electronics

Article

TrafSched: Integrating Bayesian Adaptation with LLMs for Traffic Scheduling Optimization

Wentian Fan [1], Li Xu [2], Yongcheng Zeng [3], Siyu Xia [3], Xinyu Cui [3], Junyan Shi [4], Shu Lin [3], Mengyao Zhang [3], Yiwei Guo [5], Xin Zhang [5] and Haifeng Zhang [3,4,*]

1 School of Computer Science, Nanjing University of Posts and Telecommunications, Nanjing 210003, China
2 AI Laboratory, Chongqing Changan Automobile Co., Ltd., Chongqing 400023, China
3 The Key Laboratory of Cognition and Decision Intelligence for Complex Systems, Institute of Automation, Chinese Academy of Sciences, Beijing 100190, China
4 Nanjing Artificial Intelligence Research of IA, Nanjing 210008, China
5 China Academy of Railway Sciences Co., Ltd., Beijing 100081, China
* Correspondence: haifeng.zhang@ia.ac.cn

Abstract

Railway timetabling requires resolving complex scheduling conflicts arising from shared tracks, station capacity limits, and strict safety intervals. Existing optimization or learning-based approaches often struggle to scale or generalize across diverse operational scenarios. We present **TrafSched**, a novel hybrid decision framework that combines a curated strategy library, multi-dimensional conflict prioritization, Bayesian strategy adaptation and an optional Large Language Models (LLMs) integration module. **TrafSched** iteratively detects and resolves conflicts through adaptive strategy selection and backtracking, enabling robust exploration of feasible timetables without costly model retraining. Experiments on real-world-scale datasets involving 50–120 trains show that **TrafSched** consistently outperforms heuristic and reinforcement learning baselines, achieving up to 85.05% conflict-resolution success in the most challenging cases. These results demonstrate **TrafSched**'s effectiveness and scalability for modern railway scheduling operations.

Keywords: railway timetabling; conflict resolution; scheduling; large language models

Academic Editor: Simeone Marino

Received: 28 December 2025
Revised: 24 January 2026
Accepted: 27 January 2026
Published: 5 February 2026

1. Introduction

Railway traffic transportation systems [1] constitute critical infrastructure in modern transportation networks, serving billions of passengers and handling massive freight volumes globally. As urbanization accelerates and environmental sustainability becomes paramount, rail transport has emerged as an indispensable mode characterized by exceptional capacity, operational efficiency, and substantially lower carbon emissions compared to road and air alternatives. Traffic behavior analysis and prediction methods have been extensively studied across various transportation modes [2,3], providing valuable insights for system optimization.

The operational effectiveness of railway systems depends fundamentally on their timetable schedules. The Train Timetabling Problem (TTP) [4,5] represents a complex combinatorial optimization challenge that determines spatiotemporal resource allocation across entire rail networks. A well-optimized timetable not only maximizes infrastructure utilization but also significantly impacts passenger satisfaction and operational costs.

When formulating train timetables [6], operators face critical challenges in resolving conflicts between trains competing for shared tracks while maintaining operational effi-

ciency. These challenges are amplified by multiple factors: diverse train types operating on shared infrastructure, varying track configurations, line-specific constraints, and strict adherence to safety regulations. The complexity intensifies when conflicts are interrelated, requiring coordinated resolution strategies that consider cascading effects across the network.

Currently, train timetable compilation and conflict resolution remain predominantly manual processes completed by dispatchers, leading to several limitations. First, the manual approach suffers from low operational efficiency in timetable creation and modification. Second, the quality varies significantly with individual operator expertise, resulting in inconsistent performance. Third, dispatchers have limited ability to rapidly respond to disruptions or explore alternative solutions. Finally, manual processes struggle to optimize across multiple conflicting objectives simultaneously.

Traditional approaches to address these challenges have primarily relied on mathematical programming models [7], including branch-and-bound methods enhanced with Lagrangian relaxation techniques [8]. Researchers have explored integer programming methods for real-time railway scheduling [9,10] and heuristic approaches such as iterative greedy scheduling [11], dispatcher-focused conflict handling [12], and predictive conflict management [13]. More recently, reinforcement learning (RL) [14] has demonstrated promising performance in complex decision-making problems, leading to applications in railway timetable optimization [15–17].

However, existing approaches face significant limitations. Mathematical optimization methods guarantee optimality under well-defined constraints but suffer from exponential computational complexity for large-scale networks [18]. Heuristic methods offer computational efficiency but often converge to local optima in complex scenarios with interrelated conflicts. Reinforcement learning approaches require extensive training data and struggle to generalize across diverse operational scenarios without costly model retraining [19]. Despite rapid advances in artificial intelligence, limited research has explored the potential of large language models (LLMs) [20] for railway scheduling applications.

As railway networks expand in scale and train frequencies increase to meet growing demand, effective conflict resolution methods become increasingly crucial. Modern railway operations require systems that can maintain reliable service quality, reduce dispatcher workload, minimize delays, and optimize infrastructure utilization. These systems must balance the robustness of algorithmic optimization with the flexibility to adapt to varying operational contexts.

To address these challenges, we present TrafSched, a hybrid decision framework that combines algorithmic rigor with adaptive strategy selection. TrafSched integrates four key components: a curated strategy library containing specialized resolutions for different conflict types, multi-dimensional conflict prioritization to focus on critical issues, Bayesian strategy adaptation for continuous learning from resolution outcomes, and backtracking mechanisms to escape local optima. Additionally, TrafSched incorporates an optional LLMs integration module that augments the base algorithmic framework with contextual reasoning capabilities.

This paper makes four principal contributions to the field of railway conflict resolution:

- Develop a comprehensive railway traffic schedule environment model specifically designed for conflict simulation and resolution, providing a standardized framework for timetabling research.
- Create a diverse strategy library containing 32 specialized resolution strategies for four fundamental conflict types: dwell time violations, running time violations, insufficient intervals, and overtaking conflicts.

- Propose the TrafSched decision framework that significantly outperforms traditional methods through its integration of Bayesian strategy adaptation, multi-dimensional conflict prioritization, and hierarchical backtracking mechanisms.
- Pioneer the integration of LLMs into railway scheduling, demonstrating how natural language reasoning can enhance conflict resolution in complex operational scenarios.

The remainder of this paper is organized as follows. Section 2 reviews related work in mathematical optimization, reinforcement learning, and large language model applications for railway scheduling or transportation. Section 3 formulates the railway timetabling problem and presents our comprehensive conflict detection methodology based on matrix operations. Section 4 details the TrafSched framework, including the curated strategy library, multi-dimensional conflict prioritization, Bayesian strategy adaptation, hierarchical backtracking mechanisms, and the optional LLM-enhanced decision module. Section 5 presents experimental results on real-world datasets involving 50–120 trains, demonstrating TrafSched's effectiveness compared to heuristic and reinforcement learning baselines. Section 6 discusses the contributing factors to TrafSched's performance, current limitations, and future research directions. Section 7 concludes the paper with a summary of key findings and contributions.

2. Related Work

2.1. Mathematical Optimization for Railway Scheduling

Mathematical optimization has been the predominant approach for railway scheduling since the early development of operations research. Mixed-Integer Linear Programming (MILP) formulates railway scheduling as a constrained optimization problem [21], where decision variables represent train movements and constraints encode operational requirements. MILP guarantees optimal solutions under well-defined constraints, making it attractive for safety-critical railway applications.

However, MILP suffers from exponential computational complexity as network size increases [18]. To address this scalability challenge, researchers have developed enhancement strategies that trade optimality guarantees for computational efficiency. Branch-and-bound methods [8] combined with Lagrangian relaxation techniques [22] have achieved significant improvements in solution quality while maintaining computational tractability for medium-scale networks.

Alternative mathematical formulations offer different tradeoffs. Constraint programming [23] provides greater modeling flexibility for complex operational rules but still faces scalability limitations. Dynamic priority rules [24] improve computational efficiency through simplified decision-making processes, sacrificing optimality for speed. Recent advances have integrated train speed adjustments [9] and developed sophisticated integer programming models for multi-line scheduling scenarios [25].

