RailSched: A Hybrid Decision Framework for Railway Scheduling Optimization

Abstract

Railway transportation systems worldwide face increasing pressure to maximize capacity while maintaining safety and operational efficiency. Complex scheduling conflicts frequently arise due to the inherent strict constraints in railway networks, including track sharing, station capacity limitations, and various train operational requirements. Effective conflict resolution methods are crucial for maintaining reliable railway services and optimizing infrastructure utilization. To address these challenges, we propose the **RailSched**, a novel intelligent decision framework for railway scheduling optimization. Our research by providing an intelligent and multi-strategy approach that reduces dispatcher workload, minimizes conflicts, and improves overall system reliability. The key contributions of this paper are fourfold: 1) we propose a railway schedule environment model for simulation; 2) develop a diverse strategy library targeting conflicts; 3) present an intelligent algorithmic framework for policy decision making and 4) explore the integration of large language models(LLMs) for railway schedule, an optional AI-enhanced decision module. Our evaluation using real railway data validates **RailSched**'s effectiveness, achieving a 85.05% improvement over expert-designed solutions in the most complex scenario, highlighting its practical value for modern railway dispatching.

CCS Concepts

• **Applied computing** → **Transportation**; • **Computing methodologies** → *Planning and scheduling*; Natural language processing.

Keywords

Railway Scheduling, Conflict Resolution, Decision, Large Language Models

ACM Reference Format:

. 2025. RailSched: A Hybrid Decision Framework for Railway Scheduling Optimization. In *Proceedings of (The Seventh International Conference on Distributed Artificial Intelligence).* ACM, New York, NY, USA, 10 pages. https://doi.org/10.1145/nnnnnnn.nnnnnnn

1 Introduction

Railway transportation systems [25] 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.

The operational effectiveness of railway systems hinges fundamentally on their timetable schedules.The Train Timetabling Problem (TTP) [8, 9] 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 two primary challenges: conflict resolution between trains competing for the same track section, and optimizing operational efficiency to enhance passenger experience and resource utilization. These challenges are amplified by several factors such as multiple train types operating on shared infrastructure, varying track configurations, line-specific constraints, and strict adherence to numerous operational and safety regulations.

Currently, train timetable compilation and conflict resolution remain predominantly manual processes completed by dispatchers, leading to several limitations: 1) Low operational efficiency in timetable creation and modification; 2) Inconsistent quality that varies significantly with individual operator expertise; 3) Limited ability to rapidly respond to disruptions; 4) Difficulty in optimizing across multiple objectives.

Traditional approaches to address these challenges have primarily relied on mathematical programming models [21], including branch-and-bound methods enhanced with Lagrangian relaxation techniques [48]. Other researchers have explored integer programming methods for real-time railway scheduling problems [12, 14], and heuristic approaches including parallel algorithms for iterative greedy scheduling [4], dispatcher-focused conflict handling methods [28], and predictive conflict management systems [35]. More recently, reinforcement learning (RL) [18] has demonstrated outstanding performance in solving complex decision-making problems, leading researchers to apply these techniques to railway timetable scheduling optimization [19, 34, 42]. However, despite rapid advances in artificial intelligence, limited research has explored the potential of large language models (LLMs) [47] for railway scheduling applications.

Despite these advances, traditional approaches continue to struggle with the inherent complexity of modern railway scheduling networks [3], particularly when handling interrelated conflicts across multiple train services simultaneously. As railway networks expand in scale and train frequencies increase to meet growing demand, effective conflict resolution methods become increasingly crucial for maintaining reliable railway services, reducing dispatcher workload, minimizing delays, and optimizing infrastructure utilization.

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

The Seventh International Conference on Distributed Artificial Intelligence, London

ACM ISBN 978-x-xxxx-xxxx-x/YYYY/MM
https://doi.org/10.1145/nnnnnnn.nnnnnnn

- Develop a comprehensive railway schedule environment model specifically designed for conflict simulation and resolution.
- Create a diverse strategy library containing specialized resolution for four types of conflicts.
- Propose the **RailSched** algorithmic decision framework that significantly outperforms traditional methods through its integration of Bayesian strategy adaptation, multi dimensional conflict prioritization, and backtracking mechanisms
- Pioneer the integration of LLMs into railway scheduling, exploring a new frontier in AI-assisted railway dispatching.

2 Related Work

2.1 Traditional Methods for Railway Scheduling

Traditional methods for resolving railway scheduling conflicts have primarily relied on mathematical optimization [31] and rule-based heuristics [26, 39]. Mixed-Integer Linear Programming (MILP) models railway scheduling as a constrained optimization problem [7]. While MILP guarantees optimal solutions under well-defined constraints, it suffers from exponential computational complexity for large-scale networks [22]. To address the computational challenges of exact methods, researchers have developed sophisticated enhancement strategies,they pioneered the integration of branch-and-bound methods [48] with Lagrangian relaxation techniques [5], achieving significant improvements in solution quality while maintaining computational traceability.

Other alternative methods, such as constraint programming [27] offer greater modeling flexibility for complex operational rules, while dynamic priority rules [13] improved scalability through simplified decision-making processes. Recent advances have integrated train speed adjustments [14] and developed sophisticated integer programming models for multi-line scheduling scenarios [46], demonstrating the ongoing evolution of mathematical approaches.

