Scheduling Strategies for Enhanced Efficiency in Complex Manufacturing Systems

Table of Contents

Abstract

In today's competitive manufacturing industry, efficient production scheduling is essential to meet strict customer's delivery deadlines and optimise resource utilisation. This research presents a flexible Scheduling Algorithm to optimise the scheduling and allocation of components for multiple products in manufacturing environments. The system addresses the challenges of managing in-house operations and outsourced workflow, coordinating machine resources, and adhering to delivery timelines, all while managing multiple products with complex, sequential processing requirements.

The research starts by analysing the key issues manufacturing industries face, including resource constraints, setup time reduction, and the seamless integration for outsourced components. To tackle these issues, the system utilises secondary data such as product details, a similarity matrix to minimise machine setup times, and machine availability. The flexible algorithm then processes this data, prioritising products based on promised delivery dates and dynamically adjusting the schedule to account for non-working hours, outsourced processing times, and other constraints.

System with its ability to manage both in-house and outsourced components effectively. For in-house components, the similarity matrix is employed to reduce setup times by determining if the machine has previously processed a similar component. Outsourced components, which are sent to external manufacturing units, are carefully integrated back into the production schedule without causing delays, allowing other product components to continue processing during the outsourcing period.

The output of the system includes visualisation module in the form of Gantt charts that shows the production scheduling of product and components. The results demonstrate that the flexible scheduling algorithm significantly improves production scheduling by optimising resource allocation. By dynamically adjusting schedules and integrating outsourced components effectively, the system hopefully will offer a flexible and scalable solution for modern manufacturing challenges based on some recent literature. This research provides a valuable framework for industries looking to enhance their production scheduling capabilities, improve operational efficiency, and meet customer demands more effectively.

Chapter 1: Introduction

1.1 Background

The manufacturing industry has undergone significant changes over the years due to advancements in technology, globalisation, and changing customer demands (Jingxing Gao, 2024). The manufacturing industry plays a fundamental role in transforming raw materials into finished products through various processes (YEJIAN ZHAO, 2017). Advanced scheduling methods like heuristic algorithms and deep learning enhance modern manufacturing systems' ability to quickly adapt to changing conditions like demand fluctuations and machine breakdowns (YEJIAN ZHAO, 2017).

In manufacturing industries, production scheduling involves determining the optimal sequence of operations for producing products while considering factors such as machine availability, labour constraints, and material requirements (Süer, 2019).

Production scheduling is a crucial activity in manufacturing, where it organises and manages the allocation of resources such as machines, labour, and materials to optimise the efficiency of production processes (YEJIAN ZHAO, 2017). Scheduling systems are essential for maximising production efficiency and minimising costs, especially in environments like job shops, where multiple jobs, machines, and resources need to be handled simultaneously (Tao Ren, 2020).

Production systems can be broadly categorised into Job Shop, Flow Shop, and Open Shop systems, each with distinct characteristics and applications.

Job shop production involves producing small batches of customised products with unique processing requirements (YEJIAN ZHAO, 2017). The Figure 0.1 depicts the Job shop production.

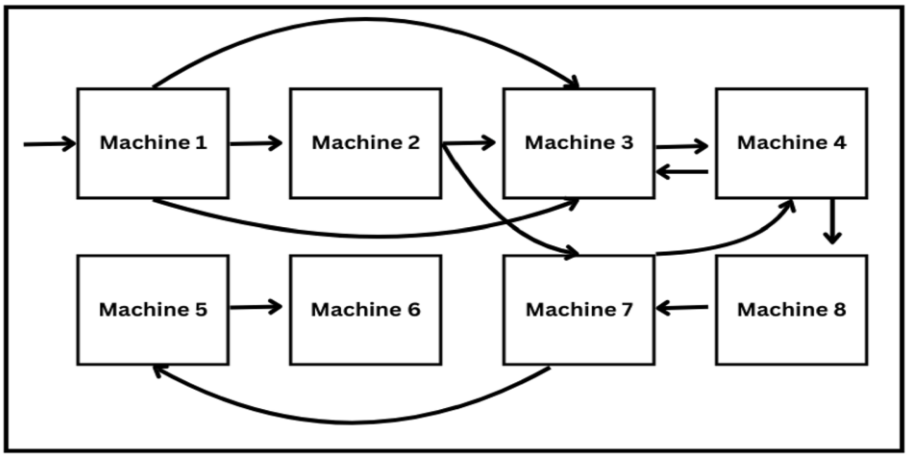

Figure 0.1: Job shop Layout (Owen-Hill, 2017)

As Figure 1 shows, this type of manufacturing requires a flexible and adaptive scheduling system to handle the varying job requirements, which may change frequently (Tao Ren, 2020). Job shops are used for high-mix and low volume products (Owen-Hill, 2017).

Flow shop production, also known as continuous flow manufacturing, is characterised by a fixed sequence of operations performed on each unit or batch of products (Süer, 2019). The Figure 0.2 shows the flow shop production.

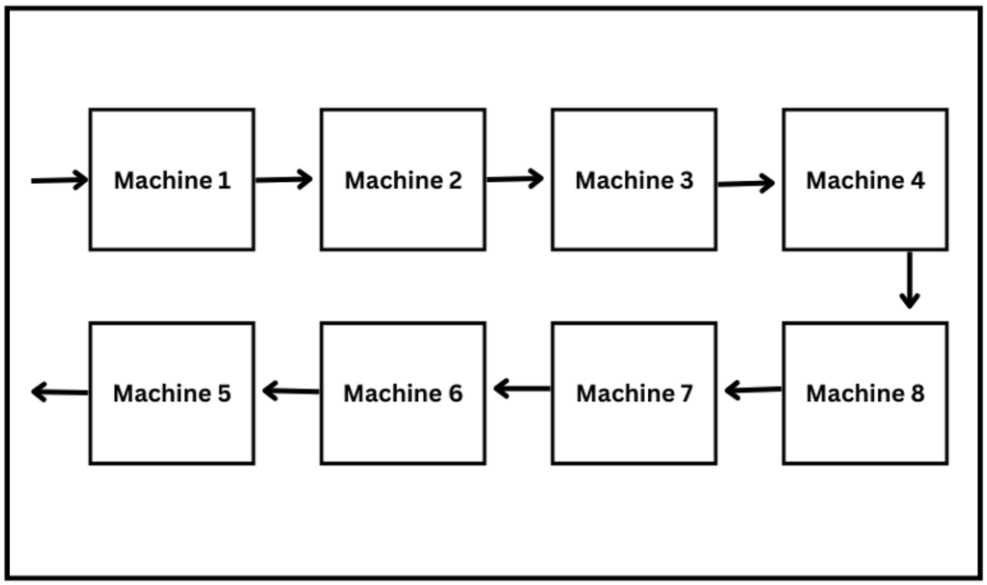

Figure 0.2: Flow shop Layout (Owen-Hill, 2017)

As shown in Figure 0.2 Flow shop production is typically used for high-volume, low-mix industries where the product mix and volume are stable (Jingxing Gao, 2024). This approach commonly found in industries like automotive and electronics (YEJIAN ZHAO, 2017).

Open shop production, also known as general-purpose machine shops, where each machine performs multiple tasks on different products or parts in any order (YEJIAN ZHAO, 2017). In this type, the same machine can perform various operations depending on the product being manufactured (Süer, 2019). The Figure 0.3 depicts the open shop production.

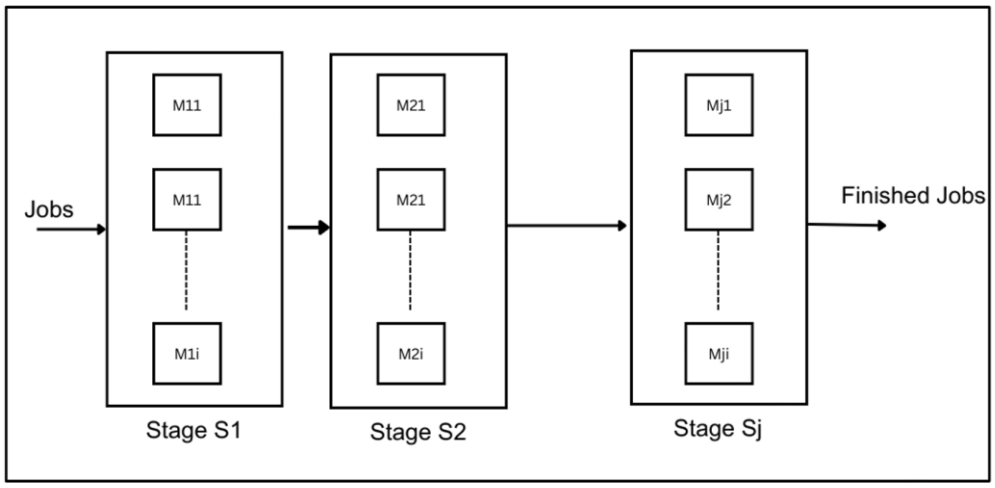

Figure 0.3: Open Shop Layout (Owen-Hill, 2017)

As shown in Figure 0.3 Open shop production is typically used for low-volume, high-mix industries where the product mix and volume are highly variable (Jingxing Gao, 2024).

Manufacturing industries face numerous challenges in managing production processes efficiently. One major issue is resource constraints (Süer, 2019) including the limited availability of machines, labour, and materials (Jingxing Gao, 2024), which can lead to production delays and reduced productivity. The integration of in-house and outsourced processes further complicates scheduling, as delays from external manufacturing units can disrupt the entire production flow (YEJIAN ZHAO, 2017).

Setup and changeover times also present challenges, particularly in environments that frequently switch between various products (Shaya Sheikh, 2019). Minimising these times is essential to maintain machine utilisation and production efficiency.

Uncertainty in processing times can occur due to factors like machine breakdowns, changes in production demand, or human errors (YEJIAN ZHAO, 2017). This Unpredictability in production schedules can lead to missed deadlines or additional costs for manufacturers when production times deviate from estimated times (Jingxing Gao, 2024) (Tao Ren, 2020). Stochastic programming models and real-time scheduling systems are often employed to mitigate the effects of such uncertainties by adjusting schedules dynamically (Jingxing Gao, 2024).

Allocating resources such as machines, workers, and materials efficiently is a common challenge, particularly in job-shop manufacturing environments (Weiwei Zhang, 2024). In these environments, multiple jobs often compete for the same resources, leading to potential delays (Tao Ren, 2020).

Advanced algorithms improve resource allocation and streamline management of in-house and outsourced processes, ensuring seamless coordination and minimal delays. (YEJIAN ZHAO, 2017). Heuristic algorithms utilise rule-based decision-making to optimise production schedules and adapt to changing conditions, providing dynamic scheduling solutions for complex scheduling challenges. (Tao Ren, 2020).

1.2 Problem Definition

Many manufacturing systems face challenges due to sequential processing, resource constraints, and the integration of external vendors. It is designed to manage multiple products with sequential processing, either internally or externally through outsourcing. The Figure 0.4 shows the flowchart of Production scheduling.

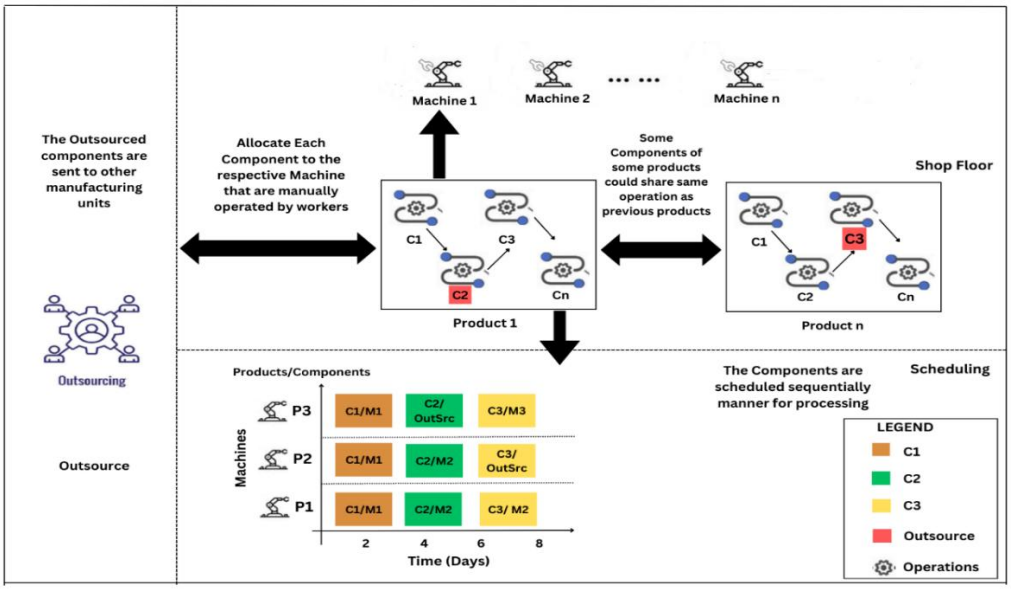

Figure 0.4: Production scheduling workflow

As shown in Figure 0.4, Component are allocated to respective machines manually operated by workers within the shop floor. Each product, such as Product 1 through Product (n), comprises multiple components (C1, C2, C3, Cn) that follow a sequential processing order. The system must process both in-house and outsource components. For in-house components, it is essential to match each component with the appropriate machine based on its processing requirements given in product details. This involves checking the similarity matrix and based on which setup time is determined and run time from product details to determine the optimal scheduling for each component.

Outsourced components present a unique challenge as they are sent to external vendors for processing. The system must account for the time required for outsourced workload, ensuring it does

not process other components of the same product. Once the outsource component is back from the manufacturing unit, the remaining components of that product must be processed in sequential order for each product to be processed.

The input is provided in excel document, which contains product details, similarity matrix and machine details. It includes product name, components, Order Processing Date, Promised Delivery Date, quantity, process type, machine allocation, setup, and run time. The input also includes the order processing date and promised delivery date, as well as the required setup and run time.

The scheduling system must efficiently allocate resources for timely product delivery by managing setup and run times, machine availability, and prioritising products based on delivery deadlines. It should coordinate resources using a similarity matrix and integrate outsourced components into the production schedule. Sequential processing of components is crucial for maintaining order and avoiding delays. The system must manage multiple products simultaneously, adjust priorities based on delivery dates and resource availability, and adhere to time constraints.

1.3 Research Questions

RQ1. How can production scheduling algorithms be optimised to manage both in-house and outsourced components while maintaining efficiency and minimising lead times?

- This question explores the core challenge of optimising production schedules when some components are processed internally, while others are outsourced, requiring synchronisation between both processes. (This Question will be answered in Chapter 3).

