

Robust scheduling policy learning for ore-flow blending in dynamic and uncertain environments

Hao Tang^{a,d,e}, Changhe Li^{b,c}*, Guoyu Chen^{b,c}, Shoufei Han^{b,c}, Michalis Mavrovouniotis^{g,h},
Miqing Liⁱ, Liang Yuan^f

^a School of Automation, China University of Geosciences, Wuhan 430074, China

^b State Key Laboratory of Digital Intelligent Technology for Unmanned Coal Mining, Anhui University of Science & Technology, Huainan, 232001, China

^c School of Artificial Intelligence, Anhui University of Science & Technology, Hefei, 231131, China

^d Hubei Key Laboratory of Advanced Control and Intelligent Automation for Complex Systems, Wuhan 430074, China

^e Engineering Research Center of Intelligent Technology for Geo-Exploration, Ministry of Education, Wuhan 430074, China

^f State Key Laboratory for Safe Mining of Deep Coal Resources and Environment Protection, Anhui University of Science & Technology, Huainan, 232001, China

^g ERATOSTHENES Centre of Excellence, Limassol, Cyprus

^h Department of Civil Engineering and Geomatics, Cyprus University of Technology, Limassol, Cyprus

ⁱ School of Computer Science, University of Birmingham, Birmingham, B15 2TT, UK

ARTICLE INFO

Keywords:

Ore-flow blending
Truck scheduling
Dynamic optimization under uncertainty
Deep reinforcement learning

ABSTRACT

Ore-flow blending is essentially a real-time scheduling decision problem where equipment failures, sudden weather changes, and time-varying road network exist during production. This problem faces core challenges including high-dimensional spatiotemporal dynamics, real-time decision response, and long-horizon decision dependencies. To overcome these challenges, this paper designs an end-to-end ore-flow blending and truck scheduling framework based on a spatio-temporal graph policy network. By encoding the spatial topology of the road network and the dynamic temporal sequences of the ore flow into a unified state representation, the proposed model achieves scalable learning across road networks of varying scales and structures. Furthermore, a two-stage training strategy combining local short-horizon pre-training and global long-horizon fine-tuning is proposed, effectively resolving the computational and memory consumption challenges in large-scale, long-horizon scenarios. Experimental results on multiple real-world mine cases and dynamic scenarios demonstrate that this method significantly outperforms state-of-the-art techniques in ore flow stability control and exhibits strong robustness and generalization capabilities under different road network structures, equipment scales, and dynamic events.

1. Introduction

Open-pit mining production systems are inherently complex spatiotemporal systems characterized by high dynamism and uncertainty. In actual operations, these systems are not only constrained by non-stationary variables that change continuously over time, such as fluctuations in truck travel times caused by sudden weather changes, but are also replete with various discrete sudden disturbance events. These include sudden failures of trucks and electric shovels, the migration of loading and unloading points as mining progresses, and the resulting real-time evolution of the road network topology. To address these challenges, this paper proposes a fundamental paradigm shift from static resource allocation to dynamic ore-flow blending (as shown in Fig. 1). While traditional approaches treat blending as an offline planning task (Liu et al., 2021; Nikbin and Moradi Afrapoli, 2023;

Chen et al., 2022) (which often fails due to unmodeled operational variability), our approach aims to achieve temporal stability control of the ore grade flow sent to the processing plant through real-time intervention in the truck scheduling process. Fundamentally, this is a real-time scheduling decision problem under dynamic and uncertain environments. Solving such NP-hard problem faces the following core challenges:

- High-dimensional spatiotemporal dynamism and uncertainty: The state space of the mining system is extremely vast and complex. It involves not only non-stationary variables continuously changing over time, such as road segment travel times affected by weather, but also discrete sudden events like equipment failures and road

* Corresponding author at: School of Artificial Intelligence, Anhui University of Science & Technology, Hefei, 231131, China.
E-mail address: lichanghe@aust.edu.cn (C. Li).

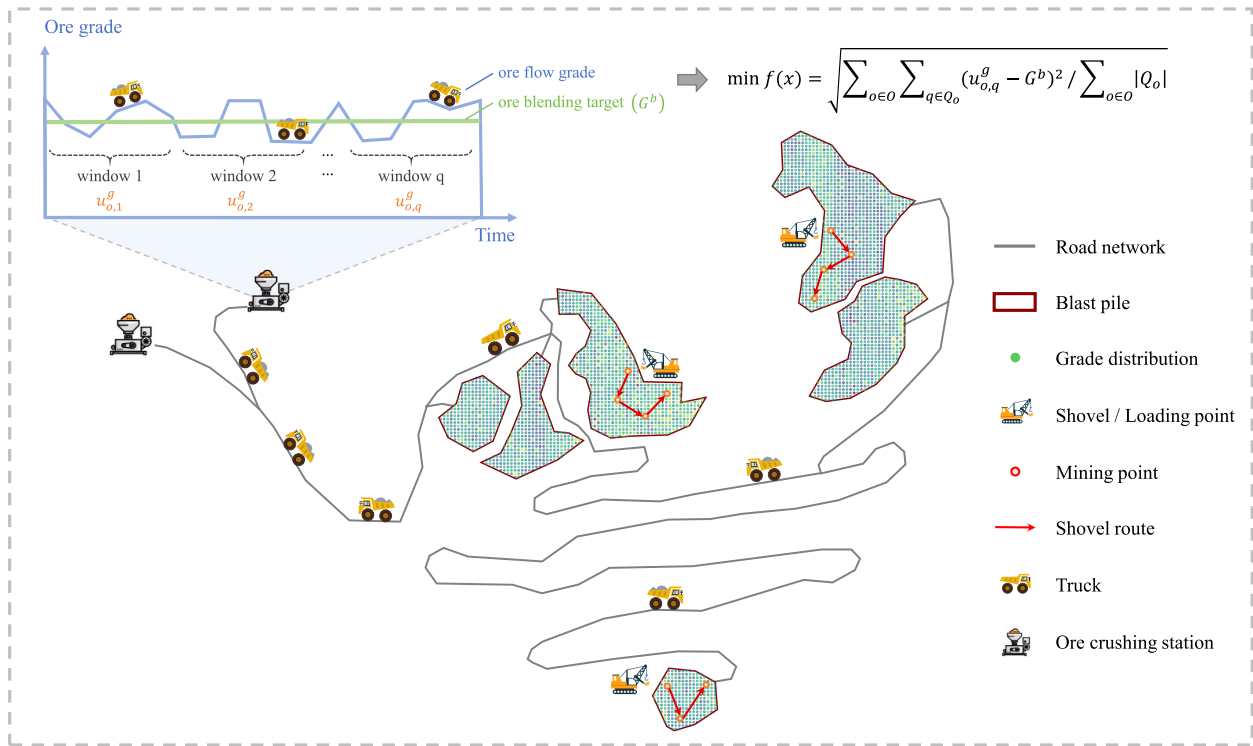


Fig. 1. Ore-flow blending. By simulating the truck's full cycle, including queueing, loading, heavy-haul transport, and unloading, the ore flow entering the crushing station/plant is grouped by sliding time windows. The objective is to minimize the deviation between the average grade within each window and the target grade, aiming to achieve temporal stability control of the ore flow grade delivered to the plant.

network topology changes. The superposition of this spatiotemporal deep coupling and stochastic disturbances results in typical high-dimensional nonlinear system characteristics.

- **Extreme decision timeliness:** The occurrence of dynamic events, such as equipment failures or road closures, is often sudden and unpredictable. This requires the scheduling system to possess a “perceive-and-decide” second-level response capability to avoid production interruptions or drastic grade fluctuations. Furthermore, the mining road network structure changes continuously with mining progress, requiring the decision model to adapt to scenarios with varying scales and topological structures.
- **Long-horizon decision dependency:** The objective of ore-flow optimization is to minimize the grade time-series deviation over a long cycle. This essentially constitutes a dynamic time-linkage optimization problem (Zhang et al., 2022), where decisions made at the current time step are strongly coupled with the constraints and objective values of future moments. This goal possesses significant “prospective” and “cumulative” properties, requiring the decision strategy to maintain a long-term non-greedy vision. This cross-temporal trade-off and coordination constitute a critical difficulty.

Current research on dynamic optimization under uncertainty generally falls into four categories, yet each faces inherent limitations when applied to the real-time ore-flow blending problem. Model-based exact optimization methods, such as stochastic programming (Schlenkrich et al., 2024; Dering and Swartz, 2022) and robust optimization (Koster et al., 2024; Lu and Shen, 2021), aim for theoretical optimality by explicitly modeling uncertainty. However, they often suffer from the curse of dimensionality, where the exponential growth of scenario trees creates a massive computational burden that precludes the second-level response capability required for online intervention. Conversely, rule-based reactive strategies (Hamzadayi and Yildiz, 2016) are widely adopted for their low latency but are inherently myopic. By relying on

greedy logic (Đurasević and Jakobović, 2018) (e.g., Shortest Processing Time), they prioritize immediate local gains at the expense of global long-term stability, often resulting in local optima. Swarm intelligence methods (Wang et al., 2024; Yazdani et al., 2021; Mavrovouniotis et al., 2020), while offering robustness, depend fundamentally on iterative search processes. In high-frequency dynamic environments, the necessity to re-optimize for every disturbance is computationally prohibitive, preventing the formation of a generalizable, fast-response policy. Furthermore, although existing learning-based methods enable end-to-end decision-making (Chiarot Villegas et al., 2024; Noriega et al., 2025), they typically employ “flat” multi-layer perceptron (MLP) architectures (de Carvalho and Dimitrakopoulos, 2021; Huo et al., 2024). These structures fail to capture the complex evolving graph topology of mining road networks and the long-horizon temporal dependencies of ore flow, severely limiting their scalability across varying fleet sizes and mine structures. Coupled with the challenges of sparse rewards and memory bottlenecks during long-horizon training (Luo et al., 2023, 2024), these constraints severely limit their scalability and practical deployment value.

To address these limitations, this paper proposes an ore-flow blending and truck scheduling framework based on a spatio-temporal graph policy network (STGPN). Unlike traditional methods, our approach leverages a graph neural network (GNN) to natively represent the evolving topology of the mining road network and a recurrent neural network (RNN) to maintain a memory of global temporal ore-flow dynamics. This architecture decouples the state representation into spatial and temporal components, allowing the agent to perceive global graph-structured information and historical trends. Consequently, it enables the learning of a robust policy that is invariant to fleet size and road topology, achieving sub-second end-to-end responses for large-scale dynamic scheduling. The main contributions of this paper are summarized as follows:

- A spatio-temporal graph policy network (STGPN) to handle high-dimensional dynamics and real-time decision response is designed. By integrating a recurrent temporal encoder with a gated

graph convolutional backbone, the model captures global ore-flow dynamics and spatial topological features. This architecture yields a robust and generalizable policy representation that is invariant to fleet size and road topology, enabling scalable deployment across different mines and dynamic scenarios.

- A two-stage training strategy that combines local short-scale pre-training with global long-scale fine-tuning is proposed, to address the computational and memory bottlenecks of large-scale, long-horizon decision-making. This divide-and-conquer approach efficiently transfers local knowledge to global tasks, making the solution computationally feasible for real-world on-site deployment.
- The proposed strategy is validated using multiple real-world mining datasets under dynamic and uncertain scenarios. Experimental results demonstrate that our approach significantly outperforms existing baselines in ore flow stability control, exhibiting superior robustness and generalization across varying road structures, equipment scales, and dynamic events.

The remainder of this paper is organized as follows. Section 2 provides a literature review of related work. Section 3 introduces an ore-flow blending model for open-pit mining. Section 4 presents the proposed framework in detail. Sections 5 and 6 carry out a set of experiments to comprehensively evaluate the performance of the proposed framework. Finally, Section 7 summarizes the main findings and future work of this study.

2. Related work

Due to the ubiquitous nature of dynamic and uncertain factors, many “optimal solutions” that appear perfect during the offline planning phase prove extremely fragile during actual execution. They may even trigger systemic nervousness characterized by oscillations due to frequent rescheduling. This section reviews four categories of methods for dynamic optimization under uncertainty and outlines the motivation for this paper.

2.1. Model-based exact optimization

Early model-driven methods explicitly characterize uncertainty (via probability distributions or uncertainty sets) to secure statistical or worst-case performance guarantees. Two-stage or multi-stage stochastic programming (Schlenkrich et al., 2024; Dering and Swartz, 2022) designs main decisions and recourse strategies based on expectation, while robust optimization (RO) and distributionally robust optimization (DRO) (Koster et al., 2024; Lu and Shen, 2021; Mohajerin Esfahani and Kuhn, 2018) emphasize conservatism regarding distributional uncertainty. Meanwhile, rolling horizon control (Glomb et al., 2022; Herding and Mönch, 2024) implementations have become a common engineering solution to balance global and local decisions. This approach decomposes long-horizon problems into short-term repetitive optimizations, resolving the problem in each cycle using the latest information. The advantages of model-driven methods lie in their interpretability and theoretical guarantees. However, their primary bottlenecks are the explosive growth of scenario or state spaces and online solution latency. Consequently, scenario reduction, decomposition, or approximation techniques are typically required in scenarios involving long horizons or high-dimensional uncertainty.

2.2. Rule-based reactive strategies

In high-frequency and event-driven real-world systems, rapidly providing feasible adjustments is often more critical than seeking global optimality, making reactive and rescheduling strategies the engineering preference (Holguin Jimenez et al., 2024). Among these, priority-based or dispatching rules (Đurasević and Jakobović, 2018; Zhang

et al., 2021) rapidly generate or adjust plans based on rules such as arrival time, shortest processing time, or urgency. These are commonly used for online production and on-site scheduling due to their ease of implementation and low latency, although they are limited regarding complex constraints and global performance. Conversely, event-triggered or period-triggered rescheduling strategies (Hamzadayi and Yildiz, 2016; Pillac et al., 2013) design trigger conditions (such as major disturbances or time intervals) for re-planning. Research in this area focuses on trigger timing and the trade-off between rescheduling cost and stability (known as alteration cost), proposing the introduction of alteration penalties into the optimization objective to control frequent reordering. While reactive strategies are optimal for real-time performance, they lack long-term optimality or robustness guarantees. In engineering practice, a common approach is to utilize reactive scheduling as a “safety layer” upon which more complex offline or rolling optimization is executed.

2.3. Swarm intelligence methods

When proactive measures fail to withstand drastic disturbances, the system must undergo reactive adjustments. In this context, swarm intelligence methods (Wang et al., 2024; Jia et al., 2017; Yazdani et al., 2021, 2023), represented by ant colony optimization (ACO) (Mavrovouniotis et al., 2017a, 2020; Rezvanian et al., 2023; de Oliveira et al., 2021), offer inherent advantages for dynamic problems due to their distributed parallel search and pheromone-based adaptive mechanisms. Such algorithms focus on how to rapidly converge and maintain population diversity in dynamic and uncertain environments. Early works enhanced algorithm robustness against periodic or minor changes through strategies such as restart strategies (Zhou et al., 2009; Angus and Hendtlass, 2005; Samà et al., 2016; Prakasam and Savarimuthu, 2019), pheromone adjustment (Eyckelhof and Snoek, 2002; Mavrovouniotis and Yang, 2014; Su et al., 2021), immigrant schemes (Mavrovouniotis and Yang, 2013; Gao et al., 2016; Mavrovouniotis and Yang, 2015), and memory pools (Guntsch and Middendorf, 2002; Mavrovouniotis and Yang, 2012; Shi et al., 2022), attempting to balance retaining historical search experience with rapid adaptation to new environments. To further address the loss of population diversity in dynamic environments, multi-population mechanisms (Mavrovouniotis et al., 2024; González et al., 2022) adopt a divide-and-conquer strategy to partition the search space into sub-regions and assign a sub-population to each, maintaining global and local diversity while enhancing the algorithm’s ability to adapt to environmental changes and track multiple optima simultaneously (Yazdani et al., 2020; F. Li et al., 2024; Liu et al., 2020). Recently, combining local search or machine learning to assist optimization algorithms in decision-making has become a new trend. On one hand, embedding local search operators or heuristics into ACO to form memetic algorithms (Mavrovouniotis et al., 2016, 2019; Skackauskas, 2022) significantly improves local optimization capabilities. On the other hand, the “learning + search” paradigm has emerged, attempting to extract structured knowledge from historical data; for instance, PPL-ACO (Xiang et al., 2021) utilizes neural networks to predict node visitation sequences after dynamic changes, thereby guiding ants in solution construction.

