

Review

Deep Reinforcement Learning for Combinatorial Optimization Problems: A Challenge-Driven Methodology and Systematic Review

Shengyun Wei ^{1,2,†} , Chuibing Huang ¹ , Zhenyi Wang ^{2,†} , Yang Wang ¹ , Dekang Kong ¹ , Haibo Mi ^{2,*}  and Zhaolong Sun ^{1,*} 

- ¹ School of Electrical Engineering, Naval University of Engineering, Wuhan 430030, China; d23180821@nue.edu.cn (S.W.); 1920191119@nue.edu.cn (C.H.); wangyang8828@stu.ouc.edu.cn (Y.W.); 2000510@nue.edu.cn (D.K.)
- ² State Key Laboratory of Complex and Critical Software Environment, Information Support Force Engineering University, Wuhan 430017, China; wangzhenyi@nudt.edu.cn
- * Correspondence: haibo_mihb@nudt.edu.cn (H.M.); 1110061090@nue.edu.cn (Z.S.)
- † These authors contributed equally to this work.

Abstract

Combinatorial optimization problems (COPs) offer essential mathematical frameworks and algorithmic foundations for modeling complex real-world decision-making tasks. Recent advances in deep reinforcement learning (DRL) have shown promising results for solving COPs, offering the potential to reduce dependence on domain-specific expertise and improve generalization across problem instances. These developments have accelerated research in the field and spurred the emergence of numerous innovative methods. Nevertheless, significant theoretical and practical challenges remain. A systematic synthesis of these challenges and their corresponding solutions is critical to guiding the future development of DRL-based approaches. To address this need, we propose a unified challenge-driven framework consisting of four core components: an environment, a state–action–reward mechanism, a solver, and an evaluation module. Using this framework, we conduct a systematic review of approximately 300 recent studies, mapping the evolution of challenges and the progress made in addressing them. We provide a multidimensional analysis of solver designs, training paradigms, and state-of-the-art (SOTA) performance, while documenting publicly available code repositories. Finally, we identify key open problems within the proposed framework to stimulate novel research directions.

Keywords: combinatorial optimization problems; deep reinforcement learning; review; challenge

MSC: 90-02; 8T05



Academic Editor: Pablo Adasme

Received: 28 May 2026

Revised: 6 July 2026

Accepted: 13 July 2026

Published: 14 July 2026

Copyright: © 2026 by the authors.

Licensee MDPI, Basel, Switzerland.

This article is an open access article distributed under the terms and

conditions of the [Creative Commons Attribution \(CC BY\)](https://creativecommons.org/licenses/by/4.0/) license.

1. Introduction

Combinatorial optimization problems (COPs) aim to identify optimal solutions from discrete combinations of objects that satisfy a set of constraints while minimizing or maximizing a given objective function. These problems constitute a central research domain in computer science, operations research, and industrial engineering, particularly in the areas of algorithm design, computational complexity, and real-world applications. COPs encompass a wide variety of structures and are commonly classified into categories such as routing, scheduling, layout, assignment, location, and planning problems [1,2]. Among

these, routing and scheduling problems have attracted the most extensive and in-depth research, owing to their foundational importance in critical industries, including logistics and manufacturing, and their inherent status as NP-hard problems. Routing problems focus on optimizing vehicle paths within spatial networks, while scheduling problems emphasize the allocation and sequencing of tasks across limited resources. These problems have been widely applied in transportation logistics [3], semiconductor manufacturing [4–6], flexible robotics [7], network function virtualization [8], power systems [9], and healthcare services [10]. The majority of COPs are NP-hard, implying that the size of the solution space grows exponentially with problem scale. Consequently, exact methods, such as branch-and-bound [11] and integer linear programming [12], can guarantee optimality but are typically limited to small- or medium-scale instances due to computational intractability [13]. In contrast, heuristic and metaheuristic algorithms [14–16] produce high-quality approximate solutions within reasonable runtime. However, these approaches often require extensive domain-specific tuning for each problem variant and frequently fail to meet real-time response requirements in dynamic environments [17]. Additionally, traditional optimization methods struggle to capture transferable knowledge under varying environmental conditions, often requiring repeated re-optimization for each new scenario. In contrast, recent advances in deep reinforcement learning (DRL)-based neural combinatorial optimization (NCO) solvers have demonstrated substantial improvements in solving COPs, not only enhancing solution quality and reducing computational time, but also significantly improving generalization across unseen problem instances. For example, NeuOpt [18] outperforms heuristic solvers such as LKH in terms of average route length when solving capacitated vehicle routing problem (CVRP) instances with $N = 100$. GOLP [19] represents the first neural solver capable of effectively scaling to traveling salesman problem (TSP) instances with 100,000 nodes (TSP100K), achieving a 5.1% optimality gap and a $174\times$ speedup relative to LKH-3. Furthermore, MTNCO [20], MVMoE [21], and RouteFinder [22] enable multi-task learning across CVRP variants, facilitating rapid adaptation to new problem configurations with altered constraints or objectives. RLACO [2] provides a comprehensive empirical evaluation across four distinct classes of COPs, systematically demonstrating the superior performance of DRL-based approaches over traditional exact solvers and metaheuristics. The emergence of these methods, coupled with sustained research momentum, indicates that DRL-based NCO solvers represent an active and rapidly developing research direction for addressing complex COPs. However, their practical advantages over well-established exact and heuristic solvers remain context-dependent and require careful empirical validation on standardized benchmarks.

1.1. Motivation

This study was motivated by two core research questions (RQs), which were aimed at elucidating the underlying motivations and conceptual foundations of our work.

1. Why is this review being conducted?
2. What are the differences between this review and the previous ones?

This review was motivated by four specific gaps in the existing literature: (1) the absence of a unified analytical framework that captures interdependencies among environment, state, action, reward, and solver design; (2) domain fragmentation, where surveys focus narrowly on either routing or scheduling without capturing cross-domain patterns; (3) limited performance benchmarking, with many prior reviews lacking systematic, multi-benchmark evaluations; and (4) insufficient coverage of foundational challenges such as state representation design and reward engineering that underlie higher-level issues like the curse of dimensionality.

In recent years, several review articles have summarized advances in DRL for solving COPs. However, the existing reviews exhibit several limitations. First, most studies lack a unified conceptual framework that systematically decomposes COPs into core components, namely environment modeling, state–action–reward design, solver architecture, and performance evaluation, impeding a deeper understanding of their intrinsic relationships and evolutionary patterns. For instance, the reviews by [23,24] primarily adopt solver categorization as their central analytical thread, whereas in scheduling problems the state–action–reward design components carry equal technical significance [25–28]. Second, many existing reviews concentrate on specific problem domains, such as scheduling [29–36] or routing [24,37], but with the rapid emergence of multi-task learning and foundation models [20–22] this narrow focus fails to capture critical advances in cross-problem generalization and unified modeling. Moreover, many reviews [1,23,30,32–38] have not reported performance metrics, while others [24,29,31,39,40] provide only limited dataset coverage and narrow method comparisons. Third, current reviews predominantly focus on domain-specific challenges such as the curse of dimensionality and training efficiency [30], while overlooking foundational components that underpin these challenges: the environment–state–action–reward framework (Figure 1). The literature also lacks systematic coverage of other critical challenges, including large-scale optimization [19], sample efficiency [41,42], sparse rewards [43,44], multi-objective optimization [3,45], dynamic environments [46,47], unified benchmarking [2], and real-world deployment [48]. Furthermore, several reviews addressing multiple COP classes [1,23,40] either overemphasize traditional machine learning or suffer from limited coverage.

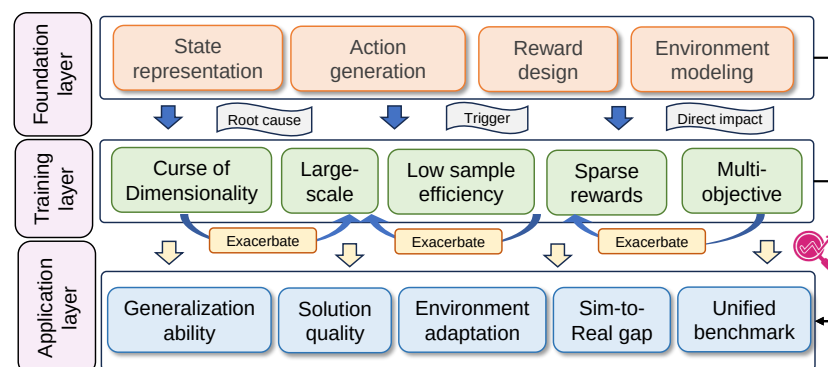


Figure 1. Three-tiered cascaded challenge framework.

To position this review against prior work, Table 1 provides a systematic comparison with existing surveys on COPs. Unlike previous studies, this review introduces a challenge-driven unified framework, which systematically organizes four core components: environment, state–action–reward mechanism, solver, and evaluation benchmark. While encompassing a broad spectrum of well-studied COPs, we focus specifically on two representative categories, scheduling and routing problems, to illustrate their conceptual typicality and methodological universality. To systematically characterize the prevailing challenges in the field, we propose a three-tiered challenge framework. Methodologically, we concentrate on state-of-the-art (SOTA) DRL approaches that have demonstrated superior performance in recent years. To ensure both breadth and depth, we conducted a comprehensive, systematic review of approximately 300 high-quality peer-reviewed publications, prioritizing methodological rigor and empirical reproducibility. Subsequently, we performed a systematic analysis of representative DRL-based solvers, synthesizing key design characteristics, including problem types, solver architectures, feature embedding strategies, training algorithms, and open-source implementations. Finally, we have com-

piled the SOTA performance results across the 17 most widely adopted benchmark datasets in the field, providing a standardized reference for future comparative studies.

Table 1. Comparison between this work and related reviews.

Ref	Problem	Methodological	Time Range	Number	Year	Features							
						UPS F	Env	S	A	R	DO	MA	BM
Lv et al. [29]	Scheduling	DRL	2019–2024	60	2025		✓	✓	✓		✓	✓	✓
Zhang et al. [30]	Scheduling	DRL	2016–2023	93	2025	✓						✓	
Xu et al. [31]	Scheduling	DRL&GP	2003–2024	128	2025	✓		✓	✓	✓	✓		✓
Ngwu et al. [32]	Scheduling	DRL	2010–2024	53	2025						✓		
Wang et al. [33]	Scheduling	DRL	2013–2023	44	2024		✓	✓	✓	✓			
Kayhan et al. [34]	Scheduling	DRL	1995–2020	80	2023						✓	✓	
Smit et al. [35]	Scheduling	DRL	2020–2024	57	2025						✓		
Khadivi et al. [36]	Scheduling	DRL	1995–2024	112	2025		✓			✓	✓	✓	
Li et al. [39]	Routing	ML	2015–2021	68	2022								✓
Shahbazian et al. [37]	Routing	ML	2020–2023	57	2024		✓				✓		
Wu et al. [24]	Routing	ML	2017–2025	230	2025		✓	✓	✓				✓
Zhang et al. [38]	COPs	ML	2017–2025	195	2023		✓	✓	✓	✓	✓		
Garmendia et al. [40]	COPs	DRL	2015–2022	10	2024		✓						✓
Wang et al. [23]	COPs	ML	2015–2023	29	2024		✓						
Chung et al. [1]	COPs	DRL	2017–2024	34	2025	✓	✓				✓		
Ours	COPs	DRL	2017–2025	303		✓	✓	✓	✓	✓	✓	✓	✓

Notation: Symbol ✓ indicates an in-depth elaboration. Abbreviations correspond to the following terms: reference (Ref); unified problem-solving framework (UPS F); DRL environments (Env); state (S); action (A); reward (R); dynamic optimization (DO); multi-agent (MA); benchmarks (BM); machine learning (ML); genetic programming (GP).

1.2. Research Method

This study conducted a systematic search. The search strategy (Figure 2) employed a two-level keyword assembly: (i) COP-related terms; (ii) DRL-related terms connected by OR. The two groups were combined with AND. Searches were conducted in Web of Science, IEEE Xplore, SpringerLink, ACM Digital Library, Elsevier ScienceDirect, MDPI, and arXiv. The literature cut-off date was May 2025.

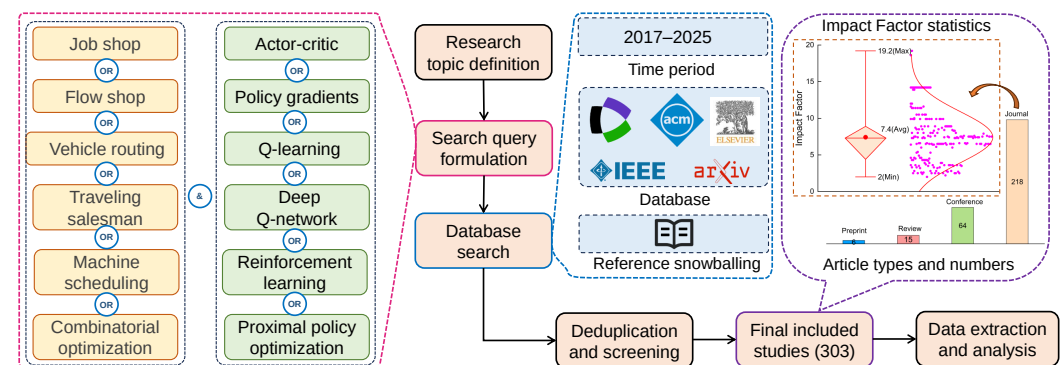


Figure 2. Research method.

Inclusion criteria: (i) studies applying DRL methods to COPs; (ii) peer-reviewed journal articles, conference proceedings, and high-quality preprints (arXiv); (iii) publication year 2017–2025. **Exclusion criteria:** (i) studies using only traditional RL without deep neural networks; (ii) studies applying DRL to non-COP domains; (iii) duplicate publications or non-peer-reviewed technical reports. Screening followed a two-stage process: title and abstract screening to remove clearly irrelevant records, followed by full-text review to confirm eligibility.

To reduce selection bias, supplementary backward and forward citation searches were conducted from key publications. The final set of publications was reviewed by multiple co-authors to ensure balanced coverage across problem domains and methodological

approaches. Publication quality was assessed by prioritizing venues indexed in Web of Science, supplemented by selective inclusion of influential arXiv preprints that have demonstrated significant impact in the NCO community.

We have documented our review protocol transparently, including the explicit search strategy, documented inclusion/exclusion criteria, multi-stage screening process, and quality assessment approach, to enable readers to assess the rigor and reproducibility of the literature selection.

1.3. Contributions

This review presents a comprehensive synthesis of recent advances in DRL-based COPs, offering a systematic and panoramic overview of methodologies developed to date. The main contributions are as follows:

1. We propose a challenge-driven unified analytical framework for the systematic understanding, comparison, and design of DRL-based COP solvers.
2. Within this unified framework, we conducted a comprehensive survey of approximately 300 recent publications, offering a detailed review of methodologies and challenges in environment construction, state representation, action space design, and reward function engineering for DRL-based COPs. Through multi-dimensional analysis across solver categories, network architectures, training paradigms, and SOTA results, we systematically evaluated the current research landscape, identifying trends, limitations, and empirical gaps.
3. We identify a set of key open RQs to guide future investigations into novel paradigms, evaluation benchmarks, and scalability in DRL-based COPs.
4. We introduce Awesome-DRL4COPs, a continuously maintained, open-source repository that consolidates fragmented research on DRL for COPs into a curated, version-controlled resource. Equipped with structured metadata and automated update pipelines, the platform enables efficient literature discovery, reproducibility tracking, and serves as a dynamic, navigable knowledge hub for researchers at the intersection of DRL and COPs.

In summary, the central contribution of this paper is twofold. First, we introduce a challenge-driven unified framework that systematically decomposes the DRL-for-COPs pipeline into five interconnected dimensions: environment, state, action, reward, and solver. This framework serves as both an organizational structure for the review and a design methodology for practitioners. Second, we conducted a comprehensive systematic review of 303 publications organized within this framework, covering methodological advances, benchmark performance, and open challenges. The framework and the review are complementary: the framework provides the analytical lens through which the literature is examined, while the review validated the framework's utility by demonstrating its capacity to organize, compare, and synthesize a large and diverse body of research.

1.4. Review Structure

Unlike the existing reviews that organize literature along a single axis (typically by solver architecture or problem type), our framework (Figure 3) provides a multi-dimensional design methodology: from unified framework and COP formalization (Section 2) through environment characterization (Section 3), MDP formulation and state–action–reward design (Section 4), solver algorithms and architectures (Section 5), benchmark evaluation (Section 6), and open research directions (Section 7), followed by concluding with key findings and a critical discussion of limitations (Section 8). This end-to-end pathway enables researchers to navigate from problem specification to solution deployment within a single, coherent framework.

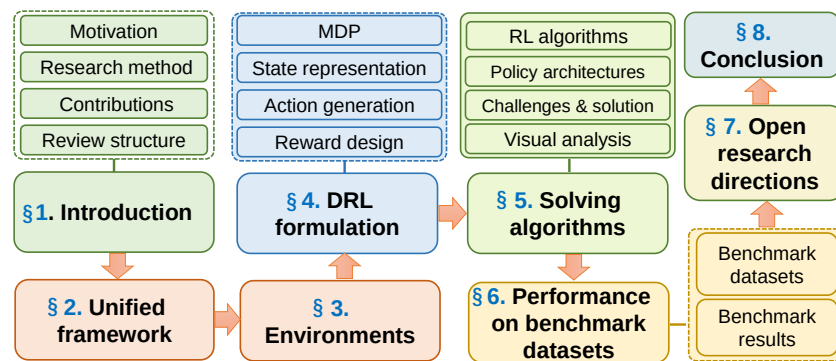


Figure 3. The structure of the challenge-driven methodology and systematic review conducted in the study.

2. Unified Framework

Formally, a combinatorial optimization problem is defined as

$$\min_{x \in \Omega} f(x) \quad \text{subject to} \quad x \in \mathcal{X} \subseteq \Omega, \quad (1)$$

where Ω is a discrete (finite or countably infinite) solution space, $f : \Omega \rightarrow \mathbb{R}$ is the objective function, and $\mathcal{X} \subseteq \Omega$ is the feasible subset defined by constraints $\mathcal{C} = \{c_1, \dots, c_m\}$ with $c_k : \Omega \rightarrow \{0, 1\}$. For NP-hard COPs, $|\Omega|$ grows exponentially with instance size, motivating DRL-based approximate methods.

This review constructs a unified DRL-based framework for COPs (Figure 4), comprising three core modules: environment, agent, and evaluation. The environment module models COP instances by specifying problem types and attribute features, establishing constraint boundaries and optimization objectives. The agent–environment interaction encompasses the state space, action space, and reward mechanism, addressing challenges such as the curse of dimensionality, suboptimal solution quality, and environmental uncertainty. The agent module integrates RL algorithms with specialized solvers to transform COP instances into learnable decision sequences. The evaluation module quantifies and compares solver performance against established benchmarks using standardized metrics. Collectively, this framework integrates the theoretical foundations, technical implementation pathways, and validation methodologies of DRL for COPs.

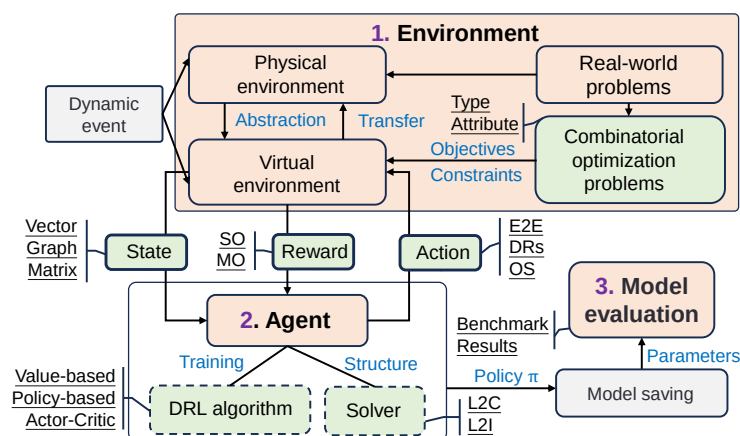


Figure 4. Unified DRL-based framework for COPs.

3. Environments

COPs encompass a broad spectrum of problem classes including routing, scheduling, assignment, packing, covering, graph coloring, and facility location [1]. Rather than attempting shallow coverage of all COP families, this review focuses on routing (TSP, VRP,

OP, PDP) and scheduling (JSP, FSP) as the two most extensively investigated domains in the DRL-for-COPs literature. This choice was data-driven: routing and scheduling collectively account for over 85% of the 303 publications surveyed. For other COP families, including graph coloring, bin packing, knapsack, and facility location, DRL methods have also been applied, as demonstrated by the RL4CO benchmark [2] which implements 27 COP environments spanning routing, scheduling, electronic design automation (mDPP), and graph problems. However, the volume of DRL research in these domains remains substantially smaller than in routing and scheduling. Our deliberate focus on the two most extensively studied domains enables the depth of analysis that a broader but shallower survey could not provide. Figure 5 presents our taxonomy categorizing environments by problem attributes and types.

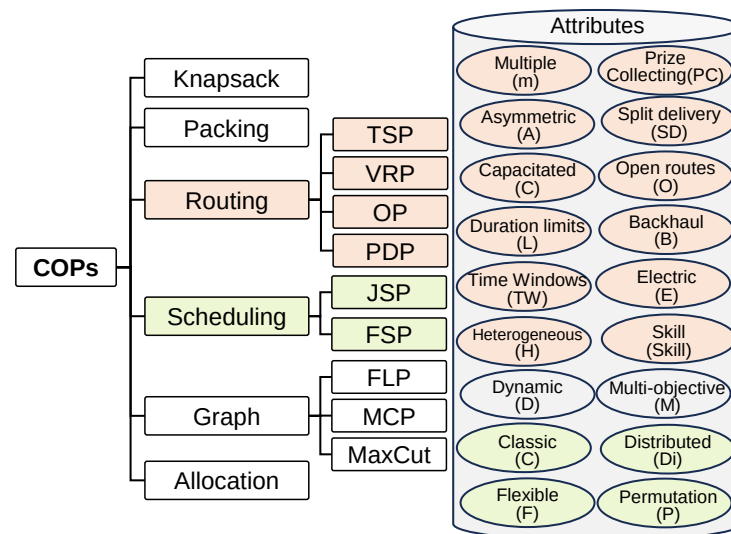


Figure 5. Typical environments and attribute set.

Additional attributes extend fundamental formulations into a wide spectrum of variants, classified into three categories: (1) Generic: dynamic (D) and multi-objective (M), applicable across domains, yielding variants such as DJSP [49,50], DTSP [51,52], DVRP [53,54], MJSP [55,56], and MTSP [57,58]. (2) Routing-specific: multiple (m) yielding mTSP [59], prize collecting (PC) yielding PCTSP [60], asymmetric (A) yielding ATSP [19]; for VRP—capacitated (C) [61], split delivery (SD) [62], open routes (O) [22], duration limits (L) [21], backhaul (B) [63], time windows (TW) [64,65], electric (E) [66–68], heterogeneous (H) [69,70], and skill constraints [71]. (3) Scheduling-specific: flexible (F) [27,72], distributed (Di) [73], and permutation (P) [74]. Figure 6 illustrates attribute-based routing problems; Figure 7 shows an EVRP instance with nine customers, six charging stations, and two vehicles. Multiple attributes combine to form composite variants: two-attribute examples include DFJSP [75], MDFSP [76], and HCVRP [69,70]; three-attribute examples include MDFJSP [46] and DiMDFSP [77]. Scheduling attributes follow the hierarchical order Di–M–P–D–F; routing attributes follow RL4CO conventions [2].

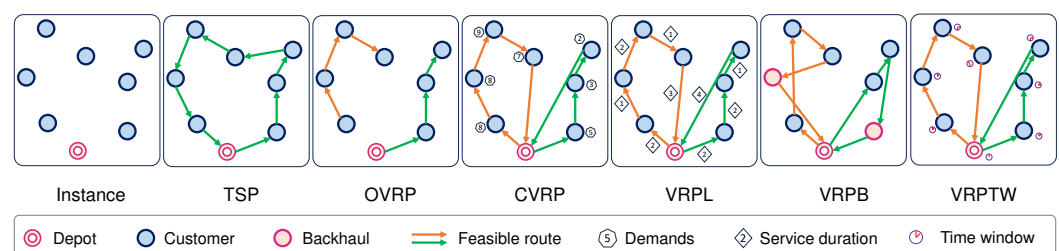


Figure 6. Different VRP attributes.

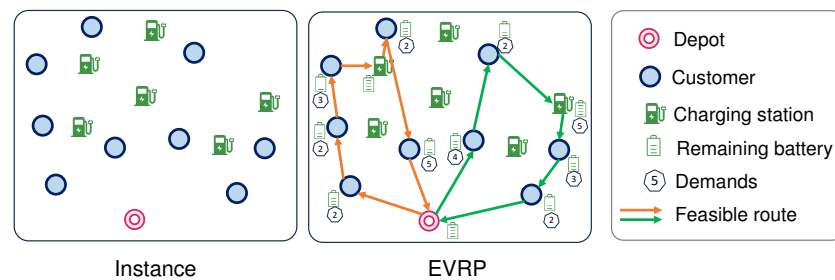


Figure 7. EVRP instance and feasible solution.

Figure 8 shows the historical progression of routing and scheduling research. Flexible scheduling, dynamic environments, multi-objective optimization, and EVRP are the most prominent directions, reflecting industrial demand for scalable, robust solutions. The routing research paradigm exhibits distinct phases: early encoder–decoder optimization [42,62,78,79], followed by universal multi-task algorithms [19,20,63,80–85], with current emphasis on unified heterogeneous frameworks [21,22].