2.2 AI-Based Methods for Railway Scheduling

The integration of Artificial Intelligence(AI) methodologies has introduced data-driven solutions for railway conflict resolution. RL has emerged as a particularly promising approach for railway scheduling due to its ability to learn optimal policies through interaction with complex environments. Early applications focused on training agents to learn optimal rescheduling policies through simulated railway environments [17], leveraging RL's sequential decision-making paradigm to adapt to changing operational conditions. And deep reinforcement learning (DRL) methods like deep Q-Networks [36] have demonstrated significant promise in handling the high-dimensional state spaces characteristic of large railway networks.

Recent studies have demonstrated how RL can effectively optimize train timetables through sophisticated simulation environments [16, 19] that model real-world operational complexities. Some researchers proposed innovative multi-task DRL frameworks for real-time scheduling that can simultaneously optimize multiple objectives[34], or combined graph neural networks [42]with RL to create extensible solutions that can adapt to varying network topologies and scales.

2.3 Large Language Models in Transportation

The application of LLMs in transportation systems represents an emerging research direction. In logistics optimization, BERT-based models demonstrate how pre-trained language models can process textual operational information to enhance decision-making [11]. And some people investigated language models to improve traffic prediction by incorporating textual descriptions alongside numerical data[20]. In the context of dispatching, research has examined human-AI collaboration frameworks in which language models facilitate communication between dispatchers and scheduling algorithms [30]. Natural language interfaces have also been proposed to support dispatcher interactions in railway systems[15], while explanatory AI frameworks have been designed to generate human-readable justifications for scheduling decisions [38].

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

Our RailSched framework may addresses this gap by exploring how LLMs can enhance conflict resolution strategies, potentially bridging the divide between data-driven AI optimization and human-understandable decision-making processes.

3 Problem Formulation

The first contribution of our work is development the railway schedule environment model, which provides a standardized framework for simulating and resolving railway conflicts, addressing a significant gap where researchers often rely on simplified models.

In optimizing rail schedules, the safety of the system is highly dependent on time constraints between trains and the strict detection of potential conflicts. We set the mathematical model of the railway timetable as follows.

3.1 Preliminary

We consider a railway network represented as a directed graph where stations correspond to nodes and track segments correspond to edges. 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 set of track segments connecting consecutive stations. For each train $i \in \mathcal{T}$ and station $s \in \mathcal{S}$, we define the event times and operational parameters that govern the timetabling problem.

3.2 The Constraints and Optimization Objective

To ensure safe operation of the railway, various constraints need to be considered. We categorize these constraints into four fundamental types: dwell time constraints, running time constraints, interval time constraints, and overtaking constraints.

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:

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

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

In equation (3), the upper bound of the running time $p_{ik}^{\max}$ for train i at segment k is obtained by adding the lower bound $p_{ik}^{\min}$ and the free running time p_{ik}^{free}. Equation (4) gives the running time range in each segment, which limits the running speed of different trains and ensures the safety of 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

Note that arrival and departure events occur in pairs always. 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, 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.

3.2.5 Optimization Objective. The optimization objective of the railway timetabling problem is to minimize the number of operational conflicts in the schedule, subject to the four constraints described above. Let C denote the total number of *conflicts* in a timetable.The problem is formulated as:

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

3.3 Timetable Matrix and Conflict Detection

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

Definition 3.1 (Railway Timetable Matrix). For a railway system with S stations, the timetable matrix $\mathbf{M} \in \mathbb{R}^{(2S-2)\times N}$ is defined as:

- **Row** $(2S-2)$: Each intermediate station (total $S-2$) corresponds to 2 rows representing arrival and departure events, while each terminal station contributes a single row
- **Column** (N): Total number of trains in the system
- **Element** m_{ij}: Event time (arrival or departure) of the train j at the station i

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

$$\mathbf{M} = \begin{pmatrix} t_{A,\text{dep}}^1 & t_{A,\text{dep}}^2 & \cdots & t_{A,\text{dep}}^N \\ t_{B,\text{arr}}^1 & t_{B,\text{arr}}^2 & \cdots & t_{B,\text{arr}}^N \\ t_{B,\text{dep}}^1 & t_{B,\text{dep}}^2 & \cdots & t_{B,\text{dep}}^N \\ t_{C,\text{arr}}^1 & t_{C,\text{arr}}^2 & \cdots & t_{C,\text{arr}}^N \end{pmatrix}$$

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

This timetable matrix representation provides a compact and computationally efficient 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 $\mathbf{P}_2, \mathbf{P}_3 \in \mathbb{R}^{(2S-3)\times(2S-2)}$:

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

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

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

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

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

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 an operator.

Definition 3.2 (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 only the relevant conflict information, we introduce a mask matrix $\mathbf{P}_4 \in \mathbb{R}^{(2S-3)\times(2S-3)}$:

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

Finally, we can obtain the number of overtaking conflicts of the local timetable matrix through the following process:

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

In the resulting matrix $\mathbf{M}_6$, negative entries indicate overtaking conflicts. And the total number of overtaking conflicts is given by the count of negative entries in $\mathbf{M}_6$. A similar analysis should be performed for insufficient interval conflicts.

4 Methodology

This section presents the **RailSched** framework, an intelligent algorithmic decision system designed to resolve railway scheduling conflicts through adaptive multi-strategy optimization. And optionally seamlessly integrate the capabilities of LLMs.

4.1 Framework Overview

The **RailSched** framework as illustrated in Figure 1, we first detail conflict resolution strategies tailored to issues in real-world rail scenarios. Then we detailed four core components in this railway schedule decision system that collaboratively detect and resolve conflicts efficiently. Finally, we introduced an optional LLMs-enhanced decision module, that combines the robustness of our algorithms with the contextual reasoning of LLMs to help select the policy.