However, despite the progress made by swarm intelligence algorithms in handling dynamic problems, they are based on iterative search leading to bottlenecks such as excessive computation time and difficulty in meeting real-time requirements when facing high-frequency, drastically changing real-time scheduling scenarios. As an iterative search method, it does not learn a generalizable policy but rather re-searches for each new instance (albeit guided by memory bias). Moreover, it relies heavily on hand-crafted heuristic rules, making it difficult to capture complex high-dimensional spatiotemporal dynamics and uncertainties and long-horizon decision dependencies in modern dynamic data streams.

2.4. Learning-based methods

The primary limitation of evolutionary algorithms lies in the computationally expensive evaluation costs introduced by the extensive iterative simulation (Patterson et al., 2017; N. Li et al., 2024). In contrast, learning-based methods enable the agent to learn scheduling strategies in a dynamic environment (Huo et al., 2023; Hazrathosseini and Moradi Afrapoli, 2024). By employing end-to-end agent models, these methods can effectively address the high computational cost challenges of truck scheduling simulation. For instance, DRL approaches based on Deep Q-Networks (de Carvalho and Dimitrakopoulos, 2021; Chiarot Villegas et al., 2024) and Double Deep Q-Networks (Huo et al., 2024; Noriega et al., 2025) have been adopted to provide rapid and effective responses to dynamic and uncertain emergent situations. To further adapt to scenarios with dynamically changing numbers of trucks during operations and large-scale heterogeneous fleets, related studies have proposed multi-agent DRL (Zhang et al., 2020) and centralized single-agent truck scheduling model (Matsui et al., 2023). In multi-agent DRL approach, each truck is treated as an agent sharing the same policy network, which receives the observed states of each agent and independently outputs actions for each agent. A shared experience training method is used to collect environment interaction sequences from all agents and learn in a centralized manner. On the other hand, the centralized approach treats scheduler as the agent, binding truck states with loading and unloading point locations to achieve a robust state space representation independent of the number of trucks.

In summary, DRL-based methods can currently handle dynamic heterogeneous truck fleets, their policy network architectures are still based on simple MLP, which typically require manual design of the state space and rely heavily on prior knowledge of the problem. These limitations prevent them from effectively capturing the high-dimensional global spatio-temporal features of mining areas and adapting to different mining area road network structures, varying equipment scales, and diverse dynamic and uncertain scheduling scenarios. Furthermore, the long-time scale simulation scheduling scenarios exacerbate the challenges for DRL methods, as they face high memory consumption during training (Luo et al., 2023, 2024).

2.5. Motivation

To summarize, existing literature falls short of providing a method that simultaneously tackles high-dimensional spatiotemporal representation, real-time responsiveness, and long-horizon decision dependencies in highly dynamic environments. While model-based approaches offer theoretically high-quality solutions, they suffer from the “re-optimization” bottleneck. This limitation is particularly evident when examining the concept of “robustness” in traditional RO paradigms. In classical RO, robustness typically refers to static solution feasibility, where an a priori schedule is optimized to remain near-optimal or strictly feasible under the worst-case realization of uncertainty sets. However, such static conservatism often leads to excessive performance loss and becomes computationally intractable when the environment undergoes frequent discrete disruptions, such as sudden equipment failures or temporary road blockages.

Furthermore, alternative optimization methodologies also exhibit certain shortcomings in dynamic settings. Although swarm intelligence methods (e.g., ACO) possess adaptive capabilities through mechanisms like pheromone updates, they fundamentally rely on instance-specific iterative search. The “knowledge” acquired is typically bound to the current problem instance and road network topology, making it difficult to directly transfer to new scenarios without computationally expensive re-searching, precluding sub-second responses. Conversely, rule-based strategies achieve the necessary low latency for real-time dispatching but are fundamentally myopic; they neglect long-term ore flow dependencies and frequently sacrifice future stability for immediate gains. Similarly, while recent learning-based methods offer

a promising avenue for end-to-end decision-making, current mining implementations are severely constrained by their reliance on “flat” MLP architectures. By reducing the dynamic environment to fixed vectors and ignoring the underlying graph topology, these models fail to capture complex spatial dependencies or generalize to varying fleet sizes.

It is precisely these limitations that necessitate the proposed STGPN framework. Unlike previous approaches that seek a single static robust schedule, our work focuses on “dynamic policy robustness” (or reactive adaptability). We define this robustness as the capability of the learned policy to perceive environmental state transitions and output high-quality, adaptive dispatching decisions in real-time when encountering unforeseen disruptions. By leveraging GNNs to natively represent the evolving topology of the mining environment and Recurrent Neural Networks (RNNs) to maintain a memory of global temporal dynamics, the STGPN yields a robust policy representation that is invariant to fleet size and road network topology. This fundamentally bridges the gap between theoretical worst-case protection and the practical necessity for instantaneous responsiveness, enabling stable transferability across different mines and dynamic, uncertain scenarios.

3. Problem formulation

Traditional ore blending is often treated as a static resource allocation problem (Liu et al., 2021; Nikbin and Moradi Afrapoli, 2023; Chen et al., 2022), optimizing the average grade while neglecting the temporal continuity of production and execution uncertainties. Even dynamic approaches like rolling horizon optimization (Moradi Afrapoli et al., 2019; Yao et al., 2023) rely on frequent re-planning, failing to model the continuous temporal evolution of ore flow. To overcome these limitations, this section fundamentally reframes ore-flow blending as a real-time scheduling decision problem under dynamic and uncertain environments. We first mathematically formulate the ore-flow blending objective and operational constraints, and then establish the markov decision process (MDP) framework to enable an end-to-end DRL solution capable of handling system dynamics and uncertainty.

3.1. Ore-flow blending objective

We propose a simulation-based ore-flow blending model where the stability of the ore grade entering the processing plant is the primary control objective. In this framework, the truck scheduling system acts as a dynamic actuator. As illustrated in Fig. 1, the arrival order and time intervals of trucks (carrying high-grade or low-grade ore) collectively determine the instantaneous fluctuations of the ore flow grade.

To quantify ore flow stability in the temporal dimension, we introduce a sliding time window mechanism. Unlike minimizing the deviation of the total shift average, our goal is to control the grade time-series. The truck flow entering the crushing station is grouped into sliding windows based on arrival times, with the window size determined by the buffer capacity. The optimization objective is to minimize the root mean squared error (RMSE) of the ore flow grade across all time windows, ensuring the grade $u_{o,q}^g$ within each window q at crushing station o remains close to the target grade G^b , as shown in Eq. (1). The notations used in the formulation are defined in Table 1.

$$\min f(x_{k,m,n}) = \sqrt{\frac{\sum_{o \in O} \sum_{q \in Q_o} (u_{o,q}^g - G^b)^2}{\sum_{o \in O} |Q_o|}}, \quad (1)$$

where O is the set of crushing stations, and Q_o is the set of time windows at station o . This objective is the first to directly link micro-level truck scheduling decisions with macro-level plant feed quality stability in the temporal dimension. Minimizing this objective is, in essence, solving a complex spatio-temporal process control problem under dynamic and uncertain environments. It requires the system's decisions to be forward-looking, as a current scheduling decision will directly impact the grade stability of multiple future time windows.

Table 1
Notations and parameters.

Notation	Definition
Index sets	
$k \in K$	Truck index and set
$i \in I$	Loading point/shovel index and set
$o \in O$	Ore crushing station index and set
$r \in R$	Rock crushing station index and set
$j \in J$	Unloading point index and set, $J = O \cup R$
$l \in L_i$	Mining point index and set for shovel i
$m \in M$	Loading and unloading point index and set, $M = I \cup J$
$n \in N_k$	Route step index and set for truck k
$q \in Q_m$	Chronologically ordered truck queue at point m
Input parameters	
T	Shift duration (minutes)
G^b	Ore blending target (%) for beneficiation plant
G^m	Marginal grade (%), used to differentiate between ore and waste rock
C	Buffer capacity (tons) of crushing station
$p_{i,l}^o, p_{i,l}^g$	Ore quantity (tons) and grade (%) at mining point l of shovel i
s_i^o	Total ore quantity (tons) of shovel i
t_k^u, t_k^c	Unloading time (minutes) and capacity (tons) of truck k
$t_{k,m,m'}^t$	Travel time (minutes) between points m and m' for truck k
$t_{k,i}^l$	Loading time (minutes) for truck k at shovel i
Decision variables & state trackers	
$x_{k,m,n}$	Binary variable for truck k visiting point m as its n th destination
t_k^r	Total route length (number of visited points) for truck k
$t_{k,n}^a, t_{k,n}^w$	Arrival time and waiting time of truck k at its n th destination
$\mathcal{K}(m, q)$	Truck identity tracking the specific truck occupying the q th position at point m
$d_{m,q}^a, d_{m,q}^o, d_{m,q}^g$	Arrival time, ore quantity, and grade of the q th truck in the queue at point m
$\mathcal{M}(i, q)$	Subset of mining points excavated to fulfill the loaded quantity of the q th truck at shovel i
$\Delta m_{i,q,l}$	Specific tonnage extracted from point l for the q th truck at shovel i
$u_{o,q}^g, u_{o,q}^l$	Average ore grade and truck queue length in the q th sliding time window at station o

3.2. Operational constraints

Fig. 2 illustrates the logical relationships and information flow among the core variables. The model is structured into four sequential logical layers corresponding to the physical scheduling lifecycle. Layer 1 defines the agent's spatial routing decisions ($x_{k,m,n}$); Layer 2 tracks the resulting temporal state transitions and kinematics for the trucks ($t_{k,n}^a, t_{k,n}^w$); Layer 3 maintains the event-driven chronologically ordered queues (Q_m) using a truck identity tracking mechanism ($\mathcal{K}$); and Layer 4 computes the physical ore flow, determining the loaded ore grade from the shovels ($d_{i,q}^g$), the conserved grade during transport ($d_{j,q}^g$), and ultimately the aggregated time-window grade ($u_{o,q}^g$) to minimize the RMSE objective.

3.2.1. Truck routing & destination

All trucks depart at the beginning of the shift. A truck must complete its operational cycles within the shift duration T and maintain a continuous sequence, visiting exactly one destination per route step n . Crucially, trucks alternate between loading points (I) and unloading points (J):

$$t_{k,0}^a = 0, \quad \forall k, \quad (2)$$

$$t_{k,i}^a \leq T < t_{k,i+1}^a, \quad \forall k, \quad (3)$$

$$\sum_{m \in M} x_{k,m,n} = 1, \quad \forall k, n, \quad (4)$$

$$\sum_{i \in I} x_{k,i,n} = 1, \quad \forall k, \forall n \text{ s.t. } n \equiv 0 \pmod{2}, \quad (5)$$

$$\sum_{j \in J} x_{k,j,n} = 1, \quad \forall k, \forall n \text{ s.t. } n \equiv 1 \pmod{2}. \quad (6)$$

3.2.2. Event-driven queuing & temporal kinematics

In traditional mathematical programming, tracking queue positions requires complex binary mapping matrices. However, since our model operates within an event-driven simulation environment (detailed in

Section 3.3), physical queuing dynamics are automatically maintained chronologically. Therefore, to reduce model complexity and improve readability, we formulate the queue as an ordered set.

Let Q_m denote the set of trucks assigned to node m , chronologically ordered by their arrival times $t_{k,n}^a$. For the q th truck in this queue, we define three key trackers: $d_{m,q}^a$ (arrival time), $d_{m,q}^o$ (loaded quantity), and $d_{m,q}^g$ (ore grade). For instance, if truck k is the 5th vehicle to arrive at shovel i ($q = 5$), $d_{i,5}^g$ captures the exact ore grade excavated and loaded into this specific truck. The sequence is strictly ordered:

$$d_{m,q}^a \leq d_{m,q+1}^a, \quad \forall m, q \in Q_m. \quad (7)$$

The waiting time $t_{k,n}^w$ for a truck is naturally derived from the departure time of the $(q-1)$ th truck in the same queue Q_m . Consequently, the arrival time of truck k at step $n+1$ depends continuously on its arrival time at step n , its waiting time $t_{k,n}^w$, the operational service time (loading or unloading), and the travel time t^t to the next node:

$$\begin{aligned} t_{k,n+1}^a &= t_{k,n}^a + t_{k,n}^w + \sum_{i \in I} x_{k,i,n} \cdot t_{k,i}^l \\ &+ \sum_{m,m' \in M} x_{k,m,n} \cdot x_{k,m',n+1} \cdot t_{k,m,m'}^t, \end{aligned} \quad (8)$$

$$\forall k, \forall n \text{ s.t. } n \equiv 0 \pmod{2},$$

$$\begin{aligned} t_{k,n+1}^a &= t_{k,n}^a + t_{k,n}^w + t_k^u \\ &+ \sum_{m,m' \in M} x_{k,m,n} \cdot x_{k,m',n+1} \cdot t_{k,m,m'}^t, \end{aligned} \quad (9)$$

$$\forall k, \forall n \text{ s.t. } n \equiv 1 \pmod{2}.$$

3.2.3. Shovel mining sequence & capacity

As trucks arrive at loading point i in the ordered sequence $q \in Q_i$, the shovel sequentially excavates consecutive mining points $l \in L_i$. Instead of formulating complex binary step-functions to track the exact mining boundary for each truck, we define this as an event-driven continuous allocation.

Let $\mathcal{M}(i, q)$ denote the subset of mining points excavated to fulfill the loaded quantity $d_{i,q}^o$ of the q th truck. Let $\Delta m_{i,q,l}$ be the specific tonnage

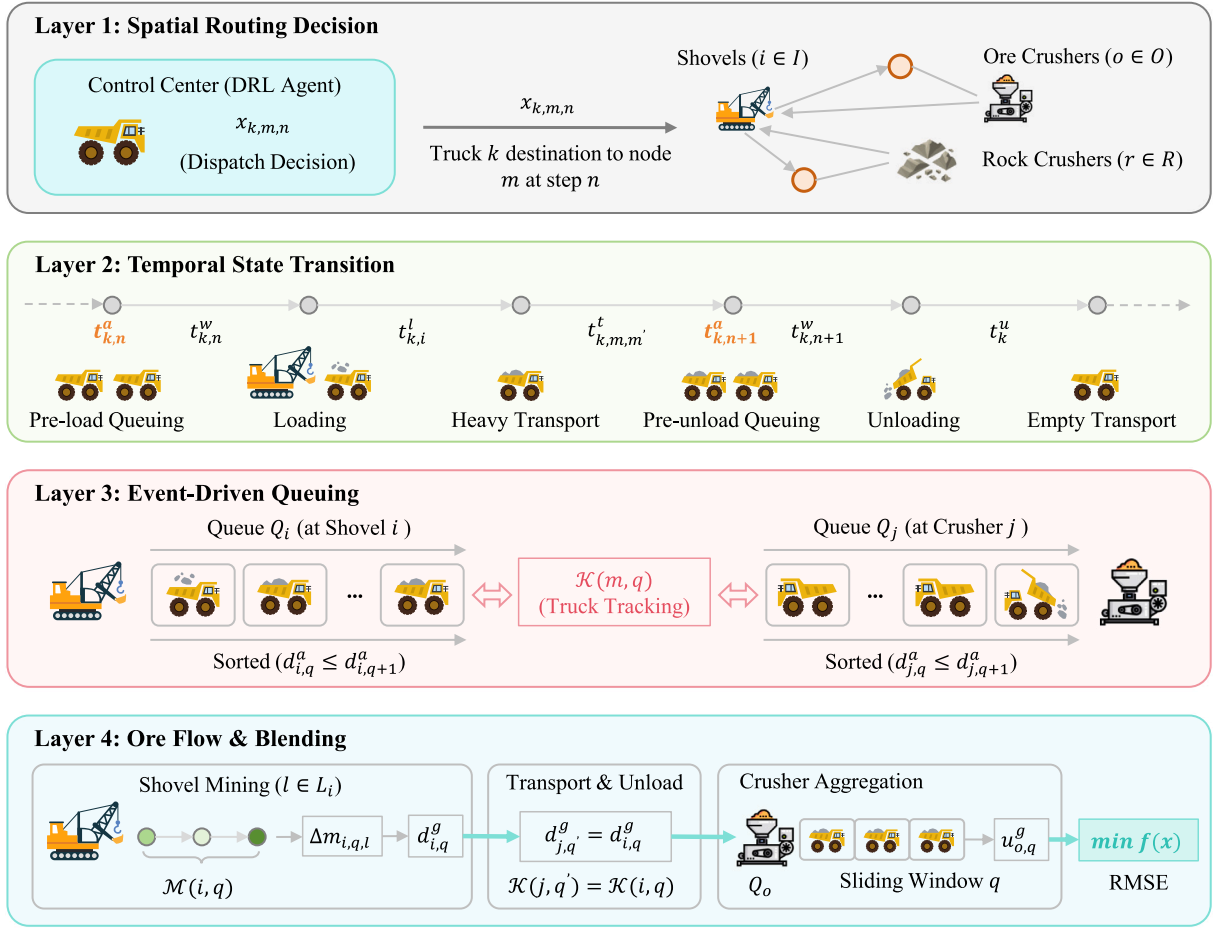


Fig. 2. Schematic diagram of ore-flow blending model structure. The framework is formulated through four logical layers: (1) Spatial Routing Decision driven by the dispatch agent; (2) Temporal State Transition capturing truck kinematics; (3) Event-Driven Queuing utilizing continuous truck identity tracking; and (4) Ore Flow & Blending evaluating mass conservation and grade aggregation to minimize the final grade RMSE objective.