Despite these advances, mathematical optimization methods face fundamental limitations in modern railway operations. First, they require complete problem specification upfront, limiting adaptability to unexpected disruptions. Second, they struggle with interrelated conflicts that span multiple trains and stations, often decomposing problems in ways that sacrifice global optimality. Third, computational requirements grow prohibitively for networks with hundreds of trains and complex operational constraints.

2.2. Reinforcement Learning for Railway Scheduling

Reinforcement learning has emerged as a promising alternative that learns scheduling policies through interaction with simulated railway environments. The sequential decision-making nature of railway operations aligns naturally with RL's framework, where an

agent learns to maximize cumulative rewards by selecting actions that resolve conflicts and improve schedule quality.

Early RL applications focused on training agents to learn rescheduling policies through simulated railway environments [26]. These approaches demonstrated the potential for adaptive conflict resolution without requiring exhaustive problem specification. Deep reinforcement learning methods such as Deep Q-Networks [27] extended this capability to handle the high-dimensional state spaces characteristic of large railway networks, learning complex policies that traditional methods struggle to encode.

Recent studies have demonstrated increasingly sophisticated RL applications for railway scheduling. Liu et al. [15] developed hierarchical deep RL approaches for real-time demand-responsive railway rescheduling. Tang et al. [16] proposed multi-task DRL frameworks that simultaneously optimize multiple objectives, including punctuality, energy consumption, and passenger satisfaction. Yue et al. [17] combined graph neural networks with RL to create extensible solutions that generalize across varying network topologies.

However, RL approaches face significant practical challenges. First, they require extensive training data and computational resources, often needing millions of simulation episodes to converge. Second, trained models struggle to generalize across diverse operational scenarios, requiring retraining when network configurations or operational rules change. Third, the black-box nature of learned policies makes them difficult to interpret and validate for safety-critical applications. Finally, RL methods often converge to local optima in complex scenarios with interrelated conflicts, lacking explicit mechanisms to escape suboptimal solutions.

2.3. Large Language Models in Transportation

The application of large language models in transportation systems represents an emerging research direction with limited but growing exploration. In traffic flow prediction, TrafficBERT has demonstrated how pre-trained transformer models can leverage large-scale traffic data to capture time-series patterns through multi-head self-attention mechanisms [28]. Guo et al. [29] further advanced this direction by proposing xTP-LLM, which transfers multi-modal traffic data into natural language descriptions, showing that language models can provide both competitive accuracy and intuitive explanations for traffic predictions.

In railway operations, LLMs are increasingly being explored for operational support and service enhancement. Li et al. [30] developed LLM4Rail, an augmented platform that integrates natural language reasoning with task-oriented actions through a Question-Thought-Action-Observation framework for railway service consulting. Chen et al. [31] proposed an LLM-based intelligent Q&A system for locomotive maintenance standardization, demonstrating how language models can transform colloquial maintenance records into structured formats. Recent work has also explored LLMs for railway intrusion tracking [32], where multi-agent cooperative frameworks enable edge devices to make real-time safety decisions through natural language reasoning.

Despite these promising applications, integrating LLMs into real-time railway decision-making poses significant challenges [33]. Current LLM applications in transportation remain largely experimental, with limited deployment in operational environments. The direct application of LLMs to railway conflict resolution and scheduling optimization remains largely unexplored in existing literature.

2.4. Summarize and Motivation

While existing approaches have advanced railway scheduling from multiple perspectives, a systematic comparison reveals fundamental limitations that motivate this work.

Mathematical optimization methods provide optimality guarantees, making them the foundation of railway operations research. However, their computational complexity scales exponentially with network size, limiting practical application to small-scale problems.

Reinforcement learning addresses some adaptability limitations by learning scheduling policies through interaction with simulated environments. But they may face three critical challenges to practical deployment: (1) sample inefficiency requiring millions of training episodes, (2) poor generalization across diverse operational scenarios necessitating costly retraining, and (3) lack of interpretability making validation difficult for railway applications.

Emerging applications of LLMs in transportation show promising capabilities in textual information processing, human–AI collaboration, and explainability. However, current LLM research in railway operations has not yet progressed to direct integration into core scheduling and optimization processes.

These gaps directly motivate the TrafSched framework. Our approach integrates curated domain strategies with Bayesian adaptation to achieve interpretable learning without training. Multi-level backtracking mechanisms enable systematic escape from local optima. Finally, our LLM-enhanced module pioneers the integration of large language models into core scheduling decisions, demonstrating how LLMs' reasoning can complement algorithmic optimization.

3. Problem Formulation and Modeling

The railway scheduling problem involves determining optimal departure and arrival times for multiple trains operating on a shared railway network while satisfying numerous operational constraints and safety requirements. This section presents our comprehensive railway timetable environment model, which provides a standardized framework for simulating and resolving railway conflicts. This formulation addresses a significant gap in existing research, where simplified models often fail to capture the complexity of real-world railway operations.

3.1. Preliminary

We model the railway network as a directed graph $\mathcal{G} = (\mathcal{S}, \mathcal{K})$, where stations correspond to nodes and track segments correspond to edges connecting consecutive stations. This graph-based representation naturally captures the sequential nature of railway operations while providing a mathematical foundation for constraint formulation and conflict analysis. Let $\mathcal{T} = \{1, 2, \ldots, N\}$ denote the set of trains, $\mathcal{S} = \{1, 2, \ldots, M\}$ represent the set of stations, and $\mathcal{K} = \{1, 2, \ldots, M-1\}$ indicate the track segments connecting consecutive stations. For each train $i \in \mathcal{T}$ and station $s \in \mathcal{S}$, we define event times (arrival, departure, or passage) that must satisfy operational and safety constraints.

3.2. The Constraints and Optimization Objective

To ensure safe railway operations, the timetable must satisfy four fundamental types of constraints: dwell time constraints, running time constraints, interval time constraints, and overtaking constraints. Violations of these constraints result in conflicts that must be resolved to produce a feasible timetable.

3.2.1. Dwell Time Constraints

Dwell time constraints refer to the limitations on the amount of time a train can spend at a station. When a train stops at a station, it must adhere to this constraint. We introduce a binary indicator variable $\delta_{i,s}$ to represent the stopping pattern of train i at station s:

$$\delta_{i,s} \in \{0, 1\}, \quad \forall i \in \mathcal{T}, \forall s \in \mathcal{S} \tag{1}$$

where $\delta_{i,s} = 1$ if train i has a scheduled stop at station s, and $\delta_{i,s} = 0$ otherwise. For trains with scheduled stops, the dwell time $st_{i,s}$ must satisfy operational bounds:

$$\delta_{i,s} \cdot r_{i,s}^{\min} \leq \delta_{i,s} \cdot st_{i,s} \leq \delta_{i,s} \cdot r_{i,s}^{\max}, \quad \forall i \in \mathcal{T}, \forall s \in \mathcal{S} \tag{2}$$

where $st_{i,s}$ means the dwell time of train i at station s, $r_{i,s}^{\min}$ and $r_{i,s}^{\max}$ represent the minimum and maximum allowable dwell times for train i at station s, respectively, these limits are determined by operational requirements.

3.2.2. Running Time Constrains

Running time constraints govern the travel time of trains between consecutive stations, ensuring that trains operate within safe speed limits while maintaining adherence to the schedule. We define a track segment $k \in \mathcal{K}$ as the railway section connecting station k to station $k+1$. For each train i and segment k, the running time $p_{i,k}$ must satisfy:

$$p_{i,k}^{\max} = p_{i,k}^{\min} + p_{i,k}^{\text{free}} \tag{3}$$

$$p_{i,k}^{\min} \leq p_{i,k} \leq p_{i,k}^{\max}, \quad \forall i \in \mathcal{T}, \forall k \in \mathcal{K} \tag{4}$$

where $p_{i,k}^{\min}$ represents the minimum running time determined by maximum safe speed, $p_{i,k}^{\text{free}}$ represents the additional running time buffer, and $p_{i,k}^{\max}$ represents the maximum allowed running time. Equation (3) defines the upper bound as the sum of the minimum running time and free running time buffer. Equation (4) constrains the actual running time to this feasible range, ensuring safe train operation.