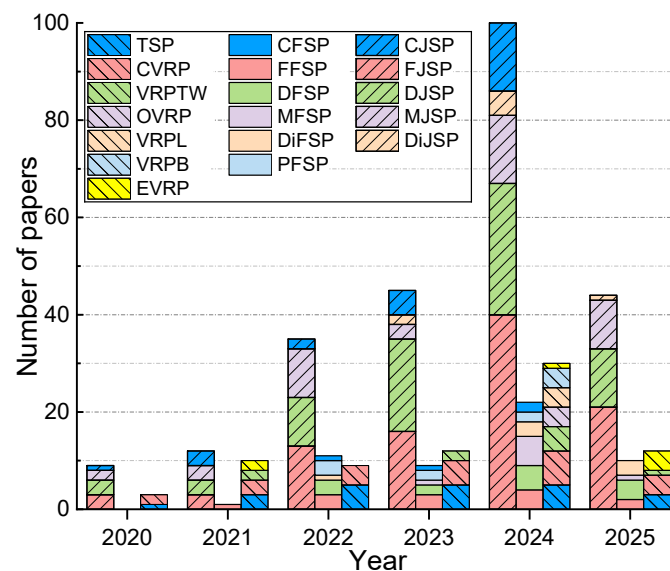


Figure 8. Publication statistics of DRL-based COPs research.

The choice of environment type directly shapes DRL agent design. In static settings (e.g., standard TSP, static JSP), state representations capture the initial instance and construction progress, with fixed-size action spaces and terminal rewards. Dynamic environments (e.g., DVRP, DJSP) require state encodings of real-time events, variable action spaces, and intermediate reward signals for timely guidance. Routing problems with Euclidean structure naturally suit attention-based architectures, while scheduling problems with precedence constraints benefit from graph-based representations (DG, HG). Multi-objective settings demand vector-valued rewards and specialized training strategies, while partially observable settings (POMDP) require belief-state representations or recurrent architectures. These connections between environment characteristics and agent design choices are systematically elaborated in the subsequent sections.

4. DRL Formulation

DRL addresses COPs by modeling them as MDPs, defined by state–action–reward dynamics. This section outlines the DRL framework design according to the following RQs:

1. RQ1: How can COPs be modeled as dynamic decision processes, and under what formal conditions do MDP, POMDP, SMDP, and Dec-POMDP formulations apply?
2. RQ2: How can state representations be designed to capture structural dependencies across COP types, and what are the trade-offs between vector-based, graph-based, matrix-based, Petri net-based, and hybrid representations?
3. RQ3: How should action spaces be structured for different solver paradigms (L2C, L2I), and how does action space dimensionality affect training efficiency?
4. RQ4: How can reward functions be engineered to address sparse feedback, multi-objective trade-offs, and dynamic constraints?

COPs introduce several distinctive characteristics that differentiate DRL formulation from standard RL settings. First, solutions are represented as partial constructions: at each decision step, the agent builds an incomplete solution (e.g., a partial tour in TSP, a partial schedule in JSP), and the state must encode both the problem instance and the current construction progress. Second, feasibility constraints are integral to COP formulations: actions violating problem constraints must be explicitly masked or penalized, making the action space dynamic and state-dependent. Third, rewards are typically delayed: in constructive NCO solvers, meaningful feedback is only available upon solution completion (sparse terminal reward), which differs fundamentally from standard RL where intermediate rewards are often available. These characteristics have driven COP-specific techniques such as encoder–decoder architectures for sequential construction, feasibility masking for constrained action spaces, and specialized baselines for variance reduction under sparse rewards.

4.1. MDP

Figure 9 illustrates the agent–environment interaction mechanisms across different MDP frameworks. MDP is a widely adopted formalism for modeling the evolution of dynamic systems over time and serves as the standard mathematical framework for DRL tasks [86]. COPs based on DRL are typically modeled using MDP or its extended variants. For foundational insights into MDP, see [86,87]. Key MDP extensions include the partially observable MDP (POMDP), the semi-MDP (SMDP), and the decentralized POMDP (Dec-POMDP). Unlike conventional MDP framework diagrams, this study incorporates routing and scheduling state transition models within the environment module based on prior research [88], enabling clearer visualization of inter-temporal state transitions where state updates are triggered by action executions. The observation model is embedded within the POMDP framework and situated along the environment-to-agent pathway, as observations are generated following state updates. The algorithm module trains the policy network to guide agent decision-making.

An MDP is defined as the 5-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma \rangle$, where $\mathcal{S}$ is a measurable state space, $\mathcal{A}$ is a measurable action space, $\mathcal{P} : \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{S})$ is a Markov transition kernel, $\mathcal{R} : \mathcal{S} \times \mathcal{A} \rightarrow \mathbb{R}$ is a bounded measurable reward function, and $\gamma \in [0, 1)$ is the discount factor. The value function is $V^\pi(s) = \mathbb{E}_\pi[\sum_{t=0}^{\infty} \gamma^t \mathcal{R}(s_t, a_t) \mid s_0 = s]$, with the Bellman equation $V^*(s) = \max_{a \in \mathcal{A}} [\mathcal{R}(s, a) + \gamma \int_{\mathcal{S}} \mathcal{P}(ds' \mid s, a) V^*(s')]$. The classic MDP (Figure 9a) is suited to fully observable environments, widely applied in static JSP [7,72] and dynamic settings with immediate event impacts [45,89,90]; multi-agent extensions maintain observability via coordinated architectures [90,91] or event-driven protocols [92,93].

A POMDP extends the MDP via the 7-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{O}, \mathcal{P}, \mathcal{Z}, \mathcal{R}, \gamma \rangle$, where $\mathcal{O}$ is a measurable observation space and $\mathcal{Z} : \mathcal{S} \times \mathcal{A} \rightarrow \Delta(\mathcal{O})$ is the observation model. The agent maintains a belief state $b_t \in \Delta(\mathcal{S})$ updated via Bayes' rule: $b_{t+1}(s') = \eta \cdot \mathcal{Z}(o_{t+1} \mid s', a_t) \int_{\mathcal{S}} \mathcal{P}(s' \mid s, a_t) b_t(s) ds$. For approximate beliefs $\hat{b}_t$, bounded optimality loss requires exponential stability of the belief filter: $\exists C > 0, \lambda \in (0, 1)$ with $\mathbb{E}[\|b_t - \hat{b}_t\|_1] \leq C\lambda^t$. Sufficient conditions

include uniform positivity of the observation likelihood ($\exists \varepsilon > 0, \mu(\cdot)$ with $\mathcal{Z}(o | s) \geq \varepsilon \mu(o)$) and the Doeblin (uniform ergodicity) condition on the transition kernel. POMDP (Figure 9b) is applied in chemical batch scheduling [94] and dynamic vehicle routing [53].

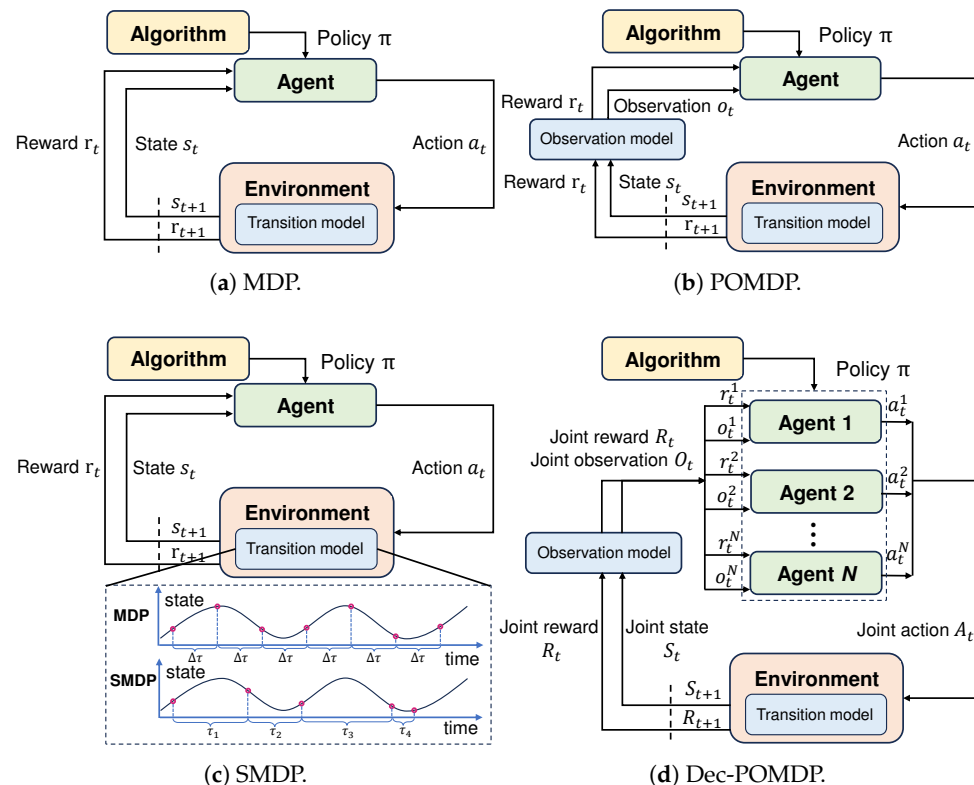


Figure 9. Agent–environment interaction paradigms.

An SMDP is defined as the 6-tuple $\langle \mathcal{S}, \mathcal{A}, \mathcal{P}, \mathcal{R}, \gamma, \mathcal{T} \rangle$, where $\mathcal{T}$ is the set of possible action durations $\tau = \tau(s, a) \in \mathbb{R}_{\geq 0}$. The expected discounted reward is $\mathcal{R}(s, a) = \mathbb{E}[\sum_{k=0}^{\tau-1} \gamma^k r(s_k, a_k) \mid s_0 = s, a_0 = a]$. The Bellman operator $(\mathcal{T}V)(s) = \max_{a \in \mathcal{A}} [\mathcal{R}(s, a) + \sum_{s'} \mathcal{P}(s' \mid s, a) \mathbb{E}[\gamma^{\tau(s,a)} V(s')]]$ is a contraction mapping when $\rho := \sup_{s,a} \mathbb{E}[\gamma^{\tau(s,a)}] < 1$, guaranteeing convergence to V^* . SMDP (Figure 9c) enables hierarchical RL with options [95], event-driven state updates [59,96], and temporal reward accumulation in stochastic environments [97].

A Dec-POMDP extends POMDP to N agents via the 8-tuple, which can be written as $\langle \mathcal{S}, \{\mathcal{A}_i\}_{i=1}^N, \{\mathcal{O}_i\}_{i=1}^N, \mathcal{P}, \mathcal{Z}, \mathcal{R}, \gamma, N \rangle$, with joint policy $\pi = (\pi_1, \dots, \pi_N)$ where $\pi_i : \mathcal{H}_i \rightarrow \Delta(\mathcal{A}_i)$. Under CTDE, a centralized critic accesses the global state during training while each agent uses only local observations during execution [4,98]. Dec-POMDP (Figure 9d) is applied in job-as-agent formulations [99], factory partitioning [100], heterogeneous multi-agent systems [101,102], and hierarchical routing [103].

4.2. State Representation

In COPs, systems are modeled as MDPs or their extended forms, where state representation plays a critical role. Its importance is evident in four key aspects: (1) As shown in Figure 9, the system state serves as the foundation for decision-making; agents perceive environmental states and rewards to select appropriate actions. A well-designed state representation captures diverse system contexts, enabling faster policy convergence. (2) Effective state representation mitigates the curse of dimensionality [74,97,102,104–106]. (3) State representations are fed into DRL algorithms as feature vectors, directly influencing

learning efficiency and solution quality. (4) They also form the basis for responding to dynamic events.

To address state representation challenges, the existing studies predominantly employ five approaches: vector-based, graph-based, matrix-based, Petri net-based, and hybrid methods.

4.2.1. Vector-Based

In routing problems, state features evolve from simple node coordinates to incorporate problem-specific constraints. For the VRP, states comprise node coordinates [62]; for the CVRP, vehicle features (position, capacity) and node demands are added, often as 3D vectors [2]. More complex variants incorporate time windows (VRPTW), route duration limits (VRPL), or dynamic battery levels (EVRP) [20,22]. Enhanced representations may include topological details such as neighbor coordinates and inter-node distances [61,80].

In scheduling contexts, vector-based state features are extracted at the entity level (machines, jobs, operations, factories, AGVs) as multidimensional attributes. As summarized in Table 2, 13 core feature categories are systematically documented. Two encoding approaches coexist: vector-based methods encode raw temporal sequences, while numerical methods summarize distributions via statistical descriptors (mean, std, min, max, range). This representation offers concise and flexible encoding that adapts well to dynamic environments, but it is limited in its capacity to model complex constraints, and it relies on co-design with the action space and reward function [107].

Table 2. Vector-based state representation features and formulas.

Feature	Sub-Feature	Formula
Processing time	Vector [105,108–113]	$\mathbf{p} = [p_{11}, p_{12}, \dots, p_{nn_i}]$
	Minimum [114]	$\min_{i,j} p_{ij}$
	Maximum [115]	$\max_{i,j} p_{ij}$
	Mean [17,43,88,100,101,106,116,117]	$\bar{p} = \frac{\sum_{i=1}^n \sum_{j=1}^{n_i} p_{ij}}{\sum_{i=1}^n n_i}$
	Std [97,118–124]	$\sigma_p = \sqrt{\frac{\sum_{i=1}^n \sum_{j=1}^{n_i} (p_{ij} - \bar{p})^2}{\sum_{i=1}^n n_i}}$
Machine utilization	Sum [4,8,92,125–128]	$\sum_{i=1}^n \sum_{j=1}^{n_i} p_{ij}$
	Range [129]	$\max_{i,j} p_{ij} - \min_{i,j} p_{ij}$
	For machine k [105,106,126,130]	$U_k = \frac{\sum_{i=1}^n \sum_{j=1}^{OP_i} x_{ijk} p_{ijk}}{MCT_k}$
	Mean [28,48,92,101,122,128,129,131,132]	$U_{ave} = \frac{\sum_{k=1}^m U_k}{m}$
	Std [5,115,118,121,130,133–144]	$U_{std} = \sqrt{\frac{\sum_{k=1}^m (U_k - U_{ave})^2}{m}}$
Remaining processing time	Range [46,140]	$\max_k U_k - \min_k U_k$
	For job i [88,90,106,107,115–117,119,136,145]	$RTJ_i = \sum_{j=OP_i+1}^{n_i} APT_{ij}$
Machine status	Binary for machine k [4,28,90,96,105,125,146–148]	$Status_k = \begin{cases} 1 & \text{if } MCT_k > T_{cur} \\ 0 & \text{otherwise} \end{cases}$
Number of jobs	Static [76,101,107,114,115,119,126,147,149–151] Dynamic [28,89,102,120,122–124,129,135,152]	n $ UC_{job} $
Job completion rate	Mean/Std [5,117,120,121,127,128,130,133–140,142,153–155]	$CRJ_{ave} = \frac{\sum_{i=1}^n CRJ_i}{n}, CRJ_{std} = \sqrt{\frac{\sum_{i=1}^n (CRJ_i - CRJ_{ave})^2}{n}}$
Tardiness rate	Actual and estimated [5,130–134,137–141,144,146,153,154,156,157]	$Tard_a = \frac{\sum_{i \in ATJ} (n_i - OP_i)}{\sum_{i \in UC_{job}} (n_i - OP_i)}, Tard_e = \frac{\sum_{i \in ETJ} ETO_i }{\sum_{i \in UC_{job}} (n_i - OP_i)}$
Completion time	For job i [104,124,149,156] For machine k [8,28,45,106,153,158] Operation [109,128]	$C_i = \max_j C_{ij}$ $MCT_k = \max_{i,j} \{C_{ij} \mid O_{ij} \text{ assigned to } M_k\}$ $C_{ij} = ST_{ij} + p_{ij}$
Machine workload	For machine k [45,90,122,126,131,133,136,155]	$W_k = \frac{\sum_{i=1}^n \sum_{j=1}^{OP_i} t_{ijk} x_{ijk}}{\max_i \sum_{j=1}^{OP_i} t_{ijk} x_{ijk}}$

Table 2. Cont.

Feature	Sub-Feature	Formula
Slack time	For job i [8,48,76,107,113,114,116,117,119,153]	$Slack_i = D_i - T_{cur} - \sum_{j=OP_i+1}^{n_i} APT_{ij}$
Idle time	For machine k [4,77,90,97,101,151,159,160]	$Idle_k = T_{cur} - MCT_k$
Number of machines	Static [28,77,89,115,124,126,131,151,161]	m
Setup time	For operation O_{ij} on machine k [4,116,147,150,151]	s_{ijk}

Notation: n : number of jobs, m : number of machines, J_i : job i , O_{ij} : operation j of job i , p_{ij} : processing time of O_{ij} , OP_i : number of scheduled operations of job i , n_i : total number of operations of job i , C_{ij} : completion time of O_{ij} , C_i : completion time of job i , D_i : due time of job i , U_k : utilization rate of machine k , MCT_k : completion time of machine k , CRJ_i : completion rate of job i , $Tard_i$: tardiness rate of job i , UC_{job} : set of uncompleted jobs, T_{cur} : current time, V : number of AGVs. APT_{ij} : average processing time of O_{ij} over compatible machines, ATJ : set of actual tardy jobs, ETJ : set of estimated tardy jobs, ETO_i : set of estimated tardy operations of job i , ST_{ij} : start time of O_{ij} , X_{ijk} : indicator if O_{ij} is assigned to machine k , t_{ijk} : processing time of O_{ij} on machine k .

4.2.2. Graph-Based

Compared to vector-based representations, graph models explicitly capture structural dependencies between operations and resources. The existing methods are classified into four graph-based subclasses: disjunctive graph (DG) [25,41,91,162], heterogeneous graph (HG) [7,27,72], directed acyclic graph (DAG) [9,163], and bipartite graph (BG) [164,165]. The structural properties and instance-solution visualizations of DG and HG are presented in Figure 10. DGs for the JSP assign each operation to a unique machine, whereas FJSP DGs allow multiple candidate machines per operation. HGs represent multiple entity types (machines, jobs, operations) as distinct node categories. Stitched disjunctive graph (SDG), directed acyclic graph (DAG), and bipartite graph (BG) representations, employed by a limited number of studies [9,163–165], are discussed in the text with citations retained.

In Figure 10c,i,k, operation nodes with compatible machine options are highlighted in orange. A feasible schedule is obtained by orienting the undirected disjunctive edges. Upon completion of scheduling, the DG transforms into a DAG, as illustrated in Figure 10b,d. A formal definition of the DG is provided in [162]. The SDG extends the DG into a dynamically stitched structure for distributed scheduling with stochastic job arrivals [166], while the static SDG constructs a global graph by concatenating individual DGs of multiple factories prior to scheduling [73,167]. The HG extends the DG by integrating heterogeneous entities, including machines and transportation resources [168,169]. The BG models machines and buffers as two disjoint node sets connected by directed edges, capturing machine–job dependencies [164,165]. The DAG captures path-scheduling relationships through its directed, acyclic structure [9,163].

Graph models inherently support structured state representation, enabling explicit modeling of complex scheduling characteristics: dependency chains, concurrent resource conflicts, and queuing behaviors. Their dynamic topology allows real-time adaptation via node/edge insertion or deletion in response to temporal events. Following graph construction, GNNs are commonly employed to learn latent representations of nodes and edges. Common architectures include: graph convolutional networks (GCNs) [44,163,170,171], graph isomorphism networks (GINs) [25,91,162,172–179], graph attention networks (GATs) [7,50,168,180–187], heterogeneous graph neural networks (HGNNs) [72,169,188–194]. However, GNN-based feature extraction introduces significant computational overhead. As the problem scale increases, this leads to prohibitive training costs and degraded real-time inference performance.

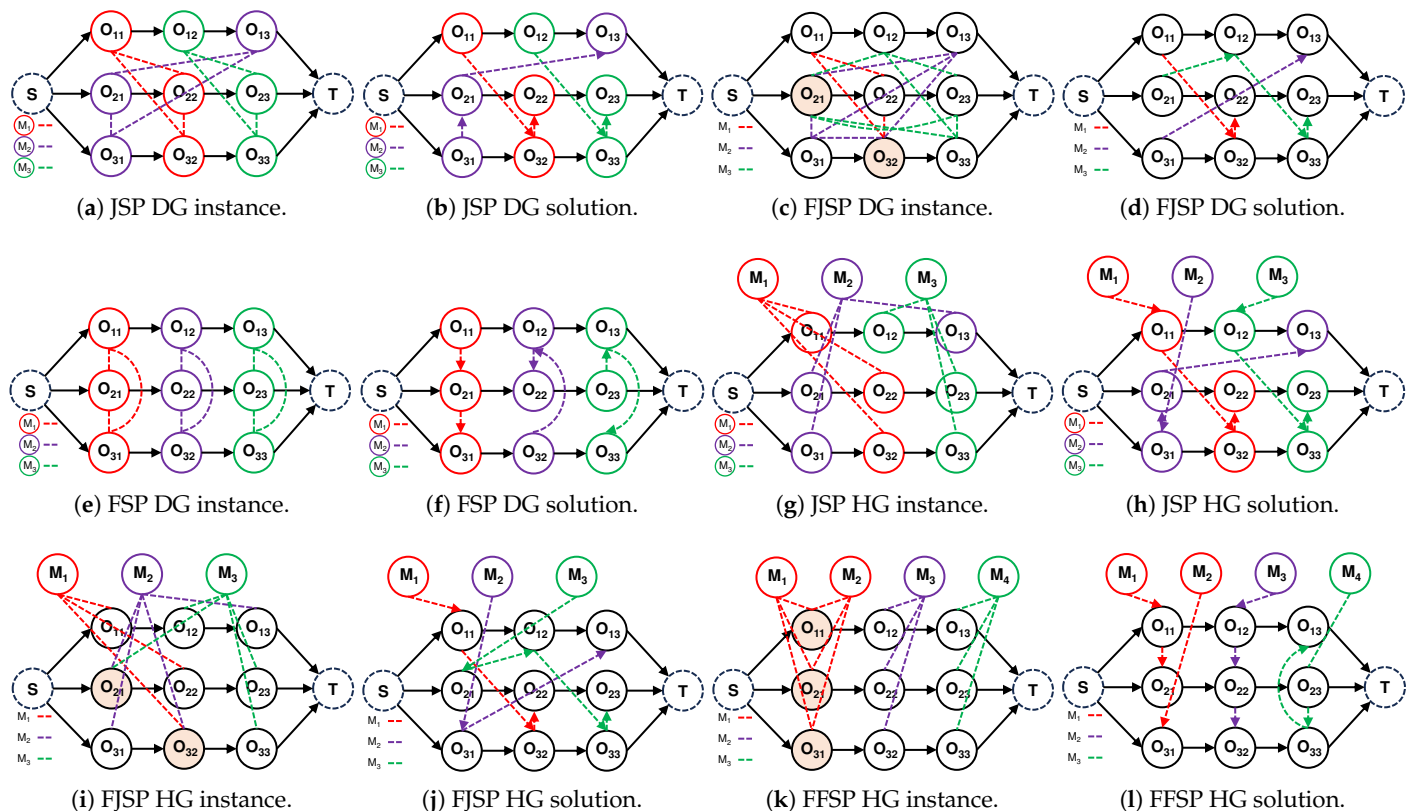


Figure 10. Graph-based state representations for scheduling: disjunctive graphs (DG, rows 1–2) and heterogeneous graphs (HG, rows 2–3). Nodes labeled o_{ji} denote the i -th operation of job j ; nodes M_k denote machines. Solid black arcs represent precedence constraints; colored dashed arcs indicate operations assigned to the same machine.

4.2.3. Matrix-Based

Matrix representations encode structural relationships among machines, jobs, and operations into 2D matrices [49,195,196] or multi-channel 3D tensors [26,197,198]. Encoded features are fed into DRL algorithms via three main approaches: direct input after flattening, feature mapping via MLPs, or input to CNNs/RNNs for automatic embedding. These nonlinear transformations extract higher-level abstractions that capture latent relationships among entities. However, fixed dimensions limit adaptability to dynamic-scale scenarios, and flattening or convolution operations may obscure higher-order topological dependencies.

4.2.4. Petri Nets-Based

Petri nets model concurrent and synchronized dynamic systems by representing resources, events, and constraints as places, transitions, and tokens, respectively. They address machine sequencing challenges by identifying minimal and maximal paths within the net [199]. Recent work has integrated Petri nets with PPO for JSP state representation [200] and applied them in robotic FSP [201,202]. While offering strong semantic interpretability, their state space grows exponentially with the number of resources and tasks, limiting scalability in large-scale problems.

4.2.5. Hybrid

Hybrid state representations combine multiple encoding paradigms, primarily in MARL settings [75,203]. For example, vectors and DGs have been used for high- and low-level agents respectively [75], and matrix-DG combinations encode categorized informa-

tion [204]. While hybrid approaches enhance representational flexibility, they significantly increase state-space complexity.

In summary, existing work has adopted diverse state representation strategies, but two key limitations remain: (1) feature design is heavily dependent on expert knowledge, particularly for heterogeneous entities (operations, jobs, machines, AGVs, factories); (2) there is no consensus on feature efficacy: although some studies suggest that increasing raw features improves performance [27], neither theoretical analysis nor systematic experimentation has identified which features are critical or how many are needed to reach performance saturation.

4.3. Action Generation

Actions are used to generate feasible solutions to the problem. A high-dimensional action space directly exacerbates the curse of dimensionality, and its design is a key factor influencing the training stability and convergence of DRL algorithms [205]. Figure 11 summarizes the primary action generation methods in the existing literature, categorized into three types: end-to-end (E2E), dispatching rules-based (DRs), and operator selection-based (OS). The numbers within the circles represent statistical data: red circles denote the number of papers in each category or subcategory, while gray circles indicate the frequency of occurrence of specific DRs across all relevant studies.

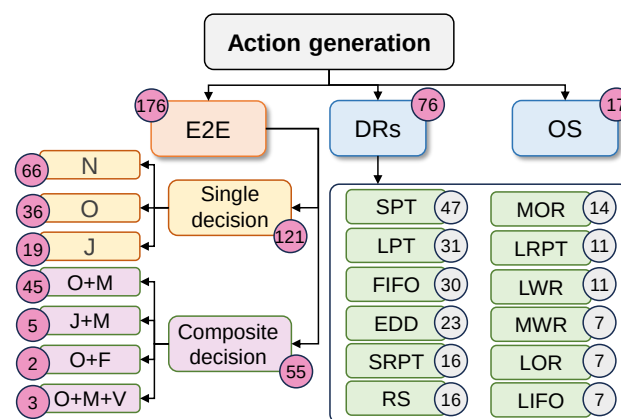


Figure 11. Classification of action generation methods and statistics on paper counts and DRs frequencies.