extracted from point l for this truck. The allocation strictly follows the chronological queue, ensuring mass conservation:

$$\sum_{l \in \mathcal{M}(i,q)} \Delta m_{i,q,l} = d_{i,q}^o \equiv t_k^c, \quad \forall i, q \in Q_i. \quad (10)$$

The total amount of ore transported from a shovel cannot exceed its total available quantity:

$$\sum_{k \in K} \sum_{n \in N_k} x_{k,i,n} \cdot t_k^c < s_i^o, \quad \forall i \in I. \quad (11)$$

The ore grade loaded into the q th truck ($d_{i,q}^g$) is naturally derived as the weighted average of the grades $p_{i,l}^g$ from the chronologically assigned mining points:

$$d_{i,q}^g = \frac{1}{d_{i,q}^o} \sum_{l \in \mathcal{M}(i,q)} \Delta m_{i,q,l} \cdot p_{i,l}^g. \quad (12)$$

3.2.4. Grade conservation & material routing

Let $\mathcal{K}(m, q) \in K$ denote the specific truck occupying the q th position in the queue Q_m . When a truck completes loading at shovel i , it carries an ore grade of $d_{i,q}^g$. Because mass and grade are conserved during transportation, when this exact same truck subsequently arrives at unloading point j and joins its queue at position q' , its unloaded grade is simply the grade it acquired at the loading point:

$$d_{j,q'}^g = d_{i,q}^g, \quad \forall (j, q'), (i, q) \text{ s.t. } \mathcal{K}(j, q') = \mathcal{K}(i, q). \quad (13)$$

Based on this conserved grade, ore and waste rock must be strictly routed to their respective crushing stations depending on the marginal

grade G^m :

$$d_{o,q'}^g \geq G^m, \quad \forall o, q' \quad d_{r,q'}^g < G^m, \quad \forall r, q'. \quad (14)$$

3.2.5. Time window grade aggregation

Finally, as trucks unload at the ore crushing station o , the material is aggregated into discrete time windows of size C . The average grade $u_{o,q}^g$ of the q th time window is the weighted mean of the arriving trucks' grades, forming the final time-series fed into the objective function (Eq. (1)):

$$\sum_{w=q}^{q+u_{o,q}^1-1} d_{o,w}^o \leq C < \sum_{w=q}^{q+u_{o,q}^1} d_{o,w}^o, \quad \forall o, q, \quad (15)$$

$$u_{o,q}^g = \frac{\sum_{w=q}^{q+u_{o,q}^1-1} d_{o,w}^o \cdot d_{o,w}^g}{\sum_{w=q}^{q+u_{o,q}^1-1} d_{o,w}^o}, \quad \forall o, q. \quad (16)$$

3.3. MDP framework

The ore-flow blending problem is characterized by high-dimensional spatiotemporal states and stochastic disturbances. To achieve sub-second response times and global optimization, we formulate the problem as a MDP.

3.3.1. Agent

We employ a centralized single-agent model representing the “central dispatch office”. The agent $\pi_\theta(a|s)$ observes the global state s_t and

makes decisions for all trucks in the mine. This centralization therefore possesses a global field of view, and its architecture is independent of the number of trucks, which allows our method to generalize to truck fleets of varying sizes.

3.3.2. State (s_t)

The state s_t at decision time t is the global spatio-temporal snapshot of the mine observed by the “central dispatch office”. It must comprehensively capture the high-dimensional spatio-temporal dynamics of the environment and satisfy scalability requirements. A flat vector cannot represent the topological structure and temporal dependencies of the mine road network. Therefore, we design a spatio-temporal graph structure for the state representation. The truck flow sequence (TFS) at loading/unloading points and the shovel mining sequence (SMS) are used as global temporal features to capture the historical cumulative effects of the ore flow and future grade trends. Concurrently, local spatial features (LSF) are introduced to capture the graph topology of the mine road network and equipment heterogeneity.

Truck Flow Sequence (TFS). The loading/unloading points maintain a truck flow sequence ordered by truck arrivals, which records the historical scheduling information of all trucks and represents the decisions made in previous time steps. The features of the truck flow sequence include: truck arrival time, $d_{m,q}^a$; start time of loading/unloading, $d_{m,q}^s$; departure time, $d_{m,q}^d$; ore quantity, $d_{m,q}^o$; and the ore grade loaded, $d_{m,q}^g$. These features describe the scheduling information for the q th truck arriving at the loading/unloading point m . While RNNs can process variable-length sequences, processing the entire shift history is computationally expensive and susceptible to vanishing gradients. Therefore, we define a sliding historical window of the N_{TFS} most recent trucks to capture the relevant temporal dependencies and facilitate efficient batch operations. Considering all $|M|$ loading and unloading points globally, the TFS feature tensor is formalized with a dimension of $\mathbb{R}^{|M| \times N_{\text{TFS}} \times 5}$.

Shovel Mining Sequence (SMS). The loading points maintain a mining sequence based on the current excavation progress of the shovels and their subsequent travel sequence. This sequence records the ore grade information for the ore that has not yet been mined by the shovel and remains in the blast pile. The features of the mining sequence include: ore volume, $p_{i,l}^o$; and ore grade, $p_{i,l}^g$ at the mining point. These features describe the information for the l th upcoming mining point at loading point i . A mining point refers to the blast pile face area within the operational radius of the shovel, where the ore volume and grade at this point represent the average values of the ore in this region. Similarly, by defining a look-ahead window of N_{SMS} upcoming mining points to represent the immediate future grade supply, and considering all $|I|$ shovels (loading points) globally, the SMS feature tensor takes a dimension of $\mathbb{R}^{|I| \times N_{\text{SMS}} \times 2}$.

Local Spatial Features (LSF). Given the heterogeneity of the transportation truck fleet and the loading shovel fleet, including differences in truck capacity, loading/unloading time, and variations in heavy-load and empty-load transport time for different trucks, the state space must capture the following information: the capacity of the currently selected truck, t_{cur}^c ; the loading/unloading time for each loading/unloading point, $\{t_m^h\}_{m \in M}$; the current truck position, $\{t_m^p = 0, 1\}_{m \in M}$; and the heavy-load and empty-load transport time between each pair of loading/unloading points, $\{t_{\text{cur},m,m'}^t\}_{m,m' \in M}$. Given the close relationship between the truck state and the road network graph, the current truck's spatial features can be represented as a combination of node and edge states, i.e., $\text{LSF} = (\text{TNode}, \text{TEdge})$. Specifically, the node features for all loading/unloading points, $\text{TNode} = \{(t_{\text{cur}}^c, t_m^h, t_m^p)\}_{m \in M} \in \mathbb{R}^{|M| \times 3}$; and the edge features, $\text{TEdge} = \{t_{\text{cur},m,m'}^t\}_{m,m' \in M} \in \mathbb{R}^{|M| \times |M| \times 1}$, where $|M|$ denotes the total number of loading and unloading points.

3.3.3. Action (a_t)

Decision-making does not occur at fixed time steps; it is event-driven. When any truck k completes its task (e.g., finishes unloading) and becomes idle, the environment triggers a decision event. At this point, the “central dispatch office” agent π_θ is activated, observes the current global state s_t , and computes an action a_t specifically for that truck k . The action a_t is to select a next destination $m \in M$ for this truck from its set of feasible nodes (e.g., a truck at a loading point must go to an unloading point). We use a masking mechanism to filter out invalid actions.

3.3.4. Transition ($p(s_{t+1}|s_t, a_t)$)

After an action a_t (assigning destination m to truck k) is executed, the environment state s_t is updated to s_{t+1} according to the simulation logic, which includes the switching of the truck's work cycle states (e.g., heavy-haul, empty-haul, queueing). This update includes changing the location of truck k , updating the TFS of the destination node m , and (if the action m is to a loading point) updating the corresponding shovel's SMS. The state transition is executed by a high-fidelity simulation environment that explicitly models uncertainty. Upon executing action a_t , the environment advances the simulation clock. Crucially, this transition is stochastic. This stochastic transition function $s_{t+1} \sim P(s_t, a_t)$ creates the dynamic and uncertain environment in which the agent must learn robust policies.

- **Parameter Uncertainty:** Truck travel times and shovel loading times are not fixed but sampled from dynamic distributions (e.g., unloading time $t_k^u \sim \mathcal{N}(3, 0.04)$) to reflect operational variability (detailed in Section 5).
- **Dynamic Events:** The environment simulates discrete disruption events, such as equipment failures or weather changes (detailed in Section 5.1), which instantaneously alter the road network topology or efficiency.

3.3.5. Reward (R)

The agent does not receive any reward until a complete episode (i.e., one shift T) is finished. At the end of the episode, all agents receive a shared global reward R , which is defined as the negative value of the ore-flow grade RMSE: $R = -f(a_{1:T})$. This is perfectly aligned with our optimization objective (Eq. (1)).

While it is common practice in DRL to employ reward decomposition or intermediate dense rewards to accelerate training and mitigate sparsity, we deliberately avoid such mechanisms. In the context of ore-flow blending, intermediate rewards (e.g., minimizing the grade deviation for a single immediate time window) inherently encourage myopic, locally greedy behavior. Because ore flow exhibits strong temporal linkage, short-sighted decisions often trap the system in severe long-term fluctuations. As demonstrated in our policy analysis (Section 6.8), achieving global stability frequently requires the agent to sacrifice short-term gains. Therefore, the sparse global reward is indispensable for forcing the agent to learn a far-sighted policy.

To mitigate the severe credit assignment problem and training instability typically caused by this sparse reward structure, we designed a specific two-stage training strategy and a variance-reduced policy gradient baseline, which are detailed in Section 4.

4. The proposed methods

To address the core challenges identified in the ore-flow blending problem, specifically high-dimensional spatiotemporal dynamism and uncertainty, extreme decision timeliness, and long-horizon decision dependencies, we propose a robust end-to-end DRL framework. The framework consists of two strategic components designed to ensure robustness and responsiveness: a spatio-temporal graph policy network and a two-stage training strategy.

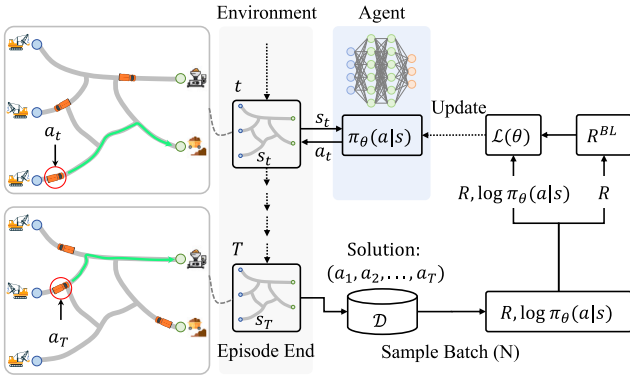


Fig. 3. DRL framework for ore-flow blending. The agent interacts with the mine environment in an event-driven manner. At each decision epoch t , the agent observes the global state s_t and outputs a dispatching action a_t . This process iterates until the shift ends at time T . The complete episode trajectory is stored in a buffer D to compute a global, sparse reward R (representing grade RMSE). Finally, the policy parameters θ are updated using a policy gradient loss $\mathcal{L}(\theta)$ centered by a moving average baseline R^{BL} to reduce training variance.

4.1. DRL framework for real-time response

To achieve the “perceive-and-decide” capability required for extreme decision timeliness, we formulate the problem as an event-driven MDP and solve it using a centralized single-agent architecture.

4.1.1. Centralized decision-making and event-driven

As illustrated in Fig. 3, we adopt a centralized single-agent model, where the agent represents the “central dispatcher” (as defined in Section 3.3.1). This agent, π_θ , is responsible for observing the global state s_t and making decisions for all trucks in the mine. This centralized architecture is a necessary prerequisite for solving strong coordination problems like grade balancing, as it provides the policy with a global perspective.

Decision-making does not occur at fixed time steps; it is purely event-driven. This aligns with realistic mine operations, as the mining system is an asynchronous system. As described in Section 3.3.3, when any truck k completes its task (e.g., finishes unloading) and becomes idle, the simulation environment automatically triggers a decision event. At this moment, the agent (central dispatcher) π_θ is activated, observes the current global state s_t , and computes an action a_t specifically for the truck k that triggered this event.

In this framework, the agent interacts with the environment by observing the global state and taking corresponding, precise actions. The action changes the environment, leading to a transition to a new state (Section 3.3.4), which prompts the agent to make subsequent decisions. This process repeats until a shift (Episode) is completed. At that point, the environment calculates the final RMSE according to Eq. (1) and returns a global, sparse reward R (Section 3.3.5). The agent π_θ then updates its network parameters θ based on this final reward R .

4.1.2. Graph-structured prior knowledge

The ore-flow blending problem can be viewed as a scheduling decision task on the mine’s road network graph. Consider a graph $G = (V, E, A)$, where $V = \{v_m\}_{m \in I \cup J}$ denotes the set of loading points I and unloading points J on the road network; $E = \{e_{ij}\}_{i \in I, j \in J}$ is the set of shortest paths between loading and unloading points, as trucks perform the scheduling transport task via these shortest paths; and $A \in \{0, 1\}^{|V| \times |V|}$ is the corresponding adjacency matrix. Given the nature of the scheduling transport task, the adjacency matrix dictates that loading points are adjacent only to unloading points, and there is no adjacency between loading points or between unloading points. To

fully leverage the inherent graph-structured prior knowledge of the ore-flow blending problem, and thus efficiently solve the truck scheduling decision task on the graph, we utilize a GNN to parameterize the agent’s stochastic policy $\pi_\theta(a|s)$, with trainable parameters θ .

4.2. STGPN architecture

Standard MLP architectures fail to capture the complex topology and temporal dependencies of mining systems. To tackle the high-dimensional spatiotemporal dynamism and uncertainty, we design the STGPN (Fig. 4). Its core innovation lies in decoupling the state representation into global temporal dynamics and spatial topology. The STGPN utilizes RNN and gated graph convolutional network (GatedGCN) (Dwivedi et al., 2023) to extract and integrate the ore flow temporal features (TFS, SMS) and the dispatching graph structural features (LSF) from the mining environment, respectively. It organizes the global spatio-temporal state features using a graph structure, making it independent of equipment scale and road network structure (i.e., possessing permutation invariance). This allows the method to learn robust and generalizable scheduling policy across different mine road network structures, equipment scales, and dynamic scheduling scenarios.

4.2.1. State encoder

The encoder is responsible for transforming raw and heterogeneous engineering data into high-dimensional graph latent representations (node features h_i^0 and edge features e_{ij}^0) that capture the dynamic evolution of the system.

Temporal encoder. Ore flow is a time-series process with cumulative effects. We employ gated recurrent unit (GRU) to encode the TFS and SMS. The hidden state of the GRU acts as a dynamic memory. It tracks historical grade trends and predicts future supply. This effectively addresses the uncertainty in ore grade fluctuations over time.

Spatial encoder. The road network topology changes as mining progresses or when roads are blocked. We encode the LSF of nodes and edges using shared MLPs. This pointwise encoding ensures the model focuses on the attributes of the graph elements rather than their fixed indices, enabling generalization to new road network structures.

Feature fusion. Finally, we fuse the features from the temporal and spatial encoders (e.g., via Sum+MLP for node-level processing) to generate the initial high-dimensional features h_i^0 and e_{ij}^0 . This fusion step is critical to the STGPN: it injects the global temporal context into the local spatial nodes. This ensures that before the GNN even begins its reasoning, every node on the graph is already equipped with global grade-awareness.