3.2.3. Interval Time Constraints

Interval time constraints ensure safe separation between consecutive trains at stations. Given the periodicity of the railway timetables with cycle time T_c, firstly, we define the periodic time difference function φ as:

$$\varphi(t_1, t_2) = (t_1 - t_2 + T_c) \bmod T_c \tag{5}$$

This function quantifies the minimum temporal distance between two events within a periodic scheduling cycle, enabling consistent constraint formulation across cycle boundaries. We consider three types of fundamental events $\mathcal{E}$ for train operations at a station:

- $\mathcal{E} = arr$: Train arrival at the station;
- $\mathcal{E} = dep$: Train departure from the station;
- $\mathcal{E} = pass$: train passage through the station without stopping.

Note that arrival and departure events occur in pairs for stopping trains. To ensure operational safety and prevent collisions, adjacent trains i and j must satisfy the interval time constraints at station s:

$$\varphi(T_{i,s}^{\mathcal{E}_i}, T_{j,s}^{\mathcal{E}_j}) \geq \tau_{\mathcal{E}_i,\mathcal{E}_j}^{s}, \quad \forall i, j \in \mathcal{T}, \forall s \in \mathcal{S} \tag{6}$$

where $T_{i,s}^{\mathcal{E}_i}$ denotes the time of event $\mathcal{E}_i$ for train i at station s, and $\tau_{\mathcal{E}_i,\mathcal{E}_j}^{s}$ represents the minimum required separation time between event types $\mathcal{E}_i$ and $\mathcal{E}_j$ at station s. In our railway model, there are a total of 7 different interval time constraints, which are combined constraint relations between *arr*, *dep* and *pass* events of two adjacent trains. Only the combination of (*dep*, *arr*) and (*arr*, *dep*) events does not apply this constraint.

3.2.4. Overtaking Constrains

Overtaking constraints prevent trains from changing their relative order within track segments, which is essential for maintaining operational safety in single-track or capacity-constrained sections. For any two consecutive trains i and j and any segment $k \in \mathcal{K}$, the relative ordering must be preserved:

$$(T_{i,k}^{dep} - T_{j,k}^{dep})(T_{i,k+1}^{arr} - T_{j,k+1}^{arr}) \geq 0, \quad \forall i, j \in \mathcal{T}, \forall k \in \mathcal{K} \tag{7}$$

where $T_{i,k}^{dep}$ represents the departure time of train i from station k, and $T_{i,k+1}^{arr}$ denotes the arrival time of train i at station $k+1$. This constraint ensures that if train i departs station k before train j, then train i must also arrive at station $k+1$ before train j, thereby preventing overtaking within the segment. An overtaking conflict occurs when the product in Equation (7) becomes negative, indicating that the relative order of trains has reversed.

3.2.5. Optimization Objective

The optimization objective of the railway timetabling problem is to minimize the total number of operational conflicts in the schedule, subject to the four constraint categories described above. Let $C_{conflicts}$ denote the total number of *conflicts* in a timetable. The problem is formulated as:

$$\min_{T_{i,s}^{\mathcal{E}}} \quad C_{conflicts}, \quad \text{s.t. Equations (2), (4), (6), (7)} \tag{8}$$

While real-world railway operations involve multiple competing objectives, including passenger delays, energy consumption, infrastructure utilization, and crew scheduling costs—we adopt conflict count minimization as the primary objective for three principled reasons.

- **Fundamental feasibility requirement.** Conflict resolution represents a prerequisite for any feasible timetable, regardless of other optimization criteria. A schedule with unresolved conflicts violates safety constraints and cannot be operationally deployed. In this sense, conflict minimization serves as a necessary foundation before multi-objective optimization can be meaningfully applied.
- **Computational tractability.** The single-objective formulation enables systematic exploration of the solution space through our adaptive decision framework without the exponential complexity introduced by Pareto frontier exploration in multi-objective optimization. This design choice aligns with our goal of developing a practical framework that can handle real-world scale problems (50–120 trains).
- **Industrial practice alignment.** In operational railway dispatching, conflict resolution is typically prioritized as the first-stage objective, with other performance metrics optimized subsequently. Our formulation reflects this hierarchical decision-making process commonly adopted in railway control centers.

3.3. Timetable Matrix and Conflict Detection

To systematically detect and analyze conflicts in the timetable, we design a matrix-based representation that captures the temporal relationships between trains across the railway network.

Definition 1 (Railway Timetable Matrix)**.** *For a railway system with S stations and N trains, the timetable matrix $\mathbf{M} \in \mathbb{R}^{(2S-2)\times N}$ is defined as follows:*

- *Rows $(2S-2)$: Each intermediate station ($S-2$ in total) contributes two rows representing arrival and departure events, while each terminal station contributes a single row (departure for the origin, arrival for the destination).*
- *Columns (N): Correspond to the set of trains $\mathcal{T} = 1, 2, \ldots, N$.*
- *Elements (m_{ij}): Denote the scheduled event time (arrival or departure) of train j at the station associated with row i.*

For example, in a 3-station network A–B–C, the matrix $\mathbf{M}$ is:

$$\mathbf{M} = \begin{pmatrix} t^1_{A,dep} & t^2_{A,dep} & \cdots & t^N_{A,dep} \\ t^1_{B,arr} & t^2_{B,arr} & \cdots & t^N_{B,arr} \\ t^1_{B,dep} & t^2_{B,dep} & \cdots & t^N_{B,dep} \\ t^1_{C,arr} & t^2_{C,arr} & \cdots & t^N_{C,arr} \end{pmatrix}$$

where $t^j_{s,e}$ denotes the scheduled time of event e for train j at the station s.

This timetable matrix representation provides a compact encoding of the entire timetable, enabling systematic analysis of temporal relationships and detection of conflicts through matrix operations.

To extract temporal relationships between adjacent trains, we define the matrix $\mathbf{P}_1 \in \mathbb{R}^{N\times(N-1)}$ to get the difference in time:

$$\mathbf{P}_1 = \begin{pmatrix} -1 & & & \\ 1 & -1 & & \\ & 1 & \ddots & \\ & & \ddots & -1 \\ & & & 1 \end{pmatrix} \tag{9}$$

Then, we can get the sequence relation about the train's time through the matrix $\mathbf{P}_1$:

$$\mathbf{M} = (m_1, m_2, m_3, \cdots, m_N) \tag{10}$$

$$\mathbf{M}_2 = \mathbf{M}\mathbf{P}_1 = (m_2 - m_1, m_3 - m_2, \cdots, m_N - m_{N-1}) \tag{11}$$

where m_j represents the j-th column of $\mathbf{M}$. Since overtaking detection requires only the relative ordering information rather than specific time differences, we map the positive number of matrix $\mathbf{M}_2$ to 1 and the negative number to -1 to get matrix $\mathbf{M}_3$:

$$\mathbf{M}_3[i,j] = \text{sign}(\mathbf{M}_2[i,j]) = \begin{cases} 1, & \text{if } \mathbf{M}_2[i,j] > 0 \\ 0, & \text{if } \mathbf{M}_2[i,j] = 0 \\ -1, & \text{if } \mathbf{M}_2[i,j] < 0 \end{cases} \tag{12}$$

To identify overtaking conflicts between consecutive track segments, we extract the ordering relationships at departure and arrival events. We define projection matrices $\mathbf{P}_2, \mathbf{P}_3 \in \mathbb{R}^{(2S-3)\times(2S-2)}$:

$$\mathbf{P}_2 = \begin{pmatrix} 1 & 0 & & \\ & 1 & 0 & \\ & & \ddots & \ddots \\ & & & 1 \quad 0 \end{pmatrix} \tag{13}$$

$$\mathbf{P}_3 = \begin{pmatrix} 0 & 1 & & & \\ & 0 & 1 & & \\ & & \ddots & \ddots & \\ & & & 0 & 1 \end{pmatrix} \tag{14}$$

For the matrices $\mathbf{P}_2$ and $\mathbf{P}_3$, we have the following derivation:

$$\mathbf{M}_4 = \mathbf{P}_2\mathbf{M}_3 = (m'_1, m'_2, m'_3, \cdots, m'_{2S-3})^T \tag{15}$$

$$\mathbf{M}_5 = \mathbf{P}_3\mathbf{M}_3 = (m'_2, m'_3, m'_4, \cdots, m'_{2S-2})^T \tag{16}$$

where $\mathbf{M}_4$ captures the departure ordering and $\mathbf{M}_5$ captures the arrival ordering at stations. Through the product of the time difference between adjacent stations, we can judge whether there is an overtaking conflict between two adjacent stations. For this purpose, we define a Hadamard Product.

