# A Study on the PRISM-Based Scheduling Model for Multi-Constrained Systems

Yujie Li, Zhengsheng Chen *

School of Computer Science, Rocket Force University of Engineering, Xi'an, China, 710025

* Corresponding Author Email: czsgeo@126.com

**Abstract.** In multi-task complex systems, how to achieve efficient task scheduling and optimal resource allocation under multiple conditions such as resource constraints, process constraints and time window limitations is a key issue that urgently needs to be solved in the fields of systems engineering and intelligent control. For typical scheduling scenarios with strong process dependence, multi-stage processes and heterogeneous resource structures, this paper proposes a priority-driven scheduling model (PRISM). Firstly, by constructing the temporal dependency graph model of the task process, the integer programming method is adopted to uniformly represent the task execution status and resource occupation. On this basis, a multi-dimensional priority evaluation function was innovatively designed. Combined with the dynamic resource state vector and the bidirectional scheduling control mechanism, the collaborative optimization of task execution periods and resource allocation was achieved. To deal with the problems of timing conflicts and resource competition in complex scenarios, an adaptive adjustment algorithm based on difference detection is proposed, which effectively guarantees the feasibility and robustness of the scheduling scheme. The experimental part was verified on a typical high-density task set. The results show that the proposed method can effectively improve the scheduling completion rate and resource utilization efficiency, and has good conflict repair ability and engineering adaptability. This method has the advantages of strong universality, stable scheduling performance and high deployability, and is suitable for the research of scheduling optimization problems in multi-type process systems.

**Keywords:** Multi Task Scheduling, Priority Driven, Resource Allocation, Integer Programming, Process Modeling.

## 1. Introduction

In multi-task complex systems, how to achieve efficient task scheduling and optimal resource allocation under multiple conditions such as resource constraints, process constraints and time window limitations is a key issue that urgently needs to be solved in the fields of systems engineering and intelligent control. For typical scheduling scenarios with strong process dependence, multi-stage processes and heterogeneous resource structures, this paper proposes a priority-driven task-resource-time collaborative optimization method (PRISM). Firstly, by constructing the temporal dependency graph model of the task process, the integer programming method is adopted to uniformly represent the task execution status and resource occupation. On this basis, a multi-dimensional priority evaluation function was innovatively designed. By combining heuristic algorithms [1], dynamic resource state vectors, and bidirectional scheduling control mechanisms, the collaborative optimization of task execution periods and resource allocation was achieved. To deal with the problems of timing conflicts and resource competition in complex scenarios, an adaptive adjustment algorithm based on difference detection is proposed, which effectively guarantees the feasibility and robustness of the scheduling scheme. The experimental part was verified on a typical high-density task set. The results show that the proposed method can effectively improve the scheduling completion rate and resource utilization efficiency, and has good conflict repair ability and engineering adaptability. This method has the advantages of strong universality, stable scheduling performance and high deployability, and is suitable for the research of scheduling optimization problems in multi-type process systems.

# 2. Related Work

## 2.1. Research on Multi-Resource Scheduling Modeling

Resource allocation and process modeling are fundamental issues in complex systems, and their significance runs through many fields, such as manufacturing, project management, logistics, etc. In multi-task scheduling problems, the choice of modeling methods is particularly crucial, usually relying on technical means such as integer programming [2], time series graph models [3], and resource partition graphs. Among them, the Job-shop Scheduling Problem (JSSP) [4] and the Project Scheduling Problem (PSP) [5], as classic modeling frameworks, have been widely applied in practical scenarios. These models focus on task-resource mapping and stage dependency, emphasizing resource exclusivity and process sequence constraints, providing a theoretical basis for solving scheduling problems. However, as system complexity increases, traditional models struggle with system coupling due to assumptions like fixed resources and homogeneous tasks, which rarely hold in real systems. Modern systems exhibit resource diversification and task heterogeneity, making traditional models inadequate. Strict processes, complex paths, and nonlinear timing further complicate modeling. Static modeling methods lack the ability to describe dynamic changes and interactions, limiting their application in modern systems. While widely used in industrial engineering, traditional models have limitations in multi-resource coupling and nonlinear processes, necessitating a more flexible and adaptable modeling mechanism.