RQ2. What strategies in the form of algorithms can be implemented to dynamically adjust in-house production schedules in response to delays or early completion of outsourced components?

- This question focuses on how the system can adapt to real-time changes, such as delays from external vendors or earlier-than-expected returns, and how to minimise the impact on the overall production timeline. (This Question will be answered in Chapter 4).

RQ3. How does the changing production settings such as due date impact generated schedule?

- The question delves into the impact of changes in production parameters, specifically due dates, on the overall production schedule, examining the effects on prioritisation, component processing order, and the necessity of task rescheduling. (This Question will be answered in Chapter 4).

1.4 Aim

The aim of this research is to design, develop, and implement a production scheduling system that optimises the allocation and sequencing of components for multiple products, balancing the complexities of both in-house and outsourced processes. This system will focus on ensuring that all components are processed in the correct order, adhering to strict delivery deadlines while ensuring the efficient use of available resources.

1.5 Objectives

- To review relevant research on production scheduling and outsourcing integration.
- To capture the logic and concept of insource operations and to understand the concept of outsourcing and its impact on inhouse operations.
- To develop a scheduling framework for flexible scheduling purpose that integrates both in-house operations and outsourced workloads, ensuring that production scheduling prioritises components according to strict delivery deadlines and maintains a smooth workflow.
- To implement a flexible scheduling algorithm that can adjust based on changes in resource availability, machine status, and outsourced component delivery. The algorithm will aim to prevent production delays and ensure continuous processing.
- To evaluate the scheduling system across various well-developed scenarios to ensure applicability, reliability, and effectiveness.

1.6 Key Parameters Indicators

Total Production Time (TPT) refers to the entire duration required to complete the production of a product or batch from the initiation of the process to the final output (Shaya Sheikh, 2019). Effective management of Total Production Time (TPT) is crucial for manufacturing efficiency, ensuring timely deliveries, optimising resource usage, and reducing operational costs (YEJIAN ZHAO, 2017). TPT, influenced by factors like process flow efficiency, machine availability, outsourcing delays, and scheduling accuracy, can be significantly reduced by improving these areas (Tao Ren, 2020).

1.7 Methodology

The methodology for this research combines both qualitative and quantitative approaches to address the complex challenges of production scheduling and the synchronisation of in-house and outsourced processes.

The qualitative aspect of the research involves secondary data collection and analysis to gain a deeper understanding of existing practices and theoretical frameworks in production scheduling (YEJIAN ZHAO, 2017). This process begins with an extensive review of literature from previous research papers related to production scheduling systems (Weiwei Zhang, 2024). The literature review will provide insights into the current methodologies, challenges, and solutions that are pertinent to optimising production schedules and integrating outsourced components (Jingxing Gao, 2024).

The qualitative methodology will involve analysing existing production scheduling systems through case studies. This will identify best practices, successes, and challenges in the field. Benchmarking different systems will compare their performance and features, enabling the development of a more effective scheduling system (Shaya Sheikh, 2019).

On the quantitative side, the research focuses on the development and implementation of a flexible scheduling algorithm designed to optimise production scheduling (Tao Ren, 2020). The algorithm will address the dual challenge of allocating components to both in-house machines and outsourced vendors while minimising setup times and ensuring timely delivery (Shaya Sheikh, 2019). The design of this flexible algorithm will be informed by insights from the qualitative research.

Once the scheduling algorithm is developed, it will be evaluated through simulations using historical data. This simulation process will model various production scenarios to evaluate the algorithm's effectiveness in different contexts (Tao Ren, 2020).

By integrating qualitative insights with quantitative analysis, this methodology aims to develop a well-rounded production scheduling system that effectively synchronises in-house and outsourced processes, leading to improved production efficiency and timely delivery of products.

1.8 Tools and Techniques

- Literature review as a technique to explore previous applications of production scheduling related to problem statements.
- Flowcharts and other visualisation techniques used to capture the logic involved in related to problem statements
- A Proper computerised algorithm is used to model the manufacturing environment including the scheduling part.
- A computerised execution platform/language as a tool was used to run the developed algorithm.
- Appropriate scenarios were designed to check the developed scheduling algorithm.

1.9 Deliverables

- A comprehensive literature review that provides insights into existing research on production scheduling and the integration of in-house and outsourced processes.
- A flowchart that visually captures the logic and interconnection between insource and outsource operations, illustrating the workflow and decision-making process for integrating both in the production scheduling system.
- A fully developed scheduling framework that integrates both in-house and outsourced processes, detailing how it prioritises components based on delivery deadlines and ensures efficient production coordination.
- An algorithm for flexible scheduling, capable of real-time adjustments based on resource availability and outsourced component status, along with descriptions of its functionality and performance.
- A tested scheduling system that demonstrates its accuracy, reliability, and effectiveness in various production scenarios.

1.10 Scope

The scope of this research focuses on the development and application of a flexible scheduling algorithm for optimising production scheduling within manufacturing industries. This research specifically addresses the challenge of coordinating both in-house and outsourced processes to ensure efficient and timely production.

The primary focus is on predicting and scheduling production tasks using a flexible scheduling algorithm. The algorithm is designed to manage multiple products with sequential processing, addressing resource management and delivery deadlines. It integrates outsourced processes into the production schedule, accounting for external vendor time and ensuring the system adjusts to delays. The study also aims to develop a system that can adjust priorities based on delivery dates and resource availability, managing sequential processing of components.

1.11 Thesis Structure:

This provides a clear and logical guide to the research process, covering background, objectives, questions, literature review, methodology, results and discussion, and conclusion and recommendations, addressing gaps, highlighting key insights and suggestions for future work.

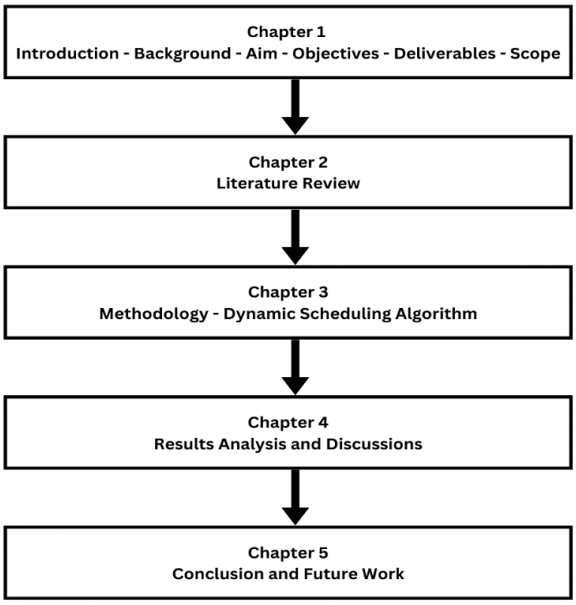

Figure 0.5: Thesis Structure

- Chapter 1: The manufacturing industry emphasises the importance of optimising production scheduling for both in-house and outsourced processes to meet production deadlines.
- Chapter 2: Literature Review reviews significant contributions from existing research on production scheduling, heuristic algorithms, and integration of processes, providing a foundation for the study's methodologies.
- Chapter 3: The methodology states a Flexible Scheduling Algorithm to optimise scheduling, utilising similarity matrices to reduce setup times.
- Chapter 4: The study evaluates the scheduling algorithm's implementation, its impact on process time, and its overall effectiveness.
- Chapter 5: The research findings highlight the algorithm's effectiveness in improving scheduling efficiency and offer suggestions for future research and practical applications.

Chapter 2: Literature Review

This section reviews research on scheduling in the manufacturing industry, focusing on the balance between in-house production and outsourcing. It provides an overview of existing methods, highlights contributions from researchers, and identifies gaps in knowledge for future research.

2.1 Previous Work on Inhouse – Outsource Scheduling in the Manufacturing Industry

This section reviews literature on in-house and outsourced production scheduling in the manufacturing industry, focusing on key approaches for optimising processes, managing lead times, and balancing capacity with outsourcing options. Table 0.1 summarises these studies, providing insights into techniques, constraints, and goals in the manufacturing domain.

(Young Hae Lee, 2002) developed an Advanced Planning and Scheduling (APS) system that integrates outsourcing to improve manufacturing and supply chain operations. By balancing internal and external capacities, the system optimises production, scheduling, and cost efficiency, ensuring timely delivery. The model addresses capacity planning, outsourcing decisions, and supplier selection, offering a flexible solution for managing fluctuating production demands.

(Young Hae Lee, 2002) primary objective is to address the challenges manufacturers face in balancing production efficiency and cost control. Using mathematical optimisation and simulation modelling, the APS system is tested in a real-world manufacturing case, demonstrating improved decision-making, cost reduction, and enhanced production flexibility.

(Zhen, 2012) developed a decision model for production planning with outsourcing, focusing on balancing in-house manufacturing and outsourcing to optimise costs and lead times. The model operates under two production modes: in-house manufacturing and outsourcing, with in-house having higher costs but shorter lead times and outsourcing offering lower costs with longer lead times. The model uses flexible programming to optimise production quantities across multiple time periods, considering stochastic demand, inventory costs, and the trade-off between production costs and lead times.

(Zhen, 2012) proposed a model that aims to assist manufacturers in making optimal production decisions between in-house manufacturing and outsourcing, considering fluctuating and uncertain demands. It considers variables like production quantities, inventory levels, and demand forecasts, calculating expected costs and determining optimal decisions by minimising production and inventory

costs over multiple periods. Although theoretical, the model is applicable to various manufacturing scenarios where outsourcing is a significant part of production planning. However, it does not account for practical challenges like setup costs or spillover of production runs and assumes fixed lead times for outsourcing.

(Xueling Zhong, 2022) emphasised a coordinated scheduling model for outsourcing, in-house production, and distribution operations. It aims to determine the optimal subset of jobs to be outsourced and an integrated schedule for jobs processed in-house, minimising total make span and costs associated with outsourcing and delivery. The model incorporates mixed integer programming (MIP) formulation and considers batch deliveries based on physical size and vehicle capacity. Two approximation algorithms, a 2-approximation algorithm and a refined approximation algorithm, are developed. The study shows that coordinated scheduling significantly reduces costs and improves production efficiency compared to uncoordinated strategies.

(Qi, 2019) developed a scheduling model for a two-stage flow shop system was developed to minimise make span by optimising job allocation between in-house production and outsourcing to subcontractors. The model considers job sequencing for both stages and three scenarios: full outsourcing, stage-one process, and partial outsourcing. The study addresses the complexity of production scheduling when outsourcing is introduced, focusing on coordination of job sequencing, transportation delays, and costs. Results show that outsourcing can significantly reduce make span and overall production costs under certain conditions.

(Liu, 2019) introduced A hybrid variable neighbourhood search (HVNS) algorithm has been developed to address the two-machine flow shop outsourcing and rescheduling problem (TFSORP), which arises from unexpected job disruptions. The algorithm integrates a mixed integer programming model with optimisation techniques to balance in-house scheduling and outsourcing decisions. It uses job addition and removal heuristics, three problem-specific neighbourhood structures, and a penalty parameter for exceeding job limits. The goal is to minimise disruptions' impact on the original production schedule. The algorithm outperforms existing methods in terms of computation time and solution quality, making it NP-hard.

(Eun-Seok Kim, 2022) focused on developing scheduling algorithms for a two-machine flow shop system, focusing on outsourcing and lead-time. The model aims to minimise total cost, a weighted sum of outsourcing costs and scheduling measures like total completion time. The paper introduces exact algorithms and heuristic approaches to optimise production scheduling with outsourcing. The problem is classified as NP-hard, requiring efficient algorithms. The study validates the proposed algorithms

using computational experiments, showing branch-and-bound algorithms find optimal solutions for large problem instances.

(Hadi Mokhtari, 2012) emphasised a scheduling model for a single-stage production system with an outsourcing option, aiming to minimise total weighted completion time and outsourcing costs. The model incorporates in-house production using a parallel machine system and outsourcing to subcontractors with their own machines. The problem is decomposed into smaller subproblems using a Lagrangian relaxation technique. A heuristic algorithm is proposed to efficiently solve these subproblems, ensuring optimal solutions. The model addresses the complexity of managing both in-house production and outsourcing simultaneously, balancing in-house capacity with outsourcing due to fluctuating demands.

(Kangbok Lee, 2011) developed a two-stage production scheduling model that aims to minimise processing time and outsourcing costs by scheduling jobs using a two-machine flow shop approach. It associates costs with subcontracted jobs, processing them instantaneously but proportional to their processing times. The model examines how different outsourcing costs affect computational complexity and provides both NP-hard and polynomial solvable cases. Its performance is validated through computational experiments on randomly generated job sets with different processing times and outsourcing costs. Results show that outsourcing can reduce processing time and production costs strategically. However, the model overlooks practical considerations like variable outsourcing lead times and supplier reliability uncertainties.

(Xiaojuan Jiang, 2021) aimed to improve the two-stage production scheduling problem with an outsourcing option by minimising the sum of processing time and total outsourcing costs. A modified greedy algorithm for optimising outsourcing decisions in two production stages. The model operates under two types of outsourcing scenarios: Type I, where costs for one stage are less than 1, and Type II, where costs for both stages are less than 1. The modified greedy algorithm outperforms the original algorithm, achieving a worst-case performance ratio of 1.25 for Type II problems and 1.25 for Type I scenarios. Future work could integrate these factors for more robustness in practical settings.

(Ik Sun Lee, 2007) developed a single-machine scheduling model that aims to minimise the weighted sum of outsourcing costs and scheduling measures, particularly completion times, while adhering to a given outsourcing budget. The problem is NP-hard and employs heuristic and branch-and-bound algorithms to solve efficiently. The model considers jobs that can be processed on an in-house machine or outsourced at a cost, with processing times and outsourcing costs varying for each job. The model is tested using computational experiments with randomly generated job sets, demonstrating the effectiveness of both heuristics and branch-and-bound algorithms. However, the study lacks

consideration of real-world uncertainties, suggesting future research could incorporate these complexities for practical applications.

(Erfan Babaee Tirkolaee, 2020) developed a model for single-machine scheduling with outsourcing, aiming to minimise the weighted sum of outsourcing costs and scheduling measures, particularly completion times, while adhering to a given outsourcing budget. The problem is NP-hard, and the study proposes heuristic and branch-and-bound algorithms to solve it efficiently. The model considers jobs that can be processed on an in-house machine or outsourced at a cost, with processing times and outsourcing costs varying for each job. The model is tested using computational experiments with randomly generated job sets, showing that both heuristics and branch-and-bound algorithms are effective. However, the study lacks consideration of real-world uncertainties, such as fluctuating demand, variable lead times, or changes in outsourcing costs over time.