4.2.2. Graph embedding

To handle the spatial coupling of the mining network, we utilize a GatedGCN. Unlike standard GCNs, GatedGCN explicitly models edge features, such as dynamic travel times. This is crucial for robustness because weather changes or congestion directly affect edge weights. Through the message-passing mechanism (Fig. 5), the agent perceives how a perturbation at one node propagates through the network. This allows it to make routing decisions that bypass bottlenecks or balance queues globally. Let h_i^l and e_{ij}^l represent the node and edge features associated with node i and edge ij at the l th layer of the GNN. The corresponding formulation for this step is:

$$h_i^{l+1} = h_i^l + \text{ReLU} \left(\text{BN} \left(U^l h_i^l + \text{AGGR}_{j \in N_i} (\sigma(e_{ij}^l) \odot V^l h_j^l) \right) \right), \quad (17)$$

$$e_{ij}^{l+1} = e_{ij}^l + \text{ReLU}(\text{BN}(A^l e_{ij}^l + B^l h_i^l + C^l h_j^l)), \quad (18)$$

where $U^l, V^l, A^l, B^l, C^l \in \mathbb{R}^{d \times d}$ represent the learnable parameters at each layer of the GatedGCN; BN denotes the batch normalization layer,

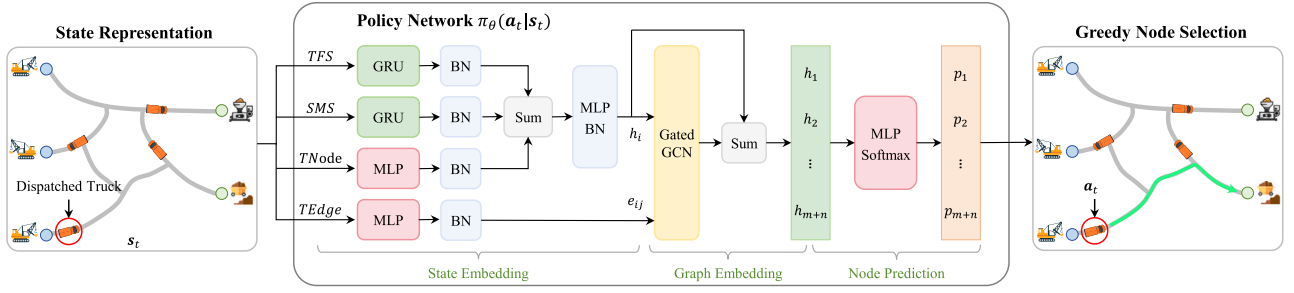


Fig. 4. The architecture of STGPN. The model consists of three main functional stages: (1) **State Embedding** explicitly decouples the input state into global temporal features (TFS and SMS, processed by GRUs) and local spatial features (TNode and TEdge, processed by MLPs). These heterogeneous features are then fused to form initial grade-aware node (h_i) and edge (e_{ij}) representations. (2) **Graph Embedding** utilizes a GatedGCN module to perform message passing on these representations, capturing complex topological dependencies and dynamic travel times across the road network. (3) **Node Prediction** decodes the refined node features using an MLP and a Softmax layer into a probability distribution ($p_1, \dots, p_{m+n}$) to guide the final truck destination selection (a_t).

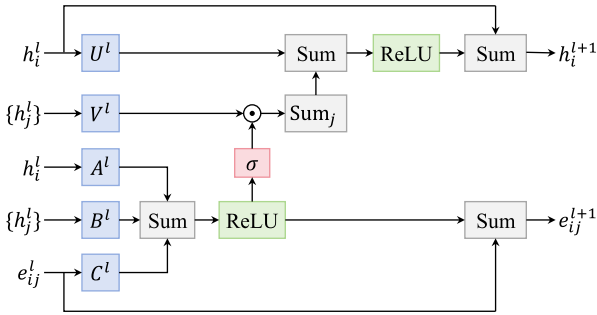


Fig. 5. GatedGCN architecture.

and AGGR refers to the neighborhood aggregation function for nodes (which can be SUM, MEAN, or MAX function); σ represents the sigmoid activation function, and $\odot$ indicates the Hadamard matrix product; h_i^0 and e_{ij}^0 are the initial inputs to the GatedGCN, corresponding to the high-dimensional features h_i and e_{ij} derived from state embedding.

4.2.3. Action decoder

Based on the final node features h_i^L learned through the GatedGCN (which have now fused temporal dynamics and spatial topology), we use an MLP to compute the initial logits z_m for each node $m \in M$. Considering the operational constraints in dispatching, for example, when the selected truck k is at a loading point, its feasible action is to travel to an unloading point. If the shovel is mining ore, the truck must be directed to the ore crushing station; otherwise, it should be directed to the waste rock crushing station. When the truck is at an unloading point, the feasible action is to return to a loading point that has ore. Consequently, the agent's action space invariably contains certain invalid actions.

To mitigate unnecessary complexity in DRL and ensure strict adherence to operational constraints, we employ a masking mechanism applied *before* the Softmax function. Specifically, we define a dynamic mask vector *mask*, where $mask_m = 0$ if node m is a valid destination for the current truck, and $mask_m = -\infty$ if it is invalid. This mask is added to the initial logits, and the final selection probability p_m is computed as $p_m = \text{Softmax}(z_m + mask_m)$. This mathematical operation ensures that invalid actions are assigned exactly zero probability, guaranteeing they are never selected during either the training or inference phases.

4.3. Training method

To train the STGPN π_θ within the MDP framework, we must overcome the two major challenges posed by long-horizon decision dependency: namely, the “credit assignment” problem (caused by the sparse

global reward) and the “high memory consumption” problem. To this end, this subsection presents a DRL training method based on policy gradient with a baseline (Kool et al., 2019), combined with an efficient two-stage training strategy (local short-horizon pre-training and global long-horizon fine-tuning) to directly mitigate the training instability associated with sparse rewards. Algorithm 1 provides an overview of the basic training procedure. The pre-training and fine-tuning processes are analogous, with the sole difference being the configuration of input parameters.

4.3.1. Local short-horizon pre-training

In the pre-training phase, the local time window W is set to a smaller range (e.g., 150 min). As shown in lines 5–6, we randomly sample a short window within the full shift duration, enabling the model to first train on a simplified, random, short-horizon problem set. The core idea is a divide-and-conquer approach, breaking down the difficult long-horizon problem into more easily solvable short-horizon subproblems. This reduces memory usage, speeds up training, and preserves effective generalization for the global problem.

4.3.2. Global long-horizon fine-tuning

Building upon the pre-trained model, we set the time window to the global shift duration. We employ a Low-Rank Adaptation (LoRA) (Hu et al., 2022) method to fine-tune a small number of parameters on the global long-horizon problem set, further enhancing model accuracy with low memory consumption. This divide-and-conquer strategy not only transfers local knowledge to the global, large-scale, long-horizon task but also overcomes the high memory consumption barrier of long-horizon simulation training, making on-site deployment feasible.

4.3.3. REINFORCE with moving average baseline

To train the policy network, we employ the REINFORCE algorithm (Williams, 1992). To mitigate the high variance inherent in the standard REINFORCE algorithm, we introduce a baseline R_b^{BL} to calculate the advantage ($R_b - R_b^{\text{BL}}$). To minimize the estimation cost, this paper uses a moving average of historical policy network rewards to update the baseline (see line 21 of Algorithm 1), rather than training a separate value network. This approach has low computational complexity and provides a stable baseline reference.

In each iteration of Algorithm 1, the local time window for training is randomly assigned in lines 5–6. Lines 7–17 sample solutions for the problem instances. Lines 9–10 assign dispatch plans based on a random distribution $u(a_{b,t}|s_{b,t})$ for periods prior to the time window, while lines 11–12 use the policy network π_θ to assign plans within the time window. Line 18 computes the reward value based on Eq. (1). Since the objective is minimization, the reward is defined as the negative of the RMSE of the ore flow grade, i.e., $R_b = -f(a_{b,1:T})$, where T is the total number of time steps, and $a_{b,1:T}$ denotes the action sequence taken within the local time window. During the pre-training phase, only the

Algorithm 1: DRL with Moving Average Baseline

Data: Local time window size W , shift duration S , max iterations I .

Result: Optimal policy network π_θ^* .

```

1 Initialize baseline reward  $R^{\text{BL}} = 0$ 
2 for  $iter = 1$  to  $I$  do
3   for  $i = 1$  to  $M$  do
4     Batch sampling  $b = N_i$ 
5      $W_s = \text{rand}(0, S - W)$ 
6      $W_e = W_s + W$ 
7     for  $t = 1$  to  $T$  do
8       Calculate environmental state  $s_{b,t}$ 
9       if  $T_t < W_s$  then
10         $a_{b,t} \sim u(a_{b,t} | s_{b,t})$ 
11      else if  $T_t < W_e$  then
12         $a_{b,t} \sim \pi_\theta(a_{b,t} | s_{b,t})$ 
13      else
14        break
15      end
16      Update environmental state  $s_{b,t+1}$ 
17    end
18     $R_b = -f(a_{b,1:T})$  base on Eq. (1)
19    Calculate gradient  $d_\theta$  based on Eq. (19)
20    Update parameters  $\theta \leftarrow \text{Adam}(\theta, d_\theta)$ 
21     $R_b^{\text{BL}} = (R_b^{\text{BL}} + R_b) / 2$ 
22  end
23 end

```

ore flow within the local window is used to evaluate the reward. Lines 19–20 use the Adam optimizer to update the policy network parameters based on the standard gradient update formula for REINFORCE with a baseline, as shown in Eq. (19). Line 21 employs the moving average method to update the baseline reward.

$$\frac{1}{|N_i|} \sum_{b=1}^{|N_i|} \sum_{t=1}^T (R_b - R_b^{\text{BL}}) \nabla_\theta \log \pi_\theta(a_{b,t} | s_{b,t}). \quad (19)$$

4.4. Sampling search strategy

A solution method based on DRL is employed for the truck scheduling problem to train the policy network of the truck agent. Given the inherent randomness of this policy, a probability distribution over actions is generated based on the current environment. During inference, multiple candidate solutions are sampled for each problem instance from the policy network, and the optimal solution is selected (Bello et al., 2016). This approach simulates the solver's search process in the solution space, facilitating the exploration of more potential optimal solutions and enhancing solution quality. Compared to single greedy selection, this method offers greater flexibility, aiding the agent in discovering better solutions for complex combinatorial optimization problems.

5. Experimental environment

To systematically validate the effectiveness, robustness, and scalability of our proposed ore-flow blending model and STGPN solution, we constructed a high-fidelity simulation environment and designed a comprehensive set of competing benchmarks and ablation studies.

5.1. Simulation environment

To ensure the validity of our experiments and the practical significance of our conclusions, the simulation environment and datasets

we constructed are highly based on real-world mine operational data. Detailed information about the mine datasets is provided in Appendix A.

5.1.1. High-fidelity simulation parameters

Our simulation environment strives to reproduce the key randomness and constraints found in actual production.

Shovel loading time. We collected actual operational videos of shovels from a real-world open-pit mine. Through statistical analysis, we found that the loading time approximately follows a normal distribution (see Appendix A.C).

Truck travel time. We did not use a simplified constant-velocity model. Instead, we adopted a prediction method based on deep learning (Ao et al., 2023). This model takes truck capacity, average speed, road segment length, grade, weather, etc., as inputs to predict the heavy-haul and empty-haul travel times for different trucks on various road segments (see Appendix A.D).

Dynamic disruption events. To test the model's robustness, the simulation environment includes five dynamic and uncertain scenarios (E1–E5): E1 (no random events), E2 (truck breakdown), E3 (shovel breakdown), E4 (weather changes causing global travel time alterations), and E5 (road blockage causing local travel time alterations).

Ore grade sequences. To enhance data diversity, the shovel's loading point location, working area, and excavation path are randomly generated within the mine's blast pile areas for each problem instance. This provides different initial ore grade sequences (TFS, SMS states) for the loading points in each instance.

5.1.2. Benchmark instance sets

We selected three real-world mine road network scenarios of increasing complexity (C1, C2, C3). C1 is the simplest, while C3 has the most complex road network topology, number of blast piles, and number of unloading points. For each mine case, we set up four equipment configuration scales (D1–D4, e.g., C3–D4 corresponds to 9 shovels and 27 trucks) based on the optimal truck-to-shovel ratio calculated in Appendix A.E (2:1 for C1/C2, 3:1 for C3). Combining the three cases, four scales, and five dynamic events, we generated a total of 24,000 problem instances. The dataset was divided into training, validation, and test sets at an 8:1:1 ratio.

5.2. Baseline methods and ablation studies

To comprehensively evaluate our method's performance, we designed two sets of experiments. The first group (Competing Baselines) compares our STGPN against existing representative algorithms. The second group (Ablation Study) compares STGPN against its internal variants to validate the necessity of our architectural design. We must point out that, to the best of our knowledge, there are no publicly available DRL benchmarks specifically for our newly proposed ore-flow blending problem. Therefore, we evaluate the capabilities of these architectures in solving this new problem through fair comparisons (e.g., using our identical spatio-temporal state representation).

5.2.1. Competing baseline methods

First, to validate the effectiveness of the ore-flow blending model, a comparison was made with traditional overall blending methods (OB) (Chen et al., 2022). Recent research on truck scheduling has predominantly employed MLPs that handle only simple local environmental states. To demonstrate the advancement of STGPN method, the state-of-the-art multi-head attention (MHA) models (Kool et al., 2019; Wu et al., 2022) in the field of combinatorial optimization and the MLP model (Noriega et al., 2025) in the field of truck scheduling are used as comparative methods. To better adapt to the ore-flow blending problem, we replace the GatedGCN module in our proposed method

with MHA and MLP. Also, dynamic optimization methods based on swarm intelligence serve as non-learning evolutionary computation methods for comparison: EA (Eiben and Smith, 2015), ACO (Bonabeau et al., 2000), MMAS-LS (Mavrovouniotis et al., 2017b), MACO (Shi et al., 2022), ACFGA (Yazdani et al., 2022), DPCGA (F. Li et al., 2024). To ensure a fair comparison, all DRL-based methods (STGPN, MHA, MLP) are trained using our proposed two-stage training strategy and evaluated using the same sampling search strategy. Hyperparameter configuration of baseline methods is provided in Appendix A.

Furthermore, to support the policy behavior analysis in Section 6.8, we introduce a Heuristic dispatching strategy as an additional behavioral baseline. This heuristic operates strictly on a locally greedy logic aimed at minimizing the immediate absolute deviation from the target grade. Specifically, at each decision epoch, if a loaded truck needs to select an unloading point, it chooses the crusher where its arrival minimizes the immediate grade deviation of that crusher's current time window. Conversely, if an empty truck needs to select a shovel, it predicts the ore grade it would load at each candidate shovel, and selects the one that minimizes the expected average grade deviation across all ore crushers. While providing a fast, rule-based response, this heuristic remains inherently short-sighted as it completely ignores the long-term cumulative effects of the ore flow, serving as a comparative benchmark to demonstrate the necessity of the forward-looking optimization capabilities exhibited by the learned DRL policies.

5.2.2. Ablation study components

One of our core innovations is the design of the spatio-temporal state representation, which includes global temporal features (TFS, SMS) and local spatial features (LSF). To validate the necessity of these new features within the STGPN architecture, we conducted an ablation study. We compare our full model against several variants where key components have been removed:

- STGPN: Our full model, i.e., GRU + GatedGCN.
- STGPN-noTFS: We remove the GRU for the TFS feature. The original spatio-temporal state retains only SMS and LSF.
- STGPN-noSMS: We remove the GRU for the SMS feature. The original spatio-temporal state retains only TFS and LSF.
- STGPN-noLSF: We remove the GatedGCN for the LSF. The temporal features from the GRU are simply flattened, concatenated, and fed directly into an MLP for the final decision.

5.3. Evaluation metrics

Performance evaluation is carried out on the test set for each problem scale. Given the small RMSE value of the ore flow grade and the inconsistent range of values between the different problem instances, the relative gap value is used to normalize the evaluation metrics to the [0, 1] range as much as possible. The average optimal solution gap is computed using Eq. (20) to evaluate the performance of the methods.

$$gap = \frac{1}{N} \sum_{i=1}^N \frac{R_i^{\text{best}} - R_i}{R_i^{\text{best}}} \times 100\%, \quad (20)$$

Given the time-consuming nature of obtaining optimal solutions for practical optimization problems, the gap is computed by comparing the reward value R_i obtained by each method for the i th test problem instance with the reward value R_i^{best} , representing the best reward found across all methods. The reward function is defined as the negative value of the objective function, as formally expressed in Eq. (1). All experiments are implemented in C++, with learning-based methods running on NVIDIA 24G RTX 4090 GPUs, and evolutionary algorithms executed on an AMD 504G 96-Core EPYC 9654 CPU.