## 2.2. Research on Intelligent Scheduling Algorithm

In the process of solving complex scheduling problems, heuristic algorithms have become important tools due to their high efficiency and practicability. Classic heuristic methods include greedy algorithm [6], genetic algorithm, simulated annealing algorithm and ant colony algorithm [7], etc. These methods explore large solution spaces using various search strategies, offering an effective approach to solve NP-hard problems. However, traditional heuristic algorithms have limitations: simple rule-based strategies struggle to adapt to complex constraints in modern engineering systems, and these algorithms often lack insight into the system's underlying structure, leading to instability and poor interpretability in scenarios with heterogeneous resources and strong process dependencies. Recent trends indicate a shift from "simple algorithm optimization" to "modeling and optimization synergy." This paradigm emphasizes combining system structure modeling with intelligent algorithms to create a structure-driven optimization framework. Structure-constraint-enhanced heuristic algorithms are gradually replacing traditional black-box methods and represent a key research direction in multi-task scheduling.

## 2.3. The innovation points of this research based on the existing work

This paper proposes the PRISM model, innovatively combining structural modeling with heuristic scheduling, and integrating the modeling expressiveness and the stability of the scheduling algorithm. Introduce a process-driven task priority evaluation [8] mechanism to dynamically adjust task priorities and significantly improve scheduling adaptability. Meanwhile, a time difference conflict detection and correction strategy is designed to monitor and quickly correct time conflicts in real time, enhancing the feasibility of the scheduling scheme and the engineering deployability. Ultimately, the PRISM model realizes the integrated modeling of task-level and system-level scheduling logic, improving the globality and scalability of scheduling decisions. Compared with traditional scheduling methods that focus on solution efficiency or modeling accuracy, the method proposed in this paper emphasizes the collaboration between modeling and scheduling more, has system adaptability and engineering implementation value, and provides a general solution framework for scheduling problems of highly complex systems.

# 3. Methodology

## 3.1. System modeling framework: Task - resource - time ternary structure

The modeling objective of this paper is to formally express the full-process constraint relationship of the task scheduling problem and transform the scheduling problem into a constrained optimization problem to support the coupled control of multiple tasks, multiple resources and multiple stages. To this end, we have designed a unified modeling structure to achieve the synchronous expression of task execution sequences, resource usage conflicts and time window limitations. The core components of the model include triples: task sets $T = \{T_1, T_2, \ldots, T_n\}$ (each task has a process stage, execution time and resource requirements); Resource set $R = \{R_1, R_2, \ldots, R_m\}$ (representing a kind of restricted resources, such as devices, channels, platforms, etc.), define the mutual exclusion matrix $M \in \{0,1\}^{m \times m}$; Time variable set S (covering the start time of the task, the duration of the process, and the period of resource occupation.)

In terms of the expression of task processes, Temporal directed graphs (DAG) [9] are adopted for modeling. The dependency time of the stages within a task is represented by the edge weight function $w(e)$, and the absence of conflicts between tasks can be achieved through binary constraint variables $x_i^t$. Edge weight function $w(e)$:

$$w(e) = \begin{cases} d_{ij} \\ \delta_{\min} \end{cases} \tag{1}$$

Among them, $d_{ij}$ represents the minimum time interval between stages, and $\delta_{min}$ represents the security buffer time between tasks.

The conflict-free relationship between tasks can be established through inequalities:

$$x_i^t + x_j^t \le 1 \tag{2}$$

Among them, $x_i^t$ indicates whether node $i$ is executed at time $t$.

We formalize the system scheduling problem as a task-resource-time ternary coupled modeling structure. In this structure, tasks describe their stage logic through flowcharts, resources describe the competitive relationship through mutual exclusion matrices, and time realizes the unified representation of full-process constraints through window functions $W_k(t)$.