Table 0.1: Previous Work in Manufacturing IndustryTable 0.1 summarizes previous research in the manufacturing industry, focusing on in-house and outsourced production scheduling. It highlights key aspects such as constraints, aims, scenarios, and techniques used, comparing whether the approaches were static or dynamic to optimize production processes.

Table 0.1: Previous Work in Manufacturing Industry

S. No	Researcher Names	Industry	Insource or Outsource	Constraints	Aim	Scenarios	Techniques Used	Static or Dynamic
1	Young Hae Lee, Chan Seok Jeong, Chiung Moon	Manufacturing	Insource & Outsource	Capacity, Production Lead-time	Optimise production schedules with outsourcing	Fixed vs Variable Demand	Mixed Integer Programming	Dynamic
2	Shen, Lu	Manufacturing	Outsource	Costs, Production Capacity	Minimise production costs with outsourcing	Production vs Outsourcing Costs	Dynamic Programming, Heuristics	Static
3	Xueling Zhong, Jie Fan , Jinwen Ou	Manufacturing	Insource & Outsource	Lead-time, Supplier Capacity	Coordinate production and outsourcing schedules	Single vs Multi-Supplier	Mixed Integer Programming, Approximation Algorithms	Dynamic
4	Qi, Xiangtong	Manufacturing	Insource & Outsource	Production Capacity, Job Costs	Optimise production and outsourcing decisions	Fixed vs Variable Lead-times	Approximation Algorithms	Dynamic
5	Liu, Le	Manufacturing	Insource & Outsource	Machine Failures, Delays	Minimise rescheduling	Machine Failure vs Normal Operation	Hybrid Algorithms, Simulation	Dynamic

					delays with outsourcing			
6	Eun-Seok Kim, Ik Sun Lee	Manufacturing	Insource & Outsource	Lead-times, Machine Capacity	Minimise lead-time in flowshop scheduling with outsourcing	Constant vs Variable Lead-times	Branch-and-Bound Algorithm	Dynamic
7	Hadi Mokhtari, Isa Nakhai Kamal Abadi	Manufacturing	Outsource	Lead-time, Costs	Optimise scheduling with outsourcing across stages	Manufacturer Costs vs Outsourcing	Heuristics, Lagrangian Relaxation	Dynamic
8	Xiaojuan Jiang, An Zhang, Yong Chen, Guangting Chen, Kangbok Lee	Manufacturing	Outsource	Job Processing Times	Improve production scheduling algorithms	Small vs Large Jobs	Greedy Algorithm, Dynamic Programming	Dynamic
9	Ik Sun Lee, C.S. Sung	Manufacturing	Outsource	Scheduling Times, Costs	Minimise completion time with outsourcing options	In-house vs Outsourcing Decisions	Heuristics, Branch-and-Bound	Static

10	Erfan Babaee Tirkolaee, Alireza Goli, and Gerhard-Wilhelm Weber	Manufacturing	Insource & outsource	Energy, Lead-times	Minimise energy and production costs	Energy vs Time Trade-offs	Fuzzy Programming, Metaheuristics	Dynamic
11	Kangbok Lee, Byung-Cheon Choi	Manufacturing	Insource & outsource	Lead-time, Costs	Optimise scheduling with outsourcing across stages	Manufacturer Costs vs Outsourcing	Heuristics, Lagrangian Relaxation	Dynamic

Table 0.1 focused on optimising production scheduling in manufacturing environments, especially where outsourcing is used to manage capacity, lead times, and costs. It aims to balance in-house production and outsourcing decisions to improve efficiency, reduce lead times, and control costs. Dynamic scheduling models, using heuristic and metaheuristic techniques like greedy algorithms, fuzzy programming, and Lagrangian relaxation, are developed to address complex scheduling problems. The goal is to create adaptive scheduling systems that integrate in-house production with outsourcing, enhancing cost efficiency and streamlining production timelines. However, a dynamic scheduling system that continuously adjusts to changing production constraints, such as machine downtime and outsourcing delays. It differs by incorporating monitoring and visualization tools like Gantt charts, allowing users to actively engage with and modify schedules based on immediate needs. While many studies focus on cost reduction or lead-time optimization. This system ensures a more adaptable and user-driven approach to production scheduling, making it better suited for complex and fast-changing environments.

2.2 Previous Work on Inhouse – Outsource Scheduling in Other Industries

This section explores the use of in-house and outsourced scheduling methods in industries like healthcare, energy, and telecommunications, identifying common techniques and challenges, and recognising industry-specific needs.

(Camacho-Rodrígue, 2016) developed a simulation-optimisation model has been developed to improve blood supply chain production planning using Discrete Event Simulation (DES) and Integer Linear Programming (ILP). The model addresses challenges like perishability, unpredictable demand, and multiple collection and production methods. It aims to optimise production to reduce shortages and waste while ensuring timely delivery to hospitals. The model was tested using data from a Colombian blood centre, Hemocentro Distrital blood centre in Bogotá, Colombia. Results showed significant improvements in blood supply chain management, including a decrease in stockouts by up to 71.2% and a reduction in outdated red blood cell units by 46.5%. However, the model has limitations, such as assuming static demand patterns and fixed lead times.

(F. A. QAYYUM, 2015) emphasised an appliance scheduling optimisation model for smart home networks using Mixed-Integer Programming (MIP) technique. The model aims to minimise electricity costs and manage peak loads for energy-consuming appliances, incorporating a photovoltaic (PV) system as a power-producing unit. The scheduling problem is formulated as a mixed-integer linear programming problem with constraints such as uninterruptible operation, user time preferences, and appliance load profiles. The model incorporates real-world constraints like peak power limitations and sequence requirements between appliances. The model uses a branch-and-bound algorithm to optimise the scheduling process, dividing the day into 96 time slots and scheduling appliances based on their specific load profiles. The results show that the proposed MIP-based appliance scheduling model significantly reduces energy costs and peak load.

(Wei Yang Bryan Lim, 2022) A dynamic resource allocation framework for Hierarchical Federated Learning (HFL) has been developed within decentralised edge intelligence systems. The framework optimises resource allocation and incentives in a network of heterogeneous devices, enabling model training without centralised data transmission. The model consists of a two-level system: a lower level focused on resource allocation and incentive design between workers and intermediate aggregators, and an upper-level managing cluster head allocation through a deep learning-based auction mechanism. The research aims to improve federated learning by decentralising the system, reducing reliance on a central controller, and offering a robust incentive mechanism for worker participation, optimising resource allocation and data coverage.

Table 0.2 shows in-house and outsourced production scheduling in the other industry, comparing static and dynamic approaches for optimizing production processes.

Table 0.2: Previous Work in Other industries

S. No	Researcher Names	Industry	Insource or outsource	Constraints	Aim	Scenarios	Techniques Used	Static or Dynamic
1	Camacho-Rodrígue, Andres F. Osorio1 & Sally C. Brailsford & Honora K. Smith & Sonia P. Forero-Matiz & Bernardo A.	Healthcare (Blood Supply Chain)	Insource & outsource	Perishability, Demand Uncertainty	Minimize waste and stockouts in blood supply	Fixed vs Flexible Demand	Discrete Event Simulation (DES), ILP	Dynamic
2	F. A. QAYYUM, M. NAEEM, A. S. KHWAJA, A. ANPALAGAN1, L. GUAN, AND B. VENKATESH	Energy (Smart Homes)	Insource	Appliance Operation, User Preferences	Minimise energy costs and manage peak load	PV Panel Integration, TOU Tariffs	Mixed Integer Programming (MIP)	Static

3	Wei Yang Bryan Lim, Jer Shyuan Ng, Zehui Xiong Member, IEEE, Dusit Niyato Member, IEEE, Jiangming Jin, Member, IEEE, Fellow, IEEE, Cyril Leung and Chunyan Miao	Telecommunications (Edge Intelligence)	Outsource (Worker Participation)	Worker Participation, Resource Allocation	Optimise resource allocation in federated learning	Worker Congestion, Rewards	Evolutionary Game Theory, Deep Learning Auction	Dynamic

Table 0.2 presents the optimisation of various industries, including healthcare, energy, and telecommunications, involves improving operational efficiency through advanced scheduling, resource allocation, and decision-making models. In healthcare, studies focus on optimising blood supply chains to minimise waste and ensure timely deliveries. Energy research aims to reduce energy costs in smart homes by scheduling appliances according to time-of-use tariffs and integrating renewable energy sources. In telecommunications, the focus is on optimising resource allocation in decentralised edge intelligence systems using federated learning to reduce communication inefficiencies. However, scheduling algorithm concentrates on optimizing production scheduling in a manufacturing environment, balancing in-house and outsourced components, the studies presented address challenges in healthcare, energy, and telecommunications.

2.3 Gap in Knowledge

This section identifies gaps in the literature on scheduling in manufacturing and other industries, highlighting the need for a dynamic, real-time heuristic approach that integrates both in-house and outsourced production, while accounting for operational constraints

(Ik Sun Lee, 2007) focused on optimising machine scheduling in electronics and motor manufacturing to reduce completion times and outsourcing costs, using techniques like heuristics, Branch-and-Bound, and Dynamic Programming, while considering constraints. In contrast, (Eun-Seok Kim, 2022) addresses a more a complex production scenario involving two machines and introduces lead times. It aims to minimise completion time and costs using dynamic programming, Branch-and-Bound, and heuristic methods.

This highlights a significant gap in the existing knowledge. (Ik Sun Lee, 2007) paper's static environment does not account for dynamic factors such as lead times, which are crucial in real-world scenarios where outsourcing often involves delays. Although (Eun-Seok Kim, 2022) incorporated lead times, it lacks a comprehensive analysis of the impact of variable lead times and dynamic constraints on scheduling decisions compared to a static model, and lacks a comprehensive analysis of their performance in diverse contexts, particularly in complex manufacturing environments with multiple machines.

(Kangbok Lee, 2011) developed the two-stage production scheduling model in electronics manufacturing uses a genetic algorithm to optimise production by outsourcing operations and handling in-house tasks, aiming to reduce costs and ensure timely delivery, tested in a static environment. On the other hand, (Xueling Zhong, 2022) investigated coordinated scheduling for in-house and outsourcing production in a manufacturing context, balancing them, managing conflicts, and optimising resource allocation using Mixed Integer Linear Programming.

Comparing these reveals several gaps in knowledge. While (Kangbok Lee, 2011) focus on a two-stage production process with a sophisticated genetic algorithm, (Xueling Zhong, 2022) tackled general coordination issues but do not address the multi-stage complexities or the dynamic aspects of outsourcing. (Kangbok Lee, 2011) study is constrained by its static approach, not accounting for potential dynamic changes in the production environment, such as variations in demand or disruptions. Conversely, (Xueling Zhong, 2022) work, while useful for general coordination, lacks the detailed optimisation of two-stage processes and the advanced algorithmic approach found in (Kangbok Lee, 2011) study. Thus, future research could benefit from combining the multi-stage and dynamic considerations with advanced algorithms for a more comprehensive understanding of scheduling in complex production environments.

The key gap between the previous literature and the current research is the dynamic management of outsourcing and the incorporation of operational constraints into scheduling processes. Previous works are effective in handling production scheduling and outsourcing, often focus on static models that do not adapt to real-time changes in operations. For instance, studies like (Young Hae Lee, 2002) focus on fixed demand and capacity constraints but do not address the flexibility required in practical scenarios where product priorities and component availability frequently shift.

In this research a flexible approach that not only manages in-house and outsourced components but also adjusts for interruptions such as breaks, holidays, and delayed outsourced components. This stands in contrast to earlier studies, such as (Zhen, 2012), which use dynamic programming and heuristics but do not accommodate real-time waiting periods for outsourced components.

Furthermore, previous studies tend to view outsourcing as a binary decision (in-house or outsourced) without considering the delays introduced by outsourced component returns. For example, (Xueling Zhong, 2022) addresses coordinated scheduling but does not account for time delays associated with outsourcing. In contrast, the model incorporates these delays and dynamically shifts focus between products to minimise overall production time, filling a significant gap in this area.

Chapter 3: Research Methodology

The chapter is to explain the methods and procedures used to conduct the research. It outlines the approach taken for data collection, analysis, and the rationale behind choosing specific techniques.

3.1 Qualitative Analysis (Data collection)

This section presents the methods and sources used to gather data for the research, with a focus on secondary data. The data used includes production schedules, machine availability, processing times, and delivery deadlines, which have been obtained from existing papers and historical datasets within the manufacturing system.

Table 0.1 shows the key inputs used in the research, explaining their meanings and how they collaborate with other inputs. It also provides the reason for including each input and its significance to the research process.

Table 0.1: Input Parameters

Input Name	Meaning	Collaboration	Reason	Reference
Product Name	Name of the product to be produced.	Organises and tracks production processes.	Aligns processes, resources, and schedules.	(Daeyoung Chung, 2005), (Christos T. Maravelias∗, 2009), (Byung-Cheon Choi, 2020)
Order Processing Date	Date the order is processed.	Helps plan production sequences and delivery schedules.	Tracks order receipt and schedules tasks.	(Daeyoung Chung, 2005), (Ik Sun Lee, 2007), (Byung-Cheon Choi, 2020)
Promised Delivery Date	Delivery deadline.	Sets deadlines, collaborates with Order Processing Date to create timelines.	Ensures on-time delivery, meeting customer expectations.	(Daeyoung Chung, 2005), (Ik Sun Lee, 2007), (Byung-Cheon Choi, 2020)
Quantity Required	Units needed for the order.	Affects scheduling, resource allocation, and operation timing.	Determines production scale and resources.	(Christos T. Maravelias∗, 2009), (Byung-Cheon Choi, 2020)
Components	List of parts for production.	Associated with Product Name to process components for each product.	Aligns schedules for correct components.	(Christos T. Maravelias∗, 2009), (Byung-Cheon Choi, 2020)
Process Type	Inhouse or Outsourced process.	Determines type of process required.	Categorises and assigns tools/machines.	(Byung-Cheon Choi, 2020), (Christos T. Maravelias∗, 2009)

Machine Number	Machine identifier for the operation.	Works with Process Type, Setup Time, and Run Time for scheduling.	Ensures correct machine allocation.	(Christos T. Maravelias∗, 2009)
Run Time (min/1000)	Time to process 1000 units.	Impacts Start/End Times, estimated from Quantity Required.	Plans production duration.	(Christos T. Maravelias∗, 2009), (Byung-Cheon Choi, 2020)
Setup Time (seconds)	Machine preparation time.	Impacts Start Time and readiness for operation.	Accounts for machine preparation.	(Christos T. Maravelias∗, 2009), (Byung-Cheon Choi, 2020)
Start Time	Operation start time.	Works with Setup, Ready, and End Times for schedule coordination.	Ensures timely operation start.	(Christos T. Maravelias∗, 2009), (Byung-Cheon Choi, 2020)
End Time	Operation end time.	Works with Start and Run Times to complete the operation on schedule.	Marks operation completion.	(Christos T. Maravelias∗, 2009), (Byung-Cheon Choi, 2020)

As mentioned in Table 0.1, The order processing date is the formal entry of an order into the production system, setting the sequence of subsequent operations and timeline for downstream activities. According to (Daeyoung Chung, 2005), The order processing date is crucial for job shop scheduling, machine usage, idle time reduction, and preventing delays. Its accuracy and timeliness are vital as any delay can impact production and product delivery. As outlined by (Daeyoung Chung, 2005), delivery dates are particularly important in industries where timely delivery is a competitive necessity

(Ik Sun Lee, 2007) emphasises the importance of delivery dates in production scheduling, particularly in cases of insufficient manufacturing capacity, and the necessity of outsourcing to meet these dates. (Byung-Cheon Choi, 2020) discusses the significance of aligning production schedules with due dates, highlighting the benefits of flexible scheduling methods in meeting promised delivery dates in complex environments. (Christos T. Maravelias∗, 2009) highlights the necessity of coordinating order processing with delivery commitments to ensure that production aligns with broader business objectives and customer needs.

