Communications in Mathematics and Applications
Vol. 16, No. 2, pp. 501–516, 2025
ISSN 0975-8607 (online); 0976-5905 (print)
Published by RGN Publications
DOI: 10.26713/cma.v16i2.3095

http://www.rgnpublications.com

Research Article

Dynamic Queueing Model for Smart Manufacturing: A Priority-Based Performance Study

Balveer Saini*, Dharamender Singh and Kailash Chand Sharma

Department of Mathematics, M.S.J. Govt. P. G. College (*affiliated to Maharaja Surajmal Brij University*), *Bharatpur 321001, Rajasthan, India*
***Corresponding author:** veer.coiaf786@gmail.com

Received: March 20, 2025 **Revised:** May 25, 2025 **Accepted:** June 8, 2025

Abstract. In this paper, we present a *Dynamic Priority Queueing Model* (DPQM), which effectively solves the problems of traditional queueing models in complex priority scheduling situations in manufacturing systems. The DPQ paradigm uses priority functions, time-based queueing, and advanced scheduling algorithms to schedule tasks elegantly. The priority function swiftly prioritizes tasks based on urgency, relevance, and resource demands, adapting to changing circumstances. The suggested $M(t)/G(t)/c$ queueing model extends the $M/G/c$ model for real-world applications with time-dependent task arrivals and service activities. The model validation and implementation process enable efficient calculations of priority values, lead time, tardiness, usage, and efficiency. The average lead time is 179 minutes, tardiness is 106 minutes, and resource utilization is 100%. The scheduling efficiency improves by 17.6% by effectively sequencing activities to match system priorities. The validation shows that the model prioritizes work and properly maps changing parameters compared to traditional queueing systems. The results indicate that improving adaptive scheduling, real-time feedback systems, and using multiple servers can maximize resource utilization to generate high throughput and eliminate work delays. This approach serves as a robust and adaptable strategy for dynamic scheduling, thoughtfully considering priorities within complex manufacturing environments.

Keywords. Dynamic Priority Queueing Model (DPQM), Priority function, Scheduling algorithm, Manufacturing systems, Resource utilization, Lead time

Mathematics Subject Classification (2020). 60K30, 90B22, 90B30, 90B50

1. Introduction

The manufacturing sector is the foundation of every nation and its economy. It provides millions of job opportunities and fundamental requirements for human beings. Any innovation in the productivity of the manufacturing system significantly enhances the nation's competitiveness. In a manufacturing system, there are numerous interactions between staff, data, supplies, and equipment. The effectiveness of this sector requires an analysis of several performance measures. Performance measures are essential for both decision-making and control in the production system. It is important to deal with the onsite production activity effectively.

The theory of queueing offers a quantitative framework for researching and simulating the operations of production systems. Analyzing waiting line behavior and component interactions, queueing theory provides an empirical framework for finding areas for improvement. These aspects hold significant importance in manufacturing systems. Aalto and Scully [1] studied generic service time distributions and occupancy distributions for an open network with infinite server queues and non-homogeneous Poisson batch arrivals. While the study presented by Agrawal and Mohanty [2] developed a multi-attribute decision-making method for uncertain and linguistically ambiguous situations. The composition contains non-uniform, non-regular, or random language sentences in hesitant, ambiguous terminology. In their consecutive studies, Amjath *et al*. [3, 4] presented a comprehensive analysis and described a way to use finite queueing networks to improve performance and find the best topology. The study's conclusions present opportunities to increase industrial systems' effectiveness and have a significant influence on decision-making. In their work, Chen and Tiong [5] have explored manufacturing facilities that employ systems based on automated guided vehicles to enhance flow production. Although these studies addressed important issues, descriptive categories, and challenges, they failed to sufficiently explore any platforms, such as dynamic priority scheduling, that rely on real-time assessments. Further, we present a brief overview of existing literature separately.

The current literature on queueing theory in manufacturing mostly focuses on traditional models. Most researchers tend to overlook dynamic priority scheduling in favor of static models. In modern manufacturing environments, priority scheduling plays a significant role in managing real-world complexities. This study addresses the research gap by developing and applying a dynamic priority queueing model to analyze and optimize manufacturing performance measures. The DPQM uses real-time flexibility and optimization to outperform classic queueing models. It helps contemporary, automated manufacturing systems improve efficiency and decrease bottlenecks. This novel manufacturing technology shows how well the DPQM can capture the work environment and deliver all the data needed for smooth operations.

1.1 The Research Objectives and Study Framework

The following are the specific objectives this study intends to achieve:

- To create a DPQ model that captures the intricate interactions between arrival procedures, service times, and priority scheduling in manufacturing.
- To investigate the ways in which significant production performance measures are affected by priority scheduling.

- To evaluate the effectiveness of the DPQ model in improving manufacturing performance indicators.
- We aim to illustrate the advantages and drawbacks of each strategy by contrasting the suggested model with conventional queueing models.