4.3.1. E2E

The E2E action generation approach directly outputs specific actions upon input of a problem instance into the DRL model. In routing problems, the most common action is selecting the next unvisited node from the set of unvisited nodes [60,64,79,206]. In contrast, action generation in scheduling problems is typically more complex. Single decisions involve either operation selection (O) [25,50,108,167,172,207–210] or job sequencing (J) [77,94,100,160,164,165,178,211]. Composite decisions combine atomic actions, such as operation selection and machine assignment (O + M) [4,7,72,176,177,182,212–215], operation selection and factory assignment (O + F) [73,166], operation selection and job selection (O + J) [114], and operation selection, machine assignment, and vehicle assignment (O + M + V) [168,169]. The combinatorial nature of these composite actions often leads to exponential growth in the action space, substantially increasing training complexity, particularly for large-scale problems [72,198,216–219].

To mitigate this challenge, the existing approaches primarily fall into two categories: invalid action masking [25,200,220] and composite action decomposition. For instance, Bonetta et al. [221] employed a state matrix to filter invalid actions in real time; Yuan et al. [222] improved training efficiency by masking illegal machine assignments and

repeated operations. In composite action decomposition, several studies [27,93,99,159,160] reformulated composite decisions as independent atomic actions within a MARL framework. For example, certain works [27,93,159] modeled each machine as an agent, allowing it to select one job or operation per step. Wang et al. [99] used job agents, each selecting a robot-related action at each decision step. An alternative approach embeds machine availability into the state representation, enabling the policy network to implicitly infer optimal machine assignments without explicitly including machine selection in the action space [160]. Furthermore, some methods cascade operation selection and machine assignment, reducing the action space from combinatorial to polynomial complexity [190].

4.3.2. DRs-Based

The DRs-based approach integrates expert knowledge into the DRL framework, where the action space is composed of heuristic DRs. This method constructs a compact, problem-scale-independent action set through experiential customization, thereby significantly reducing the search space and improving training efficiency. The statistical profile in Figure 12 illustrates this effect. Commonly employed DRs include: shortest processing time (SPT), longest processing time (LPT), first-in-first-out (FIFO), earliest due date (EDD), shortest remaining processing time (SRPT), random selection (RS), most operations remaining (MOR), longest remaining processing time (LRPT), least work remaining (LWR), most work remaining (MWR), least operations remaining (LOR), and last-in-first-out (LIFO). The model selects a single heuristic from this predefined set [49]. Composite DRs involve integrated decisions that combine multiple scheduling logics, such as operation selection and machine assignment [146], job selection and machine assignment [5,105,138,191], or joint planning of jobs, machines, and maintenance [56]. These composite rules are broadly categorized into four types. The first type employs manually predefined combinations, such as the 11 rule variants integrating job selection and factory assignment strategies [134]. The second type constructs fixed composites via Cartesian products of basic rules for operation and machine selection [138,146]. The third type uses weighted combinations, wherein agents output continuous weight vectors to dynamically adjust rule mixtures, enabling adaptive behavior [114,118]. The fourth type leverages genetic programming (GP) to automatically evolve rule structures, treating basic rules as terminals and optimizing them through evolutionary search to enhance performance and generalization [89,120].

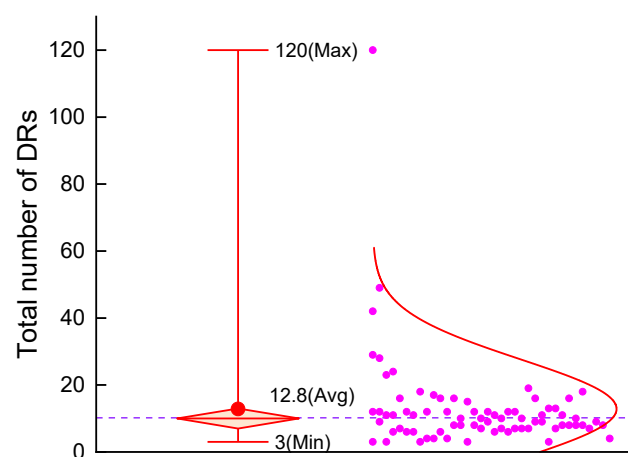


Figure 12. Distribution of the total number of DRs in the action space.

However, these methods are constrained by the expressive capacity of the rule library: if the optimal scheduling strategy cannot be represented by existing rules or their combinations, performance is inherently limited. To overcome the bottleneck of manual rule design, researchers have introduced automated synthesis techniques. For instance, niching

genetic programming generates diverse, high-performing rules, expanding the effective action space and improving adaptability in dynamic environments [205].

A further challenge lies in evaluating DR effectiveness. Current approaches primarily include: (1) comparative performance analysis between designed DRs and classical static rules [97,154,157]; (2) statistical screening of frequently selected rules via large-scale simulation experiments [145]; and (3) creation of new composite rules through combination or evolutionary evolution of basic rules [89,114,118,205].

4.3.3. OS-Based

The OS-based approach involves the agent not directly selecting the next node to visit at each decision step, but rather choosing which operator to apply to modify the current solution. L2I [61] employed one of the most comprehensive and representative action spaces, treating a curated set of operations research operators, including 2-opt, relocate, and cross, as discrete actions. It treats different parameter values of the same operator as distinct actions, enabling the agent to learn precisely which parameter of which operator to select under specific states. The actions of NLNS [223] were defined by its neural repair mechanism. Given a partially destroyed solution, the repair network decides whether to connect the end of the current route fragment to the end of another fragment or to the depot. DACT [42] selected a pair of nodes, after which a fixed pairwise operator, such as 2-opt, performs the actual transformation. N2S [224] also operated on node pairs, specifically defined as removing a pickup–delivery pair and reinserting them at new positions within the route. This design effectively enforces precedence constraints in the PDP, avoiding violations that commonly arise from traditional operators such as 2-opt. NHC [225] selected operators, including insertion, swap, and 2-opt, conditioned on the problem type. Its distinguishing feature is the encoding of both instance-level and solution-level information into edge features, allowing action selection to account for not only spatial node positions but also semantic relationships captured by edge weights. This enables the model to address problems dominated by edge-based structure, such as PRP, TSP, and GPP, by recommending targeted improvements that reduce evaluation counts and enhance solution quality. NeuOpt [18] frames its action space around k -opt exchanges. Through action factorization, it decomposes traditional k -opt moves into sequences of elementary steps, permitting dynamic determination of k and the combination of operations at varying granularities during search. Its action space includes temporarily infeasible moves, bypassing the limitations of conventional masking mechanisms and encouraging exploration in constraint boundaries, thereby improving convergence and optimization efficiency in complex constrained problems.

The OS-based action generation method significantly compresses the action space, reducing it from a combinatorial scale dependent on node count, $O(N^2)$, to a fixed, manageable size, making learning feasible and efficient. By embedding proven local search operators from operations research, it circumvents the need for the agent to learn basic optimization primitives from scratch, allowing it to focus on higher-level sequencing strategies. Consequently, the learned policy effectively orchestrates these simple operations into complex, globally optimal behaviors that surpass the performance of any single operator or their random combination. However, as the above survey reveals, the design of the action space remains heavily reliant on expert-defined operators. An operator effective for TSP, such as 2-opt, may be ill-suited for problems like PDP [224]. This dependency limits the generalizability and automation potential of neural solvers.

4.4. Reward Design

4.4.1. Sparse Rewards

In routing problems, the optimization objective is typically to minimize the total traveling distance [62]. For scheduling problems, the objective functions, as illustrated in Figure 13, mainly fall into two categories: Makespan and Tardiness. Directly transforming these objective functions into rewards in an MDP may lead to sparse rewards. Recent studies have thus focused on modifying the reward signals themselves, primarily through dense reward design [72,88,108,217], reward shaping [43,115,117,196,226], and HRL [56].

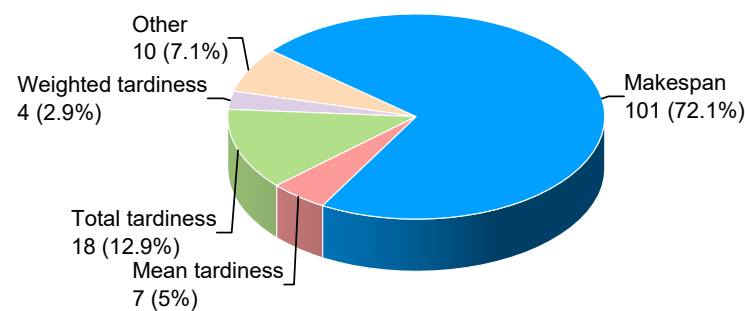


Figure 13. Objective function of scheduling problems.

Dense reward mechanisms convert sparse global objectives into immediate, step-wise rewards, and they have been widely adopted in recent research [93,214]. For instance, multiple studies have defined the reward as the Makespan difference between consecutive states [72,88,108,217]; Wu et al. [49] proposed a reward based on machine idle time and proved its negative linear correlation with Makespan; Xu et al. [105] reformulated Makespan minimization as idle time reduction and designed a stepwise computable reward function. Reward shaping enhances DRL efficiency by embedding prior knowledge into the reward structure [226]. Zhang et al. [43] assigned weighted delay rewards upon job completion to decompose the global objective into agent-specific feedback; Liu et al. [196] backtracked production processes to allocate rewards based on queue time; Liang et al. [117] used the objective function directly with relative performance scaling; Liu et al. [227] incorporated job urgency and slack-time deviation; Liu et al. [115] designed a dual-reward mechanism with distinct discount factors to balance immediate and long-term signals. Muller et al. [228] mapped inventory buffer importance to rewards instead of penalizing idle time, while Lassoued et al. [200] transformed sparse Makespan into locally optimizable machine utilization metrics. HRL mitigates sparsity by decomposing tasks into subtasks with denser intermediate rewards. Li et al. [56] employed a two-level architecture: a high-level agent dynamically switches objectives to generate multi-objective signals, while a low-level agent receives action-aligned immediate rewards.

Current dense reward designs remain largely heuristic and task-specific, limiting generalizability. Future work should focus on automating reward function synthesis via meta-learning or inverse RL and establishing unified evaluation metrics for reward density, stability, and transferability across job shop configurations. Additionally, integrating causal reasoning into reward shaping may improve interpretability and robustness in dynamic environments.

4.4.2. Multi-Objective Optimization

To address the challenges of multi-objective optimization, the existing research primarily employs four categories of strategies: the weighted sum method, decomposition-based methods, hierarchical methods, and Pareto optimization methods. These approaches collectively tackle a core challenge: how to guide agents in making efficient trade-offs

among multiple competing or conflicting objectives. The advantages, disadvantages, and representative literature for each category are systematically compared in Table 3.

Table 3. Classification and literature review comparison table of multi-objective optimization methods.

Type	A & L	Reference List
WS	A: simple, efficient, easy implementation; L: weight selection difficulty, incomplete pareto coverage	SW: [5,45,65,76,85,89,90,93,94,106,120,121,138,154,158,163,175,198,214,228–233] DW: [77,91,143,186]
DB	A: efficient pareto approximation, parallelizable, flexible; L: high computational complexity, subproblem conflicts	WS: [57,58,141,176,187] Tchebycheff: [146,234,235] Hindsight experience replay: [55]
HB	A: avoids conflicts, dynamic adaptation; L: reward design complexity, high collaboration complexity	[46,56,130,131,133,139,144,153,156]
PO	A: direct pareto front, no prior preference; L: high computational complexity, complex design	[150,151,236]

Notation: Abbreviations correspond to the following terms: advantages (A); limitations (L); action (A); weighted sum (W), decomposition-based (D); hierarchical-based (HB); Pareto optimization (PO); static weights (SW); dynamic weights (DW).

Objective conflicts are handled through three primary mechanisms: decomposition-based (DB) methods scalarize multiple objectives via weighted sum $R = \sum_k w_k r_k$ or Tchebycheff aggregation, with varying weight vectors enabling coverage of different Pareto-front regions [57,176]; hierarchical-based (HB) methods decouple objective preference selection from schedule optimization across decision levels [46,56]; and Pareto-based (PB) methods explicitly approximate the Pareto front using multi-policy or population-based training, with hindsight experience replay improving objective-space coverage [55]. Pareto-front coverage is achieved through diverse weight sampling during training (DB), varying preference configurations across episodes (HB), or diversity-preserving mechanisms such as crowding distance (PB); however, uniform coverage remains challenging for non-convex fronts or problems exceeding three objectives.

Current multi-objective optimization methods, while effective in balancing conflicting goals, face three key limitations: (1) weighted sum approaches rely on predefined preferences and lack adaptability in dynamic environments; (2) decomposition-based methods exhibit unstable convergence in high-dimensional spaces with weak inter-subproblem coordination; and (3) hierarchical and Pareto methods incur high computational costs and lack systematic support for real-time responsiveness and interpretability. A key challenge in multi-objective DRL for COPs is ensuring adequate coverage of the Pareto front: the set of non-dominated solutions capturing optimal trade-offs among conflicting objectives. DB methods sample diverse weight vectors during training, teaching the policy to map weight configurations to distinct front regions; at inference, users steer solutions via weight specification. HB methods vary the high-level preference across episodes to build a solution portfolio. PB methods explicitly enforce diversity through crowding-distance selection or multi-population training, preserving spread along the front. Yet uniform coverage remains difficult, particularly for non-convex fronts or problems with more than three objectives, where the requisite weight combinations explode combinatorially.

5. Solving Algorithms

Algorithms for solving COPs typically consist of two components: general-purpose RL algorithms and policy architectures. Following the classification in RL4CO [2] and POMO [78], solvers are categorized into two distinct types: learning to construct (L2C) and learning to improve (L2I). L2C solvers generate complete solutions by sequentially

adding nodes to partial solutions through learned iterative mechanisms. Representative methods include Ptr-Net [237], AM [62], POMO [78], ScheduleNet [59], Pointerformer [84], GLOP [19], MVMoE [21], 2D-Ptr [69], DAN [27], PolyNet [238], PARCO [70], and RouteFinder [22]. These solvers efficiently produce solutions but often yield inferior solution quality compared to L2I methods or classical optimization tools such as LKH and OR-Tools. Moreover, under complex constraints, L2C solvers frequently fail to satisfy all problem constraints during construction [2,24].

In contrast, L2I solvers learn to perform iterative optimization within the solution space by applying local search operators, such as k -opt exchanges, to refine existing solutions. Notable approaches include NeuRewriter [239], N2S [224], NLNS [223], DACT [42], NeuOpt [18], and NCS [240]. While L2I solvers significantly enhance solution quality through iterative refinement, they incur high computational costs due to repeated encoding of solution structures at each step. Furthermore, their final performance and convergence speed are highly sensitive to the quality of the initial solution [2,24].

This section is organized around three complementary dimensions. First, the solver taxonomy is established in Section 5.1 (general-purpose RL algorithms) and Section 5.2 (policy architectures), categorizing methods by construction paradigm (L2C vs. L2I), RL algorithm family, and architecture type. Second, Section 5.3 organizes five core challenges into explicit sub-subsections, namely large-scale optimization, dynamic environments, generalization capability, sample efficiency, and simulation-to-reality transfer, each surveying how different solver types address the same challenge. Third, Section 5.4 provides a visual synthesis of emerging research trends. A temporal development roadmap (Figure 14) illustrates key advances and milestone algorithms:

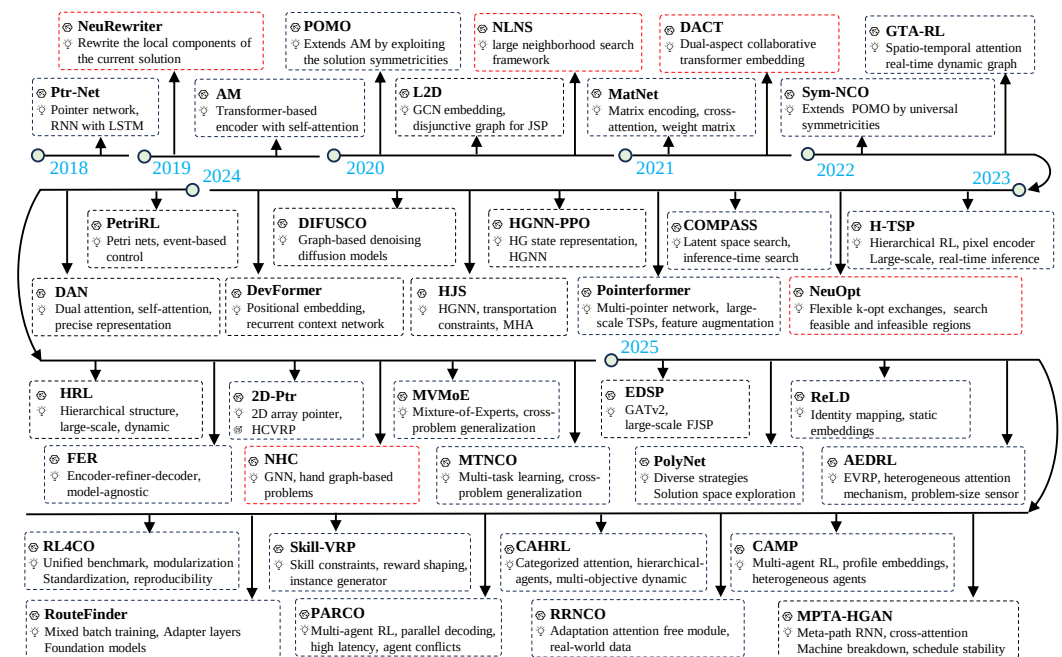


Figure 14. A roadmap of representative solver evolution.

5.1. General-Purpose RL Algorithms

Formally, the RL objective is to learn a policy π_θ that maximizes the expected cumulative reward over the distribution of problem instances $x \sim P(x)$ [2]:

$$\theta^* = \arg \max_{\theta} \mathbb{E}_{x \sim P(x)} \left[\mathbb{E}_{\pi_\theta(a|x)} \left[\sum_{t=0}^{T-1} \gamma^t \mathcal{R}(s_t, a_t) \right] \right]. \quad (2)$$

For policy gradient methods, the REINFORCE estimator with baseline $b(s)$ is [62]:

$$\nabla_{\theta} \mathcal{L}(\theta | s) = \mathbb{E}_{p_{\theta}(\pi|s)}[(L(\pi) - b(s)) \nabla_{\theta} \log p_{\theta}(\pi | s)], \quad (3)$$

where $b(s)$ is a baseline function for variance reduction.

General-purpose RL algorithms are commonly classified into three categories: (1) value-based methods, which indirectly optimize the policy by estimating the value function, such as Deep Q-Network (DQN) [150,158], Double Deep Q-Network (DDQN) [56,205], Dueling Deep Q-Network (Dueling DQN) [195], Dueling Double Deep Q-Network (D3QN) [17,43], Rainbow Deep Q-Network (RDQN) [138], Expected Deep Q-Network (EDQN) [152], and QMIX [99] for multi-agent settings; (2) policy-based methods, which directly learn the policy function, such as PPO [27,50,72], the most widely adopted in scheduling tasks; REINFORCE [62,78], commonly applied in routing problems; Evolution Strategies (ES) [172,176] for parameter optimization; and MAPPO [135,185] for multi-agent systems; and (3) actor-critic methods, which combine value and policy estimation by using a critic to guide policy updates, such as Advantage Actor-Critic (A2C) [109], Asynchronous Advantage Actor-Critic (A3C) [161,167], Soft Actor-Critic (SAC) [88,162,215], and Deep Deterministic Policy Gradient (DDPG) [96,163].

To further analyze research trends in general-purpose RL algorithms for solving COPs, Figure 15 presents a grouped bar chart showing the top three RL algorithms each year. Three key observations emerge: (1) PPO has become the dominant algorithm for COPs, owing to its versatility and training stability; (2) REINFORCE remains widely used in routing problems, largely due to early foundational works such as AM [62] and POMO [78], which integrated REINFORCE with powerful sequence models and established a prevailing research paradigm; (3) DQN has maintained sustained relevance over time, supported by its architecture-compatible design, straightforward implementation, high stability, and scalability.

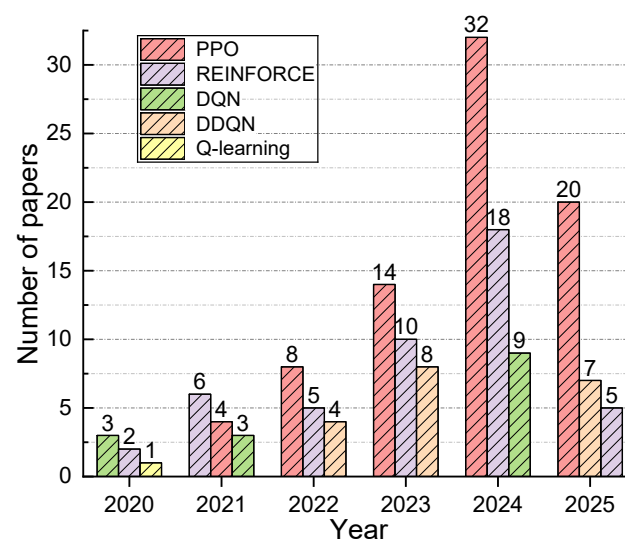


Figure 15. Annual comparison of top three RL algorithms.

5.2. Policy Architectures

Policy architectures parameterize agent behaviors within environments. Their evolution reflects a trajectory from basic designs toward general-purpose foundation models. Representative advances are summarized in Figure 14.

The predominant architectural paradigm for L2C solvers is the encoder-decoder model [62], which defines a stochastic policy $p_{\theta}(\pi | s)$ for constructing a solution $\pi = (\pi_1, \dots, \pi_n)$ given instance s , factorized as

$$p_{\theta}(\pi | s) = \prod_{t=1}^n p_{\theta}(\pi_t | s, \pi_{1:t-1}). \quad (4)$$

The encoder maps d_x -dimensional input features x_i to d_h -dimensional node embeddings $h_i^{(0)} = W^x x_i + b^x$, updated through N attention layers. Each layer consists of multi-head attention (MHA) and feed-forward (FF) sublayers with skip-connections and batch normalization:

$$\hat{h}_i = \text{BN}^{\ell} \left(h_i^{(\ell-1)} + \text{MHA}_i^{\ell} \left(h_1^{(\ell-1)}, \dots, h_n^{(\ell-1)} \right) \right), \quad (5)$$

$$h_i^{(\ell)} = \text{BN}^{\ell} \left(\hat{h}_i + \text{FF}^{\ell}(\hat{h}_i) \right). \quad (6)$$

The graph embedding is $\bar{h}^{(N)} = \frac{1}{n} \sum_{i=1}^n h_i^{(N)}$. The decoder computes per-head queries, keys, and values as $q_{(c)} = W^Q h_{(c)}$, $k_i = W^K h_i$, $v_i = W^V h_i$, with scaled dot-product attention

$$\text{Attention}(Q, K, V) = \text{softmax} \left(\frac{QK^{\top}}{\sqrt{d_k}} \right) V, \quad (7)$$

and multi-head attention $\text{MHA}(Q, K, V) = \text{Concat}(\text{head}_1, \dots, \text{head}_h) W^O$. The pointer mechanism [241] computes logits as $u_i^{(t)} = v^{\top} \tanh(W_1 e_i + W_2 d_t)$, and the output distribution is $p(\pi_t = i | s, \pi_{1:t-1}) = \text{softmax}(u^{(t)})_i$. The composition Decoder $\circ$ Encoder : $\mathcal{X} \rightarrow \Delta(\Pi)$ maps problem instances to distributions over solution permutations.

Early L2C solvers aimed to replace traditional heuristics with DRL-based E2E solvers. Ptr-Net [241] was among the first RL approaches for COPs, using policy gradients to train pointer networks while preserving input sequence invariance. Subsequent work addressed its limitations in dynamic state representations, such as in VRP, by replacing LSTM encoders with element-wise projections, enabling updated embeddings after state transitions [242]. AM [62] significantly advanced this line by adopting Transformer architectures [243], establishing itself as the dominant framework. POMO [78] extended AM by exploiting solution symmetry in TSP and CVRP. Sym-NCO [244] generalized this idea by leveraging shared symmetries across COPs, enabling class-level generalization beyond POMO's problem-specific symmetries. MDAM [79] introduced multi-decoder ensembles, though its performance remained limited by static configurations and node-scale constraints. MatNet [165] replaced self-attention with cross-attention to improve end-to-end performance on asymmetric COPs.

Subsequent research expanded into large-scale instances [19,245], cross-problem learning [60], multi-task modeling [21,22], multi-agent systems [70,246], complex constraints [71], novel applications [68], and improved architectures [247,248], yielding a range of innovative frameworks.

To bridge the solution quality gap between early L2C solvers and classical methods [239,249], L2I solvers employ policy gradients to learn selection strategies for local search operators, such as 2-opt [249]. This approach outperforms handcrafted rules and has been extended to 3-opt [250]. The evolution of L2I solvers centers on expanding action space flexibility: DACT [42] uses dual-aspect attention to guide 2-opt selection; NeuOpt [18] adapts to any $k \geq 2$ and automatically selects optimal k ; NHC [225] introduces a neural heuristic for graph COPs. Collectively, these methods integrate operations research operators with DRL adaptivity, achieving substantial gains in solution quality, efficiency, and scalability.

Representative works are categorized across dimensions in Tables 4 and 5. Below, we organize the review around key challenges as thematic pillars.

Table 4. A summary of the literature on L2C.