4.2 Strategies for Conflict Resolution

In the railway schedule system, violations of timetable constraints can lead to four distinct categories of conflicts. So we have developed dedicated resolution strategies for each category, which mainly involve adjusting the timetable of a specific train. All strategies are stored in the strategy library.

4.2.1 Strategies for Dwell Time Out-of-Range Conflicts. A dwell time conflict occurs when the actual stop time $st_{i,s}$ lies outside the permissible range $[r^{\min}_{i,s}, r^{\max}_{i,s}]$. As this issue affects only a single train at a specific station, resolution is straightforward. Two operators are provided: one to increase and one to decrease the dwell time.

4.2.2 Strategies for Running Time Out-of-Range Conflicts. A running time conflict arises when a train's travel time between two stations falls outside the allowable range, indicating it is running too fast or too slow. Similar to dwell time violations, each case involves only a single train. Two strategies are applied: increasing or decreasing the running time to bring it to the nearest boundary of the permissible range. Figure 1a illustrates an example where running time is increased to meet the lower bound.

4.2.3 Strategies for Insufficient Interval Conflicts. This conflict occurs when the time gap between events of adjacent trains at the same station is less than the required safety interval. Corresponding to seven interval constraints, there are seven conflict types. We designed eight strategies to address them. As example in Figure 1b, one direct approach cancels Train 1's stop. Strategies consider only the event type of the adjusted train, allowing them to handle multiple conflict configurations.

4.2.4 Strategies for Overtaking Conflicts. An overtaking conflict occurs when two consecutive trains enter and leave a segment in reverse order, which is prohibited in railway operations. Given the complexity of these scenarios, we have developed 13 strategies to address various configurations while minimizing the disturbance of the railway schedule. Each computes the minimal temporal adjustment required.

Figure 1c shows a typical case where Trains 1 and 2 have an overtaking conflict in Segment B. We resolve this by extending

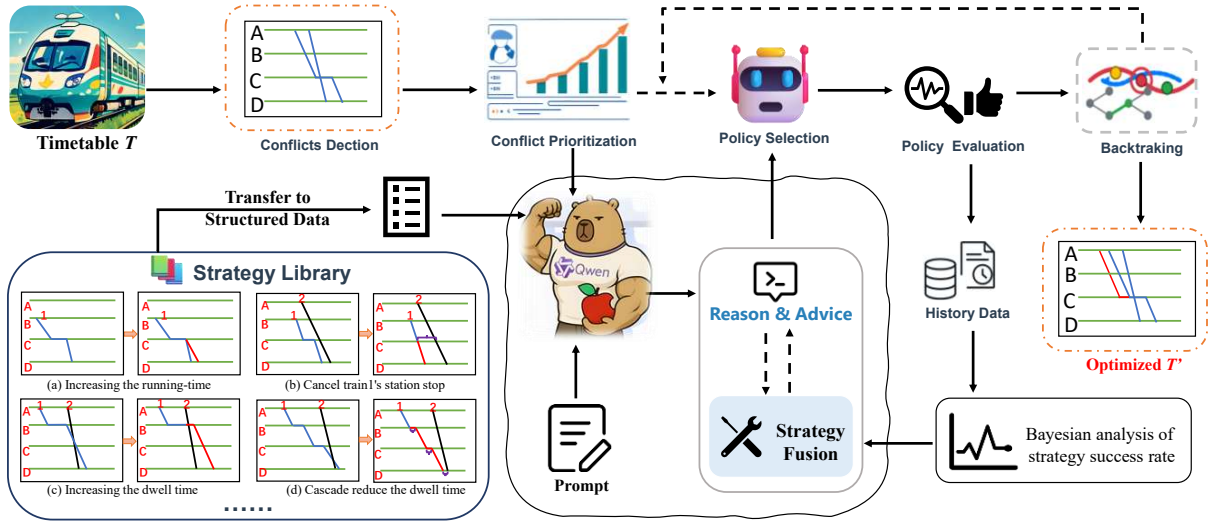

Figure 1: The RailSched Framework Architecture

Train 1's dwell time at Station B. The required extension T_{ext} is:

$$T_{\text{ext}} = T^{\text{pass}}_{2,B} - T^{\text{dep}}_{1,B} + \tau^{B}_{\text{pass}_2,\text{dep}_1} \tag{20}$$

where $T^{\text{pass}}_{2,B}$ is Train 2's passage time at Station B, $T^{\text{dep}}_{1,B}$ is Train 1's original departure time at Station B, and $\tau^{B}_{\text{pass}_2,\text{dep}_1}$ 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 target specific violations, they may cause premature convergence to local optima in complex, interrelated conflict scenarios. To overcome this, we propose five advanced strategies with broader adjustment capabilities:

Cascade Dwell Time Adjustment Strategies (A/B). These strategies adjust dwell times across multiple stations. Strategy *A* reduces dwell times from the conflict station backward to earlier stations until the conflict is resolved, as illustrated in Figure 1d. Strategy *B* extends dwell times forward, respecting upper limits. Both create cascading timetable changes.

Systematic Timetable Adjustment Strategies (C/D). These involve coordinated time shifts for multiple trains in opposite directions. Strategy *C* shifts selected trains earlier, while Strategy *D* shifts them later, maintaining minimum separations and redistributing temporal buffers to resolve complex conflicts.

Opportunity Window Insertion Strategy (E). 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

The core innovation of **RailSched** lies in its adaptive decision method, it integrates four basic components to efficiently detect and resolve train conflicts. **RailSched** combining adaptive decision-making mechanisms [1] with the ability of evolutionary search [43], enabling continuous learning from past resolutions while flexibly exploring diverse solutions.