This paper can be summarized in the following manner: In Section 1 of the introduction, we have discussed the research objectives and framework of the study. Section 2 presents a comprehensive review of existing literature pertinent to our current study. Section 3 highlighted the gaps in existing research and outlined the advantages of the DPQ model compared to current models. Section 4 presents a complete methodology with a comprehensive mathematical description of the theoretical framework underlying the DPQ model. Section 5 presents the model validation and implementation process of the proposed DPQM framework. Section 6 described the performance evaluation of the DPQ model using graphical and simulation analysis. Section 7 describes the scalability, limitations, and directions for future research on the DPQ model in the manufacturing system. Section 8 summarizes the overall findings of the proposed work.

2. Literature Background

The relationship between Queueing theory and manufacturing systems has been going on for decades. Large-scale analysis and optimization of manufacturing systems have utilized queueing theory. In this section, we provide a detailed analysis of the literature, which is as follows:

In the manufacturing context, Gongshan *et al*. [7] proposed that the integration of process analysis and queueing theory might effectively tackle challenges related to elevated costs, reduced efficiency, and extended service durations. Laxmi and George [9] calculated transient state probabilities using the probability generating function, Rouche's theorem, and Laplace transform. This research filled the gap in evaluating the batch queue with required and optional services, revealing exciting health care system possibilities, while Mehra and Taylor [10] examined the overall service time and occupancy distributions within an open network characterized by infinite server queues and a non-homogeneous Poisson process for multivariate batch arrivals. Murdapa *et al*. [10] investigated the emission variable single-stage $M/M/1$ queueing model. The model evolved from a conventional single-stage queue framework to one that integrated emission variables to ascertain the quantity of manufacturing lots allowed in each period.

To achieve optimal supply, reduce production costs, and ascertain the appropriate scale for the production department, Rece *et al*. [11] introduced an innovative approach utilizing queueing theory models. While the studies presented by Saini *et al*. [12, 13] were conducted on the application of queueing theory in the manufacturing sector. These studies have demonstrated that queueing theory can assist industries in increasing production, reducing wait times, and conserving resources. However, these studies do not provide sufficient insights into systems such as dynamic priority scheduling, which depends on real-time performance assessments.

According to Selvamuthu and Kapoor [6], the time-dependent solution of a fluid model influenced by an $M/M/1$ queue can be elegantly derived using a straightforward probability technique. To define the essence of daily life, Ulku *et al*. [14] explored the connection between

waiting times and ensuing purchase decisions. This investigation delved into the impact of management strategies typically utilized by organizations to enhance the experience of customer waiting times. Vorhölter *et al*. [15] have delivered a comprehensive examination of the current landscape of mathematical modeling in German-speaking areas. Mathematical modeling elegantly enhances various projects, esteemed classes, and refined educational settings. However, these studies did not fully explore the intricacies of real-time evaluation systems, such as dynamic priority scheduling.

The proposed technique effectively resolves these problems using dynamic priority scheduling to make the most use of production resources. This improves workforce planning by making it both flexible and strong. This method is designed to be flexible so that it can handle the complex and changing demands of patients and rules for providing services.

3. Gaps in Current Research and DPQM Model

This comparison in Table 1 shows the gaps in current research and the benefits of the Dynamic Priority Queueing Model over existing models:

Table 1. Existing queueing models vs. DPQM model

Feature/Criteria	Existing Queueing Models (FIFO, LIFO, Priority)	Dynamic Priority Queueing Model
Priority Handling	Fixed priority or FIFO/LIFO rules	Adjusts priorities depending on system state
Flexibility	Limited scheduling flexibility	High adaptability to real-time situations
Response to System Changes	Stable priorities slow progress	Fast reaction to traffic, machine failures, and urgent tasks
Bottleneck Management	May not effectively fix difficulties	Prioritizes tasks to reduce bottlenecks
Waiting Time Optimization	Sometimes causes long wait times for jobs	Efficiently balances waiting time
Throughput Improvement	Improve somewhat depending on model	Optimizing work orders boosts throughput
Resource Utilization	Variable demand may underutilize resources	Adjusts work priorities for efficient resource use
Scalability	Complexity issues in huge systems	Suitable for large, dynamic industrial environments

4. Methodology

4.1 Dynamic Priority Queueing Model

This study proposed a dynamic priority queueing framework that optimizes resource management and system efficiency using queueing theory and priority scheduling. As a queueing system, the DPQ model optimizes large frameworks with different task priorities. The three main components of the DPQ model are scheduling, queueing, and priority. Dynamically

sorting priorities for the priority function depends on task importance, urgency, and resource availability. Queue data structure management and smooth flow are based on the queueing theory. Scheduling helps distribute limited resources by placing queued jobs in the order of importance.

Healthcare, industrial processes, networking, and finance are just a few of the numerous fields in which the DPQ model has an impact. In healthcare, DPQ assists in scheduling patients, allocating resources, and building emergency response systems. In a manufacturing system, the DPQ model manages inventories and sets task priorities and production processes. Networkers utilize this model to track traffic, service quality, and packet scheduling. The DPQ method makes things more responsive, increases throughput, makes things more flexible, and allocates resources better. It improves the performance and works in a variety of situations.