Definition 2 (Hadamard Product)**.** *For matrices* $\mathbf{A}, \mathbf{B} \in \mathbb{R}^{m\times n}$ *with elements* a_{ij} *and* b_{ij}*, respectively, the Hadamard product* $\mathbf{A} \odot \mathbf{B}$ *is defined as:*

$$(\mathbf{A} \odot \mathbf{B})_{ij} = a_{ij} \cdot b_{ij}, \quad \forall i \in \{1, \ldots, m\}, \forall j \in \{1, \ldots, n\} \tag{17}$$

The product $\mathbf{M}_4 \odot \mathbf{M}_5$ yields positive values when the train order is consistent (no overtaking) and negative values when the order is reversed (overtaking conflict). To extract the relevant conflict information, we introduce a mask matrix $\mathbf{P}_4 \in \mathbb{R}^{(2S-3)\times(2S-3)}$:

$$\mathbf{P}_4 = \begin{pmatrix} 1 & & & & \\ & 0 & & & \\ & & 1 & & \\ & & & \ddots & \\ & & & & 1 \end{pmatrix} \tag{18}$$

The final conflict detection matrix is computed as:

$$\mathbf{M}_6 = \mathbf{P}_4(\mathbf{M}_4 \odot \mathbf{M}_5) \tag{19}$$

In the resulting matrix $\mathbf{M}_6$, negative entries indicate overtaking conflicts between specific train pairs in specific segments. The total number of overtaking conflicts equals the count of negative entries in $\mathbf{M}_6$.

A similar matrix-based analysis can be performed for detecting insufficient interval conflicts by comparing event time differences with minimum separation requirements $\tau^s_{\mathcal{E}_i,\mathcal{E}_j}$.

4. Methodology

This section presents the **TrafSched** framework, a hybrid decision system that resolves railway scheduling conflicts through adaptive multi-strategy optimization. First, we describe the curated strategy library for conflict resolution. Then, we detail the core algorithmic components that enable adaptive strategy selection and systematic exploration of the solution space. Finally, we present an optional LLM-enhanced decision module that augments the base framework with contextual reasoning capabilities.

4.1. Framework Overview

Figure 1 illustrates the architecture of the TrafSched framework. The system operates iteratively, beginning with conflict detection on an initial timetable T. For each prioritized

conflict, the system selects resolution strategies from a curated library, guided by Bayesian adaptation that learns from historical success rates.

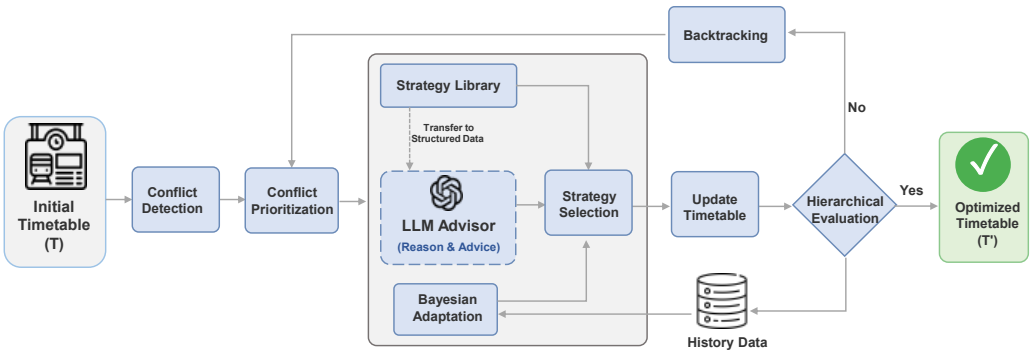

Figure 1. The architecture of the TrafSched framework. The LLM Advisor (dashed box) represents an optional enhancement module. The framework operates effectively with or without LLMs enhancement.

The **TrafSched** framework supports dual-pathway strategy selection, and sharing the same core components. **(1) Base Mode** relies purely on Bayesian statistics for strategy selection, and **(2) LLM-Enhanced Mode** augments the base framework with contextual reasoning (Section 4.4). The modes differ only in the strategy selection mechanism, where LLM-Enhanced Mode incorporates language model analysis alongside Bayesian rankings.

After applying a selected strategy, the system evaluates the resulting timetable using a hierarchical evaluation that balances local conflict resolution with global schedule quality. If the evaluation is unsatisfactory, the system employs backtracking mechanisms to explore alternative solution paths, preventing premature convergence to local optima.

4.2. Conflict Resolution Strategy Library

In the railway scheduling system, violations of operational constraints can lead to four distinct categories of conflicts. To address these issues, we have developed dedicated resolution strategies for each category. The essence of these strategies lies in targeted adjustments to train timetables that satisfy operational constraints while resolving conflicts, as shown in Figure 2. Unlike black-box learning models, our approach yields interpretable, domain-validated solutions directly informed by established operational practices.

4.2.1. Strategies for Dwell Time Conflicts

A dwell time conflict occurs when the actual stop time $st_{i,s}$ lies outside the permissible range $[r_{i,s}^{\min}, r_{i,s}^{\max}]$. As this issue affects only a single train at a specific station, resolution is straightforward. To handle this conflict by increasing or decreasing the dwell time, as illustrated in the Figure 2a.

4.2.2. Strategies for Running Time Conflicts

A running time conflict arises when a train's travel time $p_{i,k}$ between consecutive stations falls outside the allowable range $[p_{i,k}^{\min}, p_{i,k}^{\max}]$. Similar to dwell time violations, each case involves only a single train. Therefore, we also propose two corrective strategies: increasing or decreasing the running time accordingly. The goal is to adjust the train's running time to the acceptable boundary. Figure 2b illustrates an example where running time is increased to meet the lower bound.

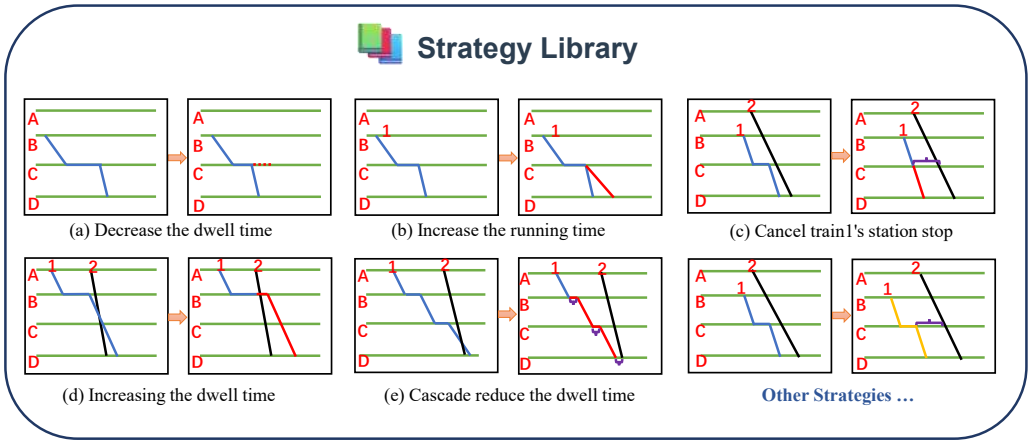

Figure 2. Some detailed examples in the Strategy Library.

4.2.3. Strategies for Insufficient Interval Conflicts

Interval conflicts occur when the temporal separation between events of adjacent trains at the same station is insufficient. Corresponding to the 7 interval constraints defined in Section 3.2.3, we developed 10 resolution strategies that adjust train schedules to satisfy minimum separation requirements.

