Liu Y, Chen Y, Jiang T. Dynamic selective maintenance optimization for multi-state systems over a finite horizon: A deep reinforcement learning approach. Eur J Oper Res 2020;283:166–81. https://doi.org/10.1016/j.ejor.2019.10.049.

[23] Chen Z, He Y, Zhao Y, Han X, Liu F, Zhou D, et al. Mission Reliability-Oriented Selective Maintenance Optimization for Intelligent Multistate Manufacturing Systems with Uncertain Maintenance Quality. IEEE Access 2019;7:109804–16. https://doi.org/10.1109/ACCESS.2019.2933580.

[24] M. Kumari, N. Chilwant, A. Prajapati MK. An Agent Based Distributed Shop Floor Control System for a Job Shop Environment. Smart Sustain Manuf Syst 2017:1–27. https://doi.org/https://doi.org/10.1520/SSMS20160001.

[25] McFarlane D, Sarma S, Chirn JL, Wong C., Ashton K. Auto ID systems and intelligent manufacturing control. Eng Appl Artif Intell 2003;16:365–76.

[26] Upasani K, Bakshi M, Pandhare V, Lad BK. Distributed maintenance planning in manufacturing industries. Comput Ind Eng 2017;108:1–14. https://doi.org/10.1016/j.cie.2017.03.027.

[27] Kumar S, Manjrekar V, Singh V, Lad BK. Integrated yet distributed operations planning approach: A next generation manufacturing planning system. J Manuf Syst 2020;54:103–22. https://doi.org/https://doi.org/10.1016/j.jmsy.2019.12.001.

[28] Kijima M. Some Results for Repairable Systems with General Repair. J Appl Probab 1989;26:89–102.

[29] Sutton RS, Barto AG. Reinforcement Learning: An Introduction. Second. The MIT Press; 2018.

[30] Adsule A, Kulkarni M, Tewari A. Reinforcement learning for optimal policy learning in condition-based maintenance. IET Collab Intell Manuf 2020;2:182–8. https://doi.org/10.1049/iet-cim.2020.0022.

[31] Khatab A, Aghezzaf EH, Djelloul I, Sari Z. Selective maintenance optimization for systems operating missions and scheduled breaks with stochastic durations. J Manuf Syst 2017;43:168–77. https://doi.org/10.1016/j.jmsy.2017.03.005.