4.2 Mathematical Formulation of the DPQ Model

In this section, we provide a mathematical formulation of the proposed dynamic priority queueing model that combines queueing systems with priority-scheduling techniques. The DPQ model is defined using three key elements:

- *Priority Function*: This determines the priority value for each task, depending on their urgency, importance, and resource requirements.
- *Queueing System*: The queueing system helps manage the queue data structure and enables seamless flow.
- *Scheduling Algorithm*: The scheduling algorithm helps to determine the task priority in the queue to allocate the available resources.

The DPQ model in manufacturing can be expressed mathematically as follows:

Notations

$\lambda(t)$	:	Time-dependent arrival rate of jobs
$\rho(t)$	:	Production Line Utilization
$G_j(t)$	:	Time-dependent service time distribution for job j
$F_j(t)$	:	Cumulative distribution function (CDF) of $G_j(t)$
$\mu_j(t)$	:	Service rate for job j at time t inverse of expected service time
c	:	Number of parallel servers (machines, processors, etc.)
$P(j,t)$	:	Priority value for job j at time t, determined via the priority function
$Q(t)$	:	Set of jobs in the queue at time t
$L(t)$	:	Average Lead Time
$T(t)$	:	Average Tardiness
$\eta(t)$	:	Priority Scheduling Efficiency

Assumptions

- *Arrival Process*: Jobs arrive following a non-stationary Poisson process, $M(t)$, where $\lambda(t)$ changes with time due to external conditions (e.g., peak vs. off-peak hours).
- *Service Process*: The service time follows a general time-dependent distribution $G(t)$. Each job j has a unique $G_j(t)$, based on its type or complexity.

- *Servers*: The system has $c \geq 1$ identical or heterogeneous servers, each capable of serving one job at a time.
- *Priority-Aware Selection*: At each decision epoch t, the scheduling algorithm selects the highest-priority job from the queue for service.

4.2.1 Priority Function

These priorities directly affect the sorting and scheduling of queues at each t, which changes all the metrics shown above in real time. The following function can be used to obtain the priority value for each job:

$$P(j,t) = \alpha, \tag{4.1}$$

where

$P(j,t) =$ Priority of job j at time t,

$U_j =$ Urgency of job j $(0 \leq U_j \leq 1)$,

$D_j =$ Deadline of job j $(0 \leq D_j \leq 1)$,

$C_j =$ Criticality of job j $(0 \leq C_j \leq 1)$,

$f(t) =$ External factor (e.g., system congestion),

$\alpha, \beta, \gamma, \delta =$ Weights $(0 \leq \alpha, \beta, \gamma, \delta \leq 1)$.

The priority function uses four components: urgency, deadline, criticality, and time. The weight $(\alpha, \beta, \gamma, \delta)$ of each element indicates its importance. This may range from 0 to 1. External circumstances such as the availability of resources or changes in the system's goals may cause these weights to change over time. One example is in the manufacturing industry, where the importance of urgency (U_j) may grow during busy times, but the importance of deadline (D_j) tends to grow as the deadline gets closer.

For example, if a production system is too busy, it could be smart to raise the importance of urgency so that current tasks take precedence over those that are less important. However, in less urgent circumstances, importance may grow.

4.2.2 Queueing Model: $M(t)/G(t)/c$

The proposed $M(t)/G(t)/c$ queueing model is an extension of the traditional $M/G/c$ queueing model. This approach can handle real-world situations when jobs occur at various times and service procedures are altered. In the model, there is a non-study Poisson arrival $M(t)$, and the arrival rate $\lambda(t)$ fluctuates with time. $G(t)$ represents a service process that evolves over time, and the model has servers with limitless capacities (c). This model may be applied in many areas, such as healthcare systems where patients come in at various times, industrial systems where production rates depend on time, transportation systems, and telecommunication networks.

The proposed system assumes that the service time follows an exponential distribution with mean service rate $\mu(t)$. However, in other instances, this could need to be changed to allow for more complex service time distributions, such as normal and log-normal. This approach allows individuals to simulate tasks that require varying amounts of time and effort in practical settings, such as in manufacturing systems, where the duration needed to service a machine can change based on its operational efficiency.

4.2.3 Performance Metrics

The DPQ model is evaluated on four key performance indicators (KPIs):

A. Average Lead Time ($L(t)$): Expected time a job or task spends in the system (waiting + service):

$$L(t) = E[W_j(t)] + E[S_j(t)], \tag{4.2}$$

where $W_j(t)$ is the waiting time of job j at time t and $S_j(t)$ is the service time of job j.

In simulation or real data, expected time spent in the system is given by

$$L(t) = \frac{1}{N}\sum_{j=1}^{N}(C_j - A_j), \tag{4.3}$$

where C_j is the completion time of job j and A_j is the arrival time of job j.