As an example, in Figure 2c, one direct approach cancels Train 1's stop at the conflict station, converting the arrival-departure pair into a single passage event. These strategies consider only the event type of the adjusted train, allowing them to handle multiple conflicts.

4.2.4. Strategies for Overtaking Conflicts

Overtaking conflicts refer to the reverse order of two adjacent trains entering and exiting a segment, violating Equation (7). Which is prohibited in railway operations. Given the complexity of overtaking scenarios, we have developed 13 distinct adjustment strategies that comprehensively address different conflict configurations while minimizing schedule perturbation. Each strategy calculates the required temporal adjustment based on the principle of minimal timetable modification.

Figure 2d shows a typical case where Trains 1 and 2 have an overtaking conflict in Segment B. We resolve this by extending Train 1's dwell time at Station B. The required extension T_{ext} is:

$$T_{\text{ext}} = T_{2,B}^{pass} - T_{1,B}^{dep} + \tau_{pass_2,dep_1}^{B} \tag{20}$$

where $T_{2,B}^{pass}$ is Train 2's passage time at Station B, $T_{1,B}^{dep}$ is Train 1's original departure time at Station B, and τ_{pass_2,dep_1}^{B} is the minimum required time interval between Train 2's passage and Train 1's departure at Station B.

4.2.5. The Advanced Strategies

While basic strategies effectively resolve isolated violations, they may cause premature convergence to local optima in scenarios with interrelated conflicts. When resolving one conflict creates others, basic strategies can become trapped in repetitive cycles. To address this limitation, we designed 5 advanced strategies with broader adjustment capabilities that coordinate changes across multiple trains.

- Cascade Dwell Time Adjustment Strategies: These strategies adjust dwell times across multiple consecutive stations, respecting dwell time bounds $[r_{i,s}^{\min}, r_{i,s}^{\max}]$ at each station while accumulating sufficient temporal shift to resolve the target conflict. An example can be seen in Figure 2e.
- Systematic Timetable Adjustment Strategies: These involve coordinated time shifts for multiple trains in opposite directions, by redistributing temporal buffers across multiple trains to resolve complex conflicts.
- Opportunity Window Insertion Strategy: This strategy searches for the top N largest time gaps within a given window, evaluates insertion feasibility for the conflicting train, and repositions it at the first conflict-free opportunity while satisfying all constraints.

4.3. The Railway Scheduling Decision Method

Having established the strategy library in Section 4.2, we now present the core algorithmic components that enable TrafSched to *intelligently* select and apply strategies for conflict resolution. The challenge lies not merely in having diverse resolution strategies available, but in determining *which* strategy to apply to *which* conflict at *which* decision point in the optimization process.

The core innovation of TrafSched lies in its adaptive decision method, which integrates four basic components to efficiently detect and resolve train conflicts. Unlike traditional optimization methods or learning-based approaches that necessitate extensive retraining, we achieve robust performance through dynamic strategy adaptation guided by accumulated operational experience.

4.3.1. Multi-Dimensional Conflict Prioritization

Railway schedules often face simultaneous conflicts of varying types and severities, making prioritization essential. The system maps conflicts to a multi-dimensional priority space, computing the priority score for conflict c using:

$$P(c) = \alpha \cdot T(c) + \beta \cdot L(c) - \gamma \cdot S(c) - \delta \cdot A(c) + \epsilon \cdot E(c) \tag{21}$$

where $T(c)$, $L(c)$, $S(c)$, $A(c)$, and $E(c)$ capture conflict type, train type, station position, attempt count, and exploration priority, respectively.

The weights $\alpha, \beta, \gamma, \delta, \epsilon \in (0, 1)$ are initialized as $\alpha^{(0)} = 0.30$, $\beta^{(0)} = 0.25$, $\gamma^{(0)} = 0.20$, $\delta^{(0)} = 0.15$, $\epsilon^{(0)} = 0.10$, and adapted periodically to reflect the effectiveness of each priority dimension. Every $K = 50$ resolution steps, we update each weight based on its contribution to successful conflict resolution:

$$w_i^{(t+1)} = w_i^{(t)} \cdot \left(1 + \eta \cdot \frac{\rho_i - \bar{\rho}}{\bar{\rho}}\right) \tag{22}$$

where $w_i \in \{\alpha, \beta, \gamma, \delta, \epsilon\}$, ρ_i measures the conflict resolution rate when prioritizing dimension i in the recent window, $\bar{\rho}$ is the average resolution rate across all dimensions, and $\eta = 0.05$ is the adaptation rate. After updating, weights are normalized to maintain $\sum_i w_i = 1$. This mechanism strengthens weights corresponding to dimensions that consistently identify resolvable conflicts, while preventing any single dimension from dominating through bounded adaptation ($\eta \ll 1$). The negative signs for γ and δ reflect that conflicts at later stations and those with many previous attempts should receive lower priority, as they often indicate structurally difficult problems.

4.3.2. Bayesian Strategy Adaptation

For each prioritized conflict, TrafSched supports dual-pathway strategy selection: (1) selecting a policy based on Bayesian statistics adaptation, or (2) an LLM-enhanced hybrid decision method that combines policy rankings with contextual reasoning. The second approach is elaborated in Section 4.4.

For the Bayesian adaptation pathway, we model the outcome of applying strategy i as a Bernoulli random variable with an unknown success probability θ_i. We assume a Beta prior over θ_i:

$$\theta_i \sim \text{Beta}(\alpha_0, \beta_0) \tag{23}$$

where α_0 and β_0 represent prior pseudo-counts of successes and failures, respectively. In our implementation, we adopt a uniform prior with $\alpha_0 = \beta_0 = 1$, reflecting no initial bias toward any strategy.

After observing s_i successful and f_i failed applications of strategy i, the posterior distribution follows the conjugate Beta update:

$$\theta_i \mid \mathcal{D}_i \sim \text{Beta}(\alpha_0 + s_i, \beta_0 + f_i) \tag{24}$$

For strategy selection, we use the posterior mean as the success rate estimate for strategy ranking:

$$SR_{i,t} = \mathbb{E}[\theta_i \mid \mathcal{D}_i] = \frac{\alpha_0 + s_i}{\alpha_0 + \beta_0 + s_i + f_i} \tag{25}$$

With the uniform prior ($\alpha_0 = \beta_0 = 1$), this posterior mean admits an efficient incremental update form:

$$SR_{i,t+1} = \frac{SR_{i,t} \cdot (n_i - 1) + \mathbb{I}(\text{success})}{n_i} \tag{26}$$

where $\mathbb{I}(\text{success})$ equals 1 if strategy i successfully resolves the target conflict and 0 otherwise, and $n_i = s_i + f_i$ denotes the total number of applications.

This Bayesian updating mechanism provides two advantages. First, it maintains exploration of all strategies rather than committing prematurely to seemingly optimal choices, as the success rate estimates continuously evolve with new evidence. Second, it provides interpretable guidance for strategy selection, as the posterior mean directly reflects cumulative empirical performance.

4.3.3. Hierarchical Strategy Evaluation

To avoid railway schedule optimization in local areas that degrade global performance, we developed a hierarchical evaluation method that considers both local and network-wide effects. The evaluation score for strategy s applied to a conflict at station station is:

$$E(s) = \begin{cases} \alpha_2 \big(C_{\text{total}} - C_{\text{down}}(s)\big), & \text{if station} = 0 \\ \alpha_1 \big(C_{\text{sec}}(\text{station} - 1) - C_{\text{up}}(s)\big) & \\ \quad + \ \alpha_2 \big(C_{\text{total}} - C_{\text{sec}}(\text{station} - 1) - \ C_{\text{down}}(s)\big), & \text{otherwise} \end{cases} \tag{27}$$

where:

- C_{total}: Total conflicts in the current timetable before applying strategy s
- $C_{\text{sec}}(\text{station} - 1)$: Conflicts in the network section up to station (station $- 1$)
- $C_{\text{up}}(s)$: New conflicts created upstream (earlier stations) after applying s
- $C_{\text{down}}(s)$: New conflicts created downstream (later stations) after applying s
- α_1, α_2: Weighting coefficients for upstream and downstream effects

For conflicts at the origin station, the evaluation considers only downstream effects. For conflicts at subsequent stations, the evaluation balances upstream and downstream impacts using weights α_1 and α_2. Typically, $\alpha_1 > \alpha_2$ to penalize upstream conflicts more heavily, as these affect earlier stages of the schedule and can cascade through the network. The hierarchical evaluation enables the system to balance local optimality with global consistency.