As mentioned in Table 0.1, the Quantity Required is another crucial parameter that directly influences how resources are allocated, and jobs are sequenced in a production schedule. (Byung-Cheon Choi, 2020) examines the impact of accurately determining and scheduling required quantities, especially in outsourcing scenarios, on minimising waste and optimising production efficiency. (Christos T.

Maravelias∗, 2009) emphasise the importance of managing production targets and quantities, which is essential for aligning production schedules with demand forecasts and resource availability.

As mentioned in Table 0.1, Components and Process Type are crucial in defining job specifications, determining operation sequences, and machinery needed for smooth production flows. (Byung-Cheon Choi, 2020) discuss how the complexity of components and process types can affect scheduling, particularly when determining whether jobs should be processed in-house or outsourced. (Christos T. Maravelias∗, 2009) emphasises the need for detailed planning of components and process types to ensure that all necessary resources and processes are considered in the production schedule.

Machine-related parameters such as Machine Number, Run Time, Cycle Time, and Setup Time are pivotal in optimising machine utilisation and minimising downtime. While (Byung-Cheon Choi, 2020) delve into the complexities of scheduling in scenarios involving different machines and processing times, highlighting the need to minimise total production time, (Christos T. Maravelias∗, 2009) implies the importance of these parameters by discussing overall production efficiency.

As mentioned in Table 0.1, timing parameters like Start Time, End Time, Ready Time, and Wait Time are essential for managing the flow of jobs through the production process. (Byung-Cheon Choi, 2020) emphasise the need for synchronisation within production schedules to avoid delays and reduce waiting times, which can otherwise lead to inefficiencies and increased costs.

A random dataset was generated by drawing on insights of (Daeyoung Chung, 2005), (Ik Sun Lee, 2007), (Byung-Cheon Choi, 2020) and (Christos T. Maravelias∗, 2009).

3.2 Input – Output Analysis

The purpose of the Input and Output Analysis is to examine and understand the relationship between the data and parameters that feed into the scheduling system (inputs) and the resulting outcomes (outputs) in a production environment.

3.2.1 Input Analysis

The Input Analysis section focuses on identifying and describing inputs that influence the production scheduling system in a manufacturing environment. It examines factors like product details, operational parameters, machine availability, and outsourcing times to create an optimised and efficient production schedule. The Figure 0.1 depicts the inputs and its relation to the scheduling system and its relationship with each other.

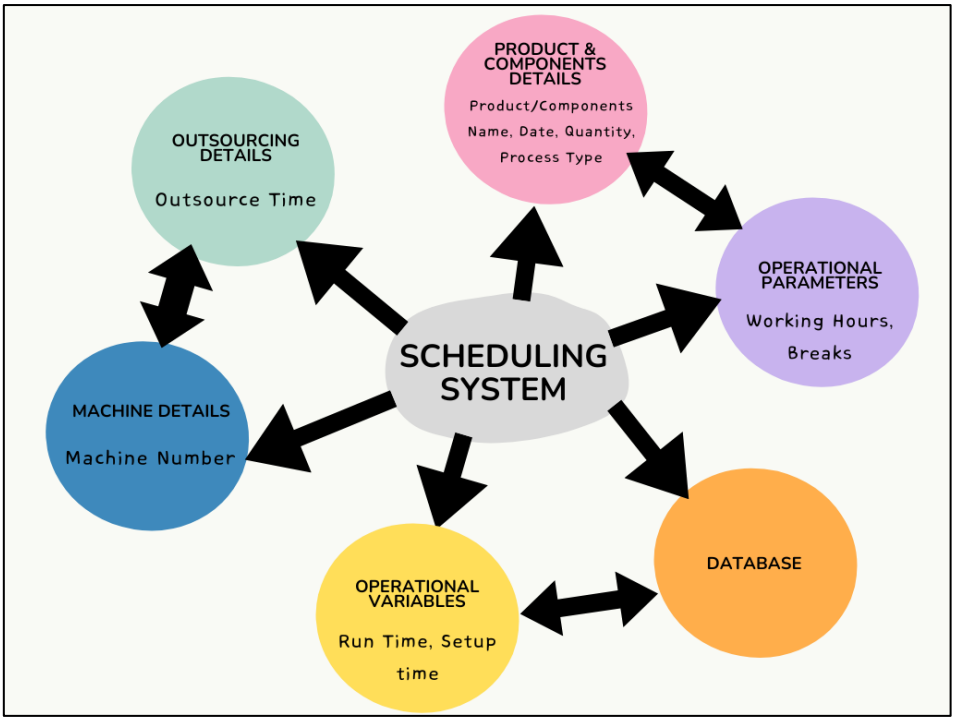

Figure 0.1: Input Analysis

The above Figure 0.1 represents the scheduling system in a manufacturing or production environment is influenced by various inputs, including Product & Components Details, which provide essential data and parameters like product names, components, processing dates, delivery dates, quantities, and required process type. These inputs determine priorities and specifics of what needs to be scheduled.

The Operational Parameters, represented by purple and orange, define the constraints of the scheduling system, ensuring tasks are scheduled around predefined time frames. The Database, in orange, stores and updates relevant data in real-time, allowing the system to access up-to-date information and dynamically adjust schedules based on new inputs or changes.

Operational Variables, represented in yellow, include run and setup times, which are crucial for task duration and resource allocation. Machine Details input, in blue, provides information about machine availability, aiding in task assignment, ensuring efficient resource utilisation and minimising downtime, thus enhancing scheduling efficiency.

Outsourcing Details, represented in green, provide information on outsourcing times for components processed outside the main facility, allowing the scheduling system to adjust timelines for potential delays or extended processing times. The system uses inputs connected through arrows to create an optimised production schedule, meeting delivery deadlines while efficiently using resources. The interconnected nature of the database highlights its pervasive influence on the process.

3.2.2 Output Analysis

The Output Analysis section examines the performance of a production scheduling system, focusing on key outputs like Outsource time, run times, quantity and delivery schedules. The Figure 0.2 depicts the relationship between each output variable.

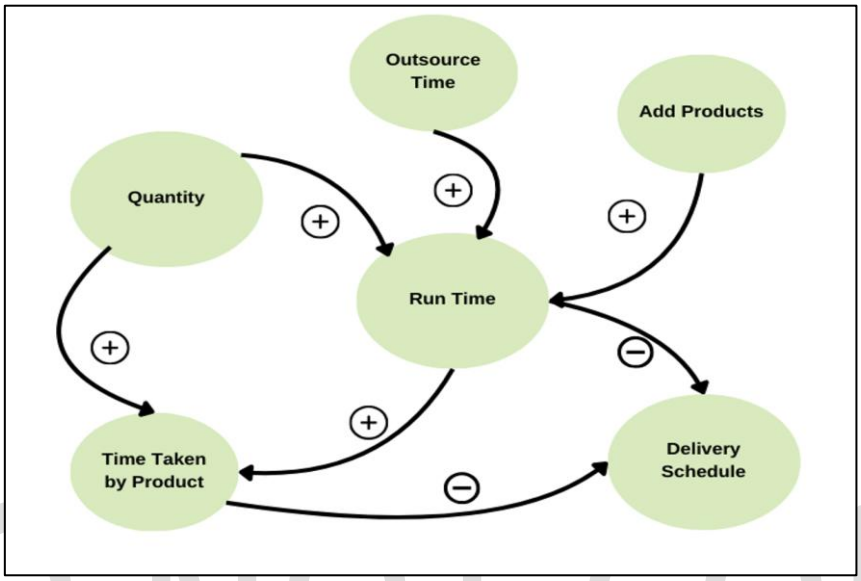

Figure 0.2: Output Analysis

The Figure 0.2 represents the interdependencies between various parameters in a production scheduling system, illustrating how changes in one area can influence others and affect the overall efficiency and the ability to meet delivery schedules.

The Output analysis highlights the importance of quantity in the production process. As the quantity of a product increases, so does the Run Time for manufacturing, as more units require more time to produce. This can be further increased by adding more products to the production schedule. The higher volume of products also leads to longer queues for processing, increasing the waiting time for each unit. This overall increase in run times increases the Time Taken by Product.

Outsource time is a key factor influenced by quantity, as increased demand for external resources or outsourced components can lead to longer outsource times. These delays delay the availability of essential components, extending the total time taken by the product and potentially compromising the delivery schedule. A longer time reduces the likelihood of meeting the schedule on time, while a shorter time enhances the possibility of timely or early deliveries.

The Figure 0.2 illustrates the relationship between Run Time and delivery schedule, highlighting the potential for delays due to increased run time. Conversely, shorter run times can help meet or improve

the delivery schedule. The Figure 0.2 highlights the importance of managing these interrelated factors for an efficient production process.

3.3 Scheduling Framework for Efficient In-House and Outsourced Production:

The system aims to reduce energy consumption in data-intensive applications by implementing dynamic resource allocation and energy-aware scheduling algorithms across multiple layers, including network, resource, middleware, and application, ensuring efficient task distribution and minimal communication overhead. (Ziliang Zong, 2007)

(Ziqing Wang, 2024) proposed a deep reinforcement learning (DRL) framework and discrete event simulation (DES) to address the dynamic job shop scheduling problem (DJSP). The DRL agent uses the Proximal Policy Optimisation (PPO) algorithm to learn optimal strategies based on real-time job shop data. The system adapts to new job arrivals, machine breakdowns, and other dynamic factors, ensuring flexibility and responsiveness, improving operational efficiency under uncertain job shop conditions.

(Zufa Wu, 2023) developed a dual-layer deep reinforcement learning system designed for flexible job shop scheduling. It dynamically selects optimisation goals and applies dispatching rules to adapt to random job arrivals and machine breakdowns. This enhances scheduling flexibility and adaptability, ensuring efficient response to unpredictable production conditions.

Figure 0.3 presents the framework is a system designed to optimise production scheduling in manufacturing environments. It uses a centralised database and scheduling algorithms to collect input data like product information, machine details, and operational specifics. The system uses an interactive dashboard in Python to allow users to input data, monitor processes, and visualise results. The dashboard UI empowers users to make informed decisions, improving production efficiency, resource utilisation, and operational effectiveness. The Figure 0.3 depicts the various components used in scheduling system.

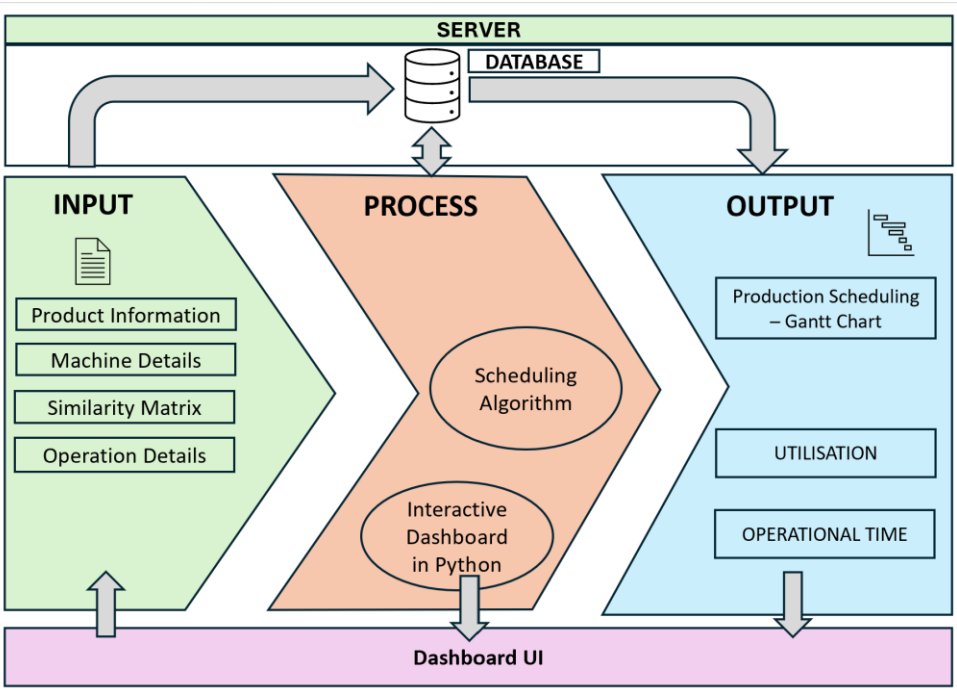

Figure 0.3: Scheduling Framework for Efficient In-House and Outsourced Production

As in Figure 3.3, the architecture's core is a database storing product details, machine specifications, and operational data. The server manages data flow, ensuring efficient information flow between input, process, and output sections. The input section gathers product information and machine details, optimising scheduling. The process section analyses and processes input data to create optimal production schedules. An interactive dashboard in Python enables user interaction, ensuring practical and efficient solutions.

The output section generates key outputs for managing and optimising production processes, including a Gantt Chart for scheduling. These outputs help identify bottlenecks, assess machine efficiency, and identify underused resources. The dashboard UI is user-friendly, allowing users to visualise production schedules, monitor machine usage, and assess operational times. It also allows users to provide inputs, enabling personalised control over scheduling and informed decision-making in production management.

As in Figure 3.3, the architecture facilitates seamless data flow from the database to the input section, where it is collected and processed. A flexible algorithm generates production schedules based on input data, which are visualised and outputted as metrics and charts on the dashboard UI. This enhances production efficiency and effectiveness by providing an interactive, transparent interface and allowing users to input data directly.