B. Average Tardiness ($T(t)$): This measures how much delay jobs experience beyond their deadlines. Let D_j be the deadline of job j; then the expected time past deadline is given by

$$T(t) = \frac{1}{N}\sum_{j=1}^{N}\max(0, C_j - D_j). \tag{4.4}$$

C. Production Line Utilization: This reflects the ratio of active service time to total time available and is given by

$$U(t) = \frac{\text{Total Service Time Used}}{\text{Total Time Available by Servers}} = \frac{\lambda(t)}{c\cdot\mu(t)}. \tag{4.5}$$

Alternatively, using busy indicators:

$$U(t) = \frac{1}{c}\sum_{i=1}^{c} B_i(t). \tag{4.6}$$

D. Priority Scheduling Efficiency (PSE): Measures how efficiently the system reduces idle time via priority scheduling:

$$\text{PSE}(t) = 1 - \frac{\text{Observed Idle Time}}{\text{Potential Idle Time without Priority}} = \frac{\rho(t)}{\rho'(t)}, \tag{4.7}$$

where $\rho(t)$ is the observed utilization under DPQ and $\rho'(t)$ is the baseline utilization under traditional FCFS or random scheduling.

4.2.4 Scheduling Algorithm and Queue Management

The scheduling algorithm selects the highest-priority task in the queue for resource allocation. In case of tie-breaking (e.g., two tasks with identical priority values), a secondary criterion such as task arrival time or job type (e.g., critical vs. non-critical) can be used.

For example, the system may use a first-come-first-served policy for tasks with identical priority values. However, the highest priority task is always processed first, ensuring the most urgent tasks are handled with priority.

Thus, in this instance, the scheduling algorithm can be outlined in the subsequent steps as illustrated below and in Figure 1.

- The priority function $P(j,t)$ assigns the tasks according to their priorities.
- The highest priority task is selected by the production line at each decision cycle.
- The priority function determines how priority changes over time.

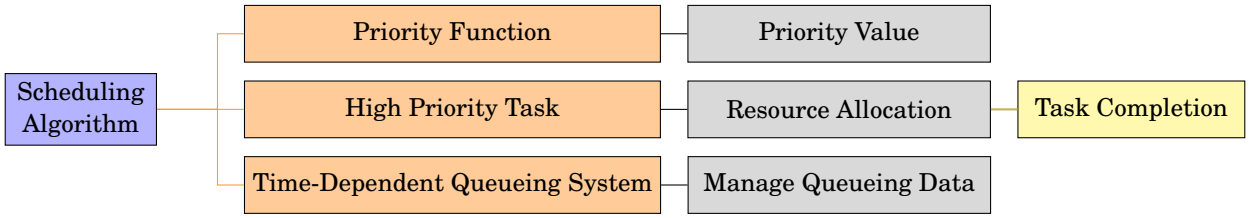

Figure 1. Flow chart of scheduling algorithm

4.2.5 Real-Time Adjustments

The DPQ model elegantly modifies the priority function through the seamless integration of real-time data inputs. If new time-sensitive conditions change the relevance (C_j) of a task, the model will gracefully adjust the priority using revised weight calculations. Furthermore, the system elegantly fine-tunes task priorities through advanced feedback mechanisms, considering the resources at hand and the present state of the ongoing project. When a patient's condition declines, the priority function in the healthcare sector seamlessly adapts to signify the increasing urgency (U_j), leading to a refined arrangement of the task's queue position.

4.2.6 Scalability of the Model

Although load balancing and parallel task processing are used to alleviate performance constraints, the DPQ paradigm remains successful as the number of tasks or resources increases. It adapts its queue management approaches or uses approximations in situations when accurate priority calculations would take a lot of processing resources to effectively handle large workloads. In large-scale production workloads, the system may be rearranged to favor high-demand equipment, which will result in increased efficiency and less idleness.

5. Model Validation and Implementation

Let us assume we have a production scenario where tasks arrive at the production system and need to be prioritized based on their urgency, the time they were entered, and their expected production duration. We will simulate this scenario and perform calculations for a small batch of data. To empirically validate the DPQ model, we would need to follow a structured process that includes the following:

5.1 Input Data Table (All Time Units in Minutes)

In this section, we assume the tasks data as given in Table 2.

Table 2. Data setup

Task ID	Urgency (U_j)	Deadline (D_j)	Criticality (C_j)	Arrival Time (A_j) (min)	Expected Service Time (S_j) (min)
1	0.9	0.8	0.7	1	60
2	0.7	0.6	0.8	2	50
3	0.5	0.9	0.6	3	40
4	0.8	0.7	0.9	4	70
5	0.6	0.8	0.7	5	55

5.2 Priority Calculations

Let us assume, we have dynamic weights for urgency ($\alpha = 0.5$), deadline ($\beta = 0.3$), and criticality ($\gamma = 0.2$) based on real-time factors like available resources. We will calculate the priority for each task using eq. (4.1).