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. Weights $\alpha, \beta, \gamma, \delta, \epsilon \in (0, 1)$ are dynamically adjusted, ensuring focus on the most critical conflicts as conditions evolve.

4.3.2 Bayesian Success Rate of Policy. After conflict prioritization, RailSched adopts a dual-pathway strategy selection approach: (1) a rule-based policy selection relying on the Base-RailSched algorithm without LLM integration, and (2) a hybrid decision method that combines LLM capabilities with rule-based mechanisms to enable more comprehensive strategy utilization. The second approach will be elaborated in detail in the subsequent sections.

Regardless of the policy selection pathway employed, RailSched maintains a unified Bayesian success rate updating mechanism to continuously optimize strategy performance based on historical outcomes. This probabilistic framework ensures that the system learns from past resolution attempts and adapts its strategy preferences accordingly. The Bayesian updating mechanism is formalized as:

$$SR_{i,t+1} = \frac{SR_{i,t} \cdot (n_i - 1) + \mathbb{I}(\text{success})}{n_i} \tag{22}$$

The equation updates the estimated success rate $SR_{i,t}$ of strategy i at time step t based on cumulative historical performance. And indicator function $\mathbb{I}(\text{success})$ equals 1 when strategy i successfully

resolves the target conflict and 0 otherwise, while n_i represents the total number of attempts using strategy i. This Bayesian approach provides a robust foundation for continuous learning, enabling the system to refine its strategy selection preferences through empirical evidence rather than relying solely on predefined method.

4.3.3 Hierarchical Strategy Evaluation. To avoid railway schedules Optimization in local that degrade global performance, RailSched evaluates policy by both local and network-wide effects:

$$E(s) = \begin{cases} \alpha_2\big(C_{\text{total}} - C_{\text{down}}(s)\big), & \text{if station} = 0 \\ \alpha_1\big(C_{\text{sec}}(\text{station}-1) - C_{\text{up}}(s)\big) & \\ \quad + \alpha_2\big(C_{\text{total}} - C_{\text{sec}}(\text{station}-1) & \text{otherwise} \\ \qquad - C_{\text{down}}(s)\big), & \end{cases} \tag{23}$$

where C_{total}, $C_{\text{down}}(s)$, $C_{\text{up}}(s)$ and $C_{\text{sec}}(\text{station}-1)$ measure conflicts in different network regions. And the α_1, α_2 weighting coefficients for upstream and downstream effects.

For conflicts at the origin station, RailSched evaluates only downstream effects. For conflicts at subsequent stations, it balances upstream ($C_{\text{up}}(s)$) and downstream ($C_{\text{down}}(s)$) impacts using weights α_1 and α_2. The hierarchical evaluation function enables the system to balance local optimality with global consistency.

4.3.4 Multi-level backtracking and exploration. Railway conflict resolution frequently encounters local optimum scenarios where standard optimization approaches fail to identify globally optimal solutions. To address this challenge, we implements backtracking and exploration mechanisms that enable strategic jumps in the solution space, allowing escape from suboptimal regions.

We can model the resolution as exploring a tree structure $T = (V, E)$ where nodes V represent different schedule states and edges E represent state transitions. Then backtracking jumps between nodes with probability:

$$P(V_{\text{target}}) = \begin{cases} 0.3, & \text{if } V_{\text{target}} = V_{\text{global\_best}} \\ 0.7 \cdot P_{\text{triangular}}(i), & \text{if } V_{\text{target}} \in \text{Checkpoints} \end{cases} \tag{24}$$

This 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 [2] favoring earlier checkpoints). This prevents the system from becoming trapped in repetitive cycles while ensuring promising solution pathways receive appropriate attention. But when progress stalls for an extended period (e.g., over 50 iterations), the system triggers an adaptive disturbance mechanism to escape persistent local optima.

Algorithm 1 presents the pseudo-code of the railway scheduling decision method, as the core of the **RailSched** framework. The exploration mechanisms work in tandem with the strategy selection and evaluation components, creating a comprehensive search approach that balances explore and utilize. This is crucial to handling the complexity of real-world railway scheduling problems.

Algorithm 1 Railway Scheduling Decision Algorithm (RSDA)

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
    Try applying strategies until resolution or all attempted
    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 LLMs-Enhanced Decision Module

With the rapid advancement of LLMs, their reasoning and problem-solving capabilities have been significantly strengthened. To address this opportunity, **RailSched** incorporates an optional LLM-enhanced decision module, designed to augment the base algorithm framework by providing advanced contextual reasoning [41] and complementary decision support.

4.4.1 Strategy Knowledge Representation. To enable the LLMs as an intelligent scheduling assistant, we transform policies in strategy library into a structured natural language presentation [10, 23, 44]. This transformation bridges the gap between algorithmic implementations and the LLMs' natural language understanding [24] capabilities, allowing the model to reason about railway operations using domain-specific knowledge.

Each conflict resolution strategy captures five essential aspects: the strategy name, identifier, a comprehensive natural language description of how the strategy operates, the required operational parameters, and the expected impact on railway schedules. This knowledge representation approach leverages the LLMs' strength in processing and reasoning with textual information, enabling it to understand not just what each strategy does, but why and when it should be applied.

4.4.2 Analysis Conflict and Make Desicion. Given a detected conflict, we generate a context-aware representation [29, 32] including conflict types, involved trains, affected stations, and severity metrics

that based on operational impact assessment. This enables LLMs to use nuanced reasoning. And for each involved train, we also analyze the classification of train types, service priority levels, current operational status, and schedule parameters. This helps LLMs to understand the possible implications of each potential resolution strategy.