Utilize binary variables $M_{ij}$:

$$M_{ij} = \begin{cases} 1, \text{Resource } R_i \text{ and resource } R_j \text{ are mutually exclusive} \\ 0, \text{Resource } R_i \text{ and Resource } R_j \text{ can be shared} \end{cases} \tag{3}$$

Construct a unified inequality of full-process constraints:

$$x_k^t \le W_k(t) \tag{4}$$

## 3.2. Prioritized scheduling strategy design

The task ranking strategy is designed based on the priority function $P_i$ [10]. Construct the comprehensive index function:

$$P_i = \alpha \cdot U_i + \beta \cdot D_i + \gamma \cdot C_i \tag{5}$$

Among them, $U_i$ represents the urgency of the task, which is the difference between the current time and the minimum window time for the next task; $D_i$ represents the task dependency density,

that is, the number of task nodes that the task subsequently depends on; $C_i$ represents the degree of resource competition, that is, the proportion of resources currently occupied by the task. The adjustable weight $\alpha$、$\beta$、$\gamma$ is used to schedule the customization and robustness of the strategy, thereby flexibly adapting to different scheduling requirements.

The resource matching strategy assists in selecting available paths through the resource state diagram. Maintain a state vector for each type of resource and update its occupancy status dynamically. Based on the resource requirements of the task, match the current idle section and calculate the candidate solution with the minimum path consumption. This method not only realizes the rationality of resource allocation, but also takes into account the load balancing of resource usage, ensuring the effective utilization of resources and the overall performance of the system. Utilize the resource matching optimization formula:

$$p^* = \arg\min_{p \in \mathrm{P}} \left( \underbrace{\sum_{k \in \mathrm{K}_i} c_k \cdot \max(0, s_i - r_k^{start})}_{\text{Time cost}} + \underbrace{\lambda \left\| \mathbf{R}(t) - \mathbf{D}_i \right\|}_{\text{Resource compatibility degree}} \right) \tag{6}$$

Among them, $p^*$ represents the set of feasible paths, $D_i$ represents the task requirement vector, and $\lambda$ is the adjustable weight parameter.

The dispatching process control structure adopts a two-stage mechanism of "forward allocation + backward adjustment". In the forward scheduling stage, based on task sorting, resources are attempted to be allocated within the execution window to quickly generate a preliminary scheduling plan. If the task fails to complete the resource binding, in the backtracking stage, the previous tasks are adjusted according to the priority, resources are released or the time is changed to ensure the feasibility and stability of the scheduling plan. Furthermore, the difference repair mechanism is used to handle micro-misalignment problems such as starting time conflicts and process overlaps, further optimizing the scheduling scheme and improving the operational efficiency of the system.

### 3.3. Constraint condition modeling and feasibility detection mechanism

For stage $v_1 \to v_2 \to \cdots \to v_n$ of Task $T_i$, it needs to be satisfied:

$$s_{k+1} \geq s_k + d_k + \delta_k^{\min}, \quad k \in \{1, \ldots, n-1\} \tag{7}$$

Among them, $s_k$、$d_k$、$\delta_k^{\min}$ respectively represent the start time of stage $v_k$, the execution duration of stage $v_k$, and the safety buffer time between the tasks of stage $v_k$ and $v_{k+1}$.

Within the same time period, at any moment $t$, each resource $R_k$ can be occupied by at most one task $T_i$:

$$\sum_{T_i \in T} x_{i,k}^t \leq 1 \tag{8}$$

And Task $T_{i+1}$ must be executed after $T_i$, that is:

$$s_{i+1} \geq s_i + d_{i,k} \tag{9}$$

To ensure that the task can only be initiated within the specified time window, the following must be met:

$$a_i \leq s_i \leq b_i - d_i \tag{10}$$

Among them, $a_i$、$b_i$、$d_i$ respectively represent the earliest start time, the latest start time and the execution duration.