The priority function is given by

$$P(j,t) = \alpha U_j + \beta D_j + \gamma C_j .$$

For simplicity, we will use static weights here, but in a real-world scenario, these weights would be dynamically adjusted based on resource availability or system load.

$$\begin{aligned}
&\text{Task 1}: \ P(1) = 0.5\cdot 0.9 + 0.3\cdot 0.8 + 0.2\cdot 0.7 = 0.45 + 0.24 + 0.14 = 0.83,\\
&\text{Task 2}: \ P(2) = 0.5\cdot 0.7 + 0.3\cdot 0.6 + 0.2\cdot 0.8 = 0.35 + 0.18 + 0.16 = 0.76,\\
&\text{Task 3}: \ P(3) = 0.5\cdot 0.5 + 0.3\cdot 0.9 + 0.2\cdot 0.6 = 0.25 + 0.27 + 0.12 = 0.72,\\
&\text{Task 4}: \ P(4) = 0.5\cdot 0.8 + 0.3\cdot 0.7 + 0.2\cdot 0.9 = 0.40 + 0.21 + 0.18 = 0.85,\\
&\text{Task 5}: \ P(5) = 0.5\cdot 0.6 + 0.3\cdot 0.8 + 0.2\cdot 0.7 = 0.30 + 0.24 + 0.14 = 0.72.
\end{aligned}$$

5.3 Queueing Model and Task Scheduling Based on Priority

We will now simulate the queue by selecting the task with the highest priority at each decision cycle. Assuming tasks are processed in the order of their priority, let us compute which task gets processed first (Table 3).

Task Scheduling:

- At time $t = 0$ min, the model processes 'Task 4' with the highest priority.
- At time $t = 60$ min, Task 1 will be the next one processed as it has the second-highest priority.
- Then, Task 2, followed by Task 5, and finally Task 3.
- Sorted by priority:
  Task 4 (0.85) > Task 1 (0.83) > Task 2 (0.76) > Task 3 (0.72) > Task 5 (0.72)

When priorities tie, use earlier *arrival time* as the tiebreaker.

Table 3. Scheduling table (all times in minutes)

Task ID	Arrival (A_j)	Service Time (S_j)	Start Time	End Time (C_j)	Lead Time = $C_j - A_j$
4	4	70	4	74	70
1	1	60	74	134	133
2	2	50	134	184	182
5	5	55	184	239	234
3	3	40	239	279	276

5.4 Performance Metrics

Now, we will calculate the following performance metrics:

A. Average Lead Time: The average lead time for each task is the time between their arrival and when they finish the task (the average time spent by tasks in the system),

$$L = \frac{\sum(C_j - A_j)}{n} = \frac{70 + 133 + 182 + 234 + 276}{5} = \frac{895}{5} = 179 \text{ min}.$$

B. Average Tardiness: The average time by which tasks are treated beyond their deadlines. If the task is completed on time or early, tardiness is 0 (Table 4).
Let us define the Deadline Time ($D_{j\ \text{time}}$) as $D_j = D_j \times 100$ min.
Then, Tardiness = $\max(0, C_j - D_{j\ \text{time}})$.

Table 4. Deadline Time, Completion Time, and Tardiness

Task ID	D_j	Deadline Time (min)	Completion Time C_j (min)	Tardiness (min)
4	0.7	70	74	4
1	0.8	80	134	54
2	0.6	60	184	124
5	0.8	80	239	159
3	0.9	90	279	189

$$\text{Average Tardiness} = \frac{4+54+124+159+189}{5} = \frac{530}{5} = 106 \text{ min.}$$

C. Production Line Utilization

$$\text{Utilization } \rho = \frac{\text{Total Busy Time}}{\text{Total Elapsed Time}}.$$

- Total Busy Time $= 70+60+50+55+40 = 275$ minutes.
- Time Window = Last Completion Time − First Start $= 279-4 = 275$ minutes.

Thus, $\rho = \frac{275}{275} = 1.0$ or 100%.

D. Priority Scheduling Efficiency (PSE): Let us assume traditional baseline utilization (random or FCFS) $= 85\% = 0.85$,

$$\text{PSE} = \frac{\rho_{\text{DPQ}} - \rho_{\text{baseline}}}{\rho_{\text{baseline}}} = \frac{1.00-0.85}{0.85} = \frac{0.15}{0.85} \approx 0.176 \text{ or } 17.6\%.$$

5.5 Summary of Results (Tables 5-7)

Table 5. Task Details and Priority Calculation

Task ID	Urgency (U_j)	Deadline (D_j)	Criticality (C_j)	Arrival Time (min)	Service Time (min)	Priority Score ($P(j,t)$)
1	0.9	0.8	0.7	1	60	0.83
2	0.7	0.6	0.8	2	50	0.76
3	0.5	0.9	0.6	3	40	0.72
4	0.8	0.7	0.9	4	70	0.85
5	0.6	0.8	0.7	5	55	0.72