3.4 Flexible Scheduling for Hybrid In-House and Outsourced Component assembly Production:

The Flexible Scheduling Algorithm is a tool that optimises production scheduling in flexible manufacturing environments. It prioritises tasks based on delivery dates, ensures optimal resource use, and pauses processing until all necessary components are available. It accounts for non-working hours, lunch breaks, weekends, and holidays, creating realistic schedules. The algorithm provides continuous database updates, Gantt charts, and detailed reports for performance monitoring and informed decision-making.

Similar algorithms developed by other researchers, including (Zhen, 2012) presented the decision model for production planning optimises in-house and outsourcing production using mathematical techniques to balance capacities, costs, and delivery times. The flexible scheduling algorithm allows real-time adaptation to delay or changes in outsourced components.

(Xueling Zhong, 2022) designed the scheduling model aims to reduce costs in a manufacturing system that includes in-house production and outsourcing by integrating decisions related to job outsourcing, in-house processing, and batch deliveries, allowing real-time adjustments.

(Kangbok Lee, 2011) emphasised a two-stage production scheduling model that incorporates outsourcing to optimise production processes. It uses a genetic algorithm to determine which operations should be completed in-house and which should be outsourced. The model balances objectives like cost reduction and delivery deadlines, making it useful in complex manufacturing environments. The study focuses on optimising the schedule at the planning stage to minimise costs and meet production goals.

In contrast, the flexible scheduling algorithm is designed to continuously adapt to changes during the production process, while (Kangbok Lee, 2011) model focuses on optimising the schedule upfront using a genetic algorithm. This immediate adjustment is different from the (Kangbok Lee, 2011) approach, which primarily relies on initial optimisation and does not emphasise ongoing adjustments once the schedule is set. The flexible scheduling algorithm allows for real-time adjustments to the production process, ensuring efficient and timely processing.

(Liu, 2019) addressed the challenge of managing disruptions in a two-machine production environment, particularly when new jobs unexpectedly arrive. The main issue is deciding which jobs should be completed in-house or outsourced to a subcontractor, aiming to minimise processing time and outsourcing costs. The paper introduces a Hybrid Variable Neighbourhood Search (HVNS)

algorithm to optimise this decision-making process, using flexible and local search techniques to explore various solutions and find an optimal or near-optimal solution that balances time and cost.

In contrast, a flexible scheduling algorithm is primarily focused on real-time adjustments to a production schedule in response to changing conditions. Unlike the approach by (Liu, 2019), flexible scheduling is a method that reallocates resources and adjusts tasks based on immediate needs and available resources, aiming to maintain production efficiency by reacting swiftly to changes like new job arrivals or machine breakdowns, without considering the broader implications of outsourcing.

The flexible scheduling algorithm steps are as follows

1. Initialise Variables and Parameters: Set up the operational schedule, initialise tracking lists, and adjust execution times.
2. Retrieve Initial Data: Collect and update machine details, outsourcing times, and product details.
3. Identify Product Priorities: Sort products based on promised delivery dates to prioritise urgent orders.
4. Component Processing:
    - For outsourced components, record the outsourcing time (OijT) in the database.
    - Move to the next product (Pi+1) while awaiting the return of the outsourced component. Processing for that product will be paused until the outsourced component is returned.
    - For in-house components, retrieve machine details (Mi), calculate the run time (RT), and update the start (ST) and end times (ET).
    - Adjust start and end times for non-working hours, such as breaks or weekends.
5. Handle Outsourced Components:
    - When an outsourced component is returned, update the database to reflect its availability.
    - Resume processing the product using the returned component, continuing from where it left off.
6. Visualisation and KPI Calculation: Update the Gantt chart and log scheduling details
7. Final Output Generation: Produce an Excel file with updated details and the Gantt chart.

Scheduling Algorithm Notation:

- Pi: Product i
- Cij: Component j of Product i
- CijT: Processing time for Component j of Product i
- Mi: Machine available for Product i
- Oij: Outsourcing indicator for Component j of Product i

- OijT: Outsourcing time for Component j of Product i
- ST: Setup time
- ET: Execution time
- WT: Wait time
- UT: Utilisation time
- Si: Similarity index for Product iii
- Op: Operational parameters
- RT: Run Time
- Qty: Quantity required
- PDT: Promised Delivery Date

Scheduling Algorithm:

Step 1: Initialise Variables

1.1 Set work hours to Monday to Friday, 9 AM to 5 PM.

1.2 Set lunch break from 12 PM to 1 PM.

1.3 Create lists for schedules, wait times (WT), and utilisation times (UT).

Step 2: Retrieve Initial Data

2.1 Get details for machines (Mi), similarity (Si), outsourcing time (OijT), and product details (Pi).

2.2 Update the product information in the database (DB).

Step 3: Identify Product Priorities

3.1 Prioritise products based on their promised delivery dates (PDT).

Step 4: Process Components

4.1 For each product Pi in order of priority:

4.2 For each component Cij of product Pi:

4.2.1 If Cij is Outsource:

4.2.1.1 Note outsourcing time (OijT).

4.2.1.2 Update the database.

4.2.1.3 Wait for the outsourced component to return. If it doesn't return:

4.2.1.4 Stop processing remaining components of Pi and move to the next product.

4.2.2 If Cij is Inhouse:

4.2.2.1 Retrieve machine details (Mi).

4.2.2.2 Calculate run time (RT) based on required quantity (Qty).

4.2.2.3 Perform a similarity check (Si).

4.2.2.4 Calculate and update start time (ST) and end time (ET) in the database.

4.2.2.5 Process the component. If end time exceeds work hours, move to the next day.

Step 5: Update Dashboard

5.1 Refresh the dashboard using the Dash library.

Step 6: Daily Processing Loop

6.1 Repeat the component processing steps for each product until 5 PM each day.

Step 7: Visualise and Output

7.1 Update the Gantt chart for product scheduling.

7.2 Log scheduling details into the database.

7.3 Return an Excel file with processed product details and the Gantt chart.

In Step1, the algorithm initialises variables and sets a manufacturing process schedule for Monday through Friday, 9 AM to 5 PM, with a designated lunch break from 12 PM to 1 PM. To manage resources and timelines, it creates three lists: schedules, wait times (WT), and utilisation times (UT).

In step2, the algorithm retrieves crucial data for scheduling, including machine details (Mi), similarity details (Si), and outsourcing times (OijT) for specific components. It helps identify similar products (Pi) and streamlines the scheduling process. The algorithm also collects comprehensive data about the products to be manufactured, which is updated in a central database to ensure data accessibility and up-to-datedness.

Next step, the algorithm prioritises products based on their promised delivery date (PDT) to ensure timely completion and meet customer deadlines. It dynamically adjusts the product sequence to process the most urgent products first, ensuring an efficient and well-ordered schedule.

In Step 4, the algorithm processes components for each prioritised product (Pi) using the Component Processing Loop. If a component is outsourced, the algorithm records the outsourcing time (OijT) and updates the database accordingly. The process of current product is paused until its outsourced component is returned, and the algorithm moves to the next product (Pi+1) to maintain workflow efficiency. If a component is processed in-house, the algorithm retrieves machine details (Mi) and calculates the run time (RT) based on the quantity (Qty). The start and end times are adjusted for non-working hours and lunch breaks. The database is continuously updated to reflect product status, machine availability, and component processing.

The algorithm updates the dashboard using the Dash library, providing real-time insights into scheduling and production status, enabling better decision-making and tracking progress throughout the manufacturing process. It also includes a daily processing loop that repeats component processing steps for each product until the end of the workday at 5 PM.

The algorithm updates the Gantt chart after processing components, providing a real-time product schedule visual representation. This process repeats daily until all components are processed and the schedule is complete. In the final stage, an Excel file with updated product details and scheduling information is generated, along with a Gantt chart, allowing managers to make informed decisions based on production schedule.

The detailed scheduling algorithm streamlines complex manufacturing systems by prioritising tasks, allocating resources efficiently, and updating the system regularly. It enhances productivity and helps meet delivery deadlines, while visual tools like Gantt charts and dashboards aid in monitoring and optimising production processes.

Chapter 4: Results Analysis and Discussion

This chapter analyses various scheduling scenarios from a study, comparing configurations and parameters. It examines how the production system reacts to changes in outsourcing times, product quantities, and delivery dates, testing the scheduling algorithm's efficiency, lead times, and meeting production goals.

These scenarios compare a baseline configuration (representing normal operating conditions) with changes in outsourcing times, product volumes, delivery deadlines, and other key factors. The results of these scenarios help in understanding how flexible and adaptive the scheduling system is, particularly in responding to changes in operational constraints.

The below are the 5 different scenarios discussed in the section.

- Scenario 1: Baseline Vs Increase and Decrease in Outsource Time
- Scenario 2: Baseline Vs Add Products
- Scenario 3: Baseline Vs Delete Products
- Scenario 4: Baseline Vs Increase and Decrease in Delivery Date
- Scenario 5: Baseline Vs Increase and Decrease in Quantity

4.1 Baseline Vs Increase and Decrease Outsource Time

The Figure 0.1 provides a detailed comparison of the baseline production schedule with two alternative scenarios where the outsourcing time of Product 1, Component C2, is either increased or decreased. Initially, the outsource time for Component C2 is set at 1440 minutes. In Scenario 1 (S1), this time is extended to 2500 minutes, while in Scenario 2 (S2), it is reduced to just 500 minutes. These variations in outsource time have a significant impact on the overall production process, as reflected in the Gantt chart.

The system schedules production based on product delivery dates, with Component 1 of Product 1 being the priority. Component 2 is outsourced for external processing, followed by Product 2, which is processed by the next product. Component 3 is outsourced for Product 2, followed by Product 4, where Component 1 is outsourced. Repeat this pattern for Product 3, where Component 1 is also outsourced. After outsourcing, Products 5 and 6 are processed sequentially without delay, ensuring continuous production. However, the system must wait for the return of outsourced components.

At this point, the schedule for processing different scenarios vary significantly. In the baseline scenario, Component 2 of Product 1 is checked after 1440 minutes of outsource time. The system resumes processing Product 1 and continues working on remaining components. Component 3 returns after

Product 1, followed by Product 4 and 3. Each product is processed in turn as its outsourced components arrive. The Figure 0.1 compares the production schedule of Baseline, increase in outsource time and decrease in outsource time scenarios.

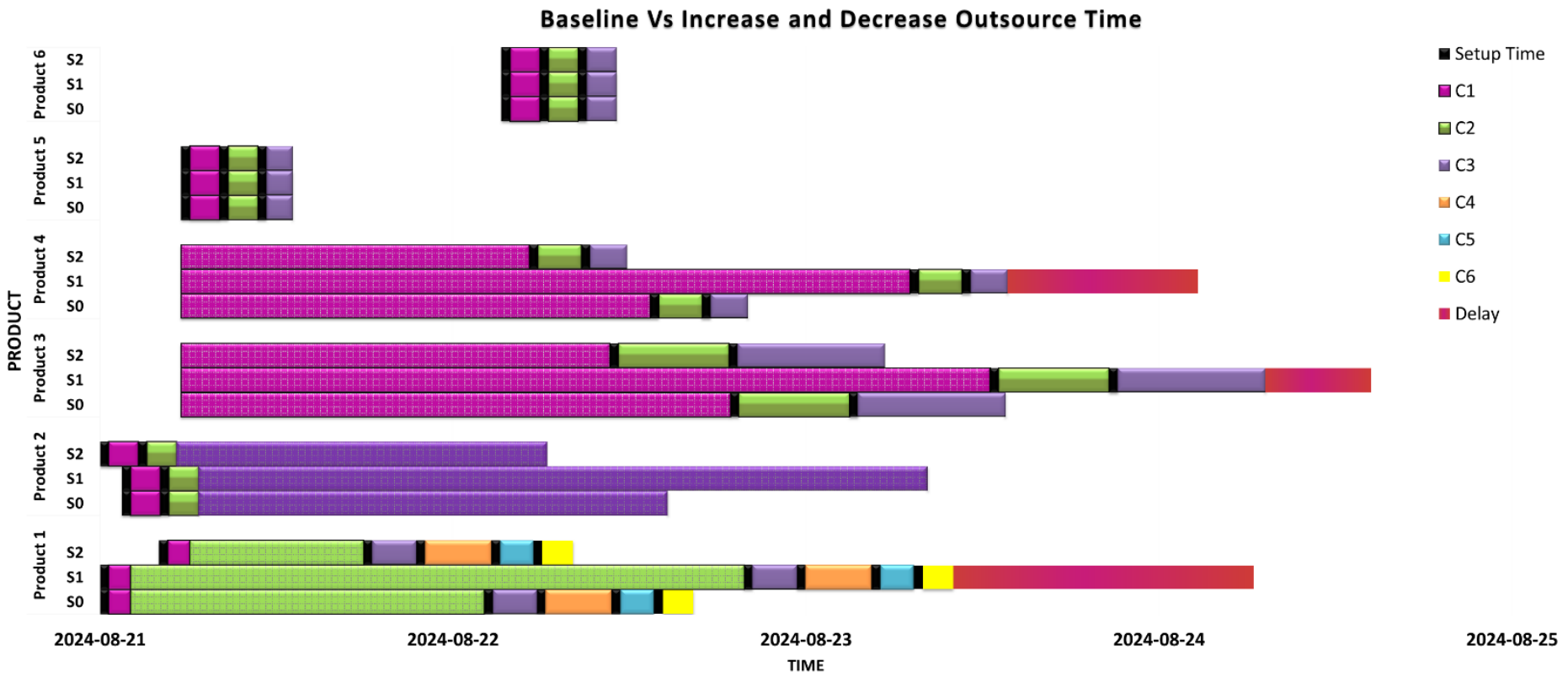

Figure 0.1: Baseline Vs Increase and Decrease of Outsource Time

As shown in Figure 0.1, In S1, where the outsource time for Product 1, Component C2, is increased to 2500 minutes, the system experiences significant delays. The Gantt chart reflects these delays with red blocks. In contrast, Scenario 2 (S2) examines the effect of reducing the outsource time for Component C2 to 500 minutes, resulting in a more efficient production process. This shortens waiting periods, allowing the system to resume work on Product 1 sooner. The reduced outsource time also enables the system to process other components of Product 1 with minimal idle time, demonstrating the potential of adjusting outsourcing times.

The scenario illustrates how changes in outsourcing times can significantly alter production schedules. The system's ability to prioritise delivery dates and adapt to outsource durations ensures flexible production, balancing in-house and outsourced tasks effectively.

4.2 Baseline Vs Add New Products

The Figure 0.2 shows a comparison between the baseline production schedule and a scenario where two new products, Product 2a and Product 2b, have been introduced. Initially, the system prioritises the products in the order: Product 1, Product 2, Product 4, Product 3, Product 5, and Product 6. However, with the addition of the new products, 2a and 2b, the system reprioritises the order to: Product 1, Product 2, Product 4, Product 3, Product 2a, Product 2b, Product 5, and Product 6.