The difference conflict detection mechanism is used to monitor whether the actual starting time difference between tasks is less than the minimum safety interval. If a conflict is detected, the local timing adjustment module is triggered. The time variable is fine-tuned through iterative optimization to restore the feasibility of the scheduling, thereby ensuring the stability and effectiveness of the scheduling scheme. For the two tasks $T_i$ and $T_j$, based on the conflict detection logic formula:

$$\text{Conflict}(T_i, T_j) = \begin{cases} 1, \text{if } \left|s_i - s_j\right| < \delta_{ij}^{\min} \wedge [s_i, s_i + d_i) \cap [s_j, s_j + d_j) \neq \varnothing \\ 0, \text{otherwise} \end{cases} \tag{11}$$

The difference detection is carried out according to the process. The flowchart is shown in Figure 1:

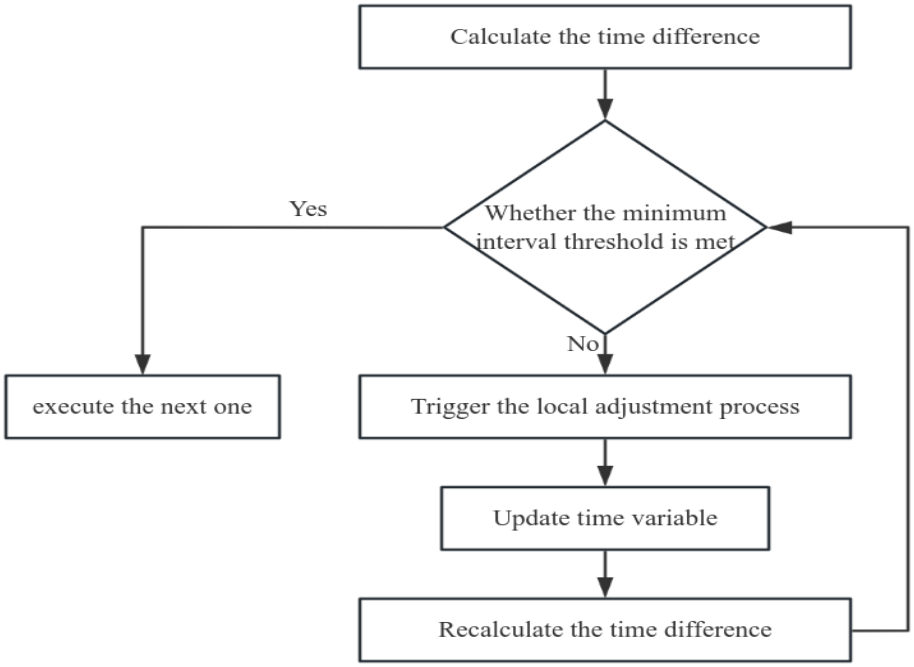


**Figure 1.** Flowchart of difference Detection

Perform local timing adjustment:

$$\min \sum_{(i,j)\in C} \left( \left| s_i' - s_i \right| + \left| s_j' - s_j \right| \right) \tag{12}$$

Among them, C、$s_i'$、$s_j'$ respectively represent the set of detected conflicting task pairs and the adjusted start time.

### 3.4. Design of scheduling Solution process

We have constructed a scheduling solution process driven by task priorities as the core, which supports the rapid matching of tasks and resources under complex constraints. Combined with difference analysis, it realizes conflict correction at the micro level, providing a model support with deployment capabilities for system scheduling problems.

# 4. Experiment

## 4.1. Experimental configuration and parameter Settings

To evaluate the generality and stability of the model, we constructed a typical example that includes multiple types of tasks, multiple process nodes, and high-density resource scheduling requirements to simulate the dynamic scheduling process in complex environments. This article conducted experiments based on the emergency transportation of materials. The experiment involves 25 heterogeneous tasks with different process stages and resource requirements. The experiment includes three types of resource sets: material warehouse, material assembly area, execution unit, etc., with a total of more than 15 resource units. The execution order of processes is irreversible, and there are delay constraints on the dependencies between stages, which increases the complexity of scheduling. The transportation process is shown in Figure 2:

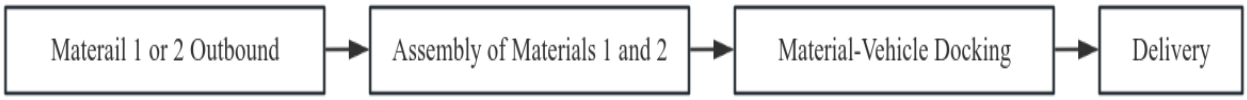


**Figure 2.** Transportation process

## 4.2. Review of the scheduling execution process

The scheduling process is implemented through a three-stage strategy: initial sorting and allocation, conflict correction, and output generation, ensuring the logical integrity of task scheduling and the engineering feasibility of resource matching. Firstly, import detailed task information from an Excel file, including task ID, execution time, resource requirements, and startup window. Defined the types and quantities of available resources (e.g. docking stations, storage areas, execution units). Clearly define the order and stages of task execution to ensure that it follows the planned workflow. Calculate the comprehensive score of each task based on urgency, dependency density, and resource competition to generate a scheduling sequence. Traverse tasks in this order to allocate resources and time windows. When attempting to allocate a start time within a feasible window, check resource requirements based on current availability. The successfully scheduled tasks and their resource usage periods (start time, end time, and utilized resources) will be recorded. In addition, calculate the time difference between adjacent tasks to verify whether they meet the minimum interval threshold. If the difference is below this threshold, a local adjustment process will be triggered to update the timing variables to address issues such as start time misalignment or resource overlap. Finally, review all tasks to ensure successful scheduling; Constraints regarding window constraints must be met without conflicts in resource usage or task flow continuity. A detailed schedule has been generated, listing the IDs of each task, their start and end times, and the resources used. Gantt chart visually represents the execution timeline and resource utilization status, making it easier to analyze scheduling results. For the convenience of analyzing and evaluating the scheduling results, the calculation process is shown in Figure 3:

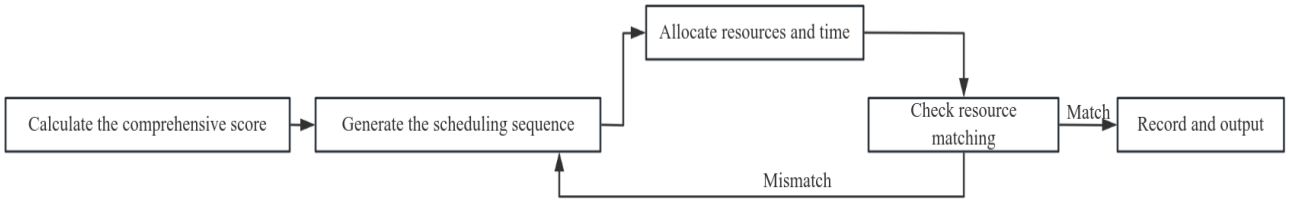


**Figure 3.** Model calculation process

Based on the known information of the task, the application model continues to calculate and conducts comparative analysis in combination with actual problems. The initial allocation sequence and resource allocation table are shown in Table 1:

**Table 1.** Allocation Sequence and Resource Allocation

| Task Code | Warehouse ID for Material 1 | Warehouse ID for Material 2 | Delivery Vehicle Number | Material-Vehicle Docking | Allocation Sequence |
|---|---|---|---|---|---|
| M1 | KF-1 | KF-1 | TZ-5 | CC-1 | 6 |
| M2 | KF-2 | KF-2 | TK-7 | CC-2 | 8 |
| M3 | KF-1 | KF-1 | TZ-2 | CC-1 | 17 |
| M4 | KF-1 | KF-1 | TZ-1 | CC-1 | 13 |
| M5 | KF-2 | KF-2 | TK-8 | CC-2 | 4 |
| M6 | KF-1 | KF-1 | TZ-6 | CC-1 | 10 |
| M7 | KF-2 | KF-2 | TZ-7 | CC-2 | 21 |
| M8 | KF-1 | KF-1 | TZ-4 | CC-1 | 2 |
| M9 | KF-2 | KF-2 | TK-9 | CC-2 | 7 |
| M10 | KF-1 | KF-1 | TK-1 | CC-1 | 20 |
| M11 | KF-1 | KF-1 | TZ-3 | CC-1 | 9 |
| M12 | KF-2 | KF-2 | TZ-9 | CC-2 | 22 |
| M13 | KF-1 | KF-1 | TK-4 | CC-1 | 25 |
| M14 | KF-2 | KF-2 | TK-7 | CC-2 | 3 |
| M15 | KF-2 | KF-2 | TZ-7 | CC-2 | 23 |
| M16 | KF-1 | KF-1 | TZ-5 | CC-1 | 18 |
| M17 | KF-1 | KF-1 | TZ-2 | CC-1 | 5 |
| M18 | KF-2 | KF-2 | TK-7 | CC-2 | 15 |
| M19 | KF-1 | KF-1 | TZ-4 | CC-1 | 14 |
| M20 | KF-1 | KF-1 | TZ-1 | CC-1 | 1 |
| M21 | KF-2 | KF-2 | TK-9 | CC-2 | 12 |
| M22 | KF-1 | KF-1 | TK-2 | CC-1 | 19 |
| M23 | KF-2 | KF-2 | TZ-8 | CC-2 | 16 |
| M24 | KF-2 | KF-2 | TZ-8 | CC-2 | 24 |
| M25 | KF-2 | KF-2 | TK-8 | CC-2 | 11 |

**Table 2.** Time Calculation Results

| Task Code | Material 1 Outbound | Material 2 Outbound | Material Assembly | Material-Vehicle Docking | Delivery |
|---|---|---|---|---|---|
| M1 | 16.73 | 17.23 | 17.78 | 27.78 | 30.00 |
| M2 | 17.78 | 18.28 | 18.80 | 30.80 | 35.00 |
| M3 | 69.79 | 70.29 | 70.83 | 80.83 | 83.00 |
| M4 | 37.79 | 38.29 | 38.83 | 48.83 | 51.00 |
| M5 | 1.78 | 2.28 | 2.80 | 14.80 | 19.00 |
| M6 | 27.73 | 28.23 | 28.78 | 38.78 | 41.00 |
| M7 | 49.87 | 50.37 | 50.89 | 60.89 | 63.00 |
| M8 | 1.73 | 2.23 | 2.78 | 12.78 | 15.00 |
| M9 | 15.87 | 16.37 | 16.89 | 28.89 | 33.00 |
| M10 | 78.82 | 79.32 | 79.87 | 91.87 | 96.00 |
| M11 | 25.79 | 26.29 | 26.83 | 36.83 | 39.00 |
| M12 | 75.84 | 76.34 | 76.86 | 86.86 | 89.00 |
| M13 | 91.78 | 92.28 | 92.83 | 104.83 | 109.00 |
| M14 | 1.87 | 2.37 | 2.89 | 14.89 | 19.00 |
| M15 | 86.87 | 87.37 | 87.89 | 97.89 | 100.00 |
| M16 | 80.73 | 81.23 | 81.78 | 91.78 | 94.00 |
| M17 | 9.79 | 10.29 | 10.83 | 20.83 | 23.00 |
| M18 | 47.78 | 48.28 | 48.80 | 60.80 | 65.00 |
| M19 | 39.73 | 40.23 | 40.78 | 50.78 | 53.00 |

| | | | | | |
|---|---|---|---|---|---|
| M20 | 1.79 | 2.29 | 2.83 | 12.83 | 15.00 |
| M21 | 33.78 | 34.28 | 34.80 | 46.80 | 51.00 |
| M22 | 94.82 | 95.32 | 95.87 | 107.87 | 112.00 |
| M23 | 64.87 | 65.37 | 65.89 | 75.89 | 78.00 |
| M24 | 99.84 | 100.34 | 100.86 | 110.86 | 113.00 |
| M25 | 24.87 | 25.37 | 25.89 | 37.89 | 42.00 |

Based on the time required for the process and the delivery sequence, the specific time for each process was calculated. The calculation results are shown in Table 2.