6. Results and discussion

6.1. Effectiveness of the Ore-Flow blending model

Before proceeding with algorithmic comparisons, we need first validate the effectiveness of our problem definition itself. Specifically, we need determine if models targeting the ore-flow RMSE (such as STGPN, MHA, and MLP) are fundamentally superior to the traditional OB model, which targets the overall average grade. As shown in Fig. 6, we compared the performance of the OB model and the three DRL architectures on two metrics: "Overall Average Error" (AvgErr) and "Ore-Flow Fluctuation" (FlowErr, i.e., RMSE). It is evident that the OB model exhibits a significant discrepancy between its AvgErr and FlowErr metrics. This result strongly substantiates our core argument from the Introduction and Related Work: a policy that only optimizes the overall average cannot guarantee process stability. Furthermore, the significant gap in FlowErr between the OB model and the DRL-based models validates the necessity and superiority of the novel ore-flow blending model (targeting RMSE) proposed in Section 3.

6.2. Ablation study on spatio-temporal state representation

We defined a set of ablation experiments in Section 5.2.2 to verify the necessity of the spatio-temporal state representation components (TFS, SMS, and LSF) within our STGPN architecture. The experimental results are shown in Fig. 7. In all three mine cases (C1, C2, and C3), the performance of STGPN-noLSF (which removes the graph embedding module and uses only GRU+MLP) decreases very significantly. This strongly proves our core argument: spatial topology information from the mine road network is critical, and flat MLP-like architectures are ineffective as they cannot capture this spatial dependency. Concurrently, removing temporal features also leads to performance degradation. In all three mine cases, both STGPN-noTFS (removing truck flow history) and STGPN-noSMS (removing shovel mining sequence) performed worse than the complete STGPN model. Compared to the historical grade information from TFS, the ability to "predict the future" (i.e., the grade supply information in SMS) is of even greater importance for achieving grade balancing.

6.3. Superiority of the STGPN architecture

We further compare the performance of different DRL architectures (STGPN, MHA, MLP) in processing this spatio-temporal state. As shown in Fig. 6, STGPN achieves the lowest RMSE (i.e., best performance) in 10 out of the 12 test cases, and the performance gap with MHA in the remaining 2 test cases was minimal. The MLP architecture performed the worst. Although we provided the MLP with the same spatio-temporal state input as the STGPN (via flattening), its RMSE was consistently and significantly higher than STGPN and MHA. This again corroborates the conclusion: the flat architecture of an MLP cannot understand the graph topology priors within the LSF, leading to poor performance. MHA's performance was superior to MLP's, but still slightly inferior to STGPN's in most cases. This indicates that while self-attention can capture relationships between nodes, the explicit graph convolution of GatedGCN (which utilizes the adjacency matrix A) is more efficient and stable when processing mine road networks with fixed topologies. The training curves in Fig. 8 further confirm the superiority of STGPN. In all three mine cases, the STGPN curve converged to a lower RMSE level than both MHA and MLP during the pre-training phase (first 200 epochs) and the fine-tuning phase (subsequent 200 epochs), demonstrating its stronger learning capability and final performance.

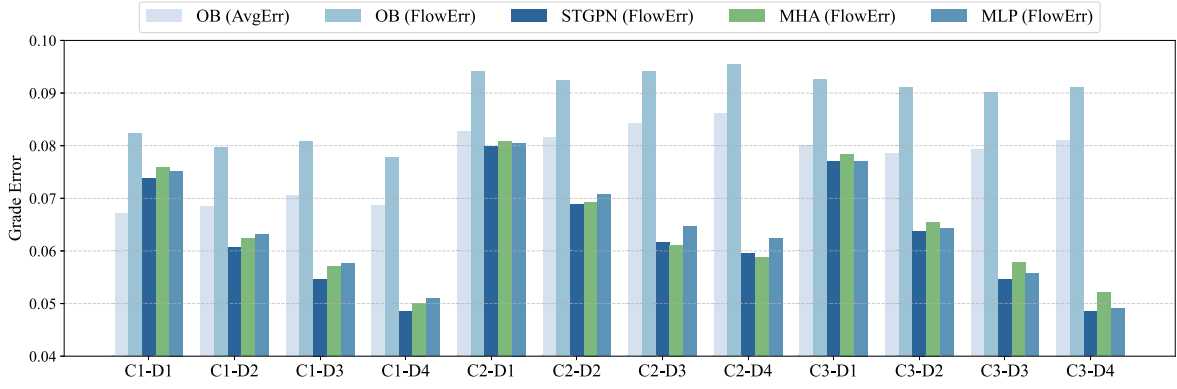


Fig. 6. A comparison of average grade error across different mine cases for different methods, where AvgErr denotes the overall ore grade error, and FlowErr represents the RMSE of ore flow grades.

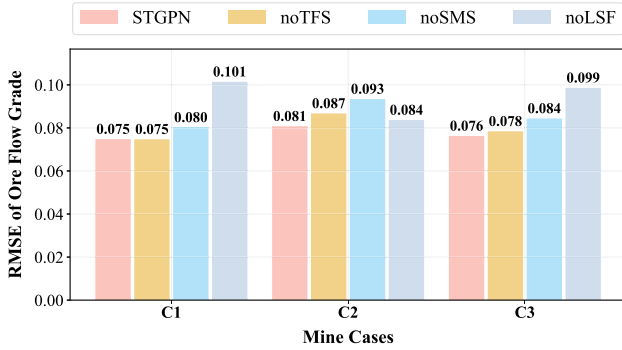


Fig. 7. Ablation study on spatio-temporal state representation.

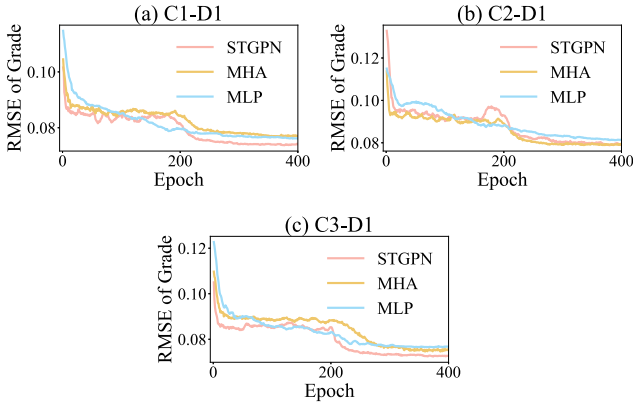


Fig. 8. Validation performance during training.

6.4. Two-stage training and sampling strategy

This section validates the effectiveness of the training and inference strategies we designed to overcome the “sparse reward” and “long-horizon” challenges. In Fig. 9, we compare the three modes of STGPN with evolutionary and swarm intelligence algorithms (EA, ACO). For the small-scale D1 dataset, the model is pre-trained on the local time window, while for the larger-scale D2–D4 datasets, the fine-tuned model from D1 is directly transferred. For the D1 dataset, after the STGPN underwent the pre-training and fine-tuning stages, its performance already surpassed that of the evolutionary computation algorithms. On the larger-scale D2–D4 problems, the performance of the transferred pre-trained model alone significantly exceeded the evolutionary computation algorithms. Building on the pre-trained

model, the performance is progressively improved through the fine-tuning and sampling search components, ultimately achieving optimal performance. The sampling phase, with only 100 sampling searches, drastically improves the model’s solution quality, far surpassing all baseline methods by mitigating the randomness of a single greedy decision. Therefore, our two-stage training strategy is both effective and necessary. Pre-training solves the cold-start and primary credit assignment problems, while fine-tuning and sampling search elevate the policy to a global optimum.

6.5. Sensitivity analysis of pre-training window size

To further explore the two-stage training strategy, we analyzed the impact of the local window size (W) during the pre-training phase on performance and memory. As shown in Fig. 10.

6.5.1. Performance impact (Fig. 10a–c)

When the window is too small (e.g., 0–30 min), the pre-training effect is poor, and the final sampling performance also fails to reach its optimum. However, once the window size exceeds a certain threshold (e.g., 60 min), the performance improvement from pre-training begins to slow down, and the final sampling performance becomes almost static. This indicates that as long as the local window is sufficiently large (e.g., 150 min), the model can learn robust local policies, and the fine-tuning and sampling stages can effectively compensate for minor performance differences from the pre-training stage.

6.5.2. Memory impact (Fig. 10d)

GPU memory consumption is almost linearly correlated with the pre-training window size. The smaller the window, the lower the memory consumption and the faster the inference speed during the pre-training phase.

There is a cost-performance trade-off for the pre-training window size. We selected $W = 150$ min because it provides robust and efficient pre-training results while maintaining controllable memory consumption.

6.6. Comprehensive comparative experiments

Table 3 presents the comprehensive performance of all methods on the test sets across all mine cases (C1–C3), different equipment scales (D1–D4), and various dynamic random events (E1–E5).

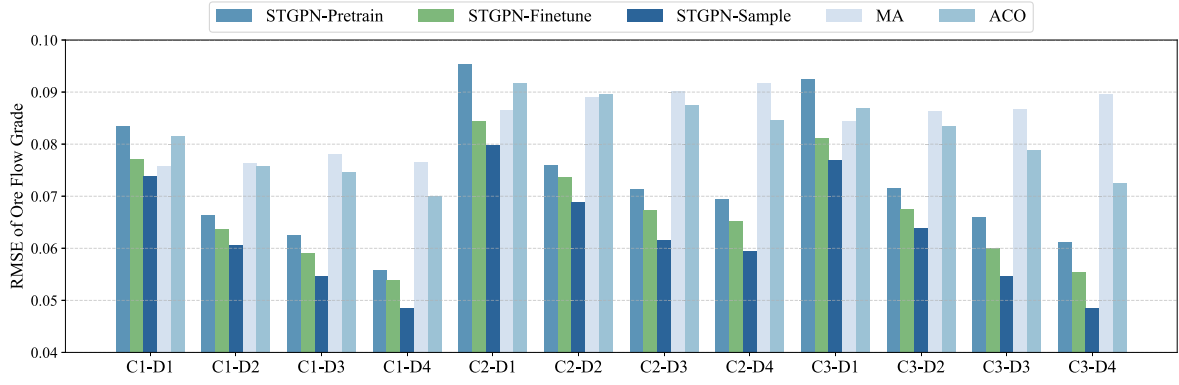


Fig. 9. A comparison of average RMSE of ore flow grade across test sets for different methods.

Table 2

Comparison of inference time (s) for different algorithms.

Case	Scale	STGPN	MHA	MLP	EA	ACO	MMAS-LS	MACO	ACFGA	DPCGA
C1	D1	0.16	0.21	0.09	34.02	28.49	29.48	32.99	65.38	47.00
	D2	0.17	0.22	0.11	142.43	72.44	94.66	98.63	174.13	183.66
	D3	0.18	0.23	0.14	261.75	132.15	153.06	140.68	324.65	344.67
	D4	0.20	0.27	0.16	409.64	212.15	175.33	179.51	462.35	482.81
C2	D1	0.15	0.19	0.11	49.83	35.17	33.96	31.04	64.28	63.78
	D2	0.17	0.21	0.19	142.03	89.27	107.09	99.60	169.48	169.71
	D3	0.20	0.22	0.26	313.79	160.79	151.47	167.10	345.99	346.95
	D4	0.20	0.23	0.34	538.12	171.87	168.80	197.25	596.00	613.15
C3	D1	0.16	0.20	0.13	72.34	52.25	57.65	57.82	98.34	78.34
	D2	0.18	0.24	0.23	319.43	129.48	146.88	120.59	360.68	330.72
	D3	0.21	0.28	0.31	637.99	309.35	221.32	294.79	757.05	750.48
	D4	0.23	0.32	0.41	1097.20	443.34	384.76	386.79	1346.31	1449.92

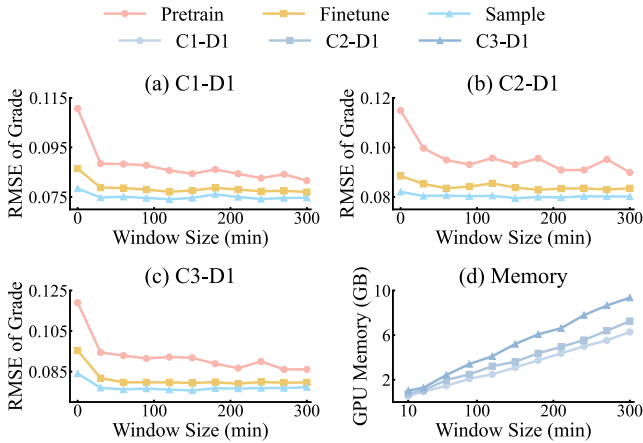


Fig. 10. An exploration of the size of the local time window.

6.6.1. Inference latency

To quantitatively validate the “second-level” real-time response capability emphasized in the Introduction, we evaluated the inference latency of all methods across different mine cases and equipment scales. The average inference times (in seconds) are summarized in Table 2.

The results clearly demonstrate that the DRL-based methods (STGPN, MHA, MLP) exhibit lightning-fast decision-making capabilities, consistently completing inference in second across all scales. In stark contrast, iterative swarm intelligence algorithms require tens to thousands of seconds to search for a solution. This quantitative analysis rigorously substantiates that STGPN completely overcomes the “re-optimization” bottleneck of traditional methods, possessing the extreme decision timeliness required for real-time online deployment in high-frequency, dynamic event-driven mining operations.

6.6.2. Scalability (D1–D4)

As the problem scale (number of equipment) increases, the performance (Gap/%) of evolutionary algorithms drops sharply. Experimental results show that these algorithms perform best only on the smallest-scale problems (D1), such as the ACO and MMAS-LS algorithms. These methods struggle to effectively cover the large-scale solution space, easily fall into local optima, and suffer a sharp decline in performance. In contrast, the performance of all DRL methods (STGPN, MHA, MLP) remains stable, with only a minor increase in inference time. This demonstrates the unparalleled scalability advantage of the DRL framework over evolutionary algorithms for handling large-scale problems. Through efficient representation learning and end-to-end optimization, the policy network compresses high-dimensional spatio-temporal states into low-dimensional representations, avoiding an explicit traversal of all combinations.

6.6.3. Robustness (E1–E5)

When facing dynamic disturbance events (E2–E5), STGPN consistently maintains the lowest Gap/% and outperforms MHA and MLP in the vast majority of scenarios. This proves the superiority of the STGPN architecture; the combination of GatedGCN and GRU enables it to understand dynamic changes in real-time (such as truck breakdowns or road blockages) and make optimal adjustments quickly.

6.6.4. Overall performance

Finally, it can be observed that all network architectures (STGPN, MHA, and MLP) achieve performance far exceeding evolutionary computation algorithms in almost all test cases. This indicates that the high-dimensional spatio-temporal environmental state representation designed in this paper can comprehensively extract and capture global ore-flow characteristics and reflect the deep impacts of varying equipment scales, network structures, and dynamic random events.

Table 3

Detailed performance comparison on 60 test scenarios. The Obj column displays the mean objective value alongside the Wins/Ties/Losses ($W/T/L$) computed via pairwise Wilcoxon signed-rank tests ($\alpha = 0.05$) against the other 8 algorithms based on the 40 instances. The Gap/% column presents the Mean $\pm$ Std of the relative gap. For each scenario, the algorithm achieving the maximum Wins has its $W/T/L$ highlighted in **bold**.