Ref	Problem	S	A	Obj	Embedding	Algorithm	Agent	Env	Link	Year
S2V-DQN [251]	MVC, TSP, MAXCUT	Graph	E2E	SO	GNN	Q-learning	SA	Static	Link	2017
PtrNet [237]	TSP	Vector	E2E	SO	LSTM	RF	SA	Static	Link	2018
VRP-RL [242]	VRP	Vector	E2E	SO	MLP	RF	SA	Dynamic	Link	2018
AM [62]	TSP *, VRP *, OP	Vector	E2E	SO	TF	RF	SA	Dynamic	Link	2019
POMO [78]	TSP, CVRP, KP	Vector	E2E	SO	TF	RF	SA	Static	Link	2020
L2D [173]	CJSP	Graph	E2E	SO	GIN	PPO	SA	Static	Link	2020
HCVRP-DRL [252]	HCVRP	Vector	E2E	SO	TF	RF	SA	Static	Link	2021
MDAM [79]	TSP *, VRP *, OP	Vector	E2E	SO	TF	RF	SA	Dynamic	Link	2021
HAM [253]	PDP	Vector	E2E	SO	TF	RF	SA	Static	Link	2021
ScheduleNet [59]	CJSP	Graph	E2E	SO	GAT	RF	MA	Static	Link	2021
MatNet [165]	ATSP, FFSP	Graph	E2E	SO	GAT	RF	SA	Static	Link	2021
GTA-RL [51]	DTSP, DVRP	Graph	E2E	SO	GAT	RF	SA	Dynamic	Link	2022
Sym-NCO [244]	TSP *, CVRP, OP	Vector	E2E	SO	TF	RF	SA	Static	Link	2022
REGAT [82]	TSP, CVRP	Graph	E2E	SO	GAT	PPO + RF	SA	Static	Link	2022
MPGN [177]	FJSP	Graph	E2E	SO	GIN	PPO	SA	Static	Link	2022
DMDDQN [141]	MDFJSP	Vector	DRs	MO	MLP	DDQN	SA	Dynamic	Link	2022
MASA [99]	FFSP	Vector	E2E	SO	RNN	QMIX	MA	Static	Link	2022
MLDRL [58]	MTSP, MVRPTW	Vector	E2E	MO	TF	AC	SA	Static	Link	2023
H-TSP [245]	TSP	Vector	E2E	SO	CNN + TF	PPO + RF	SA	Static	Link	2023
Pointerformer [84]	TSP	Vector	E2E	SO	TF	RF	SA	Static	Link	2023
Zhang et al. [52]	DTSP	Vector	E2E	SO	TF	RF	SA	Dynamic	Link	2023
NHDE [254]	MTSP	Graph	E2E	MO	HGAT	RF	SA	Static	Link	2023
COMPASS [255]	TSP, CVRP, JSP	Vector	E2E	SO	TF	RF	SA	Static	Link	2023
HADRL [256]	EVRP	Vector	E2E	SO	TF	RF	SA	Static	Link	2023
Yuan et al. [25]	CJSP	Graph	E2E	SO	GIN	PPO	SA	Static	Link	2023
Wu et al. [26]	DJSP	Matrix	DRs	SO	CNN	PPO	SA	Dynamic	Link	2023
Song et al. [72]	FJSP	Graph	E2E	SO	HGNN	PPO	SA	Static	Link	2023
HJS [190]	FJSP	Graph	DRs	SO	HGNN	RF	SA	Static	Link	2023
Lin et al. [60]	TSP *, CVRP, OP	Vector	E2E	SO	TF	RF	SA	Static	Link	2024
Wang et al. [63]	VRPB	Vector	E2E	SO	TF	RF	SA	Static	Link	2024
GLOP [19]	TSP *, CVRP	Graph	DRs	SO	TF + GNN	RF	SA	Static	Link	2024
FER [247]	TSP, CVRP	Vector	E2E	SO	TF	A2C	SA	Static	Link	2024
MD-MTA [257]	MDVRP, MDOVRP	Vector	E2E	SO	TF	RF	SA	Static	Link	2024
MVMoE [21]	16 VRPs	Vector	E2E	SO	TF	RF	SA	Static	Link	2024
Iklassov et al. [258]	VRPTW	Vector	E2E	SO	LSTM + MLP	RF	SA	Dynamic	Link	2024
MTNCO [20]	11 VRPs	Vector	E2E	SO	TF	RF	SA	Static	Link	2024
2D-Ptr [69]	HCVRP	Vector	E2E	SO	TF	RF	SA	Static	Link	2024
Wu et al. [49]	DJSP	Matrix	DRs	SO	MLP	PPO	SA	Dynamic	Link	2024
Si et al. [88]	CJSP	Vector	DRs	SO	MLP	SAC	MA	Static	Link	2024
PetriRL [200]	CJSP	Petri net	E2E	SO	MLP	PPO	SA	Static	Link	2024
Monaci et al. [108]	CJSP	Vector	E2E	SO	LSTM	PPO	SA	Static	Link	2024
Wheatley [180]	CJSP	Graph	E2E	SO	GAT	PPO	SA	Static	Link	2024
RS [220]	FJSP	Graph	E2E	SO	HGIN	RF	SA	Static	Link	2024
Hammami et al. [175]	MDJSP	Graph	E2E	MO	GNN	PPO	SA	Dynamic	Link	2024
GRL [214]	MFJSP	Graph	E2E	MO	GNN	PPO	SA	Static	Link	2024
PGRL [176]	MFJSP	Graph	E2E	MO	GIN + MLP	ES	SA	Static	Link	2024
Lei et al. [75]	DFJSP	Graph	E2E	SO	GIN + MLP	DDQN + PPO	MA	Dynamic	Link	2024
E2E-MAPPO [91]	MFJSP	Graph	E2E	MO	MLP	MAPPO	MA	Static	Link	2024
DAN [27]	FJSP	Graph	E2E	SO	GAT	PPO	SA	Static	Link	2024
Luo et al. [259]	CFSP	Vector	E2E	SO	CNN + LSTM	AC	SA	Static	Link	2024
DRQN [94]	MDFSP	Vector	E2E	MO	RNN	DQN	SA	Dynamic	Link	2024
Grumbach et al. [76]	MDFSP	Vector	E2E	MO	MLP	PPO + A2C	SA	Dynamic	Link	2024
RouteFinder [22]	48 VRPs	Vector	E2E	SO	TF	RF	SA	Static	Link	2025
AEDRL [68]	EVRP	Vector	E2E	SO	TF	RF	SA	Static	Link	2025
PARCO [70]	HCVRP, PDP, FFSP	Vector	E2E	SO	TF	RF	MA	Static	Link	2025
Skill-VRP [71]	Skill-VRP	Vector	E2E	SO	TF	RF	SA	Static	Link	2025
PolyNet [238]	TSP, CVRPTW, JSP	Vector	E2E	SO	TF	RF	SA	Static	Link	2025
ReLD [260]	16 VRPs	Vector	E2E	SO	TF	RF	SA	Static	Link	2025
CAMP [246]	PVRP	Vector	E2E	SO	TF	RF	MA	Static	Link	2025
RRNCO [261]	TSP, CVRPTW	Vector	E2E	SO	TF	RF	SA	Static	Link	2025
EDSP [219]	FJSP	Graph	E2E	SO	HGAT	PPO	SA	Static	Link	2025
MMAPPO [45]	MDFJSP	Vector	DRs	MO	MLP	MAPPO	MA	Dynamic	Link	2025
Yuan et al. [198]	MDFJSP	Matrix	E2E	MO	MLP	PPO	SA	Dynamic	Link	2025

Notation: Symbol * indicates that the term encompasses multiple variants of the problem. Abbreviations correspond to the following terms: reference (Ref); state (S); action (A); objectives (Obj), single-objective (SO); multi-objective (MO); single-agent (SA); multi-agent (MA); REINFORCE (RF); transformer (TF).

Table 5. A summary of the literature on L2I.

Ref	Problem	S	A	Obj	Embedding	Algorithm	Agent	Env	Link	Year
NeuRewriter [239]	JSP, CVRP	Vector	OS	SO	LSTM	A2C	SA	Static	Link	2019
L2I [61]	CVRP	Vector	OS	SO	TF	RF	SA	Static	Link	2020
NLNS [223]	CVRP, SDVRP	Vector	OS	SO	MLP	RF	SA	Static	Link	2020
N2OPT [249]	TSP	Graph	OS	SO	GCN	RF	SA	Static	Link	2020
N3OPT [250]	TSP	Graph	OS	SO	GCN	AC	SA	Static	Link	2021
DACT [42]	TSP, CVRP	Vector	OS	SO	TF	PPO	SA	Static	Link	2021
Wu et al. [80]	TSP, CVRP	Vector	OS	SO	TF	AC	SA	Static	Link	2022
N2S [224]	PDP	Vector	OS	SO	TF	PPO	SA	Static	Link	2022
NeuOpt [18]	TSP, CVRP	Vector	OS	SO	TF	PPO	SA	Static	Link	2023
NHC [225]	TSP, PRP, GPP	Graph	OS	SO	GNN	RF	SA	Static	Link	2024
NCS [240]	PDP	Vector	OS	SO	TF	PPO	SA	Static	Link	2024
Zhang et al. [262]	CJSP	Graph	OS	SO	GNN	RF	SA	Static	Link	2024

Notation: Abbreviations correspond to the following terms: reference (Ref); state (S); action (A); objectives (Obj), single-objective (SO); environment (Env); single-agent (SA); REINFORCE (RF); transformer (TF).

5.3. Key Challenges and Solution Strategies

5.3.1. Large-Scale Optimization

L2C and L2I solvers face challenges in real-time processing of large-scale instances due to the combined effect of quadratic growth in neural network complexity and the expansion of the MDP's state–action space [24]. To mitigate these costs, several studies have adopted efficient network architectures [47,81,84,206]. Lin et al. [206] proposed a stochastic policy parameterized by an attention-based model that integrates pointer networks with graph embedding layers. Trained via policy gradient with a roll-out baseline, this model demonstrated efficient solving of large-scale EVRPTW instances beyond the capacity of the existing methods. Building on this work, Pointerformer [84] incorporated a reversible residual network in the encoder and a multi-pointer decoder to limit memory consumption, thereby improving scalability for large-scale TSPs. Other approaches decomposed large-scale problems into smaller subproblems. H-TSP [245] employed a spatially structured HRL framework to scale E2E DRL to problems with tens of thousands of nodes, achieving a two-order-of-magnitude speedup over search-based methods. Lei et al. [75] applied an HRL architecture to large-scale DFJSP, where a high-level agent partitioned the problem into static FJSP subproblems of varying scales, and two lower-level agents solved these subproblems. GLOP [19] decomposed routing problems into sub-TSPs, further splitting each TSP into Shortest Hamiltonian Path Problems. It combined non-autoregressive and autoregressive neural heuristics, and extensive evaluations on large-scale TSP, ATSP, CVRP, and PCTSP instances demonstrated its competitive, SOTA real-time performance. However, unlike some L2I algorithms, GLOP cannot continuously improve solution quality with additional computation time.

Most architectures still rely on computationally heavy components such as Transformers, limiting their scalability to large-scale dynamic problems. Therefore, future research could pursue two directions: replacing the widely used Transformer in L2C solvers with lightweight neural architectures, and integrating L2C and L2I approaches. One promising strategy is to leverage L2C methods for rapid initial solution generation and apply L2I techniques for iterative refinement, thereby balancing computational speed with solution quality to better suit large-scale real-time applications. Regarding the curse of dimensionality, beyond the state and action space optimization methods discussed in Section 4, architectural approaches such as MARL have been employed to decompose complex scheduling problems into decoupled subproblems, mitigating dimensionality challenges arising from interdependent decision-making in large-scale systems [44,99,101,102,104,147,164,263,264].

5.3.2. Dynamic Environments

Dynamic events can alter system states in real time. In routing problems, typical dynamic events include: changes in traffic congestion [51,64,66,67], updates in customer demand [3,53,54,65,258,265], and stochastic prizes [62,79]. In scheduling problems, common dynamic events encompass: random job arrivals [17,26,50,107,117,134–136,166,209,266], machine breakdowns [28,50,98,120,130,135,136,197,209,210,230,267], variable processing times [17,49,109,117,196,264], due date variability [43,101,114,198], transportation time fluctuations [44,102,118,138], reordering of jobs or operations [138,197,230], job rework [118,135,147], and urgent job insertions [5,188].

One strategy involves adopting more effective state representation methods. As illustrated in Figure 16, we systematically summarize the state representation mechanisms employed in existing studies under dynamic event triggers. Vector-based methods typically represent dynamic environments using fixed-dimensional statistical features or global counts. When dynamic events occur, only feature values are updated, while the dimensionality of the state representation remains unchanged. This approach offers structural simplicity, computational efficiency, and strong generalization, making it the most widely adopted strategy for dynamic environments. Common neural architectures for extracting higher-level features include MLP [137,268], LSTM [136], Bi-LSTM [129], and Transformers [115]. Graph-based methods effectively model complex constraints and dependencies among entities. Under dynamic events, graph structures allow efficient state updates via local modifications, such as adding, removing, or altering nodes and edges. The size-agnostic nature of GNNs enables them to handle inputs of arbitrary scale, ensuring dimension invariance. Common graph representation frameworks include dynamic DG [41,50,168,170–172,175,209], SGD [166], HG [44,98,188,191,230], and DAG [9,163]. Popular GNN variants for embedding learning include GCN [163,170,230,269], GAT [50,168,183], and GIN [166,172,175,270]. When representing dynamic states as 2D [49,195,196] or 3D matrices [26,56,197,198,210,266], two approaches address disruptions from new job insertions. The first dynamically expands the matrix dimensions to accommodate increasing job counts [26,266]. The second fixes the matrix dimensions to the maximum anticipated number of jobs and operations, applying zero padding for insufficient entries and updating values in place when new jobs arrive [56,198].

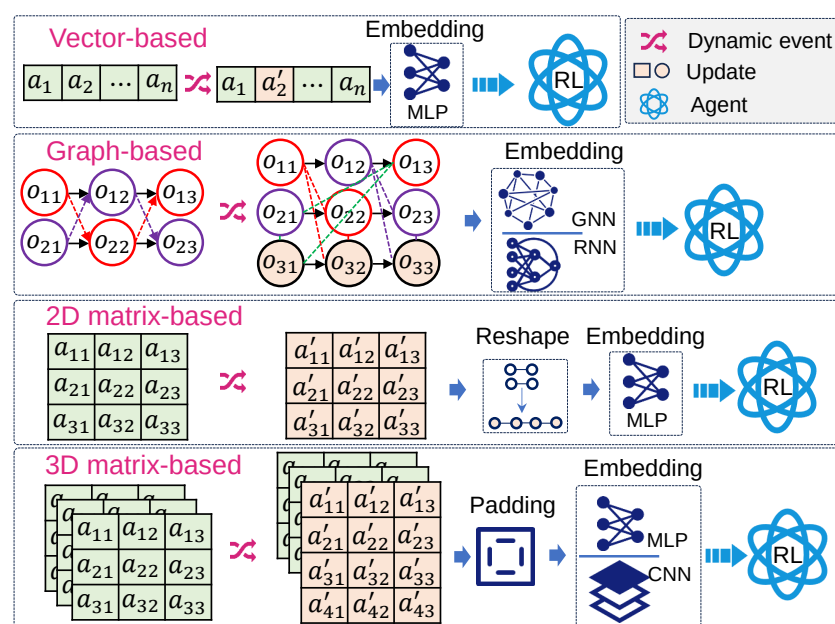


Figure 16. Typical processing patterns of different state representations after dynamic event triggering.

The second category comprises methods employing hierarchical or MARL architectures. In the context of the DFJSP, HRL [75] established a hierarchical framework to effectively manage large-scale scheduling under stochastic job arrivals. For dynamic event handling, MARL approaches commonly adopt a centralized training with decentralized execution (CTDE) framework [17,101,118,147,188,196,197,271], offering rapid response, strong fault tolerance, and operational robustness. Zhang et al. [17] enabled centralized optimization and decentralized execution through cloud-edge collaboration. Li et al. [118] decomposed the complex scheduling problem into three sub-tasks—task selection, machine assignment, and AGV scheduling—and assigned a dedicated agent to each, allowing localized decision-making. Within heterogeneous MARL frameworks, Zhang et al. [43] designed job-specific and machine-specific agents, integrating reward shaping, DRL algorithms, and dynamic adaptation mechanisms to deliver an efficient solution for complex shop floor scheduling under dynamic conditions.

While the existing approaches have proposed diverse and effective solutions from architectural and representation perspectives, they are typically trained and optimized under predefined perturbation distributions and are often restricted to specific dynamic events. To address this gap, meta-learning and transfer learning offer promising avenues. These paradigms enable models to learn not only how to solve tasks but also how to adapt their learning strategies, facilitating rapid generalization to novel perturbations with minimal samples or prior experience.

5.3.3. Generalization Capability

Regarding generalization across problem scales, many methods have enhanced generalization capabilities by designing effective model architectures. Att-GCN + MCTS [272] proposed a hybrid approach combining heatmap generation with Monte Carlo Tree Search (MCTS) to enable effective scaling of small-scale pretrained models to large-scale instances. HCVRP-DRL [252] designed a vehicle selection decoder for handling heterogeneous fleet constraints and a node selection decoder for route construction in the context of the heterogeneous capacitated vehicle routing problem (HCVRP), achieving robust generalization across various problem scales. DACT [42] adopted a dual-aspect collaborative transformer architecture to separately embed node features and positional features. FER [247] introduced an innovative encoder–optimizer–decoder framework that not only iteratively optimized feature embeddings to expand the search space but also dynamically updated probability distributions to enable more diverse exploration. Some studies have enhanced generalization through novel training methodologies: Sym-NCO [244] implemented a regularization-based training scheme leveraging common symmetries in various COPs and their solutions; 2D-Ptr [69] simultaneously selected vehicle–node pairs as actions, with its input dimensions decoupled from the number of vehicles or customers. It dynamically fused vehicle and node information through self-attention and cross-attention modules, thereby flexibly adapting to varying fleet sizes or customer counts. EL-DRL [83] addressed distribution shift limitations by adopting an ensemble-based DRL approach that learned a set of diversified sub-policies to handle different instance distributions. RIDGE [273] proposed a knowledge-inspired REINFORCE algorithm, designing an adaptive baseline-guided gradient estimator to approximate the theoretical optimal bound. ReLD [260] addressed the limitations of lightweight decoders in cross-scale generalization by enhancing the decoder architecture.

In the context of cross-problem-type generalization, COMPASS [255] introduced latent space search, enabling rapid identification of high-performance policy regions for new instances during inference via evolutionary strategies, thereby achieving policy adaptation without retraining. MTNCO [20] framed diverse routing problems as combina-

tions of shared base attributes, allowing a single model to solve multiple VRP variants simultaneously through attribute composition. Meanwhile, MVMoE [21] improved zero-shot generalization to unseen VRP variants by adopting a mixture-of-experts architecture. RouteFinder [22] adopted a framework similar to MTNCO [20], treating each VRP variant as a subset of a broader VRP problem with distinct attributes, and it achieved cross-task generalization through global feature embeddings and lightweight adapter layers.

Whether in cross-scale or cross-problem-type generalization, feature representation directly determines an agent's capacity to understand and abstract state information, serving as a core determinant of generalization performance. L2D [173] represented the state of JSP instances using domain graphs and employs GCNs to extract structural features, enabling a size-agnostic policy that generalizes effectively across large-scale instances. ScheduleNet [59] modeled the problem state as an agent-task graph and used type-aware graph attention to generate node embeddings, achieving strong generalization across diverse multi-agent scheduling tasks with varying problem sizes. HGNN-PPO [72] formulated FJSP instances as hypergraphs and decoupled state representation from decision-making via a hypergraph neural network. HJS [190] extended this modeling framework to incorporate transportation constraints, better reflecting real-world manufacturing environments. DAN [27] improved decision interpretability by separately modeling operation sequences and machine contention through a dual attention mechanism. PetriRL [200] enhanced interpretability using a Petri net-based graphical representation. EDSP [219] introduced diverse policy generation alongside a novel graph representation, outperforming both HGNN-PPO [72] and DAN [27] on large-scale instances.

These methods have collectively advanced generalization in combinatorial optimization by integrating structured representations, adaptive architectures, and dynamic learning mechanisms, ranging from transformer-based feature disentanglement to symmetry-aware training and mixture-of-experts frameworks. Yet, their success remains largely empirical, with limited theoretical grounding in how feature embeddings encode problem invariances across scales and types. Future work should establish formal connections between architectural design, representation invariance, and generalization bounds in combinatorial search spaces.

5.3.4. Sample Efficiency

To improve sample efficiency, the existing methods primarily rely on data augmentation [78,244,260], curriculum learning [42], and post-processing [18,61,224]. POMO [78] employs symmetric coordinate transformation for data augmentation; Sym-NCO [244] introduces orthogonal rotation; and ReLD [260] expands this framework to four transformations: rotation, reflection, scaling, and noise. DACT [42] uses curriculum learning to progressively supply higher-quality initial solutions during training, replacing random initialization and improving sample efficiency. For post-processing, L2I-type solvers [18,61,224] enhance solution quality via iterative refinement.

Among these, data augmentation is compatible with other methods; L2I-type approaches trade computation time for improved solution quality; and most techniques depend on specific problem structures and training distributions. Future work should prioritize lightweight adaptive architectures and more efficient data generation mechanisms.

5.3.5. Simulation-to-Reality Transfer

To bridge the gap between simulation and deployment, D-EVRP-SVR-SP [67] evaluated an electric vehicle routing algorithm using real vehicle parameters in actual geographical settings, quantifying its deployment potential and demonstrating adaptability to real-world uncertainties. RGCMA [47] conducted algorithm testing and proof-of-concept

experiments within the Beijing distribution network of JD Logistics. DRL-DT [48] developed a transfer learning framework based on digital twin technology to enable seamless transition from virtual simulations to physical production lines. GEN-DRL [179] constructed a virtual hanging workshop environment using digital twin technology and validated its algorithms with synthetic data and public benchmarks. In these studies, trained models operated exclusively within digital twin environments, with decision-making confined to simulation validation rather than physical execution.

Other studies have employed offline simulation frameworks incorporating real production data, including applications in automotive engine manufacturing [166], assembly and debugging workshops [135], semiconductor fabrication [274], prototype machining [214], intelligent gear production [131], automotive subframe processing [99], aerospace component manufacturing [46], and steel production [259]. To better reflect real-world conditions, HCVRP-DRL [252] incorporated heterogeneous vehicle constraints. MD-MTA [257] and SVRP [258] extended DRL methodologies to complex COPs by integrating multi-depot and skill constraints. Skill-VRP [71] explicitly modeled skill requirements in vehicle routing. RRNCO [261] first introduced asymmetric real-world travel distances and times into training frameworks via context-gated mechanisms and neural adaptive bias structures.

Despite extensive efforts in leveraging real-world datasets, scaling to large problems, and modeling complex constraints, deployment of these models in live production environments remains rare. Future work must prioritize the transition of trained models into operational systems and enable their continuous learning and adaptation.

5.4. Visual Analysis

We employed Sankey diagrams to visualize flow relationships and compositional preferences among solvers across multiple dimensions, as shown in Figure 17. This visualization revealed research trends, algorithmic preferences, and the alignment between problem characteristics, environmental settings, and solver architectures. By examining Figure 17 in conjunction with Tables 4 and 5, we identified the following key findings: (1) Routing and scheduling problems have attracted sustained research attention, with multi-task COPs and universal solver design emerging as recent focal points [20–22]; (2) L2C solvers dominate current approaches, with vector features paired with Transformer encoders or graph features combined with GNN encoders now standard for feature representation; (3) REINFORCE is predominantly used for routing problems, while PPO is favored in scheduling tasks; (4) E2E action generation methods significantly outnumber DRs-based approaches. These results suggest that, despite their interpretability and controlled action spaces, DR-based methods are increasingly outperformed by E2E approaches in solution quality, particularly in complex dynamic environments, where E2E methods achieve superior performance without requiring expert-crafted heuristics.

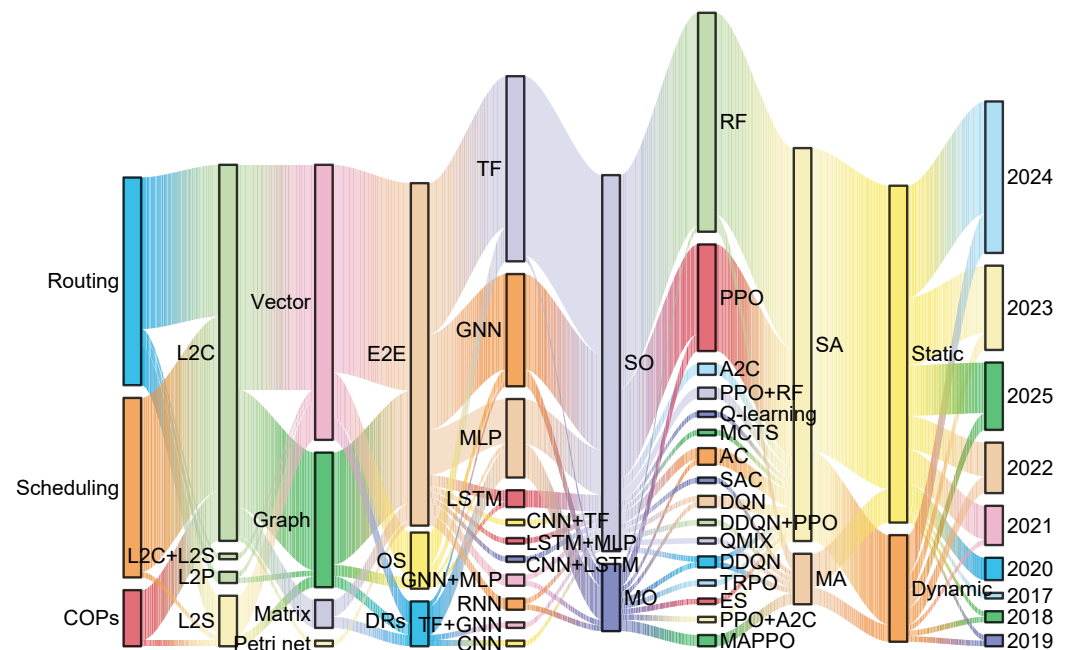


Figure 17. Sankey diagram of multi-level taxonomy for scheduling problem. Abbreviations correspond to the following terms: single-objective (SO); multi-objective (MO); single-agent (SA); multi-agent (MA); REINFORCE (RF); Transformer (TF).

6. Performance on Benchmark Datasets

6.1. Benchmark Datasets

Standardized benchmarks play a pivotal role in enabling fair comparisons and advancing research. Table 6 summarizes 17 widely adopted benchmark datasets for routing and scheduling problems, categorized into three main types: the VRP and its variants, the JSP, and the FJSP. The table also lists representative references that explicitly used these datasets and reported experimental results within the scope of this review. Collectively, these 17 datasets form the mainstream benchmarks for evaluating DRL approaches to COPs. Their varying scales and problem complexities provide a robust foundation for objectively assessing algorithmic generalization, solving efficiency, and solution optimality.

Table 6. Benchmark datasets.