Finally, through a carefully designed prompt integrate all information to the LLMs , and received the input comprising the conflict description, involved train characteristics, available strategy knowledge base, and current railway schedule status. The LLMs processes this information through Chain-of-Thought reasoning [37], and output consists of a prioritized list of recommended strategies, each accompanied by detailed justification explaining the reasoning behind the recommendation.

4.4.3 Hybrid Decision Fusion. The hybrid decision fusion mechanism represents a critical component that intelligently combines LLMs insights with algorithmic optimization to enhance conflict resolution effectiveness. When the top K strategies recommended by the LLMs fail to resolve the conflict, then will attempt to use the hybrid decision mechanism to find a viable solution.

The first is the Dynamic Weight Calculation. To effectively balance LLMs recommendations with the base algorithmic, we design an adaptive weight calculation that adjusts the influence of LLMs insights based on four factors: (1) **Base weight**, establishes the fundamental trust level in LLM recommendations, initialized at 0.4. (2) **Confidence factor**, directly incorporates the LLM's self-assessed confidence scores, increasing LLMs influence when the model expresses high certainty. (3) **Performance factor**, tracks the LLM's historical success rate by monitoring the ratio of successful recommendations to total attempts. (4) **Complexity factor**, increases LLMs influence for challenging scenarios, particularly when conflicts have required multiple resolution attempts or when optimization stagnation is detected. expresses high certainty and reducing it during uncertain predictions.

These factors combine multiplicatively to produce the final fusion weight:

$$W_{\text{LLM}} = w_{\text{base}} \cdot f_{\text{confidence}} \cdot f_{\text{performance}} \cdot f_{\text{complexity}} \tag{25}$$

Finally, The strategy fusion process creates a unified ranking by combining LLMs recommendations with algorithmic evaluations. For each strategy, the system computes a hybrid score through weighted combination, where LLMs scores reflect the model's strategic reasoning and position-based preferences, while algorithmic scores are derived from Bayesian analysis of historical success rates.

This hybrid approach ensures that the system benefits from both the contextual reasoning capabilities of LLMs and the mathematical rigor of algorithmic optimization, creating a robust decision-making framework that adapts to varying conflict complexities and operational contexts.

5 Experiments

This section evaluates our proposed **RailSched** framework for railway scheduling optimization. We conduct extensive experiments on real-world railway datasets to demonstrate the effectiveness of RailSched and systematically analyze the contribution of LLMs integration to conflict resolution performance.

5.1 Experimental Setup

We evaluate our **RailSched** framework using human-expert-designed real-world railway schedule with varying complexity levels. The evaluation metrics of experimental datasets comprise timetables ranging from 50 to 120 trains, with corresponding event counts between 1,160 and 2,890 events, covering a 24-hour scheduling horizon. Despite being designed by human experts, these timetables contain numerous unresolved conflicts.

For each experimental scenario, we measure initial conflict counts, remaining conflicts after resolution attempts, and calculate the conflict resolution rate as the primary performance metric. We compare our approach against two baseline methods: (1) traditional heuristic methods representing current state-of-practice in railway scheduling systems [45], and (2) Proximal Policy Optimization (PPO), a state-of-the-art reinforcement learning algorithm for sequential decision-making problems.

To assess the impact of incorporating LLMs into railway scheduling, we evaluate two variants of our framework, the base RailSched algorithmic framework without LLM (**RSDA**) and the **RailSched** integrating with LLMs.

All conflict resolution method are set a maximum of 1,024 running steps to ensure fair comparison and prevent excessive computational overhead. For **RailSched** integrated with LLMs, we employ the Qwen3-32B Language Model [40], selected for its advanced reasoning and contextual capabilities. This integration enhances strategic decision-making within the conflict resolution process.

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 RailSched framework compared to baseline methods across all experimental scenarios.

5.2.1 Performance Comparison Across Methods. Traditional heuristic approaches exhibit relatively poor performance, with success rates ranging from 36.48% to 88.90%. Similarly, PPO method achieves only 30.43%–64.29%, highlighting its limited effectiveness in complex railway scheduling scenarios.

In contrast, RailSched substantially outperforms these baselines. The RADA achieves success rates between 74.65% and 100%, while the LLM-enhanced RailSched consistently maintains 81.52%–100% across all scenarios. On smaller-scale problems (50–70 trains), both variants exceed 88% success rates, with perfect resolution observed in the 50-train case.

5.2.2 Scale Impact and LLM Enhancement. As timetable complexity increases, baseline methods degrade substantially, while RailSched variants maintain robust performance. The performance gap widens with larger scales, where traditional methods drop below 50% while RailSched sustains above 80% effectiveness.

LLMs integration provides consistent improvements over **RSDA**, particularly in complex scenarios. Notable enhancements include 11.27 percentage points improvement in the 90-train scenario (85.92% vs 74.65%) and practical gains of 6 additional resolved conflicts in the 120-train case.