Case	Scale	Event	STGPN		MHA		MLP		EA		ACO		MMAS-LS		MACO		ACFGA		DPCGA	
			Obj (W/T/L)	Gap/%	Obj (W/T/L)	Gap/%	Obj (W/T/L)	Gap/%	Obj (W/T/L)	Gap/%	Obj (W/T/L)	Gap/%	Obj (W/T/L)	Gap/%	Obj (W/T/L)	Gap/%	Obj (W/T/L)	Gap/%	Obj (W/T/L)	Gap/%
C1	D1	E1	0.074 (4/4/0)	3.7 ± 3.9	0.076 (3/5/0)	6.2 ± 5.5	0.075 (3/5/0)	4.5 ± 4.4	0.080 (0/2/6)	12.7 ± 9.3	0.075 (4/4/0)	4.5 ± 4.5	0.074 (4/4/0)	3.4 ± 4.0	0.076 (3/2/3)	6.7 ± 4.8	0.082 (0/2/6)	16.0 ± 12.6	0.080 (0/2/6)	13.1 ± 9.3
		E2	0.067 (5/3/0)	5.4 ± 6.5	0.072 (0/4/4)	13.3 ± 12.0	0.070 (1/4/3)	11.6 ± 13.2	0.070 (1/4/3)	13.3 ± 10.5	0.065 (6/2/0)	4.7 ± 5.4	0.065 (6/2/0)	4.5 ± 4.5	0.067 (3/3/2)	9.1 ± 6.5	0.075 (0/1/7)	18.6 ± 14.3	0.071 (1/3/4)	16.1 ± 13.3
		E3	0.080 (0/5/3)	22.9 ± 24.4	0.081 (0/4/4)	25.0 ± 25.2	0.079 (1/4/3)	21.6 ± 23.2	0.077 (0/5/3)	15.6 ± 12.9	0.070 (7/1/0)	4.1 ± 4.8	0.069 (7/1/0)	1.2 ± 1.9	0.071 (6/0/2)	6.4 ± 4.8	0.080 (0/5/3)	21.7 ± 18.3	0.077 (0/5/3)	18.0 ± 17.2
		E4	0.071 (1/5/2)	6.8 ± 8.4	0.072 (0/5/3)	9.8 ± 9.0	0.073 (0/4/4)	14.6 ± 18.9	0.072 (0/5/3)	13.1 ± 8.4	0.065 (7/1/0)	3.2 ± 3.5	0.065 (7/1/0)	3.4 ± 3.3	0.067 (5/1/2)	6.6 ± 4.9	0.073 (0/4/4)	21.1 ± 20.0	0.071 (1/4/3)	14.7 ± 12.3
		E5	0.076 (4/4/0)	3.7 ± 3.5	0.078 (3/2/3)	7.6 ± 8.6	0.079 (2/4/2)	6.4 ± 6.1	0.082 (1/1/6)	14.1 ± 12.7	0.074 (6/2/0)	3.4 ± 3.4	0.074 (6/2/0)	2.9 ± 3.6	0.076 (3/3/2)	6.2 ± 4.6	0.083 (0/0/8)	20.0 ± 18.4	0.082 (1/2/5)	18.2 ± 17.3
	D2	E1	0.057 (6/2/0)	4.2 ± 7.5	0.059 (3/4/1)	9.3 ± 8.7	0.059 (3/4/1)	9.0 ± 11.9	0.077 (0/2/6)	45.4 ± 25.1	0.058 (4/4/0)	7.2 ± 6.5	0.058 (4/4/0)	8.1 ± 6.3	0.060 (3/2/3)	13.4 ± 9.8	0.075 (0/2/6)	44.1 ± 26.1	0.076 (0/2/6)	45.9 ± 26.3
		E2	0.063 (5/3/0)	2.4 ± 3.8	0.064 (4/4/0)	4.1 ± 5.1	0.065 (4/3/1)	4.4 ± 5.0	0.079 (0/2/6)	37.0 ± 22.5	0.064 (4/4/0)	7.9 ± 6.4	0.064 (4/4/0)	8.5 ± 7.0	0.067 (3/0/5)	14.2 ± 9.6	0.080 (0/2/6)	32.5 ± 22.8	0.078 (0/2/6)	34.3 ± 23.4
		E3	0.059 (6/2/0)	4.8 ± 6.5	0.062 (3/4/1)	7.0 ± 6.1	0.062 (3/4/1)	7.6 ± 8.8	0.076 (0/2/6)	40.2 ± 25.3	0.059 (4/4/0)	5.6 ± 4.8	0.060 (4/4/0)	5.0 ± 4.7	0.061 (3/2/3)	9.5 ± 5.2	0.076 (0/2/6)	41.7 ± 26.9	0.076 (0/2/6)	39.9 ± 23.4
		E4	0.060 (6/2/0)	7.1 ± 10.9	0.062 (3/4/1)	9.6 ± 10.5	0.063 (3/4/1)	8.3 ± 11.4	0.079 (0/2/6)	46.2 ± 30.2	0.060 (4/4/0)	6.2 ± 5.9	0.061 (4/4/0)	7.4 ± 6.3	0.063 (3/2/3)	14.1 ± 6.1	0.079 (0/1/7)	43.6 ± 25.3	0.078 (1/1/6)	42.9 ± 26.6
		E5	0.063 (7/1/0)	2.5 ± 3.6	0.066 (4/3/1)	7.5 ± 7.4	0.066 (3/4/1)	8.6 ± 10.1	0.081 (0/1/7)	38.7 ± 26.3	0.065 (4/4/0)	7.1 ± 5.8	0.065 (4/3/1)	7.0 ± 4.9	0.069 (3/1/4)	13.9 ± 7.9	0.082 (0/1/7)	33.9 ± 20.3	0.079 (2/0/6)	31.1 ± 21.1
	D3	E1	0.053 (8/0/0)	1.2 ± 2.7	0.055 (4/3/1)	9.9 ± 10.2	0.057 (4/3/1)	15.0 ± 16.7	0.078 (0/1/7)	63.3 ± 35.6	0.057 (4/3/1)	10.6 ± 7.3	0.058 (4/3/1)	11.3 ± 9.5	0.061 (3/0/5)	20.9 ± 11.5	0.078 (0/1/7)	64.8 ± 36.1	0.076 (2/0/6)	61.6 ± 36.2
		E2	0.055 (8/0/0)	1.8 ± 2.8	0.058 (5/2/1)	9.1 ± 9.1	0.057 (6/1/1)	5.4 ± 5.9	0.083 (0/0/8)	63.1 ± 31.3	0.059 (4/2/2)	10.4 ± 6.7	0.060 (4/1/3)	13.8 ± 8.9	0.064 (3/0/5)	21.8 ± 10.6	0.081 (1/1/6)	58.9 ± 30.0	0.081 (1/1/6)	54.5 ± 26.3
		E3	0.057 (8/0/0)	1.6 ± 3.0	0.061 (4/3/1)	7.9 ± 7.2	0.061 (4/3/1)	7.9 ± 7.4	0.086 (0/1/7)	62.3 ± 37.1	0.061 (4/3/1)	11.3 ± 7.3	0.061 (4/3/1)	10.7 ± 8.5	0.065 (3/0/5)	20.6 ± 10.9	0.086 (0/2/6)	70.5 ± 47.3	0.085 (1/1/6)	66.3 ± 47.2
		E4	0.051 (7/1/0)	6.4 ± 7.9	0.053 (4/4/0)	9.0 ± 8.2	0.053 (4/3/1)	9.9 ± 9.7	0.078 (0/1/7)	71.8 ± 36.8	0.054 (4/3/1)	9.1 ± 7.1	0.055 (4/3/1)	12.0 ± 11.5	0.058 (3/0/5)	18.8 ± 9.5	0.078 (0/1/7)	71.3 ± 36.1	0.076 (2/0/6)	64.5 ± 32.3
		E5	0.057 (7/1/0)	2.7 ± 4.0	0.059 (4/4/0)	7.0 ± 7.6	0.060 (4/3/1)	5.4 ± 5.2	0.083 (0/1/7)	60.6 ± 35.7	0.061 (4/3/1)	10.7 ± 7.1	0.061 (4/3/1)	9.9 ± 8.5	0.064 (3/0/5)	17.8 ± 9.7	0.083 (0/1/7)	59.9 ± 34.7	0.081 (2/0/6)	57.1 ± 35.0
	D4	E1	0.049 (7/1/0)	2.2 ± 3.2	0.049 (6/2/0)	4.6 ± 4.9	0.051 (6/1/1)	8.4 ± 9.1	0.080 (0/1/7)	82.6 ± 39.7	0.056 (4/1/3)	16.5 ± 12.8	0.057 (4/1/3)	16.4 ± 9.7	0.059 (3/0/5)	25.7 ± 11.6	0.080 (0/1/7)	81.1 ± 37.8	0.077 (2/0/6)	68.1 ± 29.5
		E2	0.048 (7/1/0)	5.4 ± 6.6	0.049 (6/2/0)	6.4 ± 6.0	0.050 (6/1/1)	7.9 ± 9.2	0.080 (1/0/7)	80.0 ± 36.3	0.053 (4/1/3)	14.2 ± 9.4	0.056 (4/1/3)	16.7 ± 11.0	0.058 (3/0/5)	27.0 ± 12.1	0.081 (0/0/8)	83.8 ± 40.6	0.078 (2/0/6)	77.5 ± 37.6
		E3	0.050 (8/0/0)	2.4 ± 2.9	0.052 (6/1/1)	9.7 ± 9.1	0.053 (4/3/1)	11.4 ± 11.5	0.080 (1/0/7)	72.0 ± 30.8	0.056 (4/2/2)	15.4 ± 11.8	0.057 (4/2/2)	15.8 ± 11.6	0.061 (3/0/5)	26.6 ± 16.3	0.082 (0/0/8)	75.1 ± 31.9	0.079 (2/0/6)	67.0 ± 27.8
		E4	0.044 (8/0/0)	2.5 ± 3.7	0.047 (4/3/1)	11.5 ± 12.8	0.048 (4/3/1)	9.1 ± 7.9	0.073 (1/1/6)	77.6 ± 35.7	0.050 (4/3/1)	16.7 ± 11.8	0.051 (4/3/1)	15.9 ± 13.1	0.054 (3/0/5)	26.7 ± 16.5	0.075 (0/0/8)	79.1 ± 33.3	0.072 (1/1/6)	74.5 ± 37.2
		E5	0.052 (7/1/0)	3.1 ± 3.6	0.053 (6/2/0)	5.0 ± 5.5	0.054 (6/1/1)	7.6 ± 7.5	0.084 (0/1/7)	77.0 ± 36.4	0.058 (4/1/3)	15.8 ± 12.3	0.059 (4/1/3)	16.2 ± 12.5	0.063 (3/0/5)	25.4 ± 12.9	0.085 (0/1/7)	79.9 ± 38.0	0.082 (2/0/6)	72.3 ± 35.2
D1	E1	0.081 (7/1/0)	3.8 ± 4.4	0.083 (3/4/1)	6.2 ± 6.3	0.082 (4/3/1)	5.1 ± 5.4	0.090 (0/2/6)	14.0 ± 11.4	0.084 (4/2/2)	6.9 ± 5.7	0.083 (5/3/0)	4.3 ± 4.4	0.085 (3/1/4)	8.8 ± 6.5	0.091 (0/2/6)	13.6 ± 12.3	0.090 (0/2/6)	14.9 ± 12.8	
	E2	0.083 (4/4/0)	3.2 ± 3.8	0.084 (4/4/0)	4.4 ± 4.6	0.084 (4/4/0)	2.9 ± 3.5	0.089 (2/1/5)	9.8 ± 8.8	0.084 (4/4/0)	3.8 ± 3.8	0.084 (4/4/0)	4.3 ± 3.9	0.087 (2/1/5)	6.8 ± 5.4	0.094 (0/1/7)	16.4 ± 12.0	0.092 (0/1/7)	10.9 ± 9.2	
	E3	0.083 (4/4/0)	4.8 ± 5.2	0.085 (1/6/1)	8.1 ± 10.0	0.084 (3/5/0)	5.5 ± 7.5	0.088 (0/2/6)	16.6 ± 13.2	0.082 (4/4/0)	6.6 ± 5.8	0.081 (4/4/0)	5.0 ± 5.1	0.084 (3/3/2)	9.1 ± 5.8	0.088 (0/3/5)	14.3 ± 10.7	0.088 (0/3/5)	17.7 ± 15.9	
	E4	0.080 (3/5/0)	3.6 ± 4.0	0.081 (3/5/0)	3.6 ± 3.6	0.080 (3/5/0)	3.7 ± 3.0	0.087 (0/2/6)	10.5 ± 9.3	0.079 (4/4/0)	3.0 ± 2.1	0.079 (4/4/0)	3.4 ± 4.2	0.081 (2/4/2)	6.4 ± 5.1	0.088 (0/2/6)	14.0 ± 12.5	0.087 (0/3/5)	10.1 ± 9.8	
	E5	0.072 (3/5/0)	7.4 ± 8.0	0.072 (3/5/0)	7.0 ± 7.6	0.072 (3/5/0)	7.7 ± 8.2	0.078 (1/1/6)	14.7 ± 11.6	0.072 (3/4/1)	5.9 ± 4.2	0.071 (5/3/0)	4.3 ± 4.8	0.073 (3/4/1)	7.3 ± 5.7	0.083 (0/0/8)	21.8 ± 17.6	0.078 (1/1/6)	14.7 ± 13.5	
D2	E1	0.073 (7/1/0)	1.5 ± 2.0	0.073 (7/1/0)	2.2 ± 3.2	0.075 (6/0/2)	3.2 ± 3.8	0.097 (0/1/7)	39.9 ± 31.0	0.078 (4/1/3)	9.3 ± 8.6	0.079 (4/1/3)	10.4 ± 8.3	0.082 (3/0/5)	15.6 ± 10.6	0.097 (0/2/6)	40.7 ± 28.9	0.095 (1/1/6)	35.5 ± 26.6	
	E2	0.067 (7/1/0)	2.8 ± 3.9	0.068 (4/4/0)	4.9 ± 5.7	0.069 (4/3/1)	7.7 ± 7.1	0.087 (0/2/6)	39.2 ± 31.1	0.070 (4/3/1)	8.0 ± 7.2	0.070 (4/3/1)	7.4 ± 6.6	0.074 (3/0/5)	12.7 ± 7.5	0.088 (0/2/6)	36.9 ± 26.5	0.087 (0/2/6)	37.8 ± 28.7	
	E3	0.069 (5/3/0)	3.7 ± 6.0	0.069 (6/2/0)	4.2 ± 5.5	0.069 (6/2/0)	4.0 ± 4.6	0.088 (0/2/6)	40.0 ± 24.0	0.072 (4/1/3)	12.0 ± 8.5	0.071 (4/2/2)	11.1 ± 9.5	0.075 (3/0/5)	18.5 ± 10.5	0.088 (0/2/6)	36.5 ± 20.4	0.087 (0/2/6)	41.2 ± 25.7	
	E4	0.069 (7/1/0)	2.5 ± 3.2	0.071 (6/2/0)	3.7 ± 4.3	0.070 (6/1/1)	3.9 ± 3.9	0.092 (1/1/6)	40.0 ± 25.1	0.074 (4/1/3)	8.1 ± 4.6	0.074 (4/1/3)	7.9 ± 6.2	0.077 (3/0/5)	14.2 ± 9.4	0.094 (0/0/8)	42.9 ± 31.5	0.092 (1/1/6)	43.5 ± 29.3	
	E5	0.065 (7/1/0)	2.6 ± 3.5	0.066 (7/1/0)	2.5 ± 2.8	0.070 (3/3/2)	8.8 ± 9.2	0.088 (1/0/7)	36.9 ± 16.3	0.070 (4/1/3)	7.3 ± 5.5	0.068 (5/1/2)	6.9 ± 5.9	0.073 (3/1/4)	14.7 ± 8.4	0.090 (0/0/8)	39.7 ± 19.2	0.087 (2/0/6)	36.1 ± 16.8	
D3	E1	0.060 (7/0/1)	3.2 ± 3.2	0.059 (8/0/0)	0.7 ± 1.2	0.063 (6/0/2)	7.7 ± 7.9	0.093 (0/1/7)	60.9 ± 26.2	0.067 (4/1/3)	15.0 ± 10.1	0.068 (4/1/3)	13.2 ± 7.7	0.071 (3/0/5)	24.0 ± 11.2	0.094 (0/1/7)	65.7 ± 29.8	0.091 (2/0/6)	58.4 ± 26.6	
	E2	0.056 (7/1/0)	3.1 ± 3.9	0.056 (7/1/0)	0.8 ± 1.5	0.060 (6/0/2)	10.5 ± 11.5	0.089 (0/2/6)	63.2 ± 33.2	0.065 (4/1/3)	19.3 ± 12.3	0.066 (4/1/3)	19.8 ± 13.5	0.069 (3/0/5)	26.9 ± 12.5	0.089 (0/2/6)	64.3 ± 34.4	0.088 (0/2/6)	60.6 ± 30.8	
	E3	0.064 (8/0/0)	3.3 ± 5.4	0.065 (6/1/1)	2.5 ± 2.9	0.066 (6/1/1)	6.5 ± 7.8	0.096 (0/1/7)	63.7 ± 41.3	0.069 (4/1/3)	13.4 ± 8.1	0.070 (4/1/3)	12.9 ± 9.5	0.073 (3/0/5)	19.4 ± 9.6	0.095 (0/2/6)	65.4 ± 46.2	0.094 (1/1/6)	59.8 ± 38.8	
	E4	0.067 (7/1/0)	2.3 ± 2.7	0.066 (7/1/0)	0.8 ± 1.3	0.070 (6/0/2)	9.8 ± 10.1	0.095 (0/2/6)	60.3 ± 26.2	0.076 (4/1/3)	16.1 ± 8.4	0.076 (4/1/3)	18.5 ± 11.5	0.080 (3/0/5)	22.9 ± 10.4	0.095 (0/1/7)	52.4 ± 27.5	0.094 (1/1/6)	49.0 ± 26.4	
	E5	0.061 (7/1/0)	1.6 ± 2.0	0.060 (7/1/0)	1.6 ± 2.4	0.064 (6/0/2)	7.4 ± 9.7	0.093 (0/2/6)	57.2 ± 27.7	0.070 (4/1/3)	19.2 ± 12.1	0.071 (4/1/3)	19.5 ± 12.0	0.075 (3/0/5)	27.7 ± 10.9	0.092 (0/2/6)	63.3 ± 34.9	0.091 (0/2/6)	55.3 ± 25.9	
D4	E1	0.060 (7/1/0)	4.3 ± 5.5	0.060 (7/1/0)	3.6 ± 4.5	0.061 (6/0/2)	7.9 ± 7.1	0.094 (0/1/7)	70.2 ± 34.5	0.069 (5/0/3)	19.6 ± 12.0	0.071 (4/0/4)	22.4 ± 12.8	0.072 (3/0/5)	25.8 ± 13.7	0.094 (0/1/7)	72.5 ± 37.5	0.092 (2/0/6)	68.6 ± 37.4	
	E2	0.056 (7/1/0)	3.2 ± 4.2	0.056 (7/1/0)	1.1 ± 1.7	0.059 (6/0/2)	8.8 ± 9.7	0.096 (0/1/7)	80.7 ± 38.1	0.067 (4/1/3)	22.4 ± 14.3	0.067 (4/1/3)	20.7 ± 13.9	0.070 (3/0/5)	28.9 ± 14.8	0.095 (0/2/6)	82.0 ± 39.7	0.094 (1/1/6)	79.6 ± 40.8	
	E3	0.062 (7/1/0)	5.0 ± 6.6	0.061 (7/1/0)	2.0 ± 3.2	0.066 (6/0/2)	8.4 ± 7.1	0.097 (0/1/7)	66.8 ± 36.1	0.072 (4/1/3)	22.1 ± 2									