4.3.4. Multi-Level Backtracking and Exploration

Railway conflict resolution frequently encounters local optima where all available strategies fail to reduce conflicts or create cycles of recurring conflicts. To address this challenge, we implement backtracking and exploration mechanisms that enable strategic jumps in the solution space, allowing the search process to escape suboptimal regions.

We take the resolution process as exploring a tree structure $\mathcal{T} = (V, E)$ where each node $v \in V$ represents a distinct timetable state and each edge $(v_i, v_j) \in E$ represents a state transition caused by applying a resolution strategy. The system maintains checkpoints at regular intervals, storing promising timetable states that represent partial progress toward conflict resolution. Then, backtracking jumps between nodes with probability:

$$P(V_{\text{target}}) = \begin{cases} 0.3, & \text{if } V_{\text{target}} = V_{\text{global\_best}} \\ 0.7 \cdot P_{\text{triangular}}(i), & \text{if } V_{\text{target}} \in \text{Checkpoints} \end{cases} \tag{28}$$

This probability distribution balances exploitation (returning to the global best state with 30% probability) with exploration (visiting previous checkpoints with 70% probability, distributed according to a triangular distribution [34], favoring earlier checkpoints). This prevents the system from becoming trapped in repetitive cycles while ensuring promising solution pathways receive appropriate attention. We conducted a sensitivity analysis over the exploitation–exploration ratio (e.g., 20/80 to 40/60), observing stable performance trends. It indicates that the benefit arises from the mechanism rather than fine-grained parameter tuning. Therefore, we chose the representative 30/70 setting.

Algorithm 1 presents the complete pseudo-code for the railway scheduling decision method, integrating all four components. This approach balances exploitation of successful strategies with exploration of alternative solution paths, enabling robust performance across real-world railway scheduling scenarios.

4.4. LLM-Enhanced Decision Module

Recent advances in large language models have demonstrated remarkable capabilities in contextual reasoning, multi-step planning, and incorporating domain knowledge expressed in natural language [20]. To explore whether these capabilities can enhance railway scheduling, we developed an optional LLM-enhanced decision module that augments the base algorithmic framework.

4.4.1. Strategy Knowledge Representation

To enable LLMs to reason about railway scheduling strategies, we must bridge the gap between code implementations and natural language representations that LLMs can process. We use a structured knowledge representation scheme that transforms each strategy in the library into a natural language description while preserving essential technical details.

For example, A strategy *Overtaking Resolution by Dwell Extension* is represented as:

Identifier: Overtaking Resolution by Dwell Extension (O5)

Applicability: Overtaking conflict where Train A (slower) departs the station before Train B (faster) but Train B arrives at the next station first.

Operation: Extend Train A's dwell time at the conflict station until Train B passes, ensuring Train B departs before Train A.

Requirements: Train A ID, Train B ID, conflict station index, Train B passage time, minimum separation interval.

Effects: Resolves overtaking conflict by adjusting temporal ordering. May increase Train A's delay. Downstream conflicts are possible if extended dwell violates the maximum dwell time constraint.

This representation approach leverages application in knowledge distillation and structured prompting for LLMs [35,36]. By expressing strategies in natural language with clear operational semantics, we enable LLMs to reason about strategy applicability, consider potential issues and predict the possible impacts.

Algorithm 1 Railway Scheduling Decision Algorithm

Input: Train timetable T with conflicts, line information L, maximum steps MAX_STEP
Output: Optimized train timetable T'
Initialize strategy libraries, statistical trackers, backtracking system, and conflict history
Preprocess single-train internal conflicts
$C \leftarrow \text{GetConflicts}(T, L)$ ▷ Initial conflict set
$TS \leftarrow \text{GetTrainStops}(T)$ ▷ Retrieve train stop information
step $\leftarrow 0$
while step $<$ MAX_STEP $\wedge$ $|C| > 0$ **do**
    Update conflict set C based on modified stations
    Update conflict history and backtracking management
    if stagnation threshold exceeded **then**
        Apply backtracking mechanism
    end if
    if stagnation detected and conditions met **then**
        Apply disturbance mechanism
    end if
    if conflict duration exceeds threshold **then**
        Enable deep exploration mode
    end if
    $SC \leftarrow \text{SortConflictsByPriority}(C)$
    for each conflict $c \in SC$ **do**
        Select appropriate strategy set based on conflict type
        Sort strategies by historical success rate
        while conflict unresolved and strategies remain **do**
            Try applying strategies until resolution or all attempted
        end while
        Apply best-performing strategy based on evaluation
    end for
    step $\leftarrow$ step $+ 1$
end while
Verify final timetable state
return Optimized timetable T'

4.4.2. Contextual Conflict Analysis and Reasoning

The base algorithmic framework selects strategies primarily based on quantitative metrics such as Bayesian historical success rates and conflict counts. When the LLM module is enabled, it augments this selection process by providing contextual reasoning about strategy applicability. We employ Chain-of-Thought (CoT) prompting [37] to elicit explicit reasoning from LLMs about conflict resolution.

For each conflict requiring LLM consultation, we construct a prompt containing three components: conflict characterization (type, severity, involved trains), operational context (train priorities, station network status), and candidate strategies. The CoT prompt instructs the LLMs to: (1) identify key factors affecting the conflict, (2) analyze implications of candidate strategies considering operational context, (3) assess potential side effects, and (4) provide ranked recommendations with explicit justifications. This structured reasoning process generates interpretable outputs that capture nuances beyond algorithmic metrics, particularly for complex conflicts where multiple strategies have similar historical performance but different operational implications.

The LLM output consists of a ranked strategy list with associated confidence scores and textual justifications. These justifications enable human dispatchers to understand the reasoning behind recommendations, supporting validation by human expert.

These recommendations will be integrated into the strategy selection process described in Section 4.3.2, Specifically, when LLM consultation is triggered, the system first attempts strategies from the LLM-recommended list in order of confidence scores. If these strategies fail to resolve the conflict, the system falls back to the algorithmic ranking based on historical success rates, ensuring robust performance even when LLM recommendations are suboptimal.

5. Experiments

This section evaluates the TrafSched framework on real-world railway scheduling datasets. We first describe the experimental setup, then present comprehensive performance comparisons demonstrating TrafSched's effectiveness across varying problem scales, followed by a detailed analysis of the LLM-enhanced module's contribution to conflict resolution performance.

5.1. Experimental Setup

We evaluate TrafSched framework using a human-expert-designed real-world railway schedule with varying complexity levels. The experimental datasets comprise timetables ranging from 50 to 120 trains, with corresponding event counts between 1160 and 2890 events, covering a 24-h scheduling horizon. Despite being designed by professionals, these timetables contain numerous unresolved conflicts due to the inherent difficulty of manual conflict resolution under complex operational constraints.

For each experimental scenario, we measure initial conflict counts, remaining conflicts, and calculate the conflict resolution rate as the primary performance metric. We compare four methods: (1) **Heuristic**: rule-based scheduling methods representing current practice in railway systems, (2) **PPO**: Proximal Policy Optimization, a state-of-the-art reinforcement learning baseline for sequential decision-making, (3) **TrafSched**: our base decision framework relying purely on algorithmic optimization, and (4) **TrafSched+LLMs**: the TrafSched with LLMs enhancement using Qwen3-32B [38], selected for its strong performance on reasoning tasks and contextual understanding. All methods run for a maximum of 1024 steps. Performance is measured by conflict resolution rate.