Table 1: Performance Comparison of Railway Scheduling Method

Railway Timetable Scale			Evaluation Method							
			Remaining Conflicts				Success Rate (%)			
Trains	Events	Conflicts	Heuristic	PPO	RSDA	RailSched	Heuristic	PPO	RSDA	RailSched
50	1160	42	5	15	0	0	88.90	64.29	100	100
60	1429	53	11	21	6	5	79.24	60.38	88.68	90.57
70	1683	53	28	28	1	2	47.17	47.17	98.11	96.23
80	1906	70	40	29	14	11	42.86	58.57	80.00	84.29
90	2113	71	41	30	18	10	42.25	57.75	74.65	85.92
100	2335	79	44	44	9	8	44.30	44.30	88.61	89.87
110	2597	92	49	64	22	17	46.74	30.43	76.09	81.52
120	2890	107	68	68	22	16	36.48	36.48	79.44	85.05

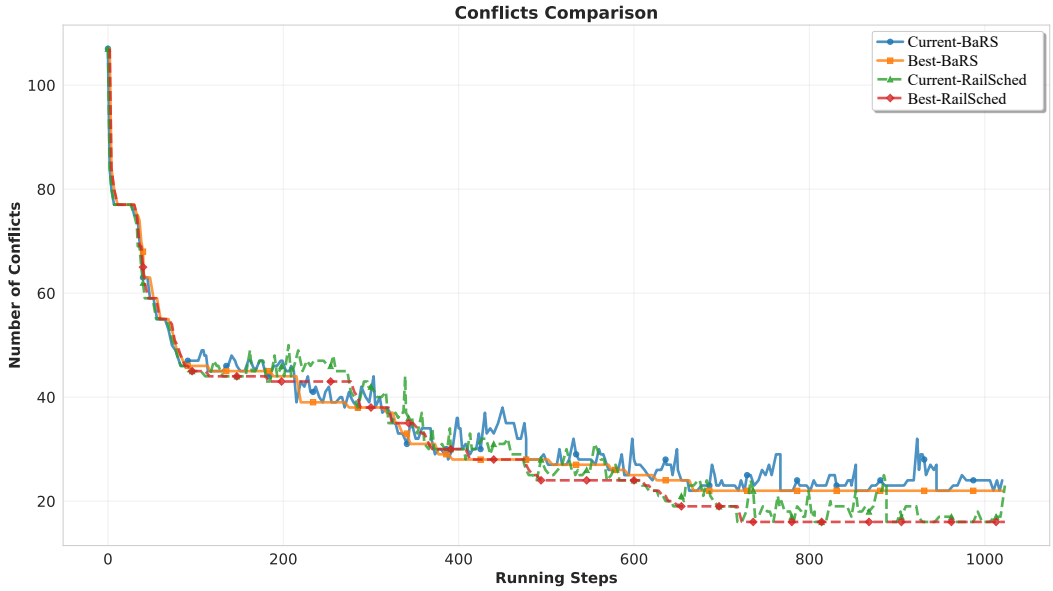

Figure 2: The Conflict Comparison In 120 Trains Scenario

Figure 2 illustrates conflict resolution dynamics for the 120-train scenario over 1,024 steps.Both methods rapidly reduce conflicts to around 45–50 within the first 100 steps.However, RailSched exhibits superior convergence behavior, maintaining more stable conflict reduction and achieving lower final conflict counts. After 500 steps, Best-RailSched reaches 16 conflicts versus 22 for Best-RSDA, demonstrating the LLM's enhanced strategic decision-making in complex optimization landscapes.

The results validate RailSched's effectiveness for modern railway dispatching, with LLM-enhanced version delivering the most significant performance gains in complex operational environments.

6 Conclusion

This paper presents RailSched, a novel algorithmic framework that addresses critical railway scheduling conflicts through intelligent automation decision. Our approach demonstrates substantial improvements over traditional methods across the datasets ranging from 50 to 120 trains. The integration of LLMs represents a breakthrough in railway scheduling optimization, providing consistent performance enhancements particularly in complex scenarios. In the most challenging 120-train case, LLM-enhanced RailSched achieves 85.05% success rate while reducing conflicts to 16 compared to 22 for the base version, demonstrating superior strategic decision-making capabilities. Our experimental validation on real-world datasets confirms RailSched's practical value for modern railway dispatching, significantly improved operational efficiency while maintaining the reliability standards required for safety-critical transportation infrastructure.

References

[1] Emmanuel Adeyemi Abaku, Tolulope Esther Edunjobi, and Agnes Clare Odimarha. 2024. Theoretical approaches to AI in supply chain optimization: Pathways to efficiency and resilience. *International Journal of Science and Technology Research Archive* 6, 1 (2024), 092–107.

[2] W Edward Back, Walter W Boles, and Gary T Fry. 2000. Defining triangular probability distributions from historical cost data. *Journal of Construction Engineering and Management* 126, 1 (2000), 29–37.

[3] Nikola Bešinović and Rob MP Goverde. 2018. Capacity assessment in railway networks. *Handbook of Optimization in the Railway Industry* (2018), 25–45.

[4] Andrea Bettinelli, Alberto Santini, and Daniele Vigo. 2017. A real-time conflict solution algorithm for the train rescheduling problem. *Transportation Research Part B: Methodological* 106 (2017), 237–265.

[5] Mikhail A Bragin. 2024. Survey on Lagrangian relaxation for MILP: importance, challenges, historical review, recent advancements, and opportunities. *Annals of Operations Research* 333, 1 (2024), 29–45.

[6] Emanuel Broman, Jonas Eliasson, and Martin Aronsson. 2022. Efficient capacity allocation on deregulated railway markets. *Journal of Rail Transport Planning Management* 21 (03 2022), 100294. doi:10.1016/j.jrtpm.2021.100294

[7] V. Cacchiani et al. 2014. An overview of recovery models and algorithms for real-time railway rescheduling. *Transportation Research Part B: Methodological* 63 (2014), 15–37.

[8] Valentina Cacchiani, Fabio Furini, and Martin Philip Kidd. 2016. Approaches to a real-world train timetabling problem in a railway node. *Omega* 58 (2016), 97–110.