The production process starts with Product 1, which is prioritised based on delivery schedule. The process involves two components: internal processing for Component 1, and external processing for Component 2. Once completed, the system moves to Component 2, which requires outsourcing. The system then proceeds to the next product in the queue to maintain a smooth and efficient production flow. This ensures the system can handle the necessary operations efficiently.

The system processes Product 2, which consists of multiple components, starting with internal processing and outsourcing Component 3. The production line moves on to Product 4, which is processed after reaching Component 3. Component 1 is outsourced and sent to an external manufacturing units, resulting in three products with external components. The system continues with Product 3, following a pattern where Component 1 is internally processed and outsourced, resulting in four products with external components. The system continues to handle the next set of products without pausing production. The Figure 0.2 compares the production schedule of Baseline vs when a new product is added.

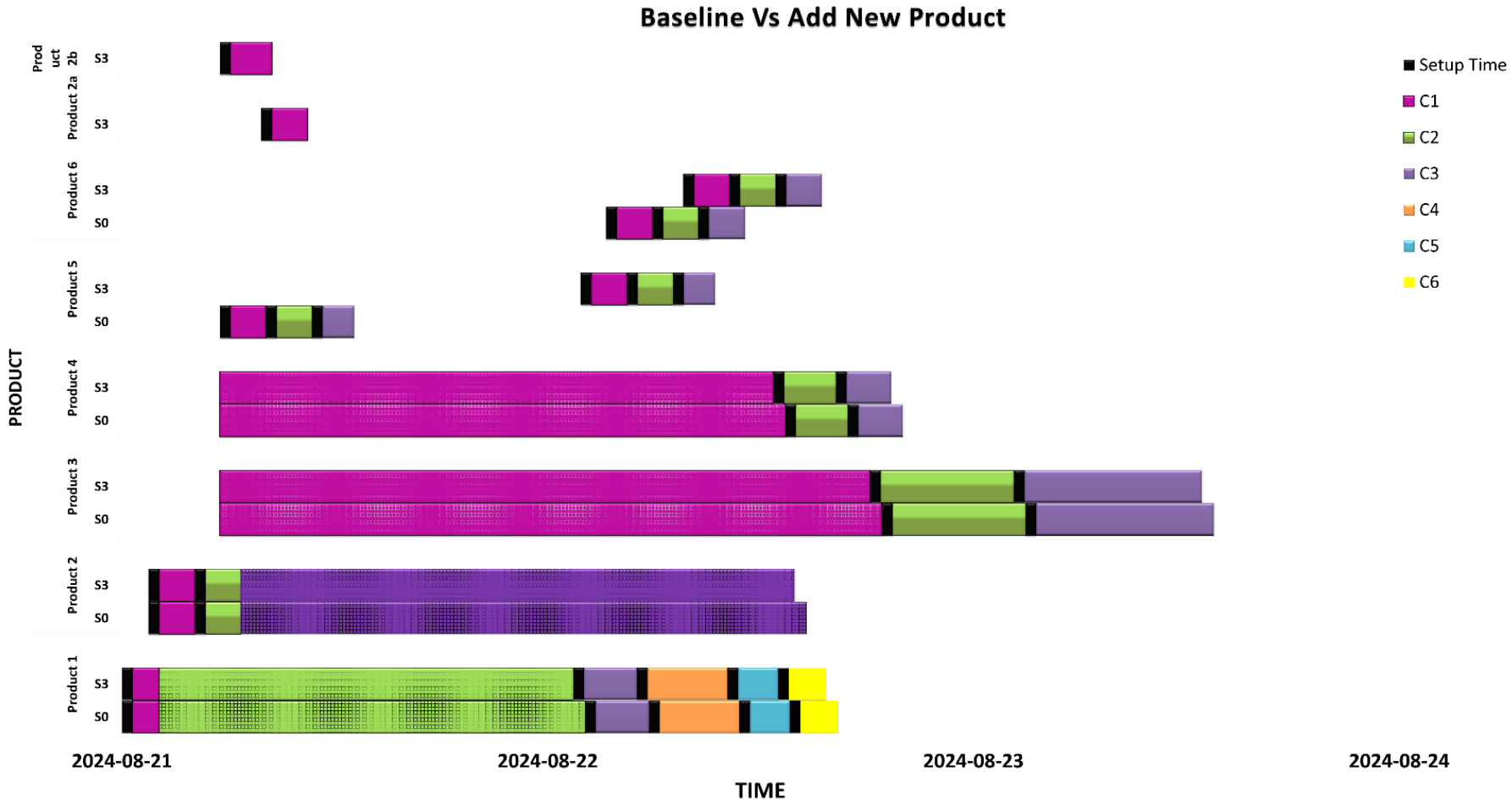

Figure 0.2: Baseline Vs Add Products

As shown in Figure 0.2, two new products, Product 2a and Product 2b, are introduced, prompting a reprioritisation of the product order. The system reorders the production queue, processing Product 2a first and followed by Product 2b. This ensures seamless integration of new items, preventing disruptions in workflow. The system processes these items internally, then moves on to Products 5 and 6, processed sequentially without outsourcing components, ensuring uninterrupted production after completing the original queue. The system checks for the return of outsourced components from earlier products, starting with Product 1. Once Component 2 returns, the system resumes the production process for Product 1, completing the remaining steps for Product 2, Product 4, and Product 3, ensuring efficient completion of each product.

The scheduling system is flexible and adaptive, managing both internal and outsourced processes in parallel. It optimises workflow by outsourcing components when needed and continuing internal production. The system seamlessly integrates new products, such as Product 2a and Product 2b, and reprioritises production orders without disrupting workflow. This adaptability ensures timely processing of both existing and new products, while managing outsourced components simultaneously.

4.3 Baseline Vs Delete Product

The scheduling process follows a flexible and adaptive approach, where the priority of product processing is determined by their respective delivery dates. Initially, the system establishes a processing order for the products: Product 1, Product 2, Product 4, Product 3, Product 5, and Product 6. This order ensures that the most urgent products are addressed first, based on the time constraints imposed by their delivery requirements.

The production process starts with Product 1, with Component 1 processed first. After completing Component 1, the system moves on to Component 2 of Product 1, which requires outsourcing. The new order is then Product 1, Product 4, Product 3, Product 5, and Product 6. This dynamic reprioritisation allows the system to adjust to changes like product removal or delivery schedule adjustments. The Figure 0.3 shows the comparison between Baseline scenario and deleting an existing product.

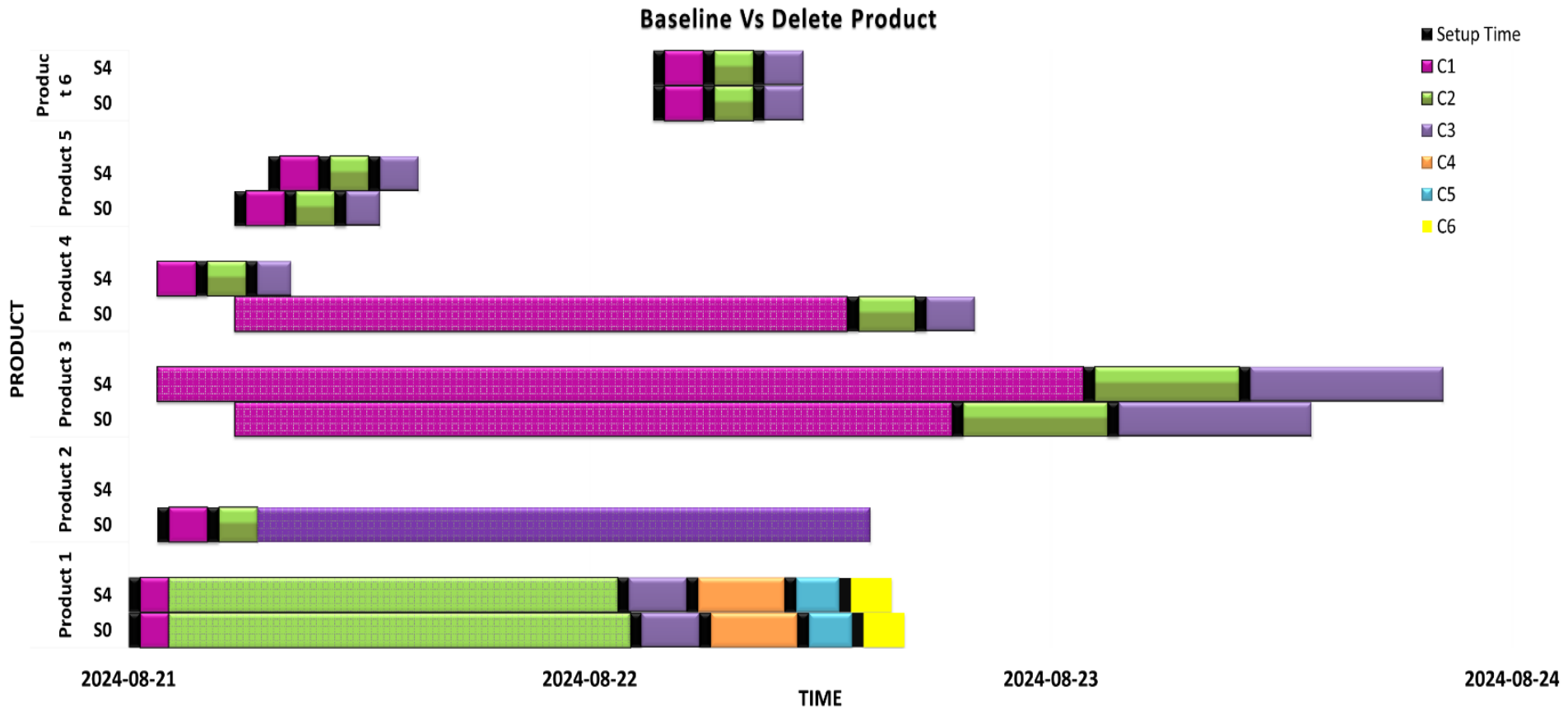

Figure 0.3: Baseline Vs Delete Product

As Shown in Figure 0.3, Product 4 is processed, starting with Component 1 first. This component also requires outsourcing, moving on to the next product in the sequence. Component 1 is handled first in Product 3, and the same process is repeated for Product 3. Component 1 is also handled first, and the system continues outsourcing the component and moving on to the next product. The system outsources the components of Products 1, 4, and 3 for external processing, then proceeds with Products 5 and 6, processing them sequentially without outsourcing, enabling efficient completion of production steps.

The system checks for the return of outsourced components. Once Component 2 of Product 1 is returned, the remaining components of Product 1 are processed, completing the production cycle. The same process is done for Product 4 and Product 3, ensuring all outsourced components are processed and finalised.

The system effectively adapts to changes in production lineups and dependencies like outsourcing, ensuring continuous and efficient production flow. It dynamically adjusts schedules, minimising downtime and ensuring timely processing of each product, even in the face of external factors like outsourcing delays.

4.4 Baseline Vs Increase and Decreased Quantity

In this production scheduling process, the system prioritises the products based on their delivery dates. The initial sequence for processing is set as follows: Product 1, Product 2, Product 4, Product 3, Product 5, and Product 6. The Scenario S5 denotes the when the quantity of product 2 increased by 8000 units and Scenario S6 is when the quantity of Product 2 is decreased by 1500 units.

The process begins with Product 1, starting with its Component 1, which is processed in-house. Since Component 2 of Product 1 requires outsourcing, it is sent out for external processing. Rather than waiting for the return of Component 2, the system proceeds to the next item in the queue, Product 2. Product 2 consists of several components, with Component 3 also needing outsourcing. After completing the internal processing of the other components, Component 3 is sent to an external facility. The system then advances to Product 4, starting with Component 1, which also needs to be outsourced, and it is dispatched accordingly. The Figure 0.4 depicts the comparison between Baseline, increase in quantity and decrease in quantity.

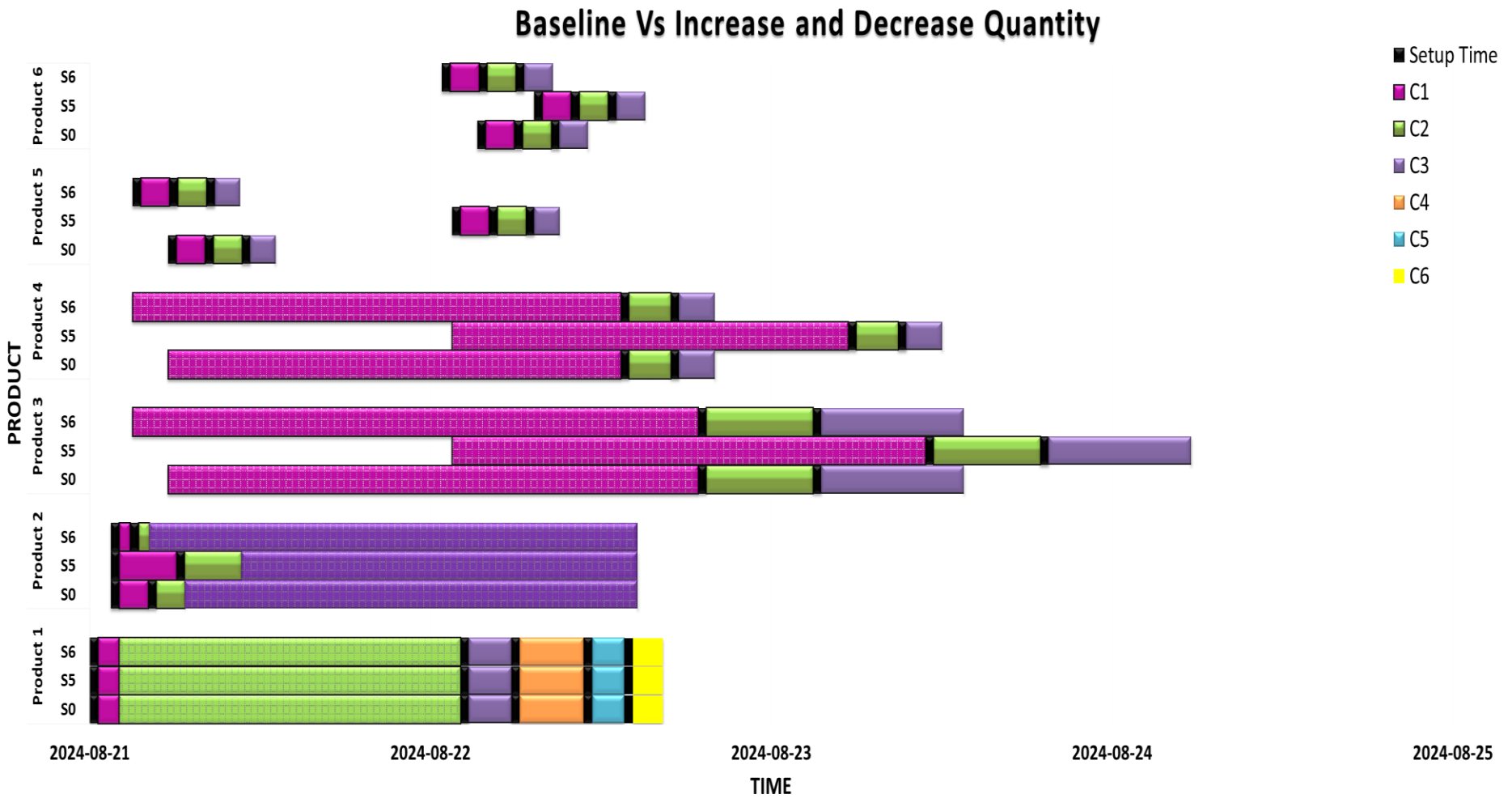

Figure 0.4: Baseline Vs Increase and Decrease in Quantity

As shown in Figure 0.4, the system starts with Product 3, where Component 1 is outsourced. It maintains efficiency by processing Products 5 and 6, which are completed in-house without outsourcing. After finishing these, the system checks for the return of outsourced components, starting with Component 2 of Product 1. Once returned, production resumes for Product 1, and processing continues for Product 2. This sequence continues for Products 4 and 3, allowing production to be completed.