Our experimental design compares TrafSched against two representative optimization paradigms rather than exhaustively evaluating all possible algorithms. This design choice reflects three considerations:

- Heuristic methods represent operational practice in railway systems. Our selected heuristic baseline implements rule-based conflict resolution commonly used in railway control centers, providing a fair representation of this paradigm's capabilities.
- Reinforcement learning (PPO) represents state-of-the-art learning-based approaches for sequential decision problems. PPO was selected as a widely-recognized effective

method, though all RL algorithms share similar limitations in our context, such as sample inefficiency, poor cross-scenario generalization, and lack of interpretability.

- Mathematical optimization methods are not included as direct baselines because: (1) our problem focuses on resolving conflicts in existing timetables rather than generating optimal schedules from scratch, it is a different problem; (2) constraint programming methods faces prohibitive computational complexity at large scale trains as documented in Section 2.1.

5.2. Result Analysis

Table 1 presents a comprehensive performance comparison of different railway scheduling methods across various timetable scales. The results demonstrate the superior effectiveness of our proposed TrafSched framework compared to baseline methods across all experimental scenarios.

Table 1. Performance Comparison of Railway Scheduling Methods.

Timetable Scale			Success Rate (%)			
Trains	**Events**	**Conflicts**	**Heur.**	**PPO**	**TrafS.**	**TrafS. + LLMs**
50	1160	42	88.90	64.29	100.00	100.00
60	1429	53	79.24	60.38	88.68	90.57
70	1683	53	47.17	47.17	98.11	96.23
80	1906	70	42.86	58.57	80.00	84.29
90	2113	71	42.25	57.75	74.65	85.92
100	2335	79	44.30	44.30	88.61	89.87
110	2597	92	46.74	30.43	76.09	81.52
120	2890	107	36.48	36.48	79.44	85.05

The traditional heuristics achieve only 36.48–88.90% success rates, with performance degrading severely as network complexity increases. The PPO baseline performs similarly poorly (30.43–64.29%), demonstrating that pure learning-based approaches struggle with the combinatorial structure and strict constraints of railway scheduling. In contrast, our method achieves 74.65–100% conflict resolution success rates through adaptive strategy selection and intelligent exploration. The LLM-enhanced TrafSched further improves performance, particularly in complex scenarios: achieving 85.92% versus 74.65% for the 90-train case and resolving 6 additional conflicts in the 120-train scenario. These improvements demonstrate that LLMs' contextual reasoning effectively complements algorithmic optimization when conflicts involve intricate operational constraints and multiple interacting trains.

To understand how TrafSched achieves superior performance, we analyze the conflict resolution dynamics over the optimization process. Figure 3 illustrates the evolution of conflict counts for the 120-train scenario, comparing TrafSched and LLM-enhanced method over 1024 iterations.

Both the TrafSched methods rapidly reduce conflicts within the first 100 steps. However, TrafSched with LLMs exhibits superior convergence in later stages. The "Best curves" show that it more consistently discovers high-quality solutions; this behavior validates the effectiveness of LLM-enhanced strategic decision-making in navigating complex search landscapes where purely algorithmic approaches may stagnate.

Figure 3. Conflict resolution dynamics for the 120-train scenario. Lines show conflict counts over 1024 iterations for TrafSched (Blue) and LLM-enhanced TrafSched (Green).

6. Discussion

6.1. The Contributing to TrafSched's Performance

TrafSched's superior performance stems from the synergistic integration of its adaptive components. Multi-dimensional prioritization focuses effort on foundational conflicts, while Bayesian adaptation learns from resolution outcomes without offline training. Hierarchical evaluation prevents local optima by considering network-wide conflict propagation, and multi-level backtracking enables strategic exploration when optimization stalls. And LLMs integration also provides consistent improvements, particularly in complex scenarios with interrelated conflicts.

6.2. Limitations and Future Directions

Current limitations include evaluation scope (up to 120 trains), focus on offline optimization rather than real-time rescheduling, and single-objective formulation that does not capture trade-offs between passenger delays, energy consumption, and crew schedules.

Future work should address scalability validation on larger networks (200+ trains), extend to real-time rescheduling with bounded latency, incorporate multi-objective optimization, and explore neuro-symbolic approaches combining LLMs' reasoning with formal constraint verification for enhanced safety guarantees.

7. Conclusions

This paper presents TrafSched, a hybrid decision framework for railway scheduling optimization that addresses critical challenges in conflict resolution through adaptive algorithmic components and optional large language model enhancement. We make four principal contributions: a comprehensive railway scheduling environment model with matrix-based conflict detection, a curated strategy library containing 32 specialized resolution strategies, an adaptive algorithmic framework integrating Bayesian learning and multi-level exploration, and pioneering LLMs integration for contextual reasoning in railway operations.

Experimental validation on real-world timetables (50–120 trains) demonstrates that TrafSched achieves 74.65–100% conflict resolution success rates, substantially outperforming traditional heuristics (36.48–88.90%) and reinforcement learning methods (30.43–64.29%). The LLM-enhanced variant achieves 81.52–100% success rates, with 5–11 percentage point improvements on complex scenarios.

Beyond railway scheduling, this work demonstrates that hybrid architectures combining algorithmic rigor with AI-enhanced reasoning can achieve superior performance while maintaining interpretability essential for safety-critical applications. TrafSched provides a practical framework for intelligent railway dispatching, demonstrating how modern AI capabilities can be effectively integrated with domain expertise to address complex optimization challenges in transportation infrastructure.

Author Contributions: Conceptualization, W.F. and H.Z.; methodology, W.F. and Y.Z.; software, W.F., J.S. and X.C.; validation, W.F., S.L., M.Z. and Y.G.; formal analysis, W.F. and X.Z.; investigation, W.F., Y.Z., S.X. and J.S.; resources, L.X. and H.Z.; data curation, W.F., M.Z. and X.Z.; writing—original draft preparation, W.F., Y.Z., S.X. and X.C.; writing—review and editing, H.Z., L.X., S.L. and Y.G.; visualization, W.F. and S.X.; supervision, H.Z.; project administration, H.Z.; funding acquisition, L.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the China State Railway Group Co.,Ltd. (P2024X002), the Innovation Joint Fund of the Chongqing Natural Science Foundation (Changan Automobile), grant number CSTB2023NSCQ-LZX0136. And the CASIA team was supported by the C^2DL (No. E5SPFZ0109).

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data supporting the conclusions of this study contain proprietary information of China Academy of Railway Sciences Corporation Limited, and are not publicly available. Requests for limited access may be submitted to the corresponding author and will be considered subject to approval by the data owner.

Conflicts of Interest: Author Li Xu was employed by the company Chongqing Changan Automobile Co., Ltd. Authors Yiwei Guo and Xin Zhang were employed by the company China Academy of Railway Sciences Corporation Limited. The remaining authors declare that the research was conducted in the absence of any commercial or financial relationships that could be construed as a potential conflict of interest.

References

1. Pyrgidis, C.N. *Railway Transportation Systems: Design, Construction and Operation*; CRC Press: Boca Raton, FL, USA, 2021.
2. Chen, S.; Piao, L.; Zang, X.; Luo, Q.; Li, J.; Yang, J.; Rong, J. Analyzing differences of highway lane-changing behavior using vehicle trajectory data. *Phys. A Stat. Mech. Its Appl.* **2023**, *624*, 128980. [CrossRef]
3. Chen, X.; Wu, S.; Shi, C.; Huang, Y.; Yang, Y.; Ke, R.; Zhao, J. Sensing Data Supported Traffic Flow Prediction via Denoising Schemes and ANN: A Comparison. *IEEE Sens. J.* **2020**, *20*, 14317–14328. [CrossRef]
4. Cacchiani, V.; Toth, P. Nominal and robust train timetabling problems. *Eur. J. Oper. Res.* **2012**, *219*, 727–737. [CrossRef]
5. Cacchiani, V.; Furini, F.; Kidd, M.P. Approaches to a real-world train timetabling problem in a railway node. *Omega* **2016**, *58*, 97–110. [CrossRef]
6. Broman, E.; Eliasson, J.; Aronsson, M. Efficient capacity allocation on deregulated railway markets. *J. Rail Transp. Plan. Manag.* **2022**, *21*, 100294. [CrossRef]
7. Lubin, M.; Dowson, O.; Garcia, J.D.; Huchette, J.; Legat, B.; Vielma, J.P. JuMP 1.0: Recent improvements to a modeling language for mathematical optimization. *Math. Program. Comput.* **2023**, *15*, 581–589. [CrossRef]
8. Zhou, X.; Zhong, M. Single-track train timetabling with guaranteed optimality: Branch-and-bound algorithms with enhanced lower bounds. *Transp. Res. Part B Methodol.* **2007**, *41*, 320–341. [CrossRef]
9. D'Ariano, A.; Pranzo, M.; Hansen, I.A. Conflict Resolution and Train Speed Coordination for Solving Real-Time Timetable Perturbations. *IEEE Trans. Intell. Transp. Syst.* **2007**, *8*, 208–222. [CrossRef]
10. Danavulapadu, V.P.; Singamsetty, P. Trains scheduling problem with multiple lines. *Sci. Rep.* **2024**, *14*, 31129. [CrossRef]
11. Bettinelli, A.; Santini, A.; Vigo, D. A real-time conflict solution algorithm for the train rescheduling problem. *Transp. Res. Part B Methodol.* **2017**, *106*, 237–265. [CrossRef]
12. Şahin, İ. Railway traffic control and train scheduling based oninter-train conflict management. *Transp. Res. Part B Methodol.* **1999**, *33*, 511–534. [CrossRef]