### 4.3. Visualization Results and Scheduling Performance Evaluation

Since we only determined the delivery order in advance and calculated the time for each link backwards from the window time of the launch task corresponding to the delivery order, we cannot guarantee that there are no conflicts in the links. Therefore, we established a difference detection model to perform difference detection on the results, ensuring that each set of data meets the constraint conditions. Through differential detection, it was found that tasks M8 and M20, M14 and M16 initially had conflicts in the time window, and the time was rescheduled. The final task execution Gantt chart is shown in Figure 4:

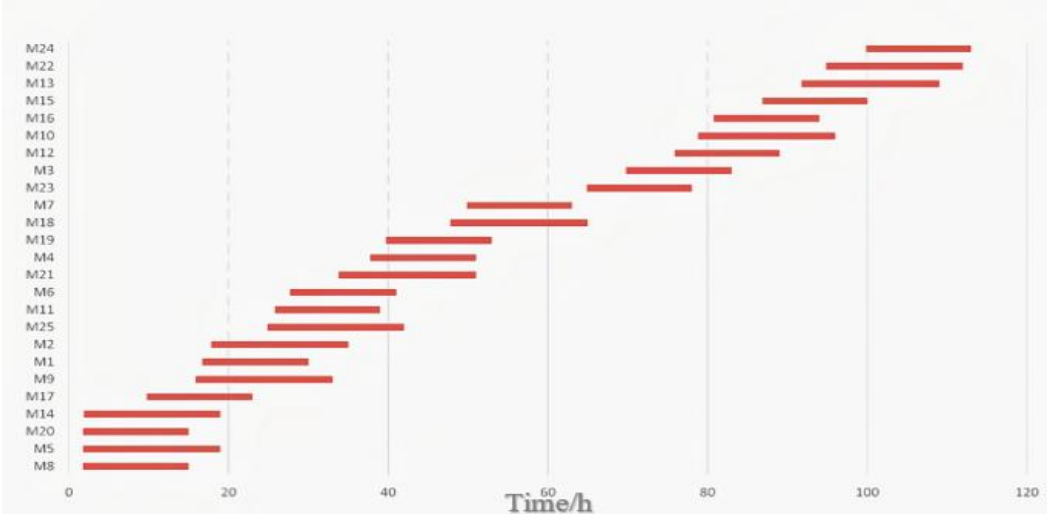


**Figure 4.** Task Execution Gantt Chart

After calculation and two successful differential detections and corrections, all delivery tasks were completed within the deadline without any time window or resource conflicts. The key indicators of the system are shown in Table 3. The scheduling system achieved the triple performance goals of high completion rate, low conflict rate, and high utilization rate in an environment with multiple constraints, resource constraints, and process coupling, verifying the engineering effectiveness and scheduling robustness of the proposed model and algorithm.

**Table 3.** Key Indicators

| Task completion rate | Time window matching rate | The number of resource conflicts | The success rate of difference conflict repair |
|---|---|---|---|
| 100% | 92% | 2 | 100% |

## 5. Conclusion

This paper proposes a unified modeling and scheduling optimization method driven by task priority for the scheduling problem under multiple tasks, multiple resources and multiple timing constraints in complex process-driven systems. By constructing a ternary modeling framework of task-resource-time, supporting the system-level expression of process constraints, resource exclusitions, and window limitations, and designing the PRISM scheduling mechanism, integrating the task priority function, resource state diagram, and bidirectional scheduling strategy, the

construction of feasible solutions for dynamic scheduling was achieved. Furthermore, this paper also proposes a conflict detection and correction mechanism based on time difference, which effectively improves the executability and robustness of the scheduling results. The experimental results show that this method successfully completes all task scheduling, has a high resource utilization rate and effective conflict repair, verifying the engineering practicability of the model. Meanwhile, this method has good generalization ability, can be extended to various task types, resource allocations or constraint forms, and has low computational complexity. It is convenient to be deployed in embedded scheduling modules, digital twin systems or scheduling decision-making platforms, providing an efficient and reliable scheduling solution for engineering practice. This study has established an effective balance among modeling expressiveness, scheduling efficiency and conflict correction mechanism, providing a structured, computable and engineering-deployable general solution path for high-complexity scheduling problems.