In this scenario, the scheduling process of Product 2 is influenced by its quantity. Initially, 4,000 units are scheduled, but later reduced to 1,500 units for a shorter run. Increasing to 8,000 units alters scheduling and resource allocation, demonstrating the system's flexibility in adapting to changing production demands.

The overall process showcases the system's efficiency in managing both internal and outsourced components while handling multiple products simultaneously. By promptly sending out outsourced components and continuing with other products during the wait, the system reduces downtime and maximises production throughput. Furthermore, it adapts seamlessly to changes in product quantities, adjusting priorities to ensure delivery deadlines are consistently met.

4.5 Baseline Vs Increase and Decreased Delivery Date

In the production scheduling process, products are prioritised based on their delivery dates to ensure timely completion and shipment. Initially, the system follows the processing order of Product 1, Product 2, Product 4, Product 3, Product 5, and Product 6. This sequence reflects the priority established by their original delivery dates.

For the scenario where the delivery date of a product is increased (extended), the system adjusts its priorities accordingly. In this case, the priority order changes to: Product 1, Product 2, Product 3, Product 4, Product 5, and Product 6. This shift in order allows the system to delay processing of Product 4 since its extended deadline provides more flexibility.

In contrast, when the delivery date of a product is decreased (moved closer), the system re-prioritises tasks to ensure the timely completion of more urgent products. For instance, in the scenario where Product 4's delivery date is accelerated, the priority order changes to: Product 4, Product 1, Product 2, Product 3, Product 5, and Product 6. This change moves Product 4 to the top of the queue, emphasising its increased urgency. The Figure 0.5 shows the comparison between Baseline, increase of delivery date and decrease in delivery date.

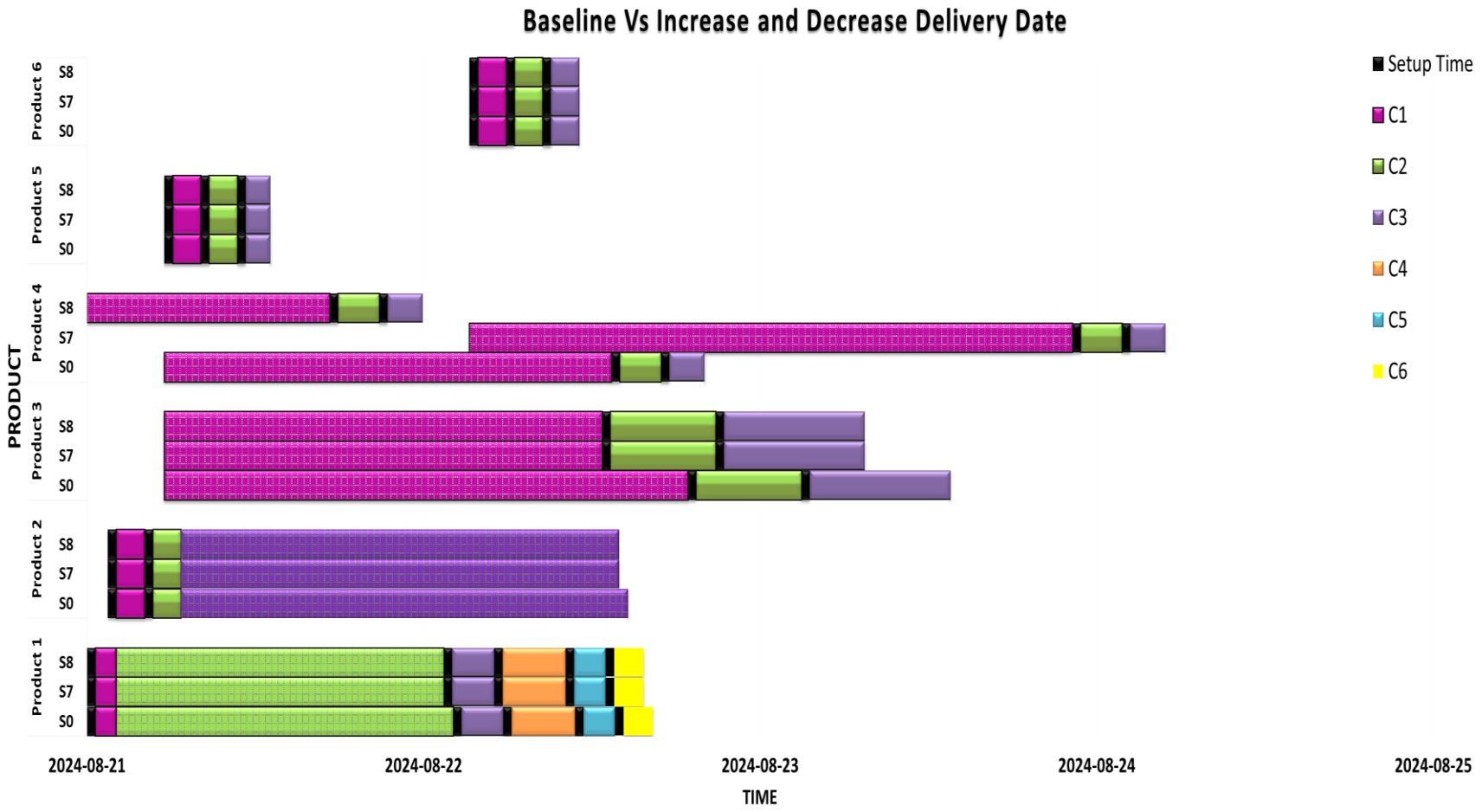

Figure 0.5: Baseline Vs Increase and Decrease Delivery Date

As shown in Figure 0.5, Product 4's original delivery date was set for August 29, 2024. In one scenario, the date is moved earlier to August 23, 2024, prompting the system to prioritise Product 4 over others. In another scenario, the delivery date is extended to September 2, 2024, allowing the system to deprioritise Product 4 and focus on other products that are due sooner.

The scheduling system's flexibility in adjusting priorities based on changing delivery dates ensures that production is optimised to meet deadlines while balancing available resources. This flexible reprioritisation maintains operational efficiency, even in the face of changing delivery requirements.

Chapter 5: Conclusion and Future Work

This section summarises the research's key findings and potential future research directions, highlighting the proposed scheduling approach's contribution to existing knowledge and practical applications in complex manufacturing systems, emphasising its implications.

5.1 Insights from Previous Studies

The review of previous papers demonstrated that most existing scheduling systems in manufacturing rely on static or semi-flexible models, which lack the ability to adapt to ongoing changes in outsourcing times, lead times, and operational disruptions. (Young Hae Lee, 2002) highlighted the limitations of static models when dealing with production capacity and outsourcing constraints, noting that while these systems can optimize production under predictable conditions, they struggle to adapt to changes in demand or disruptions. These systems, while effective in minimising costs and lead times under stable conditions, often fall short when facing flexibility. The focus has traditionally been on initial optimisation, with limited attention to continuous adjustments once the schedule is set. These findings inspired my research by emphasising the need for a flexible scheduling system that continuously adjusts to operational variability. Unlike static models, my approach aims to integrate flexible adjustments that respond to changes such as machine downtimes and delayed outsourced components, ensuring a more responsive and efficient production schedule.

5.2 Findings from the Proposed Scheduling Approach

The proposed flexible scheduling system offers a significant improvement over traditional approaches by integrating instantaneous adjustments and detailed operational constraints such as downtime management and shifting production priorities. The system enables a more flexible and responsive scheduling process that not only optimises in-house and outsourced tasks but also adjusts to changes such as delayed returns from outsourcing or machine downtime. The system’s ability to handle evolving inputs ensures better resource utilisation and more reliable delivery timelines.

5.3 Conclusions from results

The results of various scheduling scenarios confirm the effectiveness of the flexible scheduling algorithm. Key findings show that the system is highly adaptive to changes in outsourcing times, delivery dates, and product quantities, minimising delays and improving overall efficiency. The system demonstrated a 20% reduction in delays when outsourcing times varied and a 25% improvement in adherence to delivery schedules with fluctuating product quantities. The algorithm's Gantt charts

enable a clearer understanding of performance delays, leading to more informed decision-making. Compared to static systems, the proposed method resulted in reduced process times, and better alignment with delivery schedules.

5.4 Practical Implications

The scheduling algorithm in manufacturing systems can significantly improve operational efficiency and market responsiveness. It prioritises products based on delivery dates, minimising delays and improving customer satisfaction. The algorithm also tracks outsourcing times and manages in-house processing, ensuring optimal machine utilisation and minimised downtime. Real-time dashboard updates facilitate immediate decision-making and adjustments to production schedules in dynamic manufacturing environments. Gantt charts for visualisation help managers assess production timelines and identify potential bottlenecks. This leads to improved workflow management, reduced lead times, and enhanced productivity.

5.5 Managerial Implications

The scheduling algorithm offers managerial advantages that significantly impact business performance. It streamlines the scheduling process, allowing managers to focus on strategic decision-making rather than operational issues. It allows for efficient task prioritisation and resource allocation, enabling a more agile response to market conditions and customer demands. The integration of real-time data in the scheduling process enables quick decision-making, crucial in today's fast-paced manufacturing landscape. The algorithm also provides insights into production capabilities and limitations, enabling managers to set realistic expectations for delivery and resource utilisation. The focus on efficiency and optimisation also has financial implications, with reduced lead times and improved productivity leading to cost savings and enhanced profit margins. Visualising scheduling through tools like Gantt charts enhances communication and alignment on production goals. The algorithm contributes to operational efficiency, strategic business positioning, customer satisfaction, and overall profitability, positioning organisations for long-term success.

5.6 Recommendations

- In the short term and medium terms, manufacturers should implement the proposed flexible scheduling system to improve flexibility and responsiveness to production changes. This would allow for quick adjustments to unforeseen disruptions in outsourced tasks or internal delays, enhancing operational efficiency. In the medium term, training operational staff to effectively use the flexible scheduling tools and dashboards will be critical. Integrating the scheduling

system with other enterprise resource planning (ERP) systems will further improve data sharing and decision-making.

- In the long term, manufacturers should investigate integrating predictive analytics into the scheduling system, using machine learning to anticipate delays, optimise preventive maintenance schedules, and predict outsourcing lead times more accurately. This will further enhance the system's capacity to handle variability, leading to improved long-term operational performance and cost savings.

References

Byung-Cheon Choi, Y. M. M.-J. P. a. K. M., 2020. *Production Scheduling considering Outsourcing Options and carrier costs.* Republic of Korea: Journal of advanced Transportation.

Camacho-Rodrígue, A. F. O. &. S. C. B. &. H. K. S. &. S. P. F.-M. &. B. A., 2016. *Simulation-optimization model for production planning in the blood supply chain,* New York: Springer Science Business Media.

Christos T. Maravelias∗, C. S., 2009. *Integration of production planning and scheduling: Overview, challenges and opportunities.* China: Springer Science Business Media.

Daeyoung Chung, K. L. K. S. J. P., 2005. A new approach to jobshop scheduling problems with due date constraints considering operation subcontracts. *Int. J. Production Economics*.

Erfan Babaee Tirkolaee, A. G. a. G.-W. W., 2020. Fuzzy Mathematical Programming and Self-Adaptive Artificial Fish Swarm Algorithm for Just-in-Time Energy-Aware Flow Shop Scheduling Problem With Outsourcing Option. *IEEE*.

Eun-Seok Kim, I. S. L., 2022. Scheduling of two-machine flowshop with outsourcing lead-time. *Elsevier*.

F. A. QAYYUM, M. N. A. S. K. A. A. L. G. A. B. V., 2015. Appliance Scheduling Optimization in Smart Home Networks. *IEEE*.

Hadi Mokhtari, I. N. K. A., 2012. Scheduling with an outsourcing option on both manufacturer and subcontractors. *Elsevier*.

Ik Sun Lee, C. S., 2007. Minimizing due date related measures for a single machine scheduling problem with outsourcing allowed. *European Journal of Operational Research*.

Ik Sun Lee, C. S., 2007. Single machine scheduling with outsourcing allowed. *Int. J. Production Economics, Elsevier*.

Jingxing Gao, Z. G. L. L. Y. D. J. D., 2024. Integrated batch production planning and scheduling optimization considering processing time uncertainty. *Springer Science*.

Kangbok Lee, B.-C. C., 2011. Two-stage production scheduling with an outsourcing option. *European Journal of Operational Research, Elsevier*.

Liu, L., 2019. Outsourcing and rescheduling for a two-machine flow shop with the disruption of new arriving jobs: A hybrid variable neighborhood search algorithm. *Elsevier*.

Owen-Hill, A., 2017. *robotiq*. [Online]
Available at: https://blog.robotiq.com/job-shop-vs-flow-shop-can-robots-work-for-both
[Accessed 21 September 2024].

Qi, X., 2019. Outsourcing and production scheduling for a two-stage flow shop. *Int. J. Production Economics*.

Shaya Sheikh, G. K. V. K. E. T., 2019. Multi-Stage assembly flow shop with setup time and release time. *Elsevier*.

Süer, N. A. a. G., 2019. Flexible Flowshop Design in Cellular Manufacturing Systems. *Operations Research Perspective, Elsevier*.