[9] Valentina Cacchiani and Paolo Toth. 2012. Nominal and robust train timetabling problems. *European Journal of Operational Research* 219, 3 (2012), 727–737.

[10] Salvatore Carta, Alessandro Giuliani, Leonardo Piano, Alessandro Sebastian Podda, Livio Pompianu, and Sandro Gabriele Tiddia. 2023. Iterative zero-shot llm prompting for knowledge graph construction. *arXiv preprint arXiv:2307.01128* (2023).

[11] J. Chen et al. 2023. BERT-based traffic prediction. *IEEE Transactions on Intelligent Transportation Systems* 24, 3 (2023), 2876–2889.

[12] Venkata Prathap Danavulapadu and Purusotham Singamsetty. 2024. Trains scheduling problem with multiple lines. *Scientific Reports* 14, 1 (2024), 31129.

[13] A. D'Ariano et al. 2008. Conflict resolution and train speed coordination for real-time railway traffic management. *IEEE Transactions on Intelligent Transportation Systems* 9, 2 (2008), 228–244.

[14] Andrea D'Ariano, Marco Pranzo, and Ingo A. Hansen. 2007. Conflict Resolution and Train Speed Coordination for Solving Real-Time Timetable Perturbations. *IEEE Transactions on Intelligent Transportation Systems* 8, 2 (2007), 208–222. doi:10.1109/TITS.2006.888605

[15] K. Johnson et al. 2023. Explainable AI for railway operations. *IEEE Intelligent Systems* 38, 1 (2023), 45–53.

[16] Wenqing Li and Shaoquan Ni. 2022. Train timetabling with the general learning environment and multi-agent deep reinforcement learning. *Transportation Research Part B: Methodological* 157 (2022), 230–251.

[17] X. Li et al. 2020. Deep reinforcement learning for train rescheduling. *IEEE Transactions on Intelligent Transportation Systems* 21, 2 (2020), 678–691.

[18] Yuxi Li. 2017. Deep reinforcement learning: An overview. *arXiv preprint arXiv:1701.07274* (2017).

[19] Enze Liu, Zhiyuan Lin, Judith YT Wang, and Hong Chen. 2023. A Mobile Data-Driven Hierarchical Deep Reinforcement Learning Approach for Real-time Demand-Responsive Railway Rescheduling and Station Overcrowding Mitigation. *arXiv preprint arXiv:2308.11849* (2023).

[20] W. Liu et al. 2024. Human-AI collaboration in railway dispatching. *Transportation Research Part A: Policy and Practice* 179 (2024), 103789.

[21] Miles Lubin, Oscar Dowson, Joaquim Dias Garcia, Joey Huchette, Benoît Legat, and Juan Pablo Vielma. 2023. JuMP 1.0: Recent improvements to a modeling language for mathematical optimization. *Mathematical Programming Computation* 15, 3 (2023), 581–589.

[22] R. M. Lusby et al. 2018. A survey on railway timetabling. *European Journal of Operational Research* 271, 1 (2018), 1–23.

[23] Amir M. Mansourian, Rozhan Ahmadi, Masoud Ghafouri, Amir Mohammad Babaei, Elaheh Badali Golezani, Zeynab Yasamani Ghamchi, Vida Ramezanian, Alireza Taherian, Kimia Dinashi, Amirali Miri, and Shohreh Kasaei. 2025. A Comprehensive Survey on Knowledge Distillation. arXiv:2503.12067 [cs.CV] https://arxiv.org/abs/2503.12067

[24] Pablo Miralles-González, Javier Huertas-Tato, Alejandro Martín, and David Camacho. 2025. Pushing the boundary on Natural Language Inference. arXiv:2504.18376 [cs.CL] https://arxiv.org/abs/2504.18376

[25] Christos N Pyrgidis. 2021. *Railway transportation systems: design, construction and operation.* CRC press.

[26] Edwina L Rissland and David B Skalak. 1989. Combining Case-Based and Rule-Based Reasoning: A Heuristic Approach.. In *IJCAI*. 524–530.

[27] J. Rodríguez. 2007. A constraint programming model for real-time train scheduling. *INFORMS Journal on Computing* 19, 1 (2007), 50–63.

[28] İsmail Şahin. 1999. Railway traffic control and train scheduling based on inter-train conflict management. *Transportation Research Part B: Methodological* 33, 7 (1999), 511–534.

[29] Yue Shi, Alexandros Karatzoglou, Linas Baltrunas, Martha Larson, and Alan Hanjalic. 2014. CARS2: Learning Context-aware Representations for Context-aware Recommendations. In *Proceedings of the 23rd ACM International Conference on Conference on Information and Knowledge Management* (Shanghai, China) *(CIKM '14)*. Association for Computing Machinery, New York, NY, USA, 291–300. doi:10.1145/2661829.2662070

[30] J. Smith et al. 2023. Natural language interfaces for transportation systems. *ACM Transactions on Intelligent Systems and Technology* 14, 2 (2023), 1–24.

[31] Jan A Snyman, Daniel N Wilke, et al. 2005. *Practical mathematical optimization.* Vol. 97. Springer.