13. Van Thielen, S.; Corman, F.; Vansteenwegen, P. Towards a conflict prevention strategy applicable for real-time railway traffic management. *J. Rail Transp. Plan. Manag.* **2019**, *11*, 100139. [CrossRef]
14. Li, Y. Deep reinforcement learning: An overview. *arXiv* **2017**, arXiv:1701.07274.
15. Liu, E.; Lin, Z.; Wang, J.Y.; Chen, H. A Mobile Data-Driven Hierarchical Deep Reinforcement Learning Approach for Real-time Demand-Responsive Railway Rescheduling and Station Overcrowding Mitigation. *arXiv* **2023**, arXiv:2308.11849.
16. Tang, T.; Chai, S.; Wu, W.; Yin, J.; D'Ariano, A. A multi-task deep reinforcement learning approach to real-time railway train rescheduling. *Transp. Res. Part E Logist. Transp. Rev.* **2025**, *194*, 103900. [CrossRef]
17. Yue, P.; Jin, Y.; Dai, X.; Feng, Z.; Cui, D. Reinforcement Learning for Scalable Train Timetable Rescheduling with Graph Representation. *IEEE Trans. Intell. Transp. Syst.* **2024**, *25*, 6472–6485. [CrossRef]
18. Lusby, R.M.; Larsen, J.; Bull, S. A survey on robustness in railway planning. *Eur. J. Oper. Res.* **2018**, *266*, 1–15. [CrossRef]
19. Schneider, S.; Ramesh, A.; Roets, A.; Stirbu, C.; Safaei, F.; Ghriss, F.; Wülfing, J.; Güra, M.; Sibon, N.; Gentry, R.; et al. Intelligent Railway Capacity and Traffic Management Using Multi-Agent Deep Reinforcement Learning. In Proceedings of the 2024 IEEE 27th International Conference on Intelligent Transportation Systems (ITSC), Edmonton, AB, Canada, 24–27 September 2024; pp. 1743–1748.
20. Zhao, W.X.; Zhou, K.; Li, J.; Tang, T.; Wang, X.; Hou, Y.; Min, Y.; Zhang, B.; Zhang, J.; Dong, Z.; et al. A survey of large language models. *arXiv* **2023**, arXiv:2303.18223.
21. Cacchiani, V.; Huisman, D.; Kidd, M.; Kroon, L.; Toth, P.; Veelenturf, L.; Wagenaar, J. An overview of recovery models and algorithms for real-time railway rescheduling. *Transp. Res. Part B Methodol.* **2014**, *63*, 15–37. [CrossRef]
22. Bragin, M.A. Survey on Lagrangian relaxation for MILP: Importance, challenges, historical review, recent advancements, and opportunities. *Ann. Oper. Res.* **2024**, *333*, 29–45. [CrossRef]
23. Rodríguez, J. A constraint programming model for real-time train scheduling. *INFORMS J. Comput.* **2007**, *19*, 50–63. [CrossRef]
24. D'Ariano, A.; Pranzo, M.; Hansen, I.A. Conflict resolution and train speed coordination for real-time railway traffic management. *IEEE Trans. Intell. Transp. Syst.* **2008**, *9*, 228–244.
25. Zhang, Y.; Zhang, Z.; Zeng, Y.; Wu, T. Constraint programming for multi-line parallel partial disassembly line balancing problem with optional common stations. *Appl. Math. Model.* **2023**, *122*, 435–455. [CrossRef]
26. Lövei, I.; Kővári, B.; Bécsi, T.; Aradi, S. Environment Representations of Railway Infrastructure for Reinforcement Learning-Based Traffic Control. *Appl. Sci.* **2022**, *12*, 4465. [CrossRef]
27. Roderick, M.; MacGlashan, J.; Tellex, S. Implementing the Deep Q-Network. *arXiv* **2017**, arXiv:1711.07478. [CrossRef]
28. Jin, K.H.; Wi, J.A.; Lee, E.J.; Kang, S.J.; Kim, S.K.; Kim, Y.B. TrafficBERT: Pre-trained model with large-scale data for long-range traffic flow forecasting. *Expert Syst. Appl.* **2021**, *186*, 115738. [CrossRef]
29. Guo, X.; Zhang, Q.; Jiang, J.; Peng, M.; Zhu, M.; Yang, H.F. Towards explainable traffic flow prediction with large language models. *Commun. Transp. Res.* **2024**, *4*, 100150. [CrossRef]
30. Li, Z.; Deng, X.; Feng, C.; Li, H.; Wang, S.; Zhang, H.; Jia, T.; Chen, C.; Wu, L.L.; Wang, J. LLM4Rail: An LLM-Augmented Railway Service Consulting Platform. *arXiv* **2025**, arXiv:2507.23377.
31. Chen, A.; Tian, Y.; Zhang, J.; Li, C.; Zhang, H. LLM-based intelligent Q&A system for railway locomotive maintenance standardization. *Sci. Rep.* **2025**, *15*, 12953.
32. Huang, W.; Deng, X. Real-time tracking railway intruders using multiple-agent cooperated large language models with edge stream processing engine. *J. Netw. Comput. Appl.* **2025**, *242*, 104231. [CrossRef]
33. Tan, M.; Merrill, M.A.; Gottesman, Z.; Althoff, T.; Evans, D.; Hartvigsen, T. Inferring Event Descriptions from Time Series with Language Models. *arXiv* **2025**, arXiv:2503.14190. [CrossRef]
34. Back, W.E.; Boles, W.W.; Fry, G.T. Defining triangular probability distributions from historical cost data. *J. Constr. Eng. Manag.* **2000**, *126*, 29–37. [CrossRef]
35. Mansourian, A.M.; Ahmadi, R.; Ghafouri, M.; Babaei, A.M.; Golezani, E.B.; Ghamchi, Z.Y.; Ramezanian, V.; Taherian, A.; Dinashi, K.; Miri, A.; et al. A Comprehensive Survey on Knowledge Distillation. *arXiv* **2025**, arXiv:2503.12067. [CrossRef]
36. Carta, S.; Giuliani, A.; Piano, L.; Podda, A.S.; Pompianu, L.; Tiddia, S.G. Iterative zero-shot llm prompting for knowledge graph construction. *arXiv* **2023**, arXiv:2307.01128. [CrossRef]
37. Wei, J.; Wang, X.; Schuurmans, D.; Bosma, M.; Ichter, B.; Xia, F.; Chi, E.; Le, Q.; Zhou, D. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. *arXiv* **2023**, arXiv.2201.11903. [CrossRef]
38. Yang, A.; Li, A.; Yang, B.; Zhang, B.; Hui, B.; Zheng, B.; Yu, B.; Gao, C.; Huang, C.; Lv, C.; et al. Qwen3 Technical Report. *arXiv* **2025**, arXiv.2505.09388. [CrossRef]