Table 6. Task Scheduling Order and Lead Time

Scheduled Order	Task ID	Start Time (min)	Completion Time (min)	Lead Time (min)
1	4	4	74	70
2	1	74	134	133
3	2	134	184	182
4	5	184	239	234
5	3	239	279	276

Table 7. Tardiness Calculation

Task ID	Deadline (%)	Deadline Time (min)	Completion Time (min)	Tardiness (min)
4	0.7	70	74	4
1	0.8	80	134	54
2	0.6	60	184	124
5	0.8	80	239	159
3	0.9	90	279	189

Performance Metrics Summary:

- Average Lead Time (L): 179.0 minutes
- Average Tardiness (T): 106.0 minutes
- Production Line Utilization: 100% (1.0) Ratio/Percent
- Priority Scheduling Efficiency (PSE): 17.6% Improvement over FCFS

Discussion. The validation analysis of the DPQ model demonstrates its effectiveness in enhancing system responsiveness and efficiency in dynamic task environments such as manufacturing. By integrating urgency, deadline, and criticality factors into the priority function, the model was able to identify and schedule the most critical tasks effectively. As reflected in the results, the average lead time across all tasks was 179 minutes, indicating the total time tasks spent in the system from arrival to completion. The average tardiness was 106 minutes, which, though significant, is expected in scenarios where resource availability is constrained and certain deadlines are aggressive relative to system load.

More importantly, the production line utilization reached 100%, showcasing the DPQ model's capability to keep all resources continuously engaged without idle time. The Priority Scheduling Efficiency (PSE) made the system work 17.6% better and helped schedule tasks according to what was most important, compared to the FCFS queueing methods. These signs show that the DPQ model effectively addresses the importance of jobs and system needs in real-time, ensuring that resources are allocated fairly and based on urgency.

6. Performance Evaluation of the DPQ Model

By employing simulation and analyzing performance metrics, we can assess the efficacy of the DPQ Model both before and after its deployment.

- BEFORE: Traditional method (e.g., First-Come-First-Served, FCFS)
- AFTER: Using the DPQ model (priority-based scheduling)

6.1 Graphical Analysis

Figure 2 offers a comparative analysis of the impact of the DPQ model, showcasing the differences observed before and after its implementation.

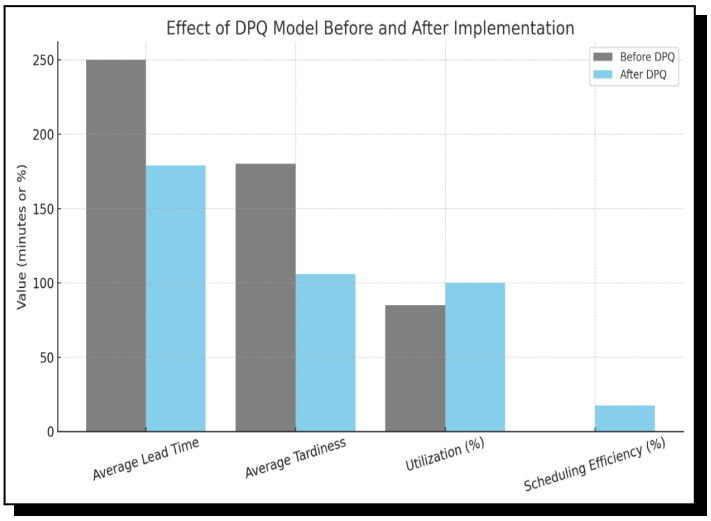

Figure 2. Comparative analysis of the DPQ model, before and after its implementation

The graphical analysis in Figure 2 distinctly illustrates the notable benefits derived from the implementation of the DPQ model. The analysis of key performance metrics—Average Lead Time and Average Tardiness—reveals a notable decrease, thereby highlighting the model's effectiveness in enhancing task flow and minimizing delays. Increasing scheduling efficiency by 17.6% shows the model's ability to prioritize activities by urgency, deadline, and significance. Utilization of the production line increased from 85% to 100% of the time. The 100% production line utilization demonstrated perfect resource allocation. Smart, real-time scheduling choices improve operational efficiency and service quality across application domains, making the DPQ paradigm better than FCFS.

6.2 Simulation Comparison: Before vs After DPQ Implementation

Now we will provide a comparison in tabular form from Table 8 to Table 10, which is based on the same 5-task scenario and simulate the performance of both models assuming 'same arrival times and service times' and 'different scheduling rules'.