Dataset	Problem	Reference List
TSPLIB [275]	TSP	[42,57,58,80,82–84,244,247,273,276,277]
CVRPLIB [278]	CVRP	[20–22,42,47,60,80,82,83,247,252,277,279,280]
Solomon [281]	CVRP	[20,21,65]
TSP/CVRP [62]	TSP, CVRP	[13,18,20–22,42,60–62,78,79,82,84,223,237–239,244,247,249–251,255,260,277,282]
ABZ [283]	JSP	[25,59,98,110,172,208,220,262,284]
FT [285]	JSP	[25,26,44,59,109,110,160,172,197,208–210,220,262,284]
ORB [286]	JSP	[25,59,98,109,110,149,172,197,208,210,220,262,284]
YN [287]	JSP	[25,59,110,172,208,220,262,284]
LA [288]	JSP	[25,26,49,59,98,109,111,149,160,172,197,208–210,220,262]
SWV [289]	JSP	[25,59,110,172,208,220,262,284]
TA [290]	JSP	[6,25,26,49,50,55,59,73,88,110,160,162,167,172–174,178,181,200,207,208,220,236,259,262,284,291,292]
DMU [293]	JSP	[6,25,50,55,88,110,128,149,162,172,173,292]
Hurink [294]	FJSP	[7,27,72,75,105,155,163,176,177,179,182,184–187,189,204,213,215,217–220,222,263,295]
MK [296]	FJSP	[7,27,72,92,98,105,121,163,169,176,179,182–187,189,192,204,212,213,215–218,220,222,230,295]
Behnke [297]	FJSP	[177,215,219,263]
Dauzere [298]	FJSP	[105,216,222]
Barnes [299]	FJSP	[216,222]

6.2. Benchmark Results

Tables 7–10 summarize experimental results on mainstream routing and scheduling benchmark datasets. For routing problems, Table 10 reports average path lengths and total testing time; for scheduling problems, average Makespan is reported. Detailed results are provided in the supplementary materials. In Table 7, CVRPLIB(A) and CVRPLIB(B) denote the Set-X(n101–n200) and Set-X(n502–n1001) subsets of the CVRPLIB dataset, respectively. The existing studies typically report results on only partial instances, hindering cross-literature comparisons on shared benchmarks. Our analysis reveals the following key observations:

- (1) Benchmark datasets exhibit substantial diversity: scheduling problems encompass a wider variety of dataset types than routing problems. Many studies evaluate only limited subsets; comprehensive evaluations across all instances within specific domains remain rare.
- (2) The key elements of the MDP model determine optimization performance. Graph-based state representation has become the dominant approach for JSP and FJSP (Table 4). The GNN feature embedding architecture directly affects solution quality. ScheduleNet [59] introduced type-aware graph attention with differentiated parameters and type-specific message aggregation for multi-type HGs, achieving SOTA on ABZ, ORB, YN, and TA datasets. Improper reward functions can reduce training efficiency and cause premature convergence. PBMP [160] used preference learning to develop a reward predictor, achieving performance matching the upper bound on the FT dataset. GMAS [163] formulated scheduling as DAG connection probability prediction with GCN-based feature extraction, achieving optimal results on the Hurink and Brandimarte datasets. Xu et al. [105] employed a transformer-based state encoder, achieving a best average Makespan of 2292.5 on the Dautzère dataset.
- (3) Exact and heuristic solvers find optimal solutions in feasible time for small-scale instances, but solution time increases sharply with problem size and complexity (Table 10).
- (4) L2I solvers generally outperform L2C methods in solution quality (e.g., L2I [61] and NeuOpt [18] on CVRP with $N = 100$), but require longer inference times. Recent L2C solvers achieve comparable results in seconds, including MTNCO [20], MV-MoE/4E [21], RF-POMO [22], and ReLD-MoEL [260]. For small-scale TSP and CVRP, the performance gap between leading NCO and classical OR solvers has narrowed to negligible levels. Recent research has shifted focus to improving generalization, addressing large-scale problems, and adapting to complex variants.

Table 7. SOTA results on public VRP benchmark datasets.

Ref	TSPLIB	CVRPLIB(A)	CVRPLIB(B)	Solomon
AM [62]	59,012	37,606	-	-
POMO [78]	32,283	30,848	131,554	1474
Wu et al. [80]	31,652	-	-	-
FER [247]	31,981	28,510	-	-
MVMoE [21]	-	28,679	116,627	1364
RF-POMO [22]	-	-	116,468	1398
MTNCO [20]	-	30,302	117,447	-
RGCMA [47]	-	29,590	-	-
Opt.	30,479	27,220	101,874	1170

Table 8. SOTA results on public JSP benchmark datasets.

Ref	ABZ	FT	ORB	YN	LA	SWV	TA	DMU
Park et al. [208]	994.8	850.3	1100.9	1139.0	1258.7	2711.3	2825.1	-
Yuan et al. [25]	957.2	831.3	1093.9	1098.3	1225.6	2653.0	2750.9	5933.3
L2D [173]	-	-	-	-	-	-	2939.8	6463.8
ScheduleNet [59]	945.4	889.3	1087.0	1081.5	1212.4	2865.8	2699.3	-
Han et al. [284]	-	747.7	-	-	1163.8	-	-	-
Zhang et al. [262]	-	723.7	-	-	-	-	-	-
PBMP [160]	-	716.7	-	-	-	-	-	-
UB	835.2	716.7	902.8	912.0	1107.3	2236.8	2366.6	4623.6

Table 9. SOTA results on public FJSP benchmark datasets.

Ref	Hurink			Brandimarte	Dauzere	Behnke	Barnes
	edata	rdata	vdata				
Song et al. [72]	1116.33	986.10	931.45	191.30	2414.89	363.53	1172.67
MPGN [177]	1260.95	1081.45	951.85	187.60	2566.06	279.27	1284.76
Yuan et al. [222]	1187.93	1041.83	963.50	189.10	2463.22	-	1133.62
GMAS [163]	1024.65	932.25	919.53	172.30	-	-	-
DAN [27]	1119.73	978.28	925.40	184.30	2412.94	278.04	-
ME-GNN [213]	1115.23	988.28	925.20	190.70	-	-	-
Liu et al. [218]	1113.28	977.90	924.78	178.30	-	-	-
GGCT [295]	1116.78	987.38	930.53	191.20	-	-	-
Xu et al. [105]	1087.58	973.10	926.70	180.10	2292.50	-	-
RS [220]	1101.48	976.20	926.50	179.00	-	-	-
Tang et al. [189]	1212.40	1061.28	963.73	197.50	-	-	-
DDAN [184]	1112.23	980.68	925.26	179.00	-	-	-
MCGA-PPO [217]	1115.05	987.60	931.35	191.00	-	-	-
GEN-DRL [179]	1171.75	1039.10	951.05	184.20	-	-	-
CARL [182]	-	-	-	186.60	2513.50	-	-
MH-RHN [216]	-	-	-	181.00	2389.83	-	1091.57
EDSP [219]	-	-	934.00	-	-	268.82	-
UB	1028.88	934.28	919.50	172.50	2211.44	271.00	997.00

Table 10. SOTA results on public TSP/CVRP benchmark datasets.

Methods	Type	TSP (N = 20)		TSP (N = 50)		TSP (N = 100)		CVRP (N = 20)		CVRP (N = 50)		CVRP (N = 100)	
		Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
Concorde (+ [277])	ES	3.83	5 m	5.69	13 m	7.76	1 h	-	-	-	-	-	-
LKH (+ [277])	HS	3.83	43 s	5.69	6 m	7.76	26 m	6.10	2 h	10.38	7 h	15.65	14 h
Gurobi (+ [277])	HS	3.83	9 s	5.69	3 m	7.76	18 m	6.10	1 m	-	-	-	-
OR Tools (+ [277])	HS	3.83	1 m	5.85	5 m	8.07	24 m	6.46	2 m	11.28	14 m	17.11	48 m
GA (+ [13])	HS	3.86	2 h	5.96	8 h	8.33	31 h	-	-	-	-	-	-
Ptr-Net (+ [62]) (G) [237]	L2C	3.89	-	5.95	-	8.3	-	-	-	-	-	-	-
Ptr-Net (+ [62]) (S-1280) [237]	L2C	-	-	5.75	-	8.00	-	-	-	-	-	-	-
S2V-DQN (G) [251]	L2C	-	-	-	-	8.31	-	-	-	-	-	-	-
AM (G) [62]	L2C	3.85	0s	5.80	2 s	8.12	6 s	6.40	1 s	10.98	3 s	16.80	8 s
AM (S-1280) [62]	L2C	3.84	5 m	5.73	24 m	7.94	1 h	6.25	6 m	10.62	28 m	16.23	2 h
N2OPT [249]	L2I	3.84	15 m	5.70	29 m	7.83	41 m	-	-	-	-	-	-
N3OPT [250]	L2I	3.84	32 m	5.70	48 m	7.82	80 m	-	-	-	-	-	-
POMO (Aug-8) [78]	L2C	3.83	3 s	5.69	16 s	7.77	1 m	6.14	5 s	10.42	26 s	15.73	2 m
NeuRewriter (+ [78]) [239]	L2I	-	-	-	-	-	-	6.16	22 m	10.51	18 m	16.10	1 h
NLNS (+ [78]) [223]	L2I	-	-	-	-	-	-	6.19	7 m	10.54	24 m	15.99	1 h
L2I (+ [78]) [61]	L2I	-	-	-	-	-	-	6.12	12 m	10.35	17 m	15.57	24 m
Wu et al. [80]	L2I	3.83	1 h	5.70	1.5 h	7.87	2 h	6.12	2 h	10.45	4 h	16.03	5 h
Sym-NCO (G) [244]	L2C	-	-	-	-	7.84	2 s	-	-	-	-	16.10	3 s
Sym-NCO (S-100) [244]	L2C	-	-	-	-	7.79	13 s	-	-	-	-	15.87	16 s
Pointerformer [84]	L2C	3.83	6s	5.69	12 s	7.77	6 s	-	-	-	-	-	-
MDAM (B-250) [79]	L2C	3.84	3 m	5.70	14 m	7.79	44 m	6.14	5 m	10.48	15 m	15.99	53 m
COMPASS [255]	L2C	-	-	-	-	7.77	2 h	-	-	-	-	15.59	4 h
DACT (+ [18]) [42]	L2I	3.83	1.5h	5.69	4.1 h	7.77	13.5 h	6.13	4 h	10.38	16 h	15.74	1.7 d

Table 10. Cont.

Methods	Type	TSP (N = 20)		TSP (N = 50)		TSP (N = 100)		CVRP (N = 20)		CVRP (N = 50)		CVRP (N = 100)	
		Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time	Obj.	Time
NeuOpt [18]	L2I	3.83	1 h	5.69	2.7 h	7.77	7 h	6.13	13.7 h	10.37	1.6 d	15.58	3.8 d
EAS (Aug-8, S-200) (+ [18]) [282]	L2C	3.83	24 m	5.70	2 h	7.77	10.9 h	6.13	38.00	10.38	3.1 h	15.61	16 h
G-DGNet (G) [13]	L2C	3.83	4 s	5.75	13 s	8.02	30 s	-	-	-	-	-	-
G-DGNet (S-1280) [13]	L2C	3.83	12 m	5.72	1 h	7.79	2.15 h	-	-	-	-	-	-
REGAT (G) [82]	L2C	3.83	1 s	5.76	4 s	8.04	11 s	-	-	-	-	-	-
REGAT (S-1280) [82]	L2C	3.83	10 m	5.74	55 m	7.90	2 h	-	-	-	-	-	-
FER-POMO (Aug-8, S-200) [247]	L2C	3.84	24 m	5.70	1.8 h	7.77	8.5 h	6.13	28 m	10.38	3 h	15.65	14 h
MTNCO [20]	L2C	-	-	-	-	-	-	-	-	10.56	3 s	15.90	11 s
MTNCO (Aug-8) [20]	L2C	-	-	-	-	-	-	-	-	10.47	20 s	15.80	1.2 m
MTNCO + SGBS [20]	L2C	-	-	-	-	-	-	-	-	10.40	2.3 m	15.66	12.6 m
MVMoE/4E [21]	L2C	-	-	-	-	-	-	-	-	10.43	4 s	15.76	11 s
Liu et al. [276]	L2C	3.83	1 m	5.70	20.4 m	7.80	1 h	-	-	-	-	-	-
SAT [277]	L2C	3.83	2 m	5.69	6 m	7.81	16 m	6.12	2 m	10.43	8 m	15.77	18 m
RF-POMO [22]	L2C	-	-	-	-	-	-	-	-	10.44	2 s	15.84	10 s
RF-MoE-L [22]	L2C	-	-	-	-	-	-	-	-	10.42	2 s	15.82	12 s
PolyNet [238]	L2C	-	-	-	-	7.77	4 m	-	-	-	-	15.64	5 m
PolyNet-EAS [238]	L2C	-	-	-	-	7.77	3 h	-	-	-	-	15.58	4 h
ReLD-MoEL [260]	L2C	-	-	-	-	-	-	-	-	-	-	15.71	10 s

Notation: Symbol † indicates that the result is taken from the subsequently given reference. Abbreviations correspond to the following terms: the number of augmentations (Aug); the number of samplings (S); the beam search width (B); greedy search (G); exact solver (ES); heuristic solver (HS).

7. Open Research Directions

Building upon a systematic analysis of technical challenges, this study reviews current DRL approaches for solving COPs. It identifies core challenges in the field and outlines future research directions across six key dimensions: fundamental modeling (state and action space design), standardized benchmark platform development, application domain expansion, COP family extension, and synergistic integration with large language models (LLMs).

A. Automated and intelligent feature extraction with efficient representation.

Current state representations in DRL-based methods predominantly rely on hand-crafted features, as detailed in Section 4.2. The design of these features depends heavily on expert knowledge, with their effectiveness, completeness, and embedding strategies significantly influencing both the learning efficiency and solution quality of DRL agents. For example, [72] extracted 10 features from graph structures comprising operation nodes, machine nodes, and arcs, while DAN [27] extended the feature set to 26 features using an HGNN and a dual attention mechanism, respectively. Performance evaluations on three subsets of the Hurink benchmark dataset [294] revealed the distinct strengths and limitations of these approaches [27]. This raises fundamental questions: which features are most informative, how many are sufficient, and which embedding model yields the most effective representation? Future research should prioritize automated feature learning by developing methods that directly derive low-dimensional, information-rich state embeddings from raw or semi-structured data to enable intelligent and efficient feature extraction.

B. More effective action space design methods.

In Section 4.3, we discuss action generation approaches. E2E methods may enable models to discover optimal solutions but often lead to combinatorial explosion in the action space, reducing learning efficiency [27,72,91,184,189,198,200,214,216–220,222]. DRs approaches rely on expert-designed rules, which may constrain solution quality. Although rule synthesis [89,114,118] and genetic programming (GP) [205] can automatically generate scheduling rules, they introduce additional computational overhead and increase algorithmic complexity. HRL [75,166] reduces action space dimensionality by decoupling composite actions, but it only optimizes the structural design of the action space. Future work should focus on designing action spaces that are scalable, learnable, and capable of generating

feasible and efficient schedules without compromising solution quality. Potential directions include developing adaptive action space frameworks with integrated constraint-handling mechanisms and exploring deep model-based approaches for action generation.

C. Establishing a fair, unified, and efficient evaluation framework.

When reviewing SOTA results on benchmark datasets, we found that only 38.4% of studies evaluated models on widely adopted benchmarks. The majority of researchers instead relied on custom-built simulation environments and randomly generated synthetic data. Furthermore, substantial variations in training protocols and evaluation criteria hinder direct comparison and reproducibility. For instance, among 190 papers, over 90 distinct performance metrics were defined and applied, many of which exhibit redundancy and inconsistent definitions. Metrics quantifying deviation from optimal solutions alone encompass more than 20 variant terminologies, including gap [88,162], gap percentage [75,169], relative percentage increase (RPI) [158], and relative percentage deviation (RPD) [25,182]. In terms of benchmark infrastructure, RL4CO [2] offers a standardized, reproducible, and comprehensive platform for evaluating DRL solutions to COPs. Future efforts should focus on expanding this open-source framework by establishing diverse COP simulation environments, standardizing input/output interfaces, and supporting large-scale parallel evaluation.

D. Expanding the application scenarios of traditional COPs.

Taking scheduling as an example, current applications of the two major problem models remain largely confined to classical manufacturing environments, including automotive engine manufacturing [166], assembly and calibration workshops [135], semiconductor fabrication facilities [274], prototype development workshops for mechanical components [214], intelligent gear production lines [131], and subframe machining workshops [99]. Among the reviewed studies, only [8] applied the FJSP framework to virtual network function mapping and scheduling in network function virtualization environments. As a classical combinatorial optimization model, the JSP and its variants exhibit greater potential when integrated with DRL algorithms. Future research should extend these models beyond traditional manufacturing contexts to domains such as logistics and transportation scheduling, cloud computing and data center resource allocation, energy system scheduling, and communication network scheduling, by fully leveraging their capacity to model resource constraints and sequence dependencies.

E. Extending DRL to under-explored COP families.

While this review focuses on routing and scheduling, which together account for over 85% of DRL-for-COPs publications, several important COP families remain comparatively under-explored. These include graph coloring, bin packing, knapsack, and facility location, for which the RL4CO benchmark [2] already provides standardized environments but the volume of published research remains limited. Dedicated surveys and systematic benchmarking studies targeting these COP families would be valuable contributions, as would the development of general-purpose DRL solvers capable of transferring knowledge across diverse COP types.

F. Exploring novel research paradigms for COPs.

The recent emergence of LLMs with remarkable capabilities across multiple domains has opened new research avenues for intelligent scheduling. Several exploratory studies have begun to explore this direction. For example, Starjob [300] addressed the high scenario dependency and reliance on expert knowledge in heuristic dispatching rules for the DiJSP by automatically generating such rules through LLMs combined with population-based self-evolution. In algorithm design, Feng et al. [301] employed LLMs to automatically construct or compose existing metaheuristic algorithms for solving the FJSP, using LLMs' capabilities in code generation and iterative refinement to enhance algorithm performance. REMoH [302] and HeurAgenix [303] developed LLM-based frameworks for generating and

adapting heuristic rules, dynamically selecting rules during problem resolution to improve solution quality and model adaptability. Future DRL frameworks could fully leverage LLMs' abilities in natural language understanding, code generation, and reasoning to advance state representation, feature extraction, routing and scheduling rule generation, reward function design, transfer learning, and simulation scenario synthesis, offering innovative solutions to systemic challenges in intelligent scheduling.

8. Conclusions

This paper systematically evaluates DRL-based methods for solving COPs from a challenge-oriented perspective, addressing gaps left by prior surveys: (1) We propose a unified framework organized around key challenges, integrating environment design, state-action-reward formulation, DRL algorithms, solvers, and evaluation metrics into a cohesive pipeline, thereby enabling systematic categorization of existing research. (2) We introduce a taxonomy based on problem attributes and types to classify existing environments. (3) We provide a detailed analysis and comparative assessment of challenges across multiple layers, foundational, training, and application, facing current DRL approaches. (4) We summarize SOTA results on 17 publicly available datasets, significantly exceeding the scope of prior surveys, which typically cover only a handful. (5) This work constitutes the most comprehensive survey to date, encompassing approximately 300 high-quality papers and covering a broad spectrum of COPs.

Through this investigation, we find that the community has developed a range of innovative and effective solutions to core challenges in the field. Significant advances have been made in model architecture design, representation learning, and solver construction, establishing a prevailing paradigm for solving COPs. Nevertheless, substantial limitations remain in DRL-based approaches, particularly in handling large-scale problems, generalizing across instances, adapting to dynamic environments, and enabling practical deployment. These constraints impede their potential as efficient, general-purpose combinatorial optimization algorithms across diverse applications. To address these gaps, we identify several critical open questions that should guide future research and innovation. Furthermore, we maintain a continuously updated open-source repository that integrates fragmented studies on DRL for COPs into a unified, version-controlled framework. Equipped with structured metadata and automated update mechanisms, this resource facilitates systematic literature tracking and provides researchers with a navigable, curated collection of tools and benchmarks. We hope this review provides a clear and organized foundation for future work, supporting more coherent and reproducible research in DRL for COPs.

Several limitations of this review should be acknowledged. First, despite our systematic search protocol, the coverage of the surveyed literature may not be exhaustive. The rapid pace of publication in DRL for COPs means that some very recent works may have been missed, particularly those published after our May 2025 cut-off date. Second, our analysis was primarily qualitative: while we compiled SOTA performance results across 17 benchmark datasets, we did not conduct a formal meta-analysis or statistical synthesis of the reported results. The heterogeneity of experimental protocols across studies, including variations in hardware, hyperparameter tuning, and evaluation metrics, precluded rigorous quantitative comparison. Third, the scope of this review was intentionally focused on routing and scheduling as the two most extensively studied COP families. Other important problem classes, such as graph coloring, bin packing, knapsack, and facility location, received only passing mention. A comprehensive treatment of DRL applications to these domains would require a dedicated review. Fourth, our discussion of solver performance relied on the reported results from the original publications; we did not independently re-implement or re-evaluate the surveyed methods. The accuracy of our synthesis was

therefore contingent on the reliability of the source publications. Fifth, while we have substantially strengthened the mathematical formalization in this revision, the treatment of several topics, including convergence guarantees for multi-agent settings and theoretical generalization bounds for neural solvers, remains at a survey level and does not provide original theoretical contributions. Finally, this review reflects the state of the field as of mid-2025; the conclusions drawn here may evolve as the field matures and as standardized benchmarks and evaluation protocols become more widely adopted.

Author Contributions: S.W.: Writing—original draft, Conceptualization, Methodology, Visualization. C.H.: Formal analysis, Conceptualization. Z.W.: Visualization, Writing—review and editing. Y.W.: Visualization, Writing—review and editing. D.K.: Formal analysis, Conceptualization. H.M.: Funding acquisition, Conceptualization. Z.S.: Project administration, Conceptualization, Methodology, Formal analysis. All authors have read and agreed to the published version of the manuscript.

Funding: This research was funded by the National Science and Technology Major Project No.2024ZD01NL00102 and the National Natural Science Foundation of China Grant No.U2441226.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

Data Availability Statement: The data presented in this study are openly available on GitHub at <https://github.com/JY00002/Awesome-DRL4COPs>.

Acknowledgments: Not applicable.

Conflicts of Interest: The authors declare no conflicts of interest.

References

1. Chung, K.T.; Lee, C.K.M.; Tsang, Y.P. Neural Combinatorial Optimization with Reinforcement Learning in Industrial Engineering: A Survey. *Artif. Intell. Rev.* **2025**, *58*, 130.
2. Berto, F.; Hua, C.; Park, J.; Luttmann, L.; Ma, Y.; Bu, F.; Wang, J.; Ye, H.; Kim, M.; Choi, S.; et al. RL4CO: An Extensive Reinforcement Learning for Combinatorial Optimization Benchmark. In Proceedings of the 31st ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Canada, 3–7 August, **2025**.
3. Boggyrbayeva, A.; Yoon, T.; Ko, H.; Lim, S.; Yun, H.; Kwon, C. A Deep Reinforcement Learning Approach for Solving the Traveling Salesman Problem with Drone. *Transp. Res. Part C Emerg. Technol.* **2023**, *148*, 103981.
4. Park, I.B.; Huh, J.; Kim, J.; Park, J. A Reinforcement Learning Approach to Robust Scheduling of Semiconductor Manufacturing Facilities. *IEEE Trans. Autom. Sci. Eng.* **2019**, *17*, 1420–1431.
5. Lu, S.; Wang, Y.; Kong, M.; Wang, W.; Tan, W.; Song, Y. A Double Deep Q-Network Framework for a Flexible Job Shop Scheduling Problem with Dynamic Job Arrivals and Urgent Job Insertions. *Eng. Appl. Artif. Intell.* **2024**, *133*, 108487.
6. Dong, Z.; Ren, T.; Qi, F.; Weng, J.; Bai, D.; Yang, J.; Wu, C.C. A Reinforcement Learning-Based Approach for Solving Multi-Agent Job Shop Scheduling Problem. *Int. J. Prod. Res.* **2025**, *63*, 3512–3537.
7. Wang, D.; Liu, S.; Zou, J.; Qiao, W.; Jin, S. Flexible Robotic Cell Scheduling with Graph Neural Network Based Deep Reinforcement Learning. *J. Manuf. Syst.* **2025**, *78*, 81–93.
8. Kuai, Z.; Wang, T.; Wang, S. Fair Virtual Network Function Mapping and Scheduling Using Proximal Policy Optimization. *IEEE Trans. Commun.* **2022**, *70*, 7434–7445.
9. Su, J.; Huang, H.; Li, G.; Li, X.; Hao, Z. Self-Organizing Neural Scheduler for the Flexible Job Shop Problem With Periodic Maintenance and Mandatory Outsourcing Constraints. *IEEE Trans. Cybern.* **2023**, *53*, 5533–5544.
10. Sarfaraj, N.; Lingkon, M.L.; Zahan, N. Applying flexible job shop scheduling in patients management to optimize processing time in hospitals. *Int. J. Res. Ind. Eng.* **2021**, *10*, 46–55.
11. Brucker, P.; Jurisch, B.; and, B.S. A branch and bound algorithm for the job-shop scheduling problem. *Discret. Appl. Math.* **1994**, *49*, 107–127.
12. Wagner, H.M. An integer linear-programming model for machine scheduling. *Nav. Res. Logist.* **1959**, *6*, 131–140.
13. Fellek, G.; Farid, A.; Fujimura, S.; Yoshie, O.; Gebreyesus, G. G-DGNet: Gated Deep Graph Attention Network with Reinforcement Learning for Solving Traveling Salesman Problem. *Neurocomputing* **2024**, *579*, 127392.
14. Gelders, L.F.; Sambandam, N. Four Simple Heuristics for Scheduling a Flow-Shop. *Int. J. Prod. Res.* **1978**, *16*, 221–231.