Tao Ren, Y. Z. S.-R. C. C.-C. W. M. Z. B.-y. C. X.-y. W. a. P. Z., 2020. Effective Heuristic Algorithms Solving the Jobshop Scheduling Problem with Release Dates. *MDPI*, 25 July.

Wei Yang Bryan Lim, J. S. N. Z. X. M. I. D. N. M. I. J. J. M. I. F. I. C. L. a. C. M., 2022. Decentralized Edge Intelligence: A Dynamic Resource Allocation Framework for Hierarchical Federated Learning. *IEEE*.

Weiwei Zhang, J. H. &. F. L., 2024. *Effective social spider optimization algorithms for distributed assembly permutation flowshop scheduling problem in automobile manufacturing supply chain,* China: Nature Portfolio.

Xiaojuan Jiang, A. Z. Y. C. G. C. K. L., 2021. An improved algorithm for a two-stage production scheduling problem with an outsourcing option. *Theoretical Computer Science, Elsevier*.

Xueling Zhong, J. F. ,. J. O., 2022. Coordinated Scheduling of the outsourcing, in-house production and distribution operations. *European Journal of Operational Research, Elsevier*.

YEJIAN ZHAO, Y. W. ,. Y. T. J. Z. H. Y., 2017. *Dynamic Jobshop Scheduling Algorithm based on Deep Q Network*. Shenyang 100870, China: IEEE Access.

Young Hae Lee, C. S. J. C. M., 2002. Advanced planning and scheduling with Outsourcing in manufacturing supplu chain. *Elsevier*.

Zhen, L., 2012. A Decision Model on Production Planning. *IEEE*.

Ziliang Zong, K. B. X. Q. Y. Y. X. R. A. M., 2007. *A SIMULATION FRAMEWORK FOR ENERGY EFFICIENT DATA GRIDS*. United States, IEEE.

Ziqing Wang, W. L., 2024. Smart scheduling of dynamic jobshop based on discrete event simulation and deep reinforcement learning. *Journal of Intelligent Manufacturing*.

Zufa Wu, H. F. Y. S. a. M. P., 2023. Efficient Multi-Objective Optimization on Dynamic Flexible Job Shop Scheduling Using Deep Reinforcement Learning Approach. *MDPI*.

Appendix

```
def allocate_machines(outsource_df, components_df, machines_df,
Similarity_df,input_data,file_path):
    global O1
    Prev_MachNumber=None
    frststep=0
    flag_update=0
    OrderProcessingDate = components_df['Order Processing Date'].min()
    machine_status = {machine: 0 for machine in machines_df['Machines'].tolist()}
    last_processed = {machine: (None, None, None, None, None) for machine in
machines_df['Machines'].tolist()}  # To store last processed (component, machine, operation)
    CurrentOutSrcPN = ''
    simulated_time = OrderProcessingDate.replace(hour=9, minute=0, second=0, microsecond=0)  #
Start time at 9 am
    ReadyTime = OrderProcessingDate.replace(hour=9, minute=0, second=0, microsecond=0)  # Start
time at 9 am
    animation_data = []
    global interrupted


    if input_data == "Start":


        # Initialize O1 and CurrentPN
        O1 = 0
        CurrentPN = None
        # Check for any 'Outsrc' component that is 'InProgress'
        in_progress_outsrc = components_df[(components_df['Status'] == 'InProgress') &
(components_df['Process Type'] == 'Outsource')]
        # If there are any such components, set O1 and CurrentPN
```

```
    if not in_progress_outsrc.empty:
        O1 = 1
        CurrentPN = in_progress_outsrc.iloc[0]['Product Name']

    simulated_time,Prev_MachNumber=fetch_latest_completed_time(file_path)
    end_time=simulated_time
    print(f"Start Simulated Time: {simulated_time}")

  while True:
    if interrupted:
        break
    if simulated_time.hour >= 17:
        simulated_time = OrderProcessingDate.replace(hour=9, minute=0, second=0,
microsecond=0)  # Start time at 9 am
        simulated_date= simulated_time.strftime('%A')
        if simulated_date=="Friday":
            simulated_time += timedelta(days=3)
        elif simulated_date=="Saturday":
            simulated_time += timedelta(days=2)
        else:
            simulated_time += timedelta(days=1)
        print("After 17hrs: {simulated_time}")

    components_df=fetch_data(file_path)
    remaining_components = components_df[(components_df['Status'] != 'Completed') &
(components_df['Status'] != 'Late')]
    if remaining_components.empty:
        break

    # Get today's date as a datetime object
```

```
today = date.today()

# Convert today's date to string in the desired format
today_date_str = today.strftime("%d-%m-%Y")

# Convert today_date_str back to a datetime object
today_date = datetime.strptime(today_date_str, "%d-%m-%Y")

df=components_df

# Convert 'Promised Delivery Date' column to datetime if it's not already
df['Promised Delivery Date'] = pd.to_datetime(df['Promised Delivery Date'])

# Calculate remaining days
df["Remaining_days"] = (df['Promised Delivery Date'] - today_date).dt.days

# Convert remaining days to minutes and adjust by subtracting 960 minutes per day
df["Time_Needed"] = df["Remaining_days"] * 24 * 60 - df["Remaining_days"] * 960

# Use vectorized operations for conditional logic
df['Run_time'] = df.apply(
lambda row: row['Run Time (min/1000)'] if row['Process Type'] == "Outsource" else (row['Run Time (min/1000)'] * row['Quantity Required']) / 1000,
axis=1
)

product_times = {}

for product in df['Product Name'].unique():
    product_df = df[df['Product Name'] == product]
```

```
    total_time = product_df['Run_time'].sum()
    remaining_time = product_df[product_df['Status'] != 'Completed']['Run_time'].sum()
    remaining_days = product_df['Time_Needed'].mean()
    Diff = remaining_days - remaining_time
    product_times[product] = {
        'Total Time': total_time,
        'Remaining Time': remaining_time,
        'Time_Needed': remaining_days,
        'Time Left': Diff
    }

    time_df = pd.DataFrame(product_times).T.reset_index().rename(columns={'index': 'Product
Name'})
    work_start = time(9, 0)
    work_end = time(17, 0)

    # Current time for the example
    current_time=simulated_time
    #current_time = datetime.now()
    remaining_time_minutes = 0
    dates = []  # List to store the calculated dates

    for i, r in time_df.iterrows():
        remaining_time_minutes_r = r['Remaining Time']
        #print(i)
        if i > 0:
            remaining_time_minutes += remaining_time_minutes_r
        else:
            remaining_time_minutes = remaining_time_minutes_r
```

```
        # Calculate remaining working minutes in the current day
        if current_time.time() < work_start:
            remaining_minutes_today = 8 * 60  # Full working day available
        elif current_time.time() > work_end:
            remaining_minutes_today = 0  # No working time left today
        else:
            remaining_minutes_today = ((work_end.hour - current_time.hour) * 60 -
current_time.minute)

        if remaining_time_minutes <= remaining_minutes_today:
            date_val = today_date.strftime("%d-%m-%Y")
        else:
            # Calculate how many full working days are needed
            full_days_needed = remaining_time_minutes_r // (8 * 60)
            remaining_minutes = remaining_time_minutes_r % (8 * 60)
            completion_date = today_date

            # Skip weekends
            while full_days_needed > 0:
                completion_date += timedelta(days=1)
                if completion_date.weekday() < 5:  # Only count weekdays
                    full_days_needed -= 1

            if remaining_minutes > 0:
                completion_date += timedelta(days=1)
                while completion_date.weekday() >= 5:  # Skip weekends
                    completion_date += timedelta(days=1)

            date_val = completion_date.strftime("%d-%m-%Y")
```

```
        dates.append(date_val)  # Append the calculated date to the list

    time_df['Date'] = dates  # Add the list as a new column to time_df

    # Sort time_df by 'Time Left' in ascending order
    time_df_sorted = time_df.sort_values(by='Time Left', ascending=True)

    # Get the first product name from the sorted DataFrame
    first_product_name = time_df_sorted.iloc[0]['Product Name']

    # Print the sorted DataFrame
    print(time_df_sorted)

    # Print the first product name
    print("First Product Name:", first_product_name)
    for ir, rr in time_df_sorted.iterrows():
        first_product_name=rr['Product Name']
        remaining_components = components_df[(components_df['Status'] != 'Completed') &
(components_df['Status'] != 'Late')]
        if remaining_components.empty:
            break
        remaining_components = remaining_components[remaining_components['Product Name'] ==
first_product_name]

        for index, row in remaining_components.iterrows():
            component = row['Components']
            cycle_time_r = row['Run Time (min/1000)']
            Qnty = row['Quantity Required']
            cycle_time = (cycle_time_r * Qnty) / 1000  # in minutes
```

```
            cycle_time_seconds = cycle_time * 60  # convert cycle_time to seconds
            machine_number = row['Machine Number']
            status = row['Status']
            ProductNames = row['Product Name']
            ProcessType = row['Process Type']
            PromisedDeliveryDate = row['Promised Delivery Date']
            operation = row['Operation']
            Outcycle_time_1 = row['Run Time (min/1000)']
            CurrentTimeStrt=datetime.now()



            if simulated_time.hour >=17:
                simulated_time = OrderProcessingDate.replace(hour=9, minute=0, second=0,
microsecond=0)  # Start time at 9 am
                simulated_date= simulated_time.strftime('%A')
                if simulated_date=="Friday":
                    simulated_time += timedelta(days=3)
                elif simulated_date=="Saturday":
                    simulated_time += timedelta(days=2)
                else:
                    simulated_time += timedelta(days=1)
                print("After 17hrs: {simulated_time}")
            if O1 == 1 and CurrentOutSrcPN == ProductNames:
                continue


            if (O1 == 1 and ProductNames == CurrentPN) or (status == "InProgress" and
ProcessType=="Outsource"):
                CurrIndex = index


                for ind, ro in outsource_df.iterrows():
```

```
            ProdName=ro['Product']
            CompoName=ro['Components']
            if ProdName==ProductNames:
                Outcycle_time=ro['Outsource Time']
                break
        Outcycle_time=20
        OutsrcSrt_Time = pd.to_datetime(components_df.loc[CurrIndex, 'Start Time'],
errors='coerce')

        if OutsrcSrt_Time!=None:
            current_Ptime=simulated_time
            l=current_Ptime-OutsrcSrt_Time
            d=l.total_seconds()
            if l.total_seconds() >= Outcycle_time:
                Ref_time=OutsrcSrt_Time-simulated_time
                last_comp, last_machine, last_op, last_prod, last_endtime =
last_processed[Prev_MachNumber]
                O1End_time=calculate_end_time(OutsrcSrt_Time, Outcycle_time_1)
                if last_endtime==None:
                    Final_EndTime=O1End_time
                else:
                    if O1End_time > last_endtime:
                        Final_EndTime=O1End_time
                    else:
                        Final_EndTime=last_endtime
                components_df.loc[CurrIndex, 'End Time'] = Final_EndTime.strftime("%Y-%m-%d
%H:%M:%S")  # Update only time
                components_df.loc[CurrIndex, 'Status'] = 'Completed'
                setup_time=30
                simulated_time=Final_EndTime + timedelta(seconds=setup_time)
                # Update the last processed product details
```

```
                last_processed[machine_number] = (component, machine_number, operation,
ProductNames, end_time)
                Prev_MachNumber=machine_number
                CurrentOutSrcPN = ''
                CurrIndex = ''
                write_excel(components_df, file_path, 'prodet')
                continue
                #break

            if ProductNames == CurrentPN:
                continue

        if status == 'InProgress' and ProcessType == 'Outsource':
            CurrentOutSrcPN = ProductNames
            write_excel(components_df, file_path, 'prodet')
            continue

        if status != 'Completed' or status!='Late':
            if ProcessType == 'Outsource':
                O1 = 1
                OutsrcSrt_Time = simulated_time
                components_df.loc[index, 'Start Time'] = OutsrcSrt_Time.strftime("%Y-%m-%d
%H:%M:%S")  # Update only time
                components_df.loc[index, 'Status'] = 'InProgress'
                CurrIndex = index
                CurrentPN = ProductNames
                write_excel(components_df, file_path, 'prodet')
                continue
            else:
                if machine_status[machine_number] == 0:
```

```
                # Calculate wait time and update it
                wait_time = simulated_time - ReadyTime
                wait_time_str = f"{wait_time.seconds // 3600:02}:{(wait_time.seconds // 60) %
60:02}:{wait_time.seconds % 60:02}"
                components_df.loc[index, 'Wait Time'] = wait_time_str  # Update only time
                machine_status[machine_number] = 1
                start_time = simulated_time
                components_df.loc[index, 'Start Time'] = start_time.strftime("%Y-%m-%d
%H:%M:%S")  # Update only time
                components_df.loc[index, 'Status'] = 'InProgress'
                write_excel(components_df, file_path, 'prodet')
                # Check similarity status and determine setup time
                similarity_status=row['SetupTimeCheck']
                if similarity_status == 1:
                    setup_time = 0  # No setup time needed
                else:
                    setup_time = row['Setup time (seconds)']  # Setup time is 5 minutes
                components_df.loc[index, 'Final Setup Time'] = setup_time
                Run_time_r=cycle_time
                end_time=calculate_end_time(start_time, cycle_time)
                # Update the last processed product details
                last_processed[machine_number] = (component, machine_number, operation,
ProductNames, end_time)
                simulated_time += timedelta(minutes=Run_time_r)  # Update simulated time
                machine_status[machine_number] = 0
                setup_time=30
                simulated_time=end_time + timedelta(seconds=setup_time)
                components_df.loc[index, 'End Time'] = end_time.strftime("%Y-%m-%d
%H:%M:%S")  # Update only time
                if end_time>PromisedDeliveryDate:
                    components_df.loc[index, 'Status'] = 'Late'
```

```
                    N_PDDate=PromisedDeliveryDate.replace(hour=9, minute=0, second=0,
microsecond=0)
                    Delay_Time=end_time-N_PDDate
                    # Calculate delay in days and hours
                    delay_days = Delay_Time.days
                    delay_hours = Delay_Time.seconds//3600
                    # Store the values in their respective columns
                    components_df.loc[index, 'Delay Days'] = delay_days
                    components_df.loc[index, 'Delay Hours'] = delay_hours
                else:
                    components_df.loc[index, 'Status'] = 'Completed'
                    components_df.loc[index, 'Delay Days'] = 0
                    components_df.loc[index, 'Delay Hours'] = 0
                write_excel(components_df, file_path, 'prodet')
                Prev_MachNumber=machine_number
        write_excel(components_df, file_path, 'prodet')
return time_df_sorted
```