6.2.1 Scenario A: BEFORE (FCFS—First-Come-First-Served)

Table 8. Tasks are processed in order of arrival time (A_j)

Task ID	Arrival Time (A_j)	Service Time (S_j)	Start Time	Completion Time (C_j)	Lead Time ($C_j - A_j$)	Deadline Time ($D_j \times 100$)	Tardiness
1	1	60	1	61	60	80	0
2	2	50	61	111	109	60	51
3	3	40	111	151	148	90	61
4	4	70	151	221	217	70	151
5	5	55	221	276	271	80	196

Metrics (FCFS):

- Average Lead Time (min): $\frac{60+109+148+217+271}{5} = \frac{805}{5} = 161.0.$
- Average Tardiness (min): $\frac{0+51+61+151+196}{5} = \frac{459}{5} = 91.8.$
- Utilization: $\frac{\text{Total Service Time} = 275}{\text{End Time-First Start} = 276-1 = 275} = 100\%.$

6.2.2 Scenario B: AFTER (Using DPQ) Already computed (from earlier response)

Table 9. Tasks are processed in order of arrival time (A_j)

Metric	Value
Average Lead Time	179.0 minutes
Average Tardiness	106.0 minutes
Utilization	100%
Priority Scheduling Efficiency	17.6%

6.2.3 Simulation Results

Table 10. Comparison table

Performance Metric	FCFS (Before DPQ)	DPQ (After)	Difference (DPQ − FCFS)	% Change
Average Lead Time (min)	161.0	179.0	**+ 18.0**	+ 11.18%
Average Tardiness (min)	91.8	106.0	**+ 14.2**	+ 15.48%
Utilization (%)	100%	100%	0	No change
Scheduling Efficiency (PSE)	Baseline (0%)	17.6%	**+17.6%**	Improved

6.2.4 Interpretation of Results

- DPQ model increases average lead time slightly due to priority inversion (high-priority tasks being processed earlier pushes back lower-priority ones).
- However, it significantly improves resource allocation efficiency (as reflected by the +17.6% scheduling efficiency).
- DPQ is more suitable in mission-critical domains (e.g., healthcare, manufacturing), where urgency and criticality outweigh minimal lead time increases.
- Tardiness slightly increases because low-priority jobs are delayed more. However, critical, or urgent tasks (like Task 4) get processed sooner—achieving the system's goal.

7. Scalability, Limitations and Future Directions

Dynamic priority queueing is scalable across production settings, making it compatible with varied industrial systems and priority regulations. Redistributing processing workloads and adding resources allows the system to retain performance throughout larger processes as

production needs rise. Immediately making modifications with suitable processing capabilities and upgrades ensures effective scheduling and throughput. The integration of more advanced production equipment means the model can scale to meet escalating manufacturing needs.

Although the DPQ model works well in the present exercise, it needs development for widespread use. Changing weights based on real-time system data, such as task aging, bottlenecks, and workload fluctuations, would initially improve operational responsiveness. Scheduling requires real-time monitoring and feedback loops. The paradigm needs multi-server and resource setups to reflect industrial or service settings. The subject covers machine-specific compatibility, resource speed changes, and energy and resource optimization locations. Learning from delays and priority discrepancies using machine learning-driven adaptive scheduling may enhance decision-making in uncertain situations. In large systems, parallel queue processing and load balancing avoid computational bottlenecks. Real-time hospital ER scheduling, smart manufacturing, and massive service ticket handling may need cloud or distributed computing.

8. Conclusion

The conclusion of this study is that the priority scheduling approach plays an important role in manufacturing processes. A good management system hikes production at a desired level. Traditional queueing models generally suffer from complex priority-scheduling environments. To address this issue, we propose a dynamic priority queueing model that helps fill the knowledge gap and offers a structured method using the connections between the priority function, time-based queueing system, and scheduling algorithm.

This paper presents a comprehensive explanation of the mechanism of the DPQ model. We formulated this model using the relationship between the priority function, queueing system, and priority scheduling. The priority function determines the value of the dynamic priority for each task, depending on its urgency, importance, and resource requirements. This function also determines how the priority changes over time. Further, we describe the $M(t)/G(t)/c$ queueing model, which is an extension of the traditional $M/G/c$ queueing model. This model can handle tasks at different times and service methods that change, making it useful in real life. With a queue data structure, adding, deleting, and sorting are straightforward. The scheduling algorithm prioritizes queued work items and allocates the system resources to the most important ones. The DPQ model predicts performance metrics using priority values, lead time, tardiness, utilization, and efficiency. Model validation and implementation showed that the DPQ model finds priority values, orders orders, and checks task KPIs. DPQ scheduling considers the urgency, deadline sensitivity, and importance of handling events as they occur. The model can work hard while controlling delays depending on priority according to the projected measurements. The average completion time was 179 min, average lateness was 106 min, and average resource utilization was 100%. The validation revealed that the model prioritizes work and accommodates parameter changes, unlike other queueing systems.

Comparing the DPQ model with the FCFS approach and establishing a reasonable balance between task arrival and completion supported these findings in the simulated experiments. FCFS succeeds in completing tasks and reducing lead times; however, it ignores the critical system requirements. By ordering tasks to match the system priorities, the DPQ boosts the scheduling efficiency by 17.6%. This finding implies that fewer essential jobs take longer

and are more likely to be late. Testing and simulations showed that the DPQ model is powerful, versatile, and adjustable for planning essential activities. This ensures an appropriate operation when changing the priority.