15. Defersha, F.M.; Rooyani, D. An Efficient Two-Stage Genetic Algorithm for a Flexible Job-shop Scheduling Problem with Sequence Dependent Attached/Detached Setup, Machine Release Date and Lag-Time. *Comput. Ind. Eng.* **2020**, *147*, 106605.
16. Zhang, F.; Mei, Y.; Nguyen, S.; Zhang, M. Correlation Coefficient-Based Recombinative Guidance for Genetic Programming Hyperheuristics in Dynamic Flexible Job Shop Scheduling. *IEEE Trans. Evol. Comput.* **2021**, *25*, 552–566.
17. Zhang, L.; Yang, C.; Yan, Y.; Hu, Y. Distributed Real-Time Scheduling in Cloud Manufacturing by Deep Reinforcement Learning. *IEEE Trans. Ind. Inform.* **2022**, *18*, 8999–9007.
18. Ma, Y.; Cao, Z.; Chee, Y.M. Learning to Search Feasible and Infeasible Regions of Routing Problems with Flexible Neural k-Opt. In Proceedings of the Annual Conference on Neural Information Processing Systems, New Orleans, LA, USA, Dec. **2023**; Volume 36.
19. Ye, H.; Wang, J.; Liang, H.; Cao, Z.; Li, Y.; Li, F. GLOP: Learning Global Partition and Local Construction for Solving Large-Scale Routing Problems in Real-Time. *Proc. AAAI Conf. Artif. Intell.* **2024**, *38*, 20284–20292.
20. Liu, F.; Lin, X.; Wang, Z.; Zhang, Q.; Xialiang, T.; Yuan, M. Multi-Task Learning for Routing Problem with Cross-Problem Zero-Shot Generalization. In Proceedings of the 30th ACM SIGKDD Conference on Knowledge Discovery and Data Mining, Barcelona, Spain, 25–29 August, **2024**.
21. Zhou, J.; Cao, Z.; Wu, Y.; Song, W.; Ma, Y.; Zhang, J.; Xu, C. MVMoE: Multi-Task Vehicle Routing Solver with Mixture-of-Experts. In Proceedings of the International Conference on Machine Learning, Vienna, Austria, 21–27 July, **2024**.
22. Berto, F.; Hua, C.; Zepeda, N.G.; Hottung, A.; Wouda, N.; Lan, L.; Park, J.; Tierney, K.; Park, J. RouteFinder: Towards Foundation Models for Vehicle Routing Problems. *arXiv* **2024**, arXiv:2406.15007.
23. Wang, F.; He, Q.; Li, S. Solving Combinatorial Optimization Problems with Deep Neural Network: A Survey. *Tsinghua Sci. Technol.* **2024**, *29*, 1266–1282.
24. Wu, X.; Wang, D.; Wen, L.; Xiao, Y.; Wu, C.; Wu, Y.; Yu, C.; Maskell, D.L.; Zhou, Y. Neural Combinatorial Optimization Algorithms for Solving Vehicle Routing Problems: A Comprehensive Survey with Perspectives. *arXiv* **2025**, arXiv:2406.00415.
25. Yuan, E.; Cheng, S.; Wang, L.; Song, S.; Wu, F. Solving Job Shop Scheduling Problems via Deep Reinforcement Learning. *Appl. Soft Comput.* **2023**, *143*, 110436.
26. Wu, X.; Yan, X. A Spatial Pyramid Pooling-Based Deep Reinforcement Learning Model for Dynamic Job-Shop Scheduling Problem. *Comput. Oper. Res.* **2023**, *160*, 106401.
27. Wang, R.; Wang, G.; Deng, F. Flexible Job Shop Scheduling via Dual Attention Network-Based Reinforcement Learning. *IEEE Trans. Neural Networks Learn. Syst.* **2024**, *35*, 3091–3102.
28. Tiacci, L.; Rossi, A. A Discrete Event Simulator to Implement Deep Reinforcement Learning for the Dynamic Flexible Job Shop Scheduling Problem. *Simul. Model. Pract. Theory* **2024**, *134*, 102948.
29. Lv, L.; Zhang, C.; Fan, J.; Shen, W. Deep Reinforcement Learning for Job Shop Scheduling Problems: A Comprehensive Literature Review. *Knowl.-Based Syst.* **2025**, *321*, 113633.
30. Zhang, X.; Zhu, G.Y. A Literature Review of Reinforcement Learning Methods Applied to Job-Shop Scheduling Problems. *Comput. Oper. Res.* **2025**, *175*, 106929.
31. Xu, M.; Mei, Y.; Zhang, F.; Zhang, M. Learn to Optimise for Job Shop Scheduling: A Survey with Comparison between Genetic Programming and Reinforcement Learning. *Artif. Intell. Rev.* **2025**, *58*, 160.
32. Ngwu, C.; Liu, Y.; Wu, R. Reinforcement Learning in Dynamic Job Shop Scheduling: A Comprehensive Review of AI-driven Approaches in Modern Manufacturing. *J. Intell. Manuf.* **2026**, *37*, 1093–1108.
33. Wang, S.; Li, J.; Jiao, Q.; Ma, F. Design patterns of deep reinforcement learning models for job shop scheduling problems. *J. Intell. Manuf.* **2024**, *36*, 3741–3759.
34. Kayhan, B.M.; Yildiz, G. Reinforcement Learning Applications to Machine Scheduling Problems: A Comprehensive Literature Review. *J. Intell. Manuf.* **2023**, *34*, 905–929.
35. Smit, I.G.; Zhou, J.; Reijnen, R.; Wu, Y.; Chen, J.; Zhang, C.; Bukhsh, Z.; Zhang, Y.; Nuijten, W. Graph Neural Networks for Job Shop Scheduling Problems: A Survey. *Comput. Oper. Res.* **2025**, *176*, 106914.
36. Khadivi, M.; Charter, T.; Yaghoubi, M.; Jalayer, M.; Ahang, M.; Shojaeinasab, A.; Najjaran, H. Deep Reinforcement Learning for Machine Scheduling: Methodology, the State-of-the-Art, and Future Directions. *Comput. Ind. Eng.* **2025**, *200*, 110856.
37. Shahbazian, R.; Pugliese, L.D.P.; Guerriero, F.; Macrina, G. Integrating Machine Learning Into Vehicle Routing Problem: Methods and Applications. *IEEE Access* **2024**, *12*, 93087–93115.
38. Zhang, C.; Wu, Y.; Ma, Y.; Song, W.; Le, Z.; Cao, Z.; Zhang, J. A review on learning to solve combinatorial optimisation problems in manufacturing. *IET Collab. Intell. Manuf.* **2023**, *5*, e12072.
39. Li, B.; Wu, G.; He, Y.; Fan, M.; Pedrycz, W. An Overview and Experimental Study of Learning-Based Optimization Algorithms for the Vehicle Routing Problem. *IEEE/CAA J. Autom. Sin.* **2022**, *9*, 1115–1138.
40. I. Garmendia, A.; Ceberio, J.; Mendiburu, A. Applicability of Neural Combinatorial Optimization: A Critical View. *ACM Trans. Evol. Learn. Optim.* **2024**, *4*, 1–26.
41. Liu, Z.; Mao, H.; Sa, G.; Liu, H.; Tan, J. Dynamic Job-Shop Scheduling Using Graph Reinforcement Learning with Auxiliary Strategy. *J. Manuf. Syst.* **2024**, *73*, 1–18.

42. Ma, Y.; Li, J.; Cao, Z.; Song, W.; Zhang, L.; Chen, Z.; Tang, J. Learning to Iteratively Solve Routing Problems with Dual-Aspect Collaborative Transformer. *Adv. Neural Inf. Process. Syst.* **2021**, *34*, 11096–11107.
43. Zhang, L.; Yan, Y.; Yang, C.; Hu, Y. Dynamic Flexible Job-Shop Scheduling by Multi-Agent Reinforcement Learning with Reward-Shaping. *Adv. Eng. Inform.* **2024**, *62*, 102872.
44. Peng, S.; Xiong, G.; Yang, J.; Shen, Z.; Tamir, T.S.; Tao, Z.; Han, Y.; Wang, F.Y. Multi-Agent Reinforcement Learning for Extended Flexible Job Shop Scheduling. *Machines* **2023**, *12*, 8.
45. Wu, J.; Liu, Y. A Modified Multi-Agent Proximal Policy Optimization Algorithm for Multi-Objective Dynamic Partial-Re-Entrant Hybrid Flow Shop Scheduling Problem. *Eng. Appl. Artif. Intell.* **2025**, *140*, 109688.
46. Shi, J.; Qiao, F.; Liu, J.; Ma, Y.; Wang, D.; Ding, C. Production-Logistics Collaborative Scheduling in Dynamic Flexible Job Shops Using Nested-Hierarchical Deep Reinforcement Learning. *Adv. Eng. Inform.* **2025**, *65*, 103195.
47. Luo, J.; Li, C. An Efficient Encoder-Decoder Network for the Capacitated Vehicle Routing Problem. *Expert Syst. Appl.* **2025**, *278*, 127311.
48. Fang, W.; Zhang, H.; Qian, W.; Guo, Y.; Li, S.; Liu, Z.; Liu, C.; Hong, D. An Adaptive Job Shop Scheduling Mechanism for Disturbances by Running Reinforcement Learning in Digital Twin Environment. *J. Comput. Inf. Sci. Eng.* **2023**, *23*, 051013.
49. Wu, X.; Yan, X.; Guan, D.; Wei, M. A Deep Reinforcement Learning Model for Dynamic Job-Shop Scheduling Problem with Uncertain Processing Time. *Eng. Appl. Artif. Intell.* **2024**, *131*, 107790.
50. Liu, C.L.; Tseng, C.J.; Weng, P.H. Dynamic Job-Shop Scheduling via Graph Attention Networks and Deep Reinforcement Learning. *IEEE Trans. Ind. Inform.* **2024**, *20*, 8662–8672.
51. Gunarathna, U.; Borovica-Gajic, R.; Karunasekera, S.; Tanin, E. Dynamic Graph Combinatorial Optimization with Multi-Attention Deep Reinforcement Learning. In Proceedings of the 30th International Conference on Advances in Geographic Information Systems, Seattle, WA, USA, 1–4 November 2023; pp. 1–12.
52. Zhang, Z.; Liu, H.; Zhou, M.; Wang, J. Solving Dynamic Traveling Salesman Problems With Deep Reinforcement Learning. *IEEE Trans. Neural Networks Learn. Syst.* **2023**, *34*, 2119–2132. <https://doi.org/10.1109/TNNLS.2021.3105905>.
53. Pan, W.; Liu, S.Q. Deep Reinforcement Learning for the Dynamic and Uncertain Vehicle Routing Problem. *Appl. Intell.* **2023**, *53*, 405–422.
54. Mustakhov, T.; Akhmetbek, Y.; Bogrybayeva, A. Deep Reinforcement Learning for Stochastic Dynamic Vehicle Routing Problem. In Proceedings of the 2023 17th International Conference on Electronics Computer and Computation (ICECCO), Kaskelen, Kazakhstan, 1–2 June 2023; pp. 1–5.
55. Shi, Z.; Si, J.; Zhang, J.; Pang, Z.; Chen, H.; Ding, G. A Deep Reinforcement Learning Method Based on Hindsight Experience Replay for Multi-Objective Dynamic Job-Shop Scheduling Problem. *Expert Syst. Appl.* **2025**, *284*, 127989.
56. Li, Y.; Liang, X.; Guo, J.; Li, X.; Wang, L.; Du, B. Categorized Attention Based Hierarchical-agents Reinforcement Learning for Multi-objective Dynamic Job Shop Scheduling Problem With Machine Deterioration. *Appl. Soft Comput.* **2025**, *175*, 113032.
57. Zhao, S.; Gu, S. A Deep Reinforcement Learning Algorithm Framework for Solving Multi-Objective Traveling Salesman Problem Based on Feature Transformation. *Neural Netw.* **2024**, *176*, 106359. <https://doi.org/10.1016/j.neunet.2024.106359>.
58. Zhang, Z.; Wu, Z.; Zhang, H.; Wang, J. Meta-Learning-Based Deep Reinforcement Learning for Multiobjective Optimization Problems. *IEEE Trans. Neural Netw. Learn. Syst.* **2023**, *34*, 7978–7991.
59. Park, J.; Bakhtiyar, S.; Park, J. ScheduleNet: Learn to Solve Multi-Agent Scheduling Problems with Reinforcement Learning. *arXiv* **2021**, arXiv:2106.03051. <https://doi.org/10.48550/arXiv.2106.03051>.
60. Lin, Z.; Wu, Y.; Zhou, B.; Cao, Z.; Song, W.; Zhang, Y.; Jayavelu, S. Cross-Problem Learning for Solving Vehicle Routing Problems. In Proceedings of the Thirty-Third International Joint Conference on Artificial Intelligence, Jeju, Republic of Korea, 6–9 August, 2024; pp. 6958–6966.
61. Lu, H.; Zhang, X.; Yang, S. A Learning-based Iterative Method for Solving Vehicle Routing Problems. In Proceedings of the International Conference on Learning Representations (ICLR), 27 April–1 May, 2020.
62. Kool, W.; van Hoof, H.; Welling, M. Attention, Learn to Solve Routing Problems! In Proceedings of the International Conference on Learning Representations, New Orleans, LA, USA, 6–9 May, 2019.
63. Wang, C.; Cao, Z.; Wu, Y.; Teng, L.; Wu, G. Deep Reinforcement Learning for Solving Vehicle Routing Problems With Backhauls. *IEEE Trans. Neural Networks Learn. Syst.* **2025**, *36*, 4779–4793.
64. Phiboonbanakit, T.; Horanont, T.; Huynh, V.N.; Supnithi, T. A Hybrid Reinforcement Learning-Based Model for the Vehicle Routing Problem in Transportation Logistics. *IEEE Access* **2021**, *9*, 163325–163347.
65. Qiao, Y.; Miao, J.; Huang, X. A Combined Diffusion Model and Reinforcement Learning Approach for Solving the Vehicle Routing Problem With Multiple Soft Time Windows. *IEEE Access* **2025**, *13*, 113529–113543.
66. Aljohani, T.M.; Ebrahim, A.; Mohammed, O. Real-Time Metadata-Driven Routing Optimization for Electric Vehicle Energy Consumption Minimization Using Deep Reinforcement Learning and Markov Chain Model. *Electr. Power Syst. Res.* **2021**, *192*, 106962.

67. Bi, X.; Shen, M.; Gu, W.; Chung, E.; Wang, Y. Real-Time Planning of Route, Speed, and Charging for Electric Delivery Vehicles: A Deep Reinforcement Learning Approach. *IEEE Trans. Transp. Electrification*. **2025**, *11*, 7066–7082.
68. Wang, C.; Zhang, R.; Hong, R.; Wang, H. Attention-Enhanced Deep Reinforcement Learning for Electric Vehicle Routing Optimization. *IEEE Trans. Transp. Electrification*. **2025**, *11*, 11228–11242.
69. Liu, Q.; Liu, C.; Niu, S.; Long, C.; Zhang, J.; Xu, M. 2D-Ptr: 2D Array Pointer Network for Solving the Heterogeneous Capacitated Vehicle Routing Problem. In Proceedings of the AAMAS '24: 23rd International Conference on Autonomous Agents and Multiagent Systems, Auckland: IFAAMAS, 6–10 May **2024**: 1238–1246.
70. Berto, F.; Hua, C.; Luttmann, L.; Son, J.; Park, J.; Ahn, K.; Kwon, C.; Xie, L.; Park, J. PARCO: Parallel AutoRegressive Models for Multi-Agent Combinatorial Optimization. In Proceedings of the Annual Conference on Neural Information Processing Systems, San Diego, CA, USA, 2–7 December, **2025**.
71. Gast Zepeda, N.; Hottung, A.; Tierney, K. Learning to Solve the Skill Vehicle Routing Problem with Deep Reinforcement Learning. In *Proceedings of the International Conference on Learning and Intelligent Optimization; Lecture Notes in Computer Science*; Springer Nature: Cham, Switzerland, **2025**; Volume 15745, pp. 250–266. https://doi.org/10.1007/978-3-032-09192-5_17.
72. Song, W.; Chen, X.; Li, Q.; Cao, Z. Flexible Job-Shop Scheduling via Graph Neural Network and Deep Reinforcement Learning. *IEEE Trans. Ind. Inform.* **2023**, *19*, 1600–1610.
73. Huang, J.P.; Gao, L.; Li, X.Y. An End-to-End Deep Reinforcement Learning Method Based on Graph Neural Network for Distributed Job-Shop Scheduling Problem. *Expert Syst. Appl.* **2024**, *238*, 121756.
74. Pan, Z.; Wang, L.; Dong, C.; Chen, J.F. A Knowledge-Guided End-to-End Optimization Framework Based on Reinforcement Learning for Flow Shop Scheduling. *IEEE Trans. Ind. Inform.* **2024**, *20*, 1853–1861.
75. Lei, K.; Guo, P.; Wang, Y.; Zhang, J.; Meng, X.; Qian, L. Large-Scale Dynamic Scheduling for Flexible Job-Shop With Random Arrivals of New Jobs by Hierarchical Reinforcement Learning. *IEEE Trans. Ind. Inform.* **2024**, *20*, 1007–1018.
76. Grumbach, F.; Müller, A.; Reusch, P.; Trojahn, S. Robust-Stable Scheduling in Dynamic Flow Shops Based on Deep Reinforcement Learning. *J. Intell. Manuf.* **2024**, *35*, 667–686.
77. Sun, X.; Shen, W.; Fan, J.; Vogel-Heuser, B.; Bi, F.; Zhang, C. Deep Reinforcement Learning-based Multi-Objective Scheduling for Distributed Heterogeneous Hybrid Flow Shops with Blocking Constraints. *Engineering* **2025**, *46*, 278–291. <https://doi.org/10.1016/j.eng.2024.11.033>.
78. Kwon, Y.D.; Choo, J.; Kim, B.; Yoon, I.; Gwon, Y.; Min, S. POMO: Policy optimization with multiple optima for reinforcement learning. In Proceedings of the 34th Conference on Neural Information Processing Systems, NeurIPS 2020, 6–12 December, **2020**.
79. Xin, L.; Song, W.; Cao, Z.; Zhang, J. Multi-Decoder Attention Model with Embedding Glimpse for Solving Vehicle Routing Problems. *Proc. AAAI Conf. Artif. Intell.* **2021**, *35*, 12042–12049.
80. Wu, Y.; Song, W.; Cao, Z.; Zhang, J.; Lim, A. Learning Improvement Heuristics for Solving Routing Problems. *IEEE Trans. Neural Networks Learn. Syst.* **2022**, *33*, 5057–5069.
81. Xu, Y.; Fang, M.; Chen, L.; Xu, G.; Du, Y.; Zhang, C. Reinforcement Learning With Multiple Relational Attention for Solving Vehicle Routing Problems. *IEEE Trans. Cybern.* **2022**, *52*, 11107–11120.
82. Lei, K.; Guo, P.; Wang, Y.; Wu, X.; Zhao, W. Solve Routing Problems with a Residual Edge-Graph Attention Neural Network. *Neurocomputing* **2022**, *8*, 79–98.
83. Jiang, Y.; Cao, Z.; Wu, Y.; Song, W.; Zhang, J. Ensemble-based Deep Reinforcement Learning for Vehicle Routing Problems under Distribution Shift. In Proceedings of the 37th Conference on Neural Information Processing Systems, NeurIPS 2023, New Orleans, United States, 10–16 December, **2023**.
84. Jin, Y.; Ding, Y.; Pan, X.; He, K.; Zhao, L.; Qin, T.; Song, L.; Bian, J. Pointerformer: Deep Reinforced Multi-Pointer Transformer for the Traveling Salesman Problem. *Proc. AAAI Conf. Artif. Intell.* **2023**, *37*, 8132–8140.
85. Deinek, E.; Kehrt, C. Learn to Solve Vehicle Routing Problems ASAP: A Neural Optimization Approach for Time-Constrained Vehicle Routing Problems with Finite Vehicle Fleet. *arXiv* **2024**, arXiv:2411.04777.
86. Wang, X.; Wang, S.; Liang, X.; Zhao, D.; Huang, J.; Xu, X.; Dai, B.; Miao, Q. Deep Reinforcement Learning: A Survey. *IEEE Trans. Neural Networks Learn. Syst.* **2024**, *35*, 5064–5078.
87. Pateria, S.; Subagdja, B.; Tan, A.H.; et al.. Hierarchical Reinforcement Learning: A Comprehensive Survey. *ACM Comput. Surv.* **2022**, *54*, 1–35.
88. Si, J.; Li, X.; Gao, L.; Li, P. An Efficient and Adaptive Design of Reinforcement Learning Environment to Solve Job Shop Scheduling Problem with Soft Actor-Critic Algorithm. *Int. J. Prod. Res.* **2024**, *62*, 8260–8275.
89. Gu, W.; Liu, S.; Guo, Z.; Yuan, M.; Pei, F. Dynamic Scheduling Mechanism for Intelligent Workshop with Deep Reinforcement Learning Method Based on Multi-Agent System Architecture. *Comput. Ind. Eng.* **2024**, *191*, 110155.
90. Zhang, Y.; Zhu, H.; Tang, D.; Zhou, T.; Gui, Y. Dynamic Job Shop Scheduling Based on Deep Reinforcement Learning for Multi-Agent Manufacturing Systems. *Robot. Comput.-Integr. Manuf.* **2022**, *78*, 102412.
91. Wang, R.; Jing, Y.; Gu, C.; He, S.; Chen, J. End-to-End Multi-Target Flexible Job Shop Scheduling With Deep Reinforcement Learning. *IEEE Internet Things J.* **2024**, *12*, 4420–4434.

92. Yuan, M.; Huang, H.; Li, Z.; Zhang, C.; Pei, F.; Gu, W. A Multi-Agent Double Deep-Q-network Based on State Machine and Event Stream for Flexible Job Shop Scheduling Problem. *Adv. Eng. Inform.* **2023**, *58*, 102230.
93. Di, Y.; Deng, L.; Zhang, L. A Collaborative-Learning Multi-Agent Reinforcement Learning Method for Distributed Hybrid Flow Shop Scheduling Problem. *Swarm Evol. Comput.* **2024**, *91*, 101764.
94. Rangel-Martinez, D.; Ricardez-Sandoval, L.A. A Recurrent Reinforcement Learning Strategy for Optimal Scheduling of Partially Observable Job-Shop and Flow-Shop Batch Chemical Plants under Uncertainty. *Comput. Chem. Eng.* **2024**, *188*, 108748.
95. Liao, Z.; Li, Q.; Dai, Y.; Zhang, Z. Learning to Schedule Job-Shop Problems via Hierarchical Reinforcement Learning. In Proceedings of the 2022 IEEE International Conference on Systems, Man, and Cybernetics (SMC), Prague, Czech Republic, 9–12 October 2022; pp. 3222–3227.
96. Park, I.B.; Park, J. Scalable Scheduling of Semiconductor Packaging Facilities Using Deep Reinforcement Learning. *IEEE Trans. Cybern.* **2023**, *53*, 3518–3531.
97. Chen, L.; Liu, H.; Jia, N.; Ren, N.; Cui, R.; Wei, W. Real-Time Stochastic Flexible Flow Shop Scheduling in a Credit Factory with Model-Based Reinforcement Learning. *Int. J. Prod. Res.* **2025**, *6*, 845–864.
98. Lv, L.; Fan, J.; Zhang, C.; Shen, W. A Multi-Agent Reinforcement Learning Based Scheduling Strategy for Flexible Job Shops under Machine Breakdowns. *Robot. Comput.-Integr. Manuf.* **2025**, *93*, 102923.
99. Wang, X.; Zhang, L.; Lin, T.; Zhao, C.; Wang, K.; Chen, Z. Solving Job Scheduling Problems in a Resource Preemption Environment with Multi-Agent Reinforcement Learning. *Robot. Comput.-Integr. Manuf.* **2022**, *77*, 102324.
100. Li, H.; Gao, L.; Fan, Q.; Li, X.; Han, B. An End-to-End Decentralised Scheduling Framework Based on Deep Reinforcement Learning for Dynamic Distributed Heterogeneous Flowshop Scheduling. *Int. J. Prod. Res.* **2025**, *63*, 4368–4388 1–21.
101. Gui, Y.; Zhang, Z.; Tang, D.; Zhu, H.; Zhang, Y. Collaborative Dynamic Scheduling in a Self-Organizing Manufacturing System Using Multi-Agent Reinforcement Learning. *Adv. Eng. Inform.* **2024**, *62*, 102646.
102. Wang, H.; Lin, W.; Peng, T.; Xiao, Q.; Tang, R. Multi-Agent Deep Reinforcement Learning-Based Approach for Dynamic Flexible Assembly Job Shop Scheduling with Uncertain Processing and Transport Times. *Expert Syst. Appl.* **2025**, *270*, 126441.
103. Zhang, Z.; Qi, G.; Guan, W. Coordinated Multi-agent Hierarchical Deep Reinforcement Learning to Solve Multi-trip Vehicle Routing Problems with Soft Time Windows. *IET Intell. Transp. Syst.* **2023**, *17*, 2034–2051.
104. Zhang, N.; Shen, Y.; Du, Y.; Chen, L.; Zhang, X. Counterfactual-Attention Multi-Agent Reinforcement Learning for Joint Condition-Based Maintenance and Production Scheduling. *J. Manuf. Syst.* **2023**, *71*, 70–81.
105. Xu, S.; Li, Y.; Li, Q. A Deep Reinforcement Learning Method Based on a Transformer Model for the Flexible Job Shop Scheduling Problem. *Electronics* **2024**, *13*, 3696.
106. Heo, C.Y.; Seo, J.; Kim, Y.; Kim, Y.; Kim, T. Estimated Tardiness-Based Reinforcement Learning Solution to Repeatable Job-Shop Scheduling Problems. *Processes* **2024**, *13*, 62.
107. Wang, Z.; Liao, W. Smart Scheduling of Dynamic Job Shop Based on Discrete Event Simulation and Deep Reinforcement Learning. *J. Intell. Manuf.* **2024**, *35*, 2593–2610.
108. Monaci, M.; Agasucci, V.; Grani, G. An Actor-Critic Algorithm with Policy Gradients to Solve the Job Shop Scheduling Problem Using Deep Double Recurrent Agents. *Eur. J. Oper. Res.* **2024**, *312*, 910–926.
109. Yang, S.; Guo, H.; Huang, J.; Han, K. A Deep Reinforcement Learning Based Approach for Dynamic Job Shop Scheduling Considering Variable Processing Time. In Proceedings of the International Conference on Artificial Intelligence, Automation and High Performance Computing, Zhuhai, China, Zhuhai, China, 19–21 July, **2024**; pp. 368–374.
110. Lee, J.; Kee, S.; Janakiram, M.; Runger, G. Attention-Based Reinforcement Learning for Combinatorial Optimization: Application to Job Shop Scheduling Problem. *arXiv* **2024**, <https://doi.org/10.48550/arXiv.2401.16580>.
111. Zhao, L.; Shen, W.; Zhang, C.; Peng, K. An End-to-End Deep Reinforcement Learning Approach for Job Shop Scheduling. In Proceedings of the 2022 IEEE 25th International Conference on Computer Supported Cooperative Work in Design (CSCWD), Hangzhou, China, 4–6 May, **2022**; pp. 841–846.
112. Ma, W.; Shao, W.; Liu, K. Reinforcement Learning for Solving the Job Shop Scheduling Problem Based on Traveling Salesman Problem Model. In Proceedings of the 2024 43rd Chinese Control Conference (CCC), Kunming, China, 28–31 July, **2024**; pp. 2003–2008.
113. Zhao, L.; Fan, J.; Zhang, C.; Shen, W.; Zhuang, J. A DRL-Based Reactive Scheduling Policy for Flexible Job Shops With Random Job Arrivals. *IEEE Trans. Autom. Sci. Eng.* **2024**, *21*, 2912–2923.
114. Gui, Y.; Tang, D.; Zhu, H.; Zhang, Y.; Zhang, Z. Dynamic Scheduling for Flexible Job Shop Using a Deep Reinforcement Learning Approach. *Comput. Ind. Eng.* **2023**, *180*, 109255.
115. Liu, Y.; Fan, J.; Zhao, L.; Shen, W.; Zhang, C. Integration of Deep Reinforcement Learning and Multi-Agent System for Dynamic Scheduling of Re-Entrant Hybrid Flow Shop Considering Worker Fatigue and Skill Levels. *Robot. Comput.-Integr. Manuf.* **2023**, *84*, 102605.
116. Serrano-Ruiz, J.C.; Mula, J.; Poler, R. Job Shop Smart Manufacturing Scheduling by Deep Reinforcement Learning. *J. Ind. Inf. Integr.* **2024**, *38*, 100582.