6.6.5. Statistical significance

Given the inherent stochastic nature of DRL policy sampling during inference and the diverse dynamic events in the mining environment, relying solely on a single overall average value is insufficient to fully demonstrate algorithmic superiority. To rigorously evaluate the statistical significance and robustness of the methods, we conducted a comprehensive statistical analysis based on 40 independent test instances for each scenario.

In Table 3, we present a rigorous dual-metric evaluation. First, the relative performance gap is reported as Mean $\pm$ Standard Deviation (Mean $\pm$ Std), capturing both the central tendency and the dispersion of each algorithm's performance when facing stochastic environments. Second, to rigorously verify whether the performance differences are statistically significant on the original objective values (RMSE), we conducted pairwise non-parametric Wilcoxon signed-rank tests ($\alpha = 0.05$) between each evaluated method and the other 8 baseline algorithms.

The test results are summarized in the format of $W/T/L$. For example, a record of 7/1/0 indicates that the algorithm is significantly better than 7 baselines, statistically similar to 1, and worse than 0. As shown in the overall summary of Table 3, STGPN achieves an overwhelming Win Rate of 75.4% (recording 362 Wins, 108 Ties, and only 10 Losses out of 480 total comparisons). It completely dominates all evolutionary algorithms (which suffer from near-zero win rates) and significantly outperforms the MHA and MLP architectures. This rigorously confirms that the superiority of the proposed STGPN framework is statistically highly significant and consistently stable under dynamic uncertainties.

6.7. Evaluation of cross-case generalizability

A critical challenge in applying DRL to real-world industrial environments is the performance degradation when the data used for decision-making significantly differs from the training data (i.e., Out-of-Distribution scenarios). To strictly evaluate the generalization capability of the proposed method under such conditions, we designed a cross-case transfer experiment.

In this setup, we force the training and testing distributions to be fundamentally different. The STGPN model is trained exclusively on a source mine case and is subsequently applied to an entirely unseen target mine case for zero-shot decision-making, without any fine-tuning. These distinct mine cases (C1, C2, C3) feature entirely different road network topologies, spatial distance matrices, grade distributions, and fleet configurations. We define two evaluation modes for our model: "Transfer" (trained on source, tested on target) and "No-Transfer" (trained and tested on the target case).

The experimental results for six cross-case transfer tasks across four scales (D1 to D4) are presented in Table 4, the STGPN demonstrates strong robustness when confronting significant data distribution shifts. The performance of the "Transfer" model exhibits highly graceful degradation, closely approximating the ideal performance of the locally trained "No-Transfer" model. Notably, even when operating in completely unseen environments, the transferred STGPN continues to outperform state-of-the-art meta-heuristic algorithms (e.g., ACO, MMAS-LS) in the vast majority of scenarios, particularly at larger operational scales (D2, D3, D4).

6.8. Policy network behavior analysis

The experiments conducted above validate the effectiveness of the proposed method. This study aims to explore the decision-making behaviors of the policy network, with a focus on the general patterns learned. Specifically, we compare the strategies learned by STGPN, MHA, and MLP against a heuristic strategy, which minimizes grade deviation within the current local time window. This local greedy strategy is inherently short-sighted and challenging to realize the global optimum. Fig. 11 illustrates the entropy loss curves during training. The MLP's simplistic architecture incorporates virtually no prior knowledge,

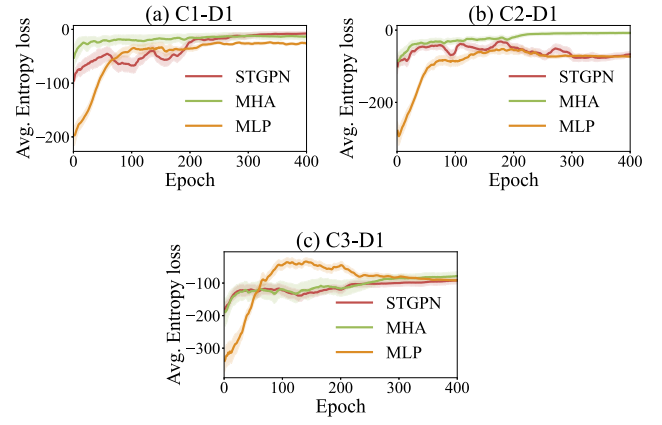


Fig. 11. Entropy loss curves during training.

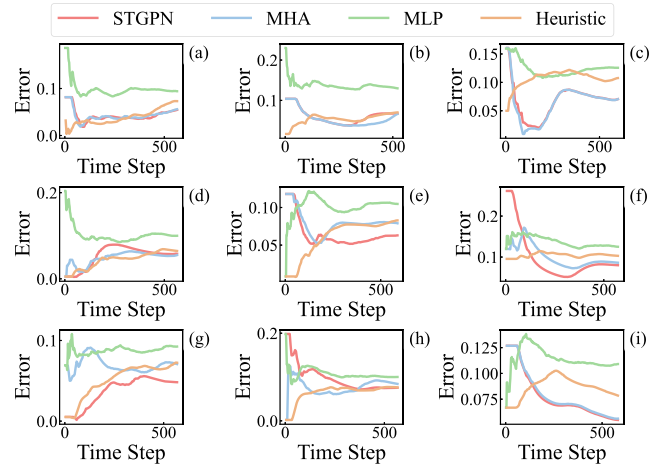


Fig. 12. Strategy behavior analysis.

resulting in highly stochastic behavior (high entropy) and blind random exploration, which corresponds to the slower convergence shown in Fig. 8. Conversely, both the MHA and STGPN architectures explicitly account for prior relationships among nodes in their designs, enabling faster initial convergence (lower entropy) that rapidly directs the search toward optimal regions.

To thoroughly analyze the temporal decision-making process, Fig. 12 examines how each decision step affects the fluctuation of the current accumulated ore grade flow. The heuristic strategy exhibits a typical "locally greedy" decision-making pattern, maximizing short-term gains. However, due to its overemphasis on early-stage local optimization and neglect of a global perspective, ore flow fluctuations gradually intensify as sequential decisions progress, demonstrating a clear cumulative effect. In contrast, the decision patterns learned by STGPN and MHA policy networks demonstrate stronger global optimization capabilities. In most cases Fig. 12(a, b, c, e, f, h, i), the decision patterns of STGPN and MHA strategies are completely opposite to those of the heuristic approach. During the early stages, their decisions typically demonstrate foresight, often sacrificing local gains to maximize future returns.

Further analysis Fig. 12(e, g) reveals that although STGPN and MHA strategies show highly similar patterns, STGPN can select superior actions, resulting in consistently better real-time performance than MHA. Additionally, scenarios Fig. 12(f, h) demonstrate that STGPN exhibits even more forward-looking considerations. Although STGPN may select seemingly suboptimal actions during specific local periods, these choices lead to sustained reward accumulation in subsequent time steps, ultimately achieving maximized overall benefits. This

Table 4

Evaluation of cross-case generalizability. To assess the generalization capability, “Transfer” denotes the STGPN model trained on a source case and directly evaluated on an unseen target case without fine-tuning, while “No-Transfer” refers to the model trained and evaluated on the same target case.

Transfer Task	Scale	STGPN		EA	ACO	MMAS-LS	MACO	ACFGA	DPCGA
		Transfer	No-Transfer						
$C1 \rightarrow C2$	D1	0.0806	0.0803	0.0864	0.0803	0.0797	0.0819	0.0887	0.0867
	D2	0.0694	0.0688	0.0903	0.0728	0.0726	0.0761	0.0915	0.0897
	D3	0.0632	0.0618	0.0931	0.0694	0.0701	0.0735	0.0930	0.0918
	D4	0.0579	0.0589	0.0953	0.0686	0.0692	0.0721	0.0951	0.0931
$C1 \rightarrow C3$	D1	0.0768	0.0760	0.0843	0.0752	0.0744	0.0768	0.0870	0.0843
	D2	0.0640	0.0626	0.0890	0.0683	0.0684	0.0712	0.0915	0.0884
	D3	0.0556	0.0539	0.0904	0.0636	0.0637	0.0670	0.0921	0.0888
	D4	0.0502	0.0489	0.0919	0.0579	0.0586	0.0616	0.0931	0.0897
$C2 \rightarrow C1$	D1	0.0747	0.0746	0.0762	0.0698	0.0695	0.0715	0.0786	0.0764
	D2	0.0617	0.0608	0.0784	0.0614	0.0618	0.0642	0.0786	0.0773
	D3	0.0559	0.0549	0.0819	0.0584	0.0588	0.0625	0.0813	0.0801
	D4	0.0505	0.0474	0.0794	0.0548	0.0558	0.0590	0.0809	0.0776
$C2 \rightarrow C3$	D1	0.0765	0.0760	0.0843	0.0752	0.0744	0.0768	0.0870	0.0843
	D2	0.0636	0.0626	0.0890	0.0683	0.0684	0.0712	0.0915	0.0884
	D3	0.0547	0.0539	0.0904	0.0636	0.0637	0.0670	0.0921	0.0888
	D4	0.0499	0.0489	0.0919	0.0579	0.0586	0.0616	0.0931	0.0897
$C3 \rightarrow C1$	D1	0.0743	0.0746	0.0762	0.0698	0.0695	0.0715	0.0786	0.0764
	D2	0.0616	0.0608	0.0784	0.0614	0.0618	0.0642	0.0786	0.0773
	D3	0.0552	0.0549	0.0819	0.0584	0.0588	0.0625	0.0813	0.0801
	D4	0.0509	0.0474	0.0794	0.0548	0.0558	0.0590	0.0809	0.0776
$C3 \rightarrow C2$	D1	0.0801	0.0803	0.0864	0.0803	0.0797	0.0819	0.0887	0.0867
	D2	0.0686	0.0688	0.0903	0.0728	0.0726	0.0761	0.0915	0.0897
	D3	0.0612	0.0618	0.0931	0.0694	0.0701	0.0735	0.0930	0.0918
	D4	0.0588	0.0589	0.0953	0.0686	0.0692	0.0721	0.0951	0.0931

characteristic indicates that the STGPN strategy can make short-term sacrifices by holistically considering long-term returns. In conclusion, both the STGPN and MHA strategies effectively address the limitations of local greedy decision-making. Notably, STGPN demonstrates more forward-looking decision-making behavior, achieving the maximization of global long-term rewards.

6.9. Discussion on practical deployment

To comprehensively evaluate the applicability of the proposed framework in real-world mining fleet management systems, it is essential to discuss the practical requirements of re-scheduling, the associated computational timeframes, and the model lifecycle management.

In realistic open-pit mining operations, predefined static schedules are highly vulnerable to continuous dynamic events such as unpredicted truck breakdowns or travel time fluctuations. Consequently, the system requires continuous, real-time re-scheduling to maintain ore-flow stability. Due to its end-to-end nature, the proposed STGPN offers highly flexible scheduling timing. The policy network can be instantly queried to output a dispatching command at the exact moment a dynamic event occurs or a truck requests an assignment, eliminating the need for rigid re-optimization intervals.

Industrial fleet management dictates that scheduling decisions must be made within seconds to avoid equipment starvation or queuing. As previously demonstrated in Table 2, the inference time of the STGPN is at the millisecond level, effortlessly satisfying these real-time deployment constraints. Conversely, the policy learning requires an initial offline training phase. Under our experimental setup (NVIDIA 24G RTX 4090 GPU and AMD 504G 96-Core EPYC 9654 CPU), training the STGPN from scratch takes approximately 0.92 h, 0.93 h, and 1.11 h for Cases C1, C2, and C3, respectively. This low computational cost makes the framework highly practical.

Determining when to repeat the training phase is governed by the operational context and performance demands. The model does not require retraining to accommodate daily uncertainties, as verified by the

dynamic scenarios (E1–E5) where the agent’s spatial-temporal perception intrinsically adapts to varying degrees of disruptions. Furthermore, as evaluated in Section 6.7, the model exhibits sufficient zero-shot transferability to maintain reasonable robustness across different mine cases without retraining. However, while cross-case transfer is feasible, it inevitably introduces a slight degradation in absolute optimal performance. Therefore, for mining scenarios with exceptionally stringent requirements on blending stability, we strongly recommend repeating the training phase when the mining topology or fleet configuration undergoes fundamental transformations. Given the low offline training time of approximately one hour, conducting a complete retraining process is computationally inexpensive and guarantees maximal decision-making quality.

7. Conclusion and future works

Current open-pit mine blending research typically treats the task as an offline and static allocation problem that optimizes the overall average grade. This methodological limitation causes severe feed grade fluctuations during execution. To address this, this paper fundamentally redefines the problem as a real-time robust scheduling control problem under dynamic and uncertain environments.

To solve the resulting MDP, which is characterized by high-dimensional spatiotemporal dynamics, demanding decision timeliness, and long-horizon dependencies, we designed an end-to-end DRL framework. Its main contributions are: (1) A spatio-temporal graph policy network that uses a GatedGCN module to process spatial topology and a GRU module to process temporal dynamics, solving the generalization and “short-sighted” decision flaws of traditional architectures. (2) A two-stage training strategy (local pre-training and global fine-tuning) that overcomes the high memory consumption and credit assignment difficulties of long-horizon simulation, making on-site deployment feasible.

Several important directions remain for further exploration. This paper primarily focuses on the single-objective optimization of grade stability. However, in actual production, energy consumption and production efficiency are equally critical, and their simultaneous optimization could yield additional benefits. In addition, currently the

shovel's excavation sequence is treated as part of the environment. Since shovel decisions also influence grade fluctuations, an important future direction is to develop a joint policy that coordinates both shovel excavation and truck dispatching for finer-grained control.