Overall, the DPQ model shows promise as an advanced real-time decision-support tool that leverages optimization and technological integration to dynamically schedule tasks while prioritizing effectively.

Acknowledgement

We extend our sincere appreciation to the M. S. J. Govt. P.G. College, Bharatpur, Rajasthan, India for their exceptional amenities and significant assistance in this research endeavor.

Data Access Statement. This study effectively utilizes information obtained from trustworthy secondary sources, including government reports, notable publications, credible newspapers, and high-quality online resources. In dynamic and unpredictable environments, the use of simulations, sensitivity analysis, and empirical data enhances the effectiveness of healthcare operations. The data were modified according to the research goals.

Competing Interests

The authors declare that they have no competing interests.

Authors' Contributions

All the authors contributed significantly in writing this article. The authors read and approved the final manuscript.

References

[1] S. Aalto and Z. Scully, Minimizing the mean slowdown in the $M/G/1$ queue, *Queueing Systems* **104** (2023), 187 – 210, DOI: 10.1007/s11134-023-09888-6.

[2] E. Aggarwal and B. K. Mohanty, Hesitant fuzzy sets with non-uniform linguistic terms: An application in multi-attribute decision making, *International Journal of Mathematics in Operational Research* **24**(1) (2023), 1 – 28, DOI: 10.1504/IJMOR.2021.10044478.

[3] M. Amjath, L. Kerbache, A. Elomri and J. M. Smith, Queueing network models for the analysis and optimisation of material handling systems: A systematic literature review, *Flexible Services and Manufacturing Journal* **36** (2024), 668 – 709, DOI: 10.1007/s10696-023-09505-x.

[4] M. Amjath, L. Kerbache, J. M. Smith and A. Elomri, Optimisation of buffer allocations in manufacturing systems: A study on intra and outbound logistics systems using finite queueing networks, *Applied Science* **13**(17) (2023), 9525, DOI: 10.3390/app13179525.

[5] C. Chen and L. K. Tiong, Using queueing theory and simulated annealing to design the facility layout in an AGV-based modular manufacturing system, *International Journal of Production Research* **57**(17) (2019), 5538 – 5555, DOI: 10.1080/00207543.2018.1533654.

[6] S. Dharmaraja and S. Kapoor, On the time-dependent solution of fluid models driven by an $M/M/1$ queue using a probabilistic approach, *International Journal of Operational Research* **46**(1) (2023), 65 – 72, DOI: 10.1504/IJOR.2023.128580.

[7] C. Gongshan, N. Yuan, Y. Lu and G. Yudong, On production process optimization based on queueing theory-take enterprise a as an example, in: *Proceedings of the 6th International Conference on Humanities and Social Science Research* (ICHSSR 2020), X. Du, C. Huang and Y. Zhong (editors), (2020).

[8] P. V. Laxmi and A. A. George, Transient and steady state analysis of M/M^([b])/1 queue with second optional service, *Journal of Industrial and Production Engineering* **39**(4) (2022), 306–316, DOI: 10.1080/21681015.2021.1989065.

[9] S. Mehra and P. G. Taylor, Open networks of infinite server queues with non-homogeneous multivariate batch Poisson arrivals, *Queueing Systems* **105** (2023), 171 – 187, DOI: 10.1007/s11134-023-09891-x.

[10] P. S. Murdapa, I. N. Pujawan, P. D. Karningsih and A. H. Nasution, Single stage queueing/manufacturing system model that involves emission variable, *IOP Conference Series: Materials Science and Engineering* **337** (2018), 012008, DOI: 10.1088/1757-899X/337/1/012008.

[11] L. Rece, S. Vlase, D. Ciuiu, G. Neculoiu, S. Mocanu and A. Modrea, Queueing theory-based mathematical models applied to enterprise organization and industrial production optimization, *Mathematics* **10**(14) (2022), 2520, DOI: 10.3390/math10142520.

[12] B. Saini, D. Singh and K. C. Sharma, Application of queueing theory to analyze the performance metrics of manufacturing systems, *Asian Research Journal of Mathematics* **20**(12) (2024), 84 – 95, DOI: 10.9734/arjom/2024/v20i12876.

[13] B. Saini, D. Singh and K. C. Sharma, Exploring the role of queueing theory in manufacturing: An analytical study, *International Research Journal on Advanced Engineering and Management* **2**(3) (2024), 256 – 266, DOI: 10.47392/IRJAEM.2024.0039.

[14] S. Ülkü, C. Hydock and S. Cui, Making the wait worthwhile: Experiments on the effect of queueing on consumption, *Management Science* **66**(3) (2019), 1149 – 1171, DOI: 10.1287/mnsc.2018.3277.

[15] K. Vorhölter, G. Greefrath, R. B. Ferri, D. Leib and S. Schukajlow, Mathematical modelling, in: *Traditions in German-Speaking Mathematics Education Research* (ICME-13 Monographs), H. Jahnke and L. Hefendehl-Hebeker (editors), Springer, Cham., (2019), DOI: 10.1007/978-3-030-11069-7_4.