117. Liang, X.; Song, W.; Wei, P. Dynamic Job Shop Scheduling via Deep Reinforcement Learning. In Proceedings of the 2023 IEEE 35th International Conference on Tools with Artificial Intelligence (ICTAI), Atlanta, GA, USA, 6–8 November, **2023**; pp. 369–376.
118. Li, Y.; Wang, Q.; Li, X.; Gao, L.; Fu, L.; Yu, Y.; Zhou, W. Real-Time Scheduling for Flexible Job Shop With AGVs Using Multiagent Reinforcement Learning and Efficient Action Decoding. *IEEE Trans. Syst. Man Cybern. Syst.* **2025**, *55*, 2120–2132.
119. Wang, Z.; Liao, W. Job Shop Scheduling Problem Using Proximal Policy Optimization. In Proceedings of the 2023 IEEE International Conference on Industrial Engineering and Engineering Management (IEEM), Singapore, 18–21 December, **2023**; pp. 1517–1521.
120. Li, Y.; Gu, W.; Yuan, M.; Tang, Y. Real-Time Data-Driven Dynamic Scheduling for Flexible Job Shop with Insufficient Transportation Resources Using Hybrid Deep Q Network. *Robot. Comput.-Integr. Manuf.* **2022**, *74*, 102283.
121. Wang, S.; Li, J.; Tang, H.; Wang, J. CEA-FJSP: Carbon Emission-Aware Flexible Job-Shop Scheduling Based on Deep Reinforcement Learning. *Front. Environ. Sci.* **2022**, *10*, 1059451.
122. Yang, S.; Wang, J.; Xin, L.; Xu, Z. Real-Time and Concurrent Optimization of Scheduling and Reconfiguration for Dynamic Reconfigurable Flow Shop Using Deep Reinforcement Learning. *CIRP J. Manuf. Sci. Technol.* **2023**, *40*, 243–252.
123. Yang, S.; Wang, J.; Xu, Z. Real-Time Scheduling for Distributed Permutation Flowshops with Dynamic Job Arrivals Using Deep Reinforcement Learning. *Adv. Eng. Inform.* **2022**, *54*, 101776.
124. Luo, L.; Yan, X. Scheduling of Stochastic Distributed Hybrid Flow-Shop by Hybrid Estimation of Distribution Algorithm and Proximal Policy Optimization. *Expert Syst. Appl.* **2025**, *271*, 126523.
125. Wang, X.; Zhong, P.; Liu, M.; Zhang, C.; Yang, S. A Novel Method-Based Reinforcement Learning with Deep Temporal Difference Network for Flexible Double Shop Scheduling Problem. *Sci. Rep.* **2024**, *14*, 9047.
126. Lin, C.C.; Peng, Y.C.; Chen, Z.Y.A.; Fan, Y.H.; Chin, H.H. Distributed Flexible Job Shop Scheduling through Deploying Fog and Edge Computing in Smart Factories Using Dual Deep Q Networks. *Mob. Netw. Appl.* **2024**, *29*, 886–904.
127. Huang, J.P.; Gao, L.; Li, X.Y.; Zhang, C.J. A Cooperative Hierarchical Deep Reinforcement Learning Based Multi-Agent Method for Distributed Job Shop Scheduling Problem with Random Job Arrivals. *Comput. Ind. Eng.* **2023**, *185*, 109650.
128. Dong, X.; Wan, G.; Zeng, P. A Heuristic-Assisted Deep Reinforcement Learning Algorithm for Flexible Job Shop Scheduling with Transport Constraints. *Complex Intell. Syst.* **2025**, *11*, 210.
129. Qiu, J.; Liu, J.; Li, Z.; Lai, X. A Multi-Level Action Coupling Reinforcement Learning Approach for Online Two-Stage Flexible Assembly Flow Shop Scheduling. *J. Manuf. Syst.* **2024**, *76*, 351–370.
130. Yue, L.; Peng, K.; Ding, L.; Mumtaz, J.; Lin, L.; Zou, T. Two-Stage Double Deep Q-network Algorithm Considering External Non-Dominant Set for Multi-Objective Dynamic Flexible Job Shop Scheduling Problems. *Swarm Evol. Comput.* **2024**, *90*, 101660.
131. Chang, J.; Yu, D.; Zhou, Z.; He, W.; Zhang, L. Hierarchical Reinforcement Learning for Multi-Objective Real-Time Flexible Scheduling in a Smart Shop Floor. *Machines* **2022**, *10*, 1195.
132. Yan, Y.; Yi, W.; Pei, Z.; Chen, Y. Multi-Agent Reinforcement Learning for Distributed Flexible Job Shop Scheduling With Random Job Arrival. *IEEE Access* **2025**, *13*, 80941–80957.
133. Luo, S.; Zhang, L.; Fan, Y. Real-Time Scheduling for Dynamic Partial-No-Wait Multiobjective Flexible Job Shop by Deep Reinforcement Learning. *IEEE Trans. Autom. Sci. Eng.* **2022**, *19*, 3020–3038.
134. Lei, Y.; Deng, Q.; Liao, M.; et al.. Deep reinforcement learning for dynamic distributed job shop scheduling problem with transfers. *Expert Syst. Appl.* **2024**, *251*, 123970.
135. Li, Y.; Li, X.; Gao, L.; Lu, Z. Multi-Agent Deep Reinforcement Learning for Dynamic Reconfigurable Shop Scheduling Considering Batch Processing and Worker Cooperation. *Robot. Comput.-Integr. Manuf.* **2025**, *91*, 102834.
136. Chen, W.; Zhang, Z.; Tang, D.; Liu, C.; Gui, Y.; Nie, Q.; Zhao, Z. Probing an LSTM-PPO-Based Reinforcement Learning Algorithm to Solve Dynamic Job Shop Scheduling Problem. *Comput. Ind. Eng.* **2024**, *197*, 110633.
137. Luo, S. Dynamic Scheduling for Flexible Job Shop with New Job Insertions by Deep Reinforcement Learning. *Appl. Soft Comput.* **2020**, *91*, 106208.
138. Chen, Y.; Liao, X.; Chen, G.; Hou, Y. Dynamic Intelligent Scheduling in Low-Carbon Heterogeneous Distributed Flexible Job Shops with Job Insertions and Transfers. *Sensors* **2024**, *24*.
139. Luo, S.; Zhang, L.; Fan, Y. Dynamic Multi-Objective Scheduling for Flexible Job Shop by Deep Reinforcement Learning. *Comput. Ind. Eng.* **2021**, *159*, 107489.
140. Ren, F.; Liu, H. Dynamic Scheduling for Flexible Job Shop Based on MachineRank Algorithm and Reinforcement Learning. *Sci. Rep.* **2024**, *14*, 29741.
141. Wang, H.; Cheng, J.; Liu, C.; Zhang, Y.; Hu, S.; Chen, L. Multi-Objective Reinforcement Learning Framework for Dynamic Flexible Job Shop Scheduling Problem with Uncertain Events. *Appl. Soft Comput.* **2022**, *131*, 109717.
142. Xu, N.; Bu, T.M. Policy Network for Solving Flexible Job Shop Scheduling Problem with Setup Times and Resource Constraints. In Proceedings of the Genetic and Evolutionary Computation Conference Companion, Boston, MA, USA, USA, 9–13 July, **2022**; pp. 208–211.

143. Cheng, W.; Zhang, C.; Meng, L.; Zhang, B.; Gao, K.; Sang, H. Deep Reinforcement Learning for Solving Efficient and Energy-Saving Flexible Job Shop Scheduling Problem with Multi-AGV. *Comput. Oper. Res.* **2025**, *181*, 107087.
144. Ding, L.; Guan, Z.; Luo, D.; Yue, L. Data-Driven Hierarchical Multi-Policy Deep Reinforcement Learning Framework for Multi-Objective Multiplicity Dynamic Flexible Job Shop Scheduling. *J. Manuf. Syst.* **2025**, *80*, 536–562.
145. Zhang, L.; Yan, Y.; Hu, Y. Dynamic Flexible Scheduling with Transportation Constraints by Multi-Agent Reinforcement Learning. *Eng. Appl. Artif. Intell.* **2024**, *134*, 108699.
146. Ding, L.; Guan, Z.; Rauf, M.; Yue, L. Multi-Policy Deep Reinforcement Learning for Multi-Objective Multiplicity Flexible Job Shop Scheduling. *Swarm Evol. Comput.* **2024**, *87*, 101550.
147. Heik, D.; Bahrpeyma, F.; Reichelt, D. Study on the Application of Single-Agent and Multi-Agent Reinforcement Learning to Dynamic Scheduling in Manufacturing Environments with Growing Complexity: Case Study on the Synthesis of an Industrial IoT Test Bed. *J. Manuf. Syst.* **2024**, *77*, 525–557.
148. Wang, M.; Zhang, J.; Zhang, P.; Cui, L.; Zhang, G. Independent Double DQN-based Multi-Agent Reinforcement Learning Approach for Online Two-Stage Hybrid Flow Shop Scheduling with Batch Machines. *J. Manuf. Syst.* **2022**, *65*, 694–708.
149. Moon, J.; Yang, M.; Jeong, J. A Novel Approach to the Job Shop Scheduling Problem Based on the Deep Q-Network in a Cooperative Multi-Access Edge Computing Ecosystem. *Sensors* **2021**, *21*, 4553.
150. Du, Y.; Li, J.Q. A Deep Reinforcement Learning Based Algorithm for a Distributed Precast Concrete Production Scheduling. *Int. J. Prod. Econ.* **2024**, *268*, 109102.
151. Du, Y.; Li, J.Q.; Chen, X.L.; Duan, P.Y.; Pan, Q.K. Knowledge-Based Reinforcement Learning and Estimation of Distribution Algorithm for Flexible Job Shop Scheduling Problem. *IEEE Trans. Emerg. Top. Comput. Intell.* **2023**, *7*, 1036–1050.
152. Yang, S.; Wang, J.; Xu, Z. Learning to Schedule Dynamic Distributed Reconfigurable Workshops Using Expected Deep Q-network. *Adv. Eng. Inform.* **2024**, *59*, 102307.
153. Zhu, X.; Xu, J.; Ge, J.; Wang, Y.; Xie, Z. Multi-Task Multi-Agent Reinforcement Learning for Real-Time Scheduling of a Dual-Resource Flexible Job Shop with Robots. *Processes* **2023**, *11*, 267.
154. Ouahabi, N.; Chebak, A.; Kamach, O.; Zegrari, M. Dynamic Production Scheduling and Maintenance Planning under Opportunistic Grouping. *Comput. Ind. Eng.* **2025**, *199*, 110646.
155. Zhao, T.; Wang, Y.; Tan, Y.; Zhang, J. Dynamic Flexible Job Shop Scheduling Algorithm Based on Deep Reinforcement Learning. In Proceedings of the 2023 35th Chinese Control and Decision Conference (CCDC), Yichang, China, 20–22 May, **2023**; pp. 5099–5104.
156. Wu, Z.; Fan, H.; Sun, Y.; Peng, M. Efficient Multi-Objective Optimization on Dynamic Flexible Job Shop Scheduling Using Deep Reinforcement Learning Approach. *Processes* **2023**, *11*, 2018.
157. Yuan, M.; Zheng, L.; Huang, H.; Zhou, K.; Pei, F.; Gu, W. Research on Flexible Job Shop Scheduling Problem with AGV Using Double DQN. *J. Intell. Manuf.* **2025**, *36*, 509–535.
158. Du, Y.; Li, J.; Li, C.; Duan, P. A Reinforcement Learning Approach for Flexible Job Shop Scheduling Problem With Crane Transportation and Setup Times. *IEEE Trans. Neural Netw. Learn. Syst.* **2024**, *35*, 5695–5709.
159. Oh, S.H.; Cho, Y.I.; Woo, J.H. Distributional Reinforcement Learning with the Independent Learners for Flexible Job Shop Scheduling Problem with High Variability. *J. Comput. Des. Eng.* **2022**, *9*, 1157–1174.
160. Liu, X.; Han, L.; Kang, L.; Liu, J.; Miao, H. Preference Learning Based Deep Reinforcement Learning for Flexible Job Shop Scheduling Problem. *Complex Intell. Syst.* **2025**, *11*, 144.
161. Pan, Z.; Wang, L.; Wang, J.; Lu, J. Deep Reinforcement Learning Based Optimization Algorithm for Permutation Flow-Shop Scheduling. *IEEE Trans. Emerg. Top. Comput. Intell.* **2023**, *7*, 983–994.
162. Zhang, W.; Zhao, F.; Yang, C.; Du, C.; Feng, X.; Zhang, Y.; Peng, Z.; Mei, X. A Novel Soft Actor–Critic Framework with Disjunctive Graph Embedding and Autoencoder Mechanism for Job Shop Scheduling Problems. *J. Manuf. Syst.* **2024**, *76*, 614–626.
163. Jing, X.; Yao, X.; Liu, M.; Zhou, J. Multi-Agent Reinforcement Learning Based on Graph Convolutional Network for Flexible Job Shop Scheduling. *J. Intell. Manuf.* **2024**, *35*, 75–93.
164. Hameed, M.S.A.; Schwung, A. Graph Neural Networks-Based Scheduler for Production Planning Problems Using Reinforcement Learning. *J. Manuf. Syst.* **2023**, *69*, 91–102.
165. Kwon, Y.D.; Choo, J.; Yoon, I.; Park, M.; Park, D.; Gwon, Y. Matrix Encoding Networks for Neural Combinatorial Optimization. In Proceedings of the 35th Conference on Neural Information Processing Systems, NeurIPS 2021, 6–14 December, **2021**.
166. Huang, J.P.; Gao, L.; Li, X.Y. A Hierarchical Multi-Action Deep Reinforcement Learning Method for Dynamic Distributed Job-Shop Scheduling Problem With Job Arrivals. *IEEE Trans. Autom. Sci. Eng.* **2024**, *22*, 2501–2513.
167. Huang, J.P.; Gao, L.; Li, X.Y.; Zhang, C.J. A Novel Priority Dispatch Rule Generation Method Based on Graph Neural Network and Reinforcement Learning for Distributed Job-Shop Scheduling. *J. Manuf. Syst.* **2023**, *69*, 119–134.
168. Zhang, M.; Wang, L.; Qiu, F.; Liu, X. Dynamic Scheduling for Flexible Job Shop with Insufficient Transportation Resources via Graph Neural Network and Deep Reinforcement Learning. *Comput. Ind. Eng.* **2023**, *186*, 109718.
169. Moon, S.; Lee, S.; Park, K.J. Learning-Enabled Flexible Job-Shop Scheduling for Scalable Smart Manufacturing. *J. Manuf. Syst.* **2024**, *77*, 356–367.

170. Yang, Z.; Bi, L.; Jiao, X. Combining Reinforcement Learning Algorithms with Graph Neural Networks to Solve Dynamic Job Shop Scheduling Problems. *Processes* **2023**, *11*, 1571.
171. Liu, Y.; Zhang, F.; Sun, Y.; Zhang, M. Evolutionary Trainer-Based Deep Q-Network for Dynamic Flexible Job Shop Scheduling. *IEEE Trans. Evol. Comput.* **2025**, *29*, 749–763.
172. Su, C.; Zhang, C.; Xia, D.; Han, B.; Wang, C.; Chen, G.; Xie, L. Evolution Strategies-Based Optimized Graph Reinforcement Learning for Solving Dynamic Job Shop Scheduling Problem. *Appl. Soft Comput.* **2023**, *145*, 110596.
173. Zhang, C.; Song, W.; Cao, Z.; Zhang, J.; Tan, P.S.; Xu, C. Learning to dispatch for job shop scheduling via deep reinforcement learning. In Proceedings of the 34th Conference on Neural Information Processing Systems, NeurIPS 2020, 6–12 December, **2020**.
174. Park, J.; Su, Y.; Ju, F. Efficient Job Shop Scheduling via Graph Based Deep Reinforcement Learning with Adaptive Multi-Action Selection. In Proceedings of the 2024 IEEE 20th International Conference on Automation Science and Engineering (CASE), Bari, Italy, 28 August–1 September, **2024**; pp. 2949–2954.
175. Hammami, N.E.H.; Lardeux, B.; B. Hadj-Alouane, A.; Jridi, M. Design and Calibration of a DRL Algorithm for Solving the Job Shop Scheduling Problem under Unexpected Job Arrivals. *Flex. Serv. Manuf. J.* **2024**, *37*, 125.
176. Su, C.; Zhang, C.; Wang, C.; Cen, W.; Chen, G.; Xie, L. Fast Pareto Set Approximation for Multi-Objective Flexible Job Shop Scheduling via Parallel Preference-Conditioned Graph Reinforcement Learning. *Swarm Evol. Comput.* **2024**, *88*, 101605.
177. Lei, K.; Guo, P.; Zhao, W.; Wang, Y.; Qian, L.; Meng, X.; Tang, L. A Multi-Action Deep Reinforcement Learning Framework for Flexible Job-shop Scheduling Problem. *Expert Syst. Appl.* **2022**, *205*, 117796.
178. Zhou, T.; Luo, L.; Ji, S.; He, Y. A Reinforcement Learning Approach to Robust Scheduling of Permutation Flow Shop. *Biomimetics* **2023**, *8*, 478.
179. Pan, J.; Zhong, R.; Hu, B.; Feng, Y.; Zhang, Z.; Tan, J. Smart Scheduling of Hanging Workshop via Digital Twin and Deep Reinforcement Learning. *Flex. Serv. Manuf. J.* **2024**, *37*, 15.
180. Infantes, G.; Roussel, S.; Pereira, P.; Jacquet, A.; Benazera, E. Learning to Solve Job Shop Scheduling Under Uncertainty. In Proceedings of the International Conference on the Integration of Constraint Programming, Artificial Intelligence, and Operations Research (CPAIOR), Uppsala, Sweden, 28–31 May, **2024**.
181. Hu, Z.; Chen, Y.; Li, H.; Deng, X. Research on Job Shop Scheduling Problem Based on Multi-Agent Reinforcement Learning. In Proceedings of the 4th International Conference on Artificial Intelligence and Computer Engineering, Dalian, China, 17–19 November, **2023**; pp. 1209–1213.
182. Zhang, W.; Zhao, F.; Li, Y.; Du, C.; Feng, X.; Mei, X. A Novel Collaborative Agent Reinforcement Learning Framework Based on an Attention Mechanism and Disjunctive Graph Embedding for Flexible Job Shop Scheduling Problem. *J. Manuf. Syst.* **2024**, *74*, 329–345.
183. Lv, L.; Zhang, C.; Shen, W. A Deep Reinforcement Learning-based Rescheduling Method for Flexible Job Shops under Machine Breakdowns. In Proceedings of the 2024 27th International Conference on Computer Supported Cooperative Work in Design (CSCWD), Tianjin, China, 8–10 May, **2024**; pp. 1734–1739.
184. Zhao, Y.; Chen, Y.; Wu, J.; Guo, C. Dual Dynamic Attention Network for Flexible Job Scheduling with Reinforcement Learning. In Proceedings of the 2024 International Joint Conference on Neural Networks (IJCNN), Yokohama, Japan, 30 June–5 July, **2024**; pp. 1–8.
185. Wan, L.; Fu, L.; Li, C.; Li, K. An Effective Multi-Agent-Based Graph Reinforcement Learning Method for Solving Flexible Job Shop Scheduling Problem. *Eng. Appl. Artif. Intell.* **2025**, *139*, 109557.
186. Tang, Y.; Shen, L.; Han, S. Low-Carbon Flexible Job Shop Scheduling Problem Based on Deep Reinforcement Learning. *Sustainability* **2024**, *16*, 4544.
187. Li, Y.; Zhong, W.; Wu, Y. Multi-Objective Flexible Job-Shop Scheduling via Graph Attention Network and Reinforcement Learning. *J. Supercomput.* **2025**, *81*, 293.
188. Johnson, D.; Chen, G.; Lu, Y. Multi-Agent Scheduler for the Continuous Dynamic Flexible Job Shop Scheduling Problem. In Proceedings of the 2024 IEEE 20th International Conference on Automation Science and Engineering (CASE), Bari, Italy, 28 August–1 September, **2024**; pp. 2924–2930.
189. Tang, H.; Dong, J. Solving Flexible Job-Shop Scheduling Problem with Heterogeneous Graph Neural Network Based on Relation and Deep Reinforcement Learning. *Machines* **2024**, *12*, 584.
190. Moon, S.; Lee, S.; Park, K.J. Graph-Based Reinforcement Learning for Flexible Job Shop Scheduling with Transportation Constraints. In Proceedings of the IECON 2023- 49th Annual Conference of the IEEE Industrial Electronics Society, Singapore, 16–19 October, **2023**; pp.1–6.
191. Pu, Y.; Li, F.; Rahimifard, S. Multi-Agent Reinforcement Learning for Job Shop Scheduling in Dynamic Environments. *Sustainability* **2024**, *16*, 3234.
192. Liu, L.; He, Y.; Mao, J. Flexible Job-shop Scheduling Method Based on Deep Reinforcement Learning. In Proceedings of the 2023 China Automation Congress (CAC), Chongqing, China, 17–19 November, **2023**; pp. 5041–5046.