[32] Daniil Sorokin and Iryna Gurevych. 2017. Context-Aware Representations for Knowledge Base Relation Extraction. In *Proceedings of the 2017 Conference on Empirical Methods in Natural Language Processing*, Martha Palmer, Rebecca Hwa, and Sebastian Riedel (Eds.). Association for Computational Linguistics, Copenhagen, Denmark, 1784–1789. doi:10.18653/v1/D17-1188

[33] Mingtian Tan, Mike A. Merrill, Zack Gottesman, Tim Althoff, David Evans, and Tom Hartvigsen. 2025. Inferring Event Descriptions from Time Series with Language Models. arXiv:2503.14190 [cs.AI] https://arxiv.org/abs/2503.14190

[34] Tao Tang, Simin Chai, Wei Wu, Jiateng Yin, and Andrea D'Ariano. 2025. A multi-task deep reinforcement learning approach to real-time railway train rescheduling. *Transportation Research Part E: Logistics and Transportation Review* 194 (2025), 103900.

[35] Sofie Van Thielen, Francesco Corman, and Pieter Vansteenwegen. 2019. Towards a conflict prevention strategy applicable for real-time railway traffic management. *Journal of Rail Transport Planning & Management* 11 (2019), 100139.

[36] Y. Wang et al. 2022. Multi-agent reinforcement learning for railway traffic management. *Transportation Research Part C: Emerging Technologies* 134 (2022), 103456.

[37] Jason Wei, Xuezhi Wang, Dale Schuurmans, Maarten Bosma, Brian Ichter, Fei Xia, Ed Chi, Quoc Le, and Denny Zhou. 2023. Chain-of-Thought Prompting Elicits Reasoning in Large Language Models. arXiv:2201.11903 [cs.CL] https://arxiv.org/abs/2201.11903

[38] R. Williams et al. 2024. Rule extraction from regulatory documents using large language models. *Artificial Intelligence in Transportation* 5 (2024), 100125.

[39] Cong Xiu, Jinyi Pan, Andrea D'Ariano, Shuguang Zhan, and Qiyuan Peng. 2024. Passenger service-oriented timetable rescheduling for large-scale disruptions in a railway network: A heuristic-based alternating direction method of multipliers. *Omega* 125 (2024), 103040.

[40] An Yang, Anfeng Li, Baosong Yang, Beichen Zhang, Binyuan Hui, Bo Zheng, Bowen Yu, Chang Gao, Chengen Huang, Chenxu Lv, Chujie Zheng, Dayiheng Liu, Fan Zhou, Fei Huang, Feng Hu, Hao Ge, Haoran Wei, Huan Lin, Jialong Tang, Jian Yang, Jianhong Tu, Jianwei Zhang, Jianxin Yang, Jiaxi Yang, Jing Zhou, Jingren Zhou, Junyang Lin, Kai Dang, Keqin Bao, Kexin Yang, Le Yu, Lianghao Deng, Mei Li, Mingfeng Xue, Mingze Li, Pei Zhang, Peng Wang, Qin Zhu, Rui Men, Ruize Gao, Shixuan Liu, Shuang Luo, Tianhao Li, Tianyi Tang, Wenbiao Yin, Xingzhang Ren, Xinyu Wang, Xinyu Zhang, Xuancheng Ren, Yang Fan, Yang Su, Yichang Zhang, Yinger Zhang, Yu Wan, Yuqiong Liu, Zekun Wang, Zeyu Cui, Zhenru Zhang, Zhipeng Zhou, and Zihan Qiu. 2025. Qwen3 Technical Report. arXiv:2505.09388 [cs.CL] https://arxiv.org/abs/2505.09388

[41] Fei Yu, Hongbo Zhang, Prayag Tiwari, and Benyou Wang. 2023. Natural Language Reasoning, A Survey. arXiv:2303.14725 [cs.CL] https://arxiv.org/abs/2303.14725

[42] Peng Yue, Yaochu Jin, Xuewu Dai, Zhenhua Feng, and Dongliang Cui. 2024. Reinforcement Learning for Scalable Train Timetable Rescheduling With Graph Representation. *IEEE Transactions on Intelligent Transportation Systems* 25, 7 (2024), 6472–6485.

[43] Hoda Zamani and Mohammad H Nadimi-Shahraki. 2024. An evolutionary crow search algorithm equipped with interactive memory mechanism to optimize artificial neural network for disease diagnosis. *Biomedical Signal Processing and Control* 90 (2024), 105879.

[44] Bowen Zhang and Harold Soh. 2024. Extract, define, canonicalize: An llm-based framework for knowledge graph construction. *arXiv preprint arXiv:2404.03868* (2024).

[45] Qin Zhang, Richard Martin Lusby, Pan Shang, and Xiaoning Zhu. 2022. A heuristic approach to integrate train timetabling, platforming, and railway network maintenance scheduling decisions. *Transportation Research Part B: Methodological* 158 (2022), 210–238. doi:10.1016/j.trb.2022.02.002

[46] Yu Zhang, Zeqiang Zhang, Yanqing Zeng, and Tengfei Wu. 2023. Constraint programming for multi-line parallel partial disassembly line balancing problem with optional common stations. *Applied Mathematical Modelling* 122 (2023), 435–455.

[47] Wayne Xin Zhao, Kun Zhou, Junyi Li, Tianyi Tang, Xiaolei Wang, Yupeng Hou, Yingqian Min, Beichen Zhang, Junjie Zhang, Zican Dong, et al. 2023. A survey of large language models. *arXiv preprint arXiv:2303.18223* 1, 2 (2023).

[48] Xuesong Zhou and Ming Zhong. 2007. Single-track train timetabling with guaranteed optimality: Branch-and-bound algorithms with enhanced lower bounds. *Transportation Research Part B: Methodological* 41, 3 (2007), 320–341. doi:10.1016/j.trb.2006.05.003

Received 20 February 2007; revised 12 March 2009; accepted 5 June 2009