Although the task-Resource-Time Integrated scheduling framework (PRISM) proposed in this paper performs well in static task and resource environments, in order to adapt to a wider range of practical application scenarios, future research will be extended to dynamic task environments including uncertain factors such as the arrival of dynamic tasks and the temporary failure of resources. By combining robust optimization and fuzzy scheduling theory, the adaptability of the system to disturbances can be significantly improved. Furthermore, the research will also expand the multi-objective scheduling mechanism. Besides optimizing the task completion rate and the shortest execution time, it will also consider multiple objectives such as minimizing energy consumption and resource switching costs, construct a multi-objective optimization framework, and utilize evolutionary algorithms such as MOEA/D and NSGA-II to improve the scheduling quality. Meanwhile, the research will also explore the integration with machine learning strategies, introduce reinforcement learning models or graph neural networks in the task ranking stage to predict the risk of task conflicts and resource pressure, and construct a "scheduling experience library" to assist in strategy transfer and adaptive adjustment. Subsequent research will further expand towards dynamic environments, nonlinear cost structures and intelligent feedback mechanisms to achieve the evolution of the task scheduling system from rule-driven to intelligent perception.

## References

[1] Wang R, Nie L, Tan Y. Integrated optimization of train line planning and timetabling: a new method of changing train operation zone and direct-service of cross-line ODs [J]. Transportation Letters, 2025, 17 (4): 666-686.

[2] Shen G, Chi K, Alfarraj O, et al. Task Offloading and Resource Allocation for Wireless Powered Multi-AP Mobile Edge Computing [J]. IEICE Transactions on fundamentals of electronics, communications & computer sciences, 2025 (2): E108/A.

[3] Francisco L.A.F., Digital Archiving of Learner's Credentials Using Discrete Cosine Transform Algorithm [J]. International Research Journal on Advanced Engineering and Management (IRJAEM), 2024, 2 (8): 2505-2516.

[4] Schmid M, Braun S, Sollacher R, et al. Highly efficient encoding for job-shop scheduling problems and its application on quantum computers [J]. Quantum Science and Technology, 2025 (1): 10.

[5] Zhang D, You Y, Liu X. Study of EWM-based project schedule management in tobacco redrying plants [J]. JOURNAL OF COMPUTATIONAL METHODS IN SCIENCES AND ENGINEERING, 2024, 24 (6): 4065–4079.

[6] Taibi S, Toumi L, Bouamama S. Complex network community discovery using fast local move iterated greedy algorithm [J]. The Journal of Supercomputing, 2025, 81 (1): 1-39.

[7] Bouhabza K, Guiatni M, Bouzid Y, et al. Energy-Efficient Passivity-Based Sliding Mode Controller for Small-Scale Quadrotor UAV for Trajectory Tracking [J]. Unmanned Systems, 2025, 13 (03): 837-859.

[8] Balke K N, Dudek C L, Thomas Urbanik I I. Development and Evaluation of Intelligent Bus Priority Concept [J]. Transportation Research Record Journal of the Transportation Research Board, 2000, 1727 (1727): 12-19.

[9] Ma Z, Qi J, Wang M, et al. Time-Varying Formation Tracking Control for Multi-UAV Systems with Directed Graph and Communication Delays [J]. 2021 40th Chinese Control Conference (CCC), 2021: 5436-5441.

[10] Xu T, Huang T Z, Deng L J, et al. Exemplar-based image inpainting using adaptive two-stage structure-tensor based priority function and nonlocal filtering [J]. Journal of Visual Communication and Image Representation, 2022, 83: 103430-.