193. Zhao, Y.; Luo, X.; Zhang, Y. The Application of Heterogeneous Graph Neural Network and Deep Reinforcement Learning in Hybrid Flow Shop Scheduling Problem. *Comput. Ind. Eng.* **2024**, *187*, 109802.
194. Dax, V.M.; Li, J.; Leahy, K.; Kochenderfer, M.J. Graph Q-Learning for Combinatorial Optimization. In Proceedings of the 36th Conference on Neural Information Processing Systems, New Orleans, LA, USA, 28 November–3 December, **2022**.
195. Qin, Z.; Johnson, D.; Lu, Y. Dynamic Production Scheduling towards Self-Organizing Mass Personalization: A Multi-Agent Dueling Deep Reinforcement Learning Approach. *J. Manuf. Syst.* **2023**, *68*, 242–257.
196. Liu, R.; Piplani, R.; Toro, C. A Deep Multi-Agent Reinforcement Learning Approach to Solve Dynamic Job Shop Scheduling Problem. *Comput. Oper. Res.* **2023**, *159*, 106294.
197. Liu, C.L.; Chang, C.C.; Tseng, C.J. Actor-Critic Deep Reinforcement Learning for Solving Job Shop Scheduling Problems. *IEEE Access* **2020**, *8*, 71752–71762.
198. Yuan, E.; Wang, L.; Song, S.; Cheng, S.; Fan, W. Dynamic Scheduling for Multi-Objective Flexible Job Shop via Deep Reinforcement Learning. *Appl. Soft Comput.* **2025**, *171*, 112787.
199. Jensen, K. *Coloured Petri Nets: Basic Concepts, Analysis Methods and Practical Use*; Springer: Berlin/Heidelberg, Germany, 1997.
200. Lassoued, S.; Schwung, A. Introducing PetriRL: An Innovative Framework for JSSP Resolution Integrating Petri Nets and Event-Based Reinforcement Learning. *J. Manuf. Syst.* **2024**, *74*, 690–702.
201. Kim, H.J.; Lee, J.H. Look-Ahead Based Reinforcement Learning for Robotic Flow Shop Scheduling. *J. Manuf. Syst.* **2023**, *68*, 160–175.
202. Lee, J.H.; Kim, H.J. Reinforcement Learning for Robotic Flow Shop Scheduling with Processing Time Variations. *Int. J. Prod. Res.* **2022**, *60*, 2346–2368.
203. Lei, K.; Guo, P.; Wang, Y.; Xiong, J.; Zhao, W. An End-to-end Hierarchical Reinforcement Learning Framework for Large-scale Dynamic Flexible Job-shop Scheduling Problem. In Proceedings of the 2022 International Joint Conference on Neural Networks (IJCNN), Padua, Italy, 18–23 July, **2022**; pp. 1–8.
204. Zhang, L.; Feng, Y.; Xiao, Q.; et al.. Deep reinforcement learning for dynamic flexible job shop scheduling problem considering variable processing times. *J. Manuf. Syst.* **2023**, *71*, 257–273.
205. Xu, M.; Mei, Y.; Zhang, F.; Zhang, M. Niching Genetic Programming to Learn Actions for Deep Reinforcement Learning in Dynamic Flexible Scheduling. *IEEE Trans. Evol. Comput.* **2024**, *30*, 61–75.
206. Lin, B.; Ghaddar, B.; Nathwani, J. Deep Reinforcement Learning for the Electric Vehicle Routing Problem With Time Windows. *IEEE Trans. Intell. Transp. Syst.* **2022**, *23*, 11528–11538.
207. Chen, R.; Li, W.; Yang, H. A Deep Reinforcement Learning Framework Based on an Attention Mechanism and Disjunctive Graph Embedding for the Job-Shop Scheduling Problem. *IEEE Trans. Ind. Inform.* **2023**, *19*, 1322–1331.
208. Park, J.; Chun, J.; Kim, S.H.; Kim, Y.; Park, J. Learning to Schedule Job-Shop Problems: Representation and Policy Learning Using Graph Neural Network and Reinforcement Learning. *Int. J. Prod. Res.* **2021**, *59*, 3360–3377.
209. Liu, C.L.; Huang, T.H. Dynamic Job-Shop Scheduling Problems Using Graph Neural Network and Deep Reinforcement Learning. *IEEE Trans. Syst. Man, Cybern. Syst.* **2023**, *53*, 6836–6848.
210. Wang, L.; Hu, X.; Wang, Y.; Xu, S.; Ma, S.; Yang, K.; Liu, Z.; Wang, W. Dynamic Job-Shop Scheduling in Smart Manufacturing Using Deep Reinforcement Learning. *Comput. Netw.* **2021**, *190*, 107969.
211. Dong, Z.; Ren, T.; Weng, J.; Qi, F.; Wang, X. Minimizing the Late Work of the Flow Shop Scheduling Problem with a Deep Reinforcement Learning Based Approach. *Appl. Sci.* **2022**, *12*, 2366.
212. Han, B.A.; Yang, J.J. A Deep Reinforcement Learning Based Solution for Flexible Job Shop Scheduling Problem. *Int. J. Simul. Model.* **2021**, *20*, 375–386.
213. Huang, D.; Zhao, H.; Tian, W.; Chen, K. A Deep Reinforcement Learning Method Based on a Multiexpert Graph Neural Network for Flexible Job Shop Scheduling. *Comput. Ind. Eng.* **2025**, *200*, 110768.
214. Rui, Z.; Zhang, X.; Liu, M.; Ling, L.; Wang, X.; Liu, C.; Sun, M. Graph Reinforcement Learning for Flexible Job Shop Scheduling under Industrial Demand Response: A Production and Energy Nexus Perspective. *Comput. Ind. Eng.* **2024**, *193*, 110325.
215. Wan, L.; Fu, L.; Li, C.; Li, K. Flexible Job Shop Scheduling via Deep Reinforcement Learning with Meta-Path-Based Heterogeneous Graph Neural Network. *Knowl.-Based Syst.* **2024**, *296*, 111940.
216. Shao, C.; Yu, Z.; Ding, H.; Cao, G.; Zhou, B. A Reinforcement Learning Method for Flexible Job Shop Scheduling Based on Multi-Head Attention and Deep Residual Network. *Comput. Electr. Eng.* **2025**, *123*, 110044.
217. Huang, D.; Zhao, H.; Cao, J.; Chen, K.; Zhang, L. Optimizing the Flexible Job Shop Scheduling Problem via Deep Reinforcement Learning with Mean Multichannel Graph Attention. *Appl. Soft Comput.* **2025**, *177*, 113128.
218. Liu, X.; Chen, X.; Chau, V.; Musial, J.; Blazewicz, J. Flexible Job Shop Scheduling Problem Using Graph Neural Networks and Reinforcement Learning. *Comput. Oper. Res.* **2025**, *182*, 107139.
219. Echeverria, I.; Murua, M.; Santana, R. Diverse Policy Generation for the Flexible Job-Shop Scheduling Problem via Deep Reinforcement Learning with a Novel Graph Representation. *Eng. Appl. Artif. Intell.* **2025**, *139*, 109488.

220. Ho, K.H.; Cheng, J.Y.; Wu, J.H.; Chiang, F.; Chen, Y.C.; Wu, Y.Y.; Wu, I.C. Residual Scheduling: A New Reinforcement Learning Approach to Solving Job Shop Scheduling Problem. *IEEE Access* **2024**, *12*, 14703–14718.
221. Bonetta, G.; Zago, D.; Cancelliere, R.; Grosso, A. Job Shop Scheduling via Deep Reinforcement Learning: A Sequence to Sequence Approach. In *Proceedings of the Learning and Intelligent Optimization*; Sellmann, M., Tierney, K., Eds.; Springer: Cham, Switzerland, 2023; Volume 14286, pp. 475–490.
222. Yuan, E.; Wang, L.; Cheng, S.; Song, S.; Fan, W.; Li, Y. Solving Flexible Job Shop Scheduling Problems via Deep Reinforcement Learning. *Expert Syst. Appl.* **2024**, *245*, 123019.
223. Hottung, A.; Tierney, K. Neural Large Neighborhood Search for the Capacitated Vehicle Routing Problem. In *Proceedings of the 24th European Conference on Artificial Intelligence (ECAI 2020)*, Santiago de Compostela, Spain, 29 August–8 September, **2020**.
224. Ma, Y.; Li, J.; Cao, Z.; Song, W.; Guo, H.; Gong, Y.; Chee, Y.M. Efficient Neural Neighborhood Search for Pickup and Delivery Problems. In *Proceedings of the Thirty-First International Joint Conference on Artificial Intelligence, IJCAI-22*, Vienna, Austria, 23–29 July 2022; pp. 4776–4784.
225. Garmendia, A.I.; Ceberio, J.; Mendiburu, A. Neural Improvement Heuristics for Graph Combinatorial Optimization Problems. *IEEE Trans. Neural Networks Learn. Syst.* **2024**, *35*, 18300–18312.
226. Hu, Y.; Wang, W.; Jia, H.; Wang, Y.; Chen, Y.; Hao, J.; Wu, F.; Fan, C. Learning to utilize shaping rewards: A new approach of reward shaping. In *Proceedings of the 34th Conference on Neural Information Processing Systems, NeurIPS 2020*, 6–12 December, **2020**.
227. Liu, R.; Piplani, R.; Toro, C. Deep Reinforcement Learning for Dynamic Scheduling of a Flexible Job Shop. *Int. J. Prod. Res.* **2022**, *60*, 4049–4069.
228. Müller, A.; Grumbach, F.; Kattenstroth, F. Reinforcement Learning for Two-Stage Permutation Flow Shop Scheduling—A Real-World Application in Household Appliance Production. *IEEE Access* **2024**, *12*, 11388–11399.
229. Fan, X.; Zhou, Y.; Liu, C.Y. Trajectory Planning for Capacitated Vehicle Routing Problem: A Deep Reinforcement Learning Approach. In *Proceedings of the 2023 IEEE International Conference on Mechatronics and Automation (ICMA)*, Harbin, China, 6–9 August, **2023**; pp. 760–765.
230. Wan, L.; Cui, X.; Zhao, H.; Fu, L.; Li, C. A Novel Method for Solving Dynamic Flexible Job-Shop Scheduling Problem via DIFFormer and Deep Reinforcement Learning. *Comput. Ind. Eng.* **2024**, *198*, 110688.
231. Liu, Z.; Wang, Y.; Liang, X.; Ma, Y.; Feng, Y.; Cheng, G.; Liu, Z. A Graph Neural Networks-Based Deep Q-learning Approach for Job Shop Scheduling Problems in Traffic Management. *Inf. Sci.* **2022**, *607*, 1211–1223.
232. Khuntiyaporn, T.; Songmuang, P.; Limprasert, W. The Multiple Objectives Flexible Jobshop Scheduling Using Reinforcement Learning. In *Proceedings of the 2021 16th International Joint Symposium on Artificial Intelligence and Natural Language Processing (ISAI-NLP)*, Ayutthaya, Thailand, 21–23 December, **2021**; pp. 1–6.
233. Lang, S.; Behrendt, F.; Lanzerath, N.; Reggelin, T.; Muller, M. Integration of Deep Reinforcement Learning and Discrete-Event Simulation for Real-Time Scheduling of a Flexible Job Shop Production. In *Proceedings of the 2020 Winter Simulation Conference (WSC)*, Orlando, FL, USA, 14–18 December, **2020**; pp. 3057–3068.
234. Liu, M.; Rui, Z.; Zhang, X.; Ge, M.; Ling, L.; Wang, X.; Liu, C. EFJSP-IBDR: Energy-efficient Flexible Job Shop Scheduling under Incentive-Based Demand Response via Graph Reinforcement Learning with Dual Attention Mechanism. *Expert Syst. Appl.* **2025**, *280*, 127340.
235. Chen, X.; Li, Y.; Wang, K.; Wang, L.; Liu, J.; Wang, J.; Wang, X.V. Reinforcement Learning for Distributed Hybrid Flowshop Scheduling Problem with Variable Task Splitting towards Mass Personalized Manufacturing. *J. Manuf. Syst.* **2024**, *76*, 188–206.
236. Zhao, F.; Di, S.; Wang, L. A Hyperheuristic With Q-Learning for the Multiobjective Energy-Efficient Distributed Blocking Flow Shop Scheduling Problem. *IEEE Trans. Cybern.* **2023**, *53*, 3337–3350.
237. Bello, I.; Pham, H.; Le, Q.V.; Norouzi, M.; Bengio, S. Neural Combinatorial Optimization with Reinforcement Learning. In *Proceedings of the International Conference on Learning Representations (Workshop)*, Toulon, France, 24–26 April, **2017**.
238. Hottung, A.; Mahajan, M.; Tierney, K. PolyNet: Learning Diverse Solution Strategies for Neural Combinatorial Optimization. In *Proceedings of the Thirteenth International Conference on Learning Representations*, Singapore, 24–28 April, **2025**.
239. Chen, X.; Tian, Y. Learning to Perform Local Rewriting for Combinatorial Optimization. In *Proceedings of the Annual Conference on Neural Information Processing Systems*, Vancouver, BC, Canada, 8–14 December 2019.
240. Kong, D.; Ma, Y.; Cao, Z.; Yu, T.; Xiao, J. Efficient Neural Collaborative Search for Pickup and Delivery Problems. *IEEE Trans. Pattern Anal. Mach. Intell.* **2024**, *46*, 11019–11034.
241. Vinyals, O.; Fortunato, M.; Jaitly, N. Pointer Networks. *Annu. Conf. Neural Inf. Process. Syst.* **2015**, *28*, 2692–2700.
242. Nazari, M.; Oroojlooy, A.; Takac, M.; Snyder, L.V. Reinforcement Learning for Solving the Vehicle Routing Problem. In *Proceedings of the 32nd Conference on Neural Information Processing Systems (NIPS)*, Montréal, Canada, 3–8 December, **2018**; pp. 9861–9871.
243. Vaswani, A.; Shazeer, N.; Parmar, N.; Uszkoreit, J.; Jones, L.; Gomez, A.N.; Kaiser, L.; Polosukhin, I. Attention is All You Need. *Annu. Conf. Neural Inf. Process. Syst.* **2017**, *30*, 5998–6008.

244. Kim, M.; Park, J.; Park, J. Sym-NCO: Leveraging Symmetricity for Neural Combinatorial Optimization. In Proceedings of the 36th Conference on Neural Information Processing Systems, NeurIPS 2022, New Orleans, LA, USA, 28 November–9 December, **2022**.
245. Pan, X.; Jin, Y.; Ding, Y.; Feng, M.; Zhao, L.; Song, L.; Bian, J. H-TSP: Hierarchically Solving the Large-Scale Traveling Salesman Problem. *Proc. AAAI Conf. Artif. Intell.* **2023**, *37*, 9345–9353.
246. Hua, C.; Berto, F.; Son, J.; Kang, S.; Kwon, C.; Park, J. CAMP: Collaborative Attention Model with Profiles for Vehicle Routing Problems. In Proceedings of the 2025 International Conference on Autonomous Agents and Multiagent Systems (AAMAS), Detroit, MI, USA, 19–23 May, **2025**.
247. Li, J.; Ma, Y.; Cao, Z.; Wu, Y.; Song, W.; Zhang, J.; Chee, Y.M. Learning Feature Embedding Refiner for Solving Vehicle Routing Problems. *IEEE Trans. Neural Networks Learn. Syst.* **2024**, *35*, 15279–15291.
248. Kim, H.; Kim, M.; Berto, F.; Kim, J.; Park, J. DevFormer: A Symmetric Transformer for Context-Aware Device Placement. In Proceedings of the International Conference on Machine Learning, Honolulu, HI, USA, 23–29 July, **2023**; pp. 16541–16566.
249. da Costa, P.R.d.O.; Rhuggenaath, J.; Zhang, Y.; Akcay, A. Learning 2-opt Heuristics for the Traveling Salesman Problem via Deep Reinforcement Learning. In Proceedings of the 12th Asian Conference on Machine Learning, ACML 2020, Bangkok, Thailand, 18–20 November, **2020**; pp. 465–480.
250. Sui, J.; Ding, S.; Liu, R.; Xu, L.; Bu, D. Learning 3-opt heuristics for traveling salesman problem via deep reinforcement learning. In Proceedings of the 13th Asian Conference on Machine Learning, ACML 2021, 17–19 November, **2021**.
251. Khalil, E.; Dai, H.; Zhang, Y.; Dilkina, B.; Song, L. Learning Combinatorial Optimization Algorithms over Graphs. In Proceedings of the Annual Conference on Neural Information Processing Systems, 4–9 December, **2017**; Volume 30, pp. 6348–6358.
252. Li, J.; Ma, Y.; Gao, R.; Cao, Z.; Lim, A.; Song, W.; Zhang, J. Deep Reinforcement Learning for Solving the Heterogeneous Capacitated Vehicle Routing Problem. *IEEE Trans. Cybern.* **2022**, *52*, 13572–13585.
253. Li, J.; Xin, L.; Cao, Z.; Lim, A.; Song, W.; Zhang, J. Heterogeneous attentions for solving pickup and delivery problem via deep reinforcement learning. *IEEE Trans. Intell. Transp. Syst.* **2021**, *23*, 2306–2315.
254. Chen, J.; Zhang, Z.; Cao, Z.; Wu, Y.; Ma, Y.; Ye, T.; Wang, J. Neural Multi-Objective Combinatorial Optimization with Diversity Enhancement. In Proceedings of the Advances in Neural Information Processing Systems, New Orleans, LA, USA, 10–16 December, **2023**.
255. Chalumeau, F.; Surana, S.; Bonnet, C.; Grinsztajn, N.; Pretorius, A.; Laterre, A.; Barrett, T.D. Combinatorial Optimization with Policy Adaptation using Latent Space Search. *Adv. Neural Inf. Process. Syst.* **2023**, *36*, 7947–7959.
256. Fan, M.; Wu, Y.; Liao, T.; Cao, Z.; Guo, H.; Sartoretti, G.; Wu, G. Deep Reinforcement Learning for UAV Routing in the Presence of Multiple Charging Stations. *IEEE Trans. Veh. Technol.* **2023**, *72*, 5732–5746.
257. Li, J.; Tian Dai, B.; Niu, Y.; Xiao, J.; Wu, Y. Multi-Type Attention for Solving Multi-Depot Vehicle Routing Problems. *IEEE Trans. Intell. Transp. Syst.* **2024**, *25*, 17831–17840.
258. Iklassov, Z.; Sobirov, I.; Solozabal, R.; Takac, M. Reinforcement Learning for Solving Stochastic Vehicle Routing Problem with Time Windows. *arXiv*, **2024**, arXiv:2402.09765.
259. Luo, Z.; Jiang, C.; Liu, L.; Zheng, X.; Ma, H. Flow-Shop Scheduling Problem With Batch Processing Machines via Deep Reinforcement Learning for Industrial Internet of Things. *IEEE Trans. Emerg. Top. Comput. Intell.* **2024**, *8*, 3518–3533.
260. Huang, Z.; Zhou, J.; Cao, Z.; Xu, Y. Rethinking Light Decoder-based Solvers for Vehicle Routing Problems. In Proceedings of the Thirteenth International Conference on Learning Representations, Singapore, 24–28 April, **2025**.
261. Son, J.; Zhao, Z.; Berto, F.; Hua, C.; Kwon, C.; Park, J. Neural Combinatorial Optimization for Real-World Routing. *arXiv* **2025**, arXiv:2503.16159.
262. Zhang, C.; Cao, Z.; Song, W.; Wu, Y.; Zhang, J. Deep Reinforcement Learning Guided Improvement Heuristic for Job Shop Scheduling. In Proceedings of the International Conference on Learning Representations, Vienna, Austria, 7–11 May, **2024**.
263. Wan, L.; Cui, X.; Zhao, H.; Li, C.; Wang, Z. An Effective Deep Actor-Critic Reinforcement Learning Method for Solving the Flexible Job Shop Scheduling Problem. *Neural Comput. Appl.* **2024**, *36*, 11877–11899.
264. Wang, Y.F. Adaptive Job Shop Scheduling Strategy Based on Weighted Q-learning Algorithm. *J. Intell. Manuf.* **2020**, *31*, 417–432.
265. Akkerman, F.; Mes, M.; Van Jaarsveld, W. A Comparison of Reinforcement Learning Policies for Dynamic Vehicle Routing Problems with Stochastic Customer Requests. *Comput. Ind. Eng.* **2025**, *200*, 110747.
266. Sun, Z.; Han, W.; Gao, L. Real-Time Scheduling for Dynamic Workshops with Random New Job Insertions by Using Deep Reinforcement Learning. *Adv. Prod. Eng. Manag.* **2023**, *18*, 137–151.
267. Zhang, M.; Lu, Y.; Hu, Y.; Amaitik, N.; Xu, Y. Dynamic Scheduling Method for Job-Shop Manufacturing Systems by Deep Reinforcement Learning with Proximal Policy Optimization. *Sustainability* **2022**, *14*, 5177.
268. Zhang, J.D.; He, Z.; Chan, W.H.; Chow, C.Y. DeepMAG: Deep Reinforcement Learning with Multi-Agent Graphs for Flexible Job Shop Scheduling. *Knowl.-Based Syst.* **2023**, *259*, 110083.
269. Kipf, T.N.; Welling, M. Semi-Supervised Classification with Graph Convolutional Networks. In Proceedings of the International Conference on Learning Representations (ICLR), France, 24–26 April, **2017**.

270. Xu, K.; Hu, W.; Leskovec, J.; Jegelka, S. How Powerful are Graph Neural Networks? In Proceedings of the International Conference on Learning Representations (ICLR), New Orleans, LA, USA, 6–9 May, **2019**.
271. Johnson, D.; Chen, G.; Lu, Y. Multi-Agent Reinforcement Learning for Real-Time Dynamic Production Scheduling in a Robot Assembly Cell. *IEEE Robot. Autom. Lett.* **2022**, *7*, 7684–7691.
272. Fu, Z.H.; Qiu, K.B.; Zha, H. Generalize a Small Pre-trained Model to Arbitrarily Large TSP Instances. In Proceedings of the AAAI Conference on Artificial Intelligence, 2–9 February, **2021**; pp. 7474–7482.
273. Duan, H.; Xiao, K.; Liu, L.; Chen, H.; Huang, H. REINFORCE with Bound-guided Gradient Estimator for the Traveling Salesman Problem toward Scale Generalization. *Eng. Appl. Artif. Intell.* **2025**, *141*, 109819.
274. Kuhnle, A.; May, M.C.; Schäfer, L.; Lanza, G. Explainable Reinforcement Learning in Production Control of Job Shop Manufacturing System. *Int. J. Prod. Res.* **2022**, *60*, 5812–5834.
275. Reinelt, G. TSPLIB—A Traveling Salesman Problem Library. *ORSA J. Comput.* **1991**, *3*, 376–384.
276. Liu, C.; Feng, X.F.; Li, F.; Xian, Q.L.; Jia, Z.H.; Wang, Y.H.; Du, Z.D. Deep Reinforcement Learning Combined with Transformer to Solve the Traveling Salesman Problem. *J. Supercomput.* **2025**, *81*.
277. Guan, Q.; Cao, H.; Jia, L.; Yan, D.; Chen, B. Synergetic Attention-Driven Transformer: A Deep Reinforcement Learning Approach for Vehicle Routing Problems. *Expert Syst. Appl.* **2025**, *274*, 126961.
278. Uchoa, E.; Pecin, D.; Pessoa, A.e.a. New Benchmark Instances for the Capacitated Vehicle Routing Problem. *Eur. J. Oper. Res.* **2017**, *257*, 845–858.
279. Kalatzantonakis, P.; Sifaleras, A.; Samaras, N. A Reinforcement Learning-Variable Neighborhood Search Method for the Capacitated Vehicle Routing Problem. *Expert Syst. Appl.* **2023**, *213*, 118812.
280. Guo, T.; Mei, Y.; Zhang, M.; Zhao, H.; Cai, K.; Du, W. Learning-Aided Neighborhood Search for Vehicle Routing Problems. *IEEE Trans. Pattern Anal. Mach. Intell.* **2025**, *47*, 5930–5944.
281. Solomon, M.M. Algorithms for the Vehicle Routing and Scheduling Problems with Time Window Constraints. *Oper. Res.* **1987**, *35*, 254–265.
282. Hottung, A.; Kwon, Y.D.; Tierney, K. Efficient active search for combinatorial optimization problems. *arXiv*, **2021**, arXiv:2106.05126.
283. Adams, J.; Zawack, B.D. The Shifting Bottleneck Procedure for Job Shop Scheduling. *Manag. Sci.* **1988**, *34*, 391–401.
284. Han, B.A.; Yang, J.J. Research on Adaptive Job Shop Scheduling Problems Based on Dueling Double DQN. *IEEE Access* **2020**, *8*, 186474–186495.
285. Fisher, H. *Probabilistic Learning Combinations of Local Job-Shop Scheduling Rules*; Prentice Hall: Englewood Cliffs, NJ, USA, **1963**.
286. Applegate, D.; Cook, W. A Computational Study of the Job-Shop Scheduling Problem. *Orsa J. Comput.* **1991**, *3*, 149–156.
287. Yamada, T.; Nakano, R. A Genetic Algorithm Applicable to Large-Scale Job-Shop Problems. In Proceedings of the Parallel Problem Solving from Nature, Brussels, Belgium, 28–30 September **1992**.
288. Lawrence, S. *Supplement to Resource Constrained Project Scheduling: An Experimental Investigation of Heuristic Scheduling Techniques*; Graduate School of Industrial Administration, Carnegie-Mellon University: Pittsburgh, PA, USA, **1984**.
289. Storer, R.H.; Wu, S.D.; Vaccari, R. New Search Spaces for Sequencing Problems with Application to Job Shop Scheduling. *Manag. Sci.* **1992**, *38*, 1495–1509.
290. Taillard, E.D. Benchmarks for Basic Scheduling Problems. *Eur. J. Oper. Res.* **1993**, *64*, 278–285.
291. Gebreyesus, G.D.; Fellek, G.T.; Farid, A.; Qiang, Y.; Fujimura, S.; Yoshie, O. A Deep Reinforcement Learning Framework with Convolution Augmented Attention and Gate Mechanism for Job Shop Scheduling. In Proceedings of the 2024 8th International Conference on Innovation in Artificial Intelligence, Tokyo, Japan, 16–18 March, **2024**; pp. 171–177.
292. Gebreyesus, G.; Fellek, G.; Farid, A.; Hou, S.; Fujimura, S.; Yoshie, O. Deep Reinforcement Learning-Based Spatio-Temporal Graph Neural Network for Solving Job Shop Scheduling Problem. *Evol. Intell.* **2025**, *18*, 6.
293. Demirkol, E.; Mehta, S.; Uzsoy, R. Benchmarks for shop scheduling problems. *Eur. J. Oper. Res.* **1998**, *109*, 137–141.
294. Hurink, J.; Jurisch, B.; Thole, M. Tabu search for the job-shop scheduling problem with multi-purpose machines. *OR Spectr.* **1994**, *15*, 205–215.
295. Huang, D.; Zhao, H.; Zhang, L.; Chen, K. Learning to Dispatch for Flexible Job Shop Scheduling Based on Deep Reinforcement Learning via Graph Gated Channel Transformation. *IEEE Access* **2024**, *12*, 50935–50948.
296. Brandimarte, P. Routing and scheduling in a flexible job shop by tabu search. *Ann. Oper. Res.* **1993**, *41*, 157–183.
297. Behnke, D.; Geiger, M.J. *Test Instances for the Flexible Job Shop Scheduling Problem with Work Centers*; Institut für Betriebliche Logistik und Organisation: Darmstadt, Germany, **2012**.
298. Dauzère-Pérès, S.; Paulli, J. An integrated approach for modeling and solving the general multiprocessor job-shop scheduling problem using tabu search. *Ann. Oper. Res.* **1997**, *70*, 281–306.
299. Barnes, J.W.; Chambers, J.B. Flexible Job Shop Scheduling by Tabu Search. In *Graduate Program in Operations and Industrial Engineering: Technical Report Series, ORP96-09*; The University of Texas at Austin: Austin, TX, USA, **1996**.
300. Abgaryan, H.; Cazenave, T.; Harutyunyan, A. Starjob: Dataset for LLM-Driven Job Shop Scheduling. *arXiv* **2025**, arXiv:2503.01877.

301. Feng, S.; Sun, W.; Li, S.; Talwalkar, A.; Yang, Y. A Comprehensive Evaluation of Contemporary ML-Based Solvers for Combinatorial Optimization. *arXiv* **2025**, arXiv:2505.16952.
302. Fornies-Tabuenca, D.; Uribe, A.; Otamendi, U.; Artetxe, A.; Rivera, J.C.; de Lacalle, O.L. REMoH: A Reflective Evolution of Multi-objective Heuristics approach via Large Language Models. *arXiv* **2025**, arXiv:2506.07759. <https://doi.org/10.48550/arXiv.2506.07759>.
303. Yang, X.; Zhang, L.; Qian, H.; Song, L.; Bian, J. HeurAgenix: Leveraging LLMs for Solving Complex Combinatorial Optimization Challenges. *arXiv* **2025**, arXiv:2506.15196. <https://doi.org/10.48550/arXiv.2506.15196>.

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.