CRedit authorship contribution statement

Hao Tang: Writing – original draft, Visualization, Software, Methodology, Investigation, Formal analysis, Data curation, Conceptualization. **Changhe Li:** Writing – review & editing, Supervision, Project administration, Methodology, Funding acquisition, Conceptualization. **Guoyu Chen:** Writing – review & editing, Software, Methodology, Formal analysis. **Shoufei Han:** Visualization, Software, Investigation, Data curation. **Michalis Mavrovouniotis:** Writing – review & editing, Validation, Resources. **Miqing Li:** Writing – review & editing, Supervision, Conceptualization. **Liang Yuan:** Writing – review & editing, Validation, Resources, Funding acquisition.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Acknowledgments

This work was supported in part by the National Natural Science Foundation of China under Grant 62476006, in part by the Hubei Provincial Natural Science Foundation of China under Grant 2023AFA049, and in part by the Fundamental Research Funds of the AUST under Grant 2024JBZD0007.

Appendix A. Supplementary data

Supplementary material related to this article can be found online at <https://doi.org/10.1016/j.cor.2026.107526>.

Data availability

Data will be made available on request.

References

- Angus, D., Hendtlass, T., 2005. Dynamic ant colony optimisation. *Appl. Intell.* 23 (1), 33–38.
- Ao, M., Li, C., Yang, S., 2023. Prediction method of truck travel time in open pit mines based on LSTM model. In: 2023 42nd Chinese Control Conference. CCC, pp. 8651–8656.
- Bello, I., Pham, H., Le, Q.V., Norouzi, M., Bengio, S., 2016. Neural combinatorial optimization with reinforcement learning. *arXiv preprint arXiv:1611.09940*.
- Bonabeau, E., Dorigo, M., Theraulaz, G., 2000. Inspiration for optimization from social insect behaviour. *Nature* 406 (6791), 39–42.
- de Carvalho, J.P., Dimitrakopoulos, R., 2021. Integrating production planning with truck-dispatching decisions through reinforcement learning while managing uncertainty. *Minerals* 11 (6), 587.
- Chen, L., Gu, Q., Wang, R., Feng, Z., Zhang, C., 2022. Comprehensive utilization of mineral resources: Optimal blending of polymetallic ore using an improved NSGA-III algorithm. *Sustainability* 14 (17), 10766.
- Chiarot Villegas, T.V., Segura Altamirano, S.F., Castro Cárdenas, D.M., Si-fuentes Montes, A.M., Chaman Cabrera, L.I., Aliaga Zegarra, A.S., Oblitas Vera, C.L., Alban Palacios, J.C., 2024. Improving productivity in mining operations: a deep reinforcement learning model for effective material supply and equipment management. *Neural Comput. Appl.* 36 (9), 4523–4535.
- Dering, D., Swartz, C.L., 2022. A stochastic optimization framework for integrated scheduling and control under demand uncertainty. *Comput. Chem. Eng.* 165, 107931.
- Dwivedi, V.P., Joshi, C.K., Luu, A.T., Laurent, T., Bengio, Y., Bresson, X., 2023. Benchmarking graph neural networks. *J. Mach. Learn. Res.* 24 (43), 1–48.
- Eiben, A.E., Smith, J., 2015. From evolutionary computation to the evolution of things. *Nature* 521 (7553), 476–482.
- Eyckelhof, C.J., Snoek, M., 2002. Ant systems for a dynamic TSP: Ants caught in a traffic jam. In: *International Workshop on Ant Algorithms*. Springer, pp. 88–99.
- Gao, S., Wang, Y., Cheng, J., Inazumi, Y., Tang, Z., 2016. Ant colony optimization with clustering for solving the dynamic location routing problem. *Appl. Math. Comput.* 285, 149–173.
- Glomb, L., Liers, F., Rösel, F., 2022. A rolling-horizon approach for multi-period optimization. *European J. Oper. Res.* 300 (1), 189–206.
- González, P., Osorio, R.R., Pardo, X.C., Banga, J.R., Doallo, R., 2022. An efficient ant colony optimization framework for HPC environments. *Appl. Soft Comput.* 114, 108058.
- Guntsch, M., Middendorf, M., 2002. Applying population based ACO to dynamic optimization problems. In: *International Workshop on Ant Algorithms*. Springer, pp. 111–122.
- Hamzadayi, A., Yildiz, G., 2016. Event driven strategy based complete rescheduling approaches for dynamic m identical parallel machines scheduling problem with a common server. *Comput. Ind. Eng.* 91, 66–84.
- Hazrathosseini, A., Moradi Afrapoli, A., 2024. Transition to intelligent fleet management systems in open pit mines: A critical review on application of reinforcement-learning-based systems. *Min. Technol.* 133 (1), 50–73.
- Herding, R., Mönch, L., 2024. A rolling horizon planning approach for short-term demand supply matching. *Cent. Eur. J. Oper. Res.* 32 (3), 865–896.
- Holguin Jimenez, S., Trabelsi, W., Sauvey, C., 2024. Multi-objective production rescheduling: A systematic literature review. *Mathematics* 12 (20), 3176.
- Hu, E.J., Shen, Y., Wallis, P., Allen-Zhu, Z., Li, Y., Wang, S., Wang, L., Chen, W., 2022. LoRA: Low-rank adaptation of large language models. In: *International Conference on Learning Representations*.
- Huo, D., Sari, Y.A., Kealey, R., Zhang, Q., 2023. Reinforcement learning-based fleet dispatching for greenhouse gas emission reduction in open-pit mining operations. *Resour. Conserv. Recycl.* 188, 106664.
- Huo, D., Sari, Y.A., Zhang, Q., 2024. Smart dispatching for low-carbon mining fleet: A deep reinforcement learning approach. *J. Clean. Prod.* 435, 140459.
- Jia, Y.-H., Chen, W.-N., Gu, T., Zhang, H., Yuan, H., Lin, Y., Yu, W.-J., Zhang, J., 2017. A dynamic logistic dispatching system with set-based particle swarm optimization. *IEEE Trans. Syst. Man Cybern.: Syst.* 48 (9), 1607–1621.
- Kool, W., van Hoof, H., Welling, M., 2019. Attention, learn to solve routing problems!. In: *International Conference on Learning Representations*.
- Koster, A.M., Segsneider, J., Ventsch, N., 2024. γ -Robust optimization of project scheduling problems. *Comput. Oper. Res.* 161, 106453.
- Li, N., Wu, Y., Ye, H., Wang, L., Wang, Q., Jia, M., 2024. Scheduling optimization of underground mine trackless transportation based on improved estimation of distribution algorithm. *Expert Syst. Appl.* 245, 123025.
- Li, F., Yue, Q., Liu, Y., Ouyang, H., Gu, F., 2024. A fast density peak clustering based particle swarm optimizer for dynamic optimization. *Expert Syst. Appl.* 236, 121254.
- Liu, Y., Liu, J., Jin, Y., Li, F., Zheng, T., 2020. An affinity propagation clustering based particle swarm optimizer for dynamic optimization. *Knowl.-Based Syst.* 195, 105711.
- Liu, B., Zhang, D., Gao, X., 2021. A method of ore blending based on the quality of beneficiation and its application in a concentrator. *Appl. Sci.* 11 (11), 5092.
- Lu, M., Shen, Z.-J.M., 2021. A review of robust operations management under model uncertainty. *Prod. Oper. Manage.* 30 (6), 1927–1943.
- Luo, F., Lin, X., Liu, F., Zhang, Q., Wang, Z., 2023. Neural combinatorial optimization with heavy decoder: Toward large scale generalization. *Adv. Neural Inf. Process. Syst.* 36, 8845–8864.
- Luo, F., Lin, X., Wang, Z., Tong, X., Yuan, M., Zhang, Q., 2024. Self-improved learning for scalable neural combinatorial optimization. *arXiv preprint arXiv:2403.19561*.
- Matsui, K., Escibano, J., Angeloudis, P., 2023. Real-time dispatching for autonomous vehicles in open-pit mining deployments using deep reinforcement learning. In: 2023 IEEE 26th International Conference on Intelligent Transportation Systems. ITSC, IEEE, pp. 5468–5475.
- Mavrovouniotis, M., Bonilha, I.S., Müller, F.M., Ellinas, G., Polycarpou, M., 2019. Effective ACO-based memetic algorithms for symmetric and asymmetric dynamic changes. In: 2019 IEEE Congress on Evolutionary Computation. CEC, IEEE, pp. 2567–2574.
- Mavrovouniotis, M., Li, C., Yang, S., 2017a. A survey of swarm intelligence for dynamic optimization: Algorithms and applications. *Swarm Evol. Comput.* 33, 1–17.
- Mavrovouniotis, M., Li, C., Yazdani, D., Handjimitsis, D., 2024. Exchange strategies for multi-colony ant algorithms in dynamic environments. In: 2024 IEEE Congress on Evolutionary Computation. CEC, IEEE, pp. 01–08.
- Mavrovouniotis, M., Müller, F.M., Yang, S., 2016. Ant colony optimization with local search for dynamic traveling salesman problems. *IEEE Trans. Cybern.* 47 (7), 1743–1756.
- Mavrovouniotis, M., Müller, F.M., Yang, S., 2017b. Ant colony optimization with local search for dynamic traveling salesman problems. *IEEE Trans. Cybern.* 47 (7), 1743–1756. <http://dx.doi.org/10.1109/TCYB.2016.2556742>.
- Mavrovouniotis, M., Yang, S., 2012. Ant colony optimization with memory-based immigrants for the dynamic vehicle routing problem. In: 2012 IEEE Congress on Evolutionary Computation. IEEE, pp. 1–8.

- Mavrovouniotis, M., Yang, S., 2013. Ant colony optimization with immigrants schemes for the dynamic travelling salesman problem with traffic factors. *Appl. Soft Comput.* 13 (10), 4023–4037.
- Mavrovouniotis, M., Yang, S., 2014. Ant colony optimization with self-adaptive evaporation rate in dynamic environments. In: 2014 IEEE Symposium on Computational Intelligence in Dynamic and Uncertain Environments. CIDUE, IEEE, pp. 47–54.
- Mavrovouniotis, M., Yang, S., 2015. Ant algorithms with immigrants schemes for the dynamic vehicle routing problem. *Inform. Sci.* 294, 456–477.
- Mavrovouniotis, M., Yang, S., Van, M., Li, C., Polycarpou, M., 2020. Ant colony optimization algorithms for dynamic optimization: A case study of the dynamic travelling salesperson problem [research frontier]. *IEEE Comput. Intell. Mag.* 15 (1), 52–63.
- Mohajerin Esfahani, P., Kuhn, D., 2018. Data-driven distributionally robust optimization using the Wasserstein metric: Performance guarantees and tractable reformulations. *Math. Program.* 171 (1), 115–166.
- Moradi Afrapoli, A., Tabesh, M., Askari-Nasab, H., 2019. A multiple objective transportation problem approach to dynamic truck dispatching in surface mines. *European J. Oper. Res.* 276 (1), 331–342.
- Nikbin, V., Moradi Afrapoli, A., 2023. A multiple objective hybrid algorithm for daily ore blend in oil sands mines. *Int. J. Min. Reclam. Environ.* 37 (9), 667–682.
- Noriega, R., Pourrahimian, Y., Askari-Nasab, H., 2025. Deep Reinforcement Learning based real-time open-pit mining truck dispatching system. *Comput. Oper. Res.* 173, 106815.
- de Oliveira, S.M., Bezerra, L.C., Stützle, T., Dorigo, M., Wanner, E.F., de Souza, S.R., 2021. A computational study on ant colony optimization for the traveling salesman problem with dynamic demands. *Comput. Oper. Res.* 135, 105359.
- Patterson, S.R., Kozan, E., Hyland, P., 2017. Energy efficient scheduling of open-pit coal mine trucks. *European J. Oper. Res.* 262 (2), 759–770.
- Pillac, V., Gendreau, M., Guéret, C., Medaglia, A.L., 2013. A review of dynamic vehicle routing problems. *European J. Oper. Res.* 225 (1), 1–11.
- Prakasam, A., Savarimuthu, N., 2019. Novel local restart strategies with hyperpopulated ant colonies for dynamic optimization problems. *Neural Comput. Appl.* 31 (Suppl 1), 63–76.
- Rezvanian, A., Vahidipour, S.M., Sadollah, A., 2023. An overview of ant colony optimization algorithms for dynamic optimization problems. In: *Optimization Algorithms-Classics and Recent Advances*. IntechOpen.
- Samà, M., Pellegrini, P., D'Ariano, A., Rodriguez, J., Pacciarelli, D., 2016. Ant colony optimization for the real-time train routing selection problem. *Transp. Res. B* 85, 89–108.
- Schlenkrich, M., Seiringer, W., Altendorfer, K., Parragh, S.N., 2024. Enhancing rolling horizon production planning through stochastic optimization evaluated by means of simulation. *arXiv preprint arXiv:2402.14506*.
- Shi, L., Zhan, Z.-H., Liang, D., Zhang, J., 2022. Memory-based ant colony system approach for multi-source data associated dynamic electric vehicle dispatch optimization. *IEEE Trans. Intell. Transp. Syst.* 23 (10), 17491–17505.
- Skackauskas, J., 2022. HEDCOS: High Efficiency Dynamic Combinatorial Optimization System using Ant Colony Optimization Algorithm (Ph.D. thesis). Brunel University London.
- Su, Y., Liu, J., Xiang, X., Zhang, X., 2021. A responsive ant colony optimization for large-scale dynamic vehicle routing problems via pheromone diversity enhancement. *Complex Intell. Syst.* 7 (5), 2543–2558.
- Đurasević, M., Jakobović, D., 2018. A survey of dispatching rules for the dynamic unrelated machines environment. *Expert Syst. Appl.* 113, 555–569. <http://dx.doi.org/10.1016/j.eswa.2018.06.053>.
- Wang, S., Sun, W., Huang, M., 2024. An adaptive large neighborhood search for the multi-depot dynamic vehicle routing problem with time windows. *Comput. Ind. Eng.* 191, 110122.
- Williams, R.J., 1992. Simple statistical gradient-following algorithms for connectionist reinforcement learning. *Mach. Learn.* 8 (3), 229–256.
- Wu, Y., Song, W., Cao, Z., Zhang, J., Lim, A., 2022. Learning improvement heuristics for solving routing problems. *IEEE Trans. Neural Netw. Learn. Syst.* 33 (9), 5057–5069.
- Xiang, X., Tian, Y., Zhang, X., Xiao, J., Jin, Y., 2021. A pairwise proximity learning-based ant colony algorithm for dynamic vehicle routing problems. *IEEE Trans. Intell. Transp. Syst.* 23 (6), 5275–5286.
- Yao, J., Wang, Z., Chen, H., Hou, W., Zhang, X., Li, X., Yuan, W., 2023. Open-pit mine truck dispatching system based on dynamic ore blending decisions. *Sustainability* 15 (4), 3399.
- Yazdani, D., Cheng, R., He, C., Branke, J., 2020. Adaptive control of subpopulations in evolutionary dynamic optimization. *IEEE Trans. Cybern.* 52 (7), 6476–6489.
- Yazdani, D., Cheng, R., He, C., Branke, J., 2022. Adaptive control of subpopulations in evolutionary dynamic optimization. *IEEE Trans. Cybern.* 52 (7), 6476–6489. <http://dx.doi.org/10.1109/TCYB.2020.3036100>.
- Yazdani, D., Cheng, R., Yazdani, D., Branke, J., Jin, Y., Yao, X., 2021. A survey of evolutionary continuous dynamic optimization over two decades—Part A. *IEEE Trans. Evol. Comput.* 25 (4), 609–629.
- Yazdani, D., Omidvar, M.N., Yazdani, D., Branke, J., Nguyen, T.T., Gandomi, A.H., Jin, Y., Yao, X., 2023. Robust optimization over time: a critical review. *IEEE Trans. Evol. Comput.*
- Zhang, L., Hu, Y., Tang, Q., Li, J., Li, Z., 2021. Data-driven dispatching rules mining and real-time decision-making methodology in intelligent manufacturing shop floor with uncertainty. *Sensors* 21 (14), 4836.
- Zhang, C., Odonkor, P., Zheng, S., Khorasgani, H., Serita, S., Gupta, C., Wang, H., 2020. Dynamic dispatching for large-scale heterogeneous fleet via multi-agent deep reinforcement learning. In: 2020 IEEE International Conference on Big Data. Big Data, IEEE, pp. 1436–1441.
- Zhang, T., Wang, H., Yuan, B., Jin, Y., Yao, X., 2022. Surrogate-assisted evolutionary Q-learning for black-box dynamic time-linkage optimization problems. *IEEE Trans. Evol. Comput.* 27 (5), 1162–1176.
- Zhou, R., Nee, A., Lee, H., 2009. Performance of an ant colony optimisation algorithm in dynamic job shop scheduling problems. *Int. J. Prod. Res.* 47 (11), 2903–2920.