WORLD MODEL-GUIDED ADAPTIVE LOOKAHEAD FOR DYNAMIC MULTI-OBJECTIVE NEURAL ROUTING PROBLEMS

Anonymous authors
Paper under double-blind review

ABSTRACT

We introduce a world-model-guided neural solver for dynamic multi-objective routing problems, where graphs change during route execution. Although neural combinatorial optimization (NCO) models can re-encode graphs after changes, training only on static instances may limit decision quality in evolving environments. Moreover, direct node selection may not fully account for the impact of alternative choices on final route quality. We establish a standardized training and evaluation framework that incorporates five event families into partially executed trajectories: node addition, node removal, attribute drift, addition with drift, and removal with drift. This enables dynamic adaptation without modifying the NCO architecture. We further propose world-model-guided adaptive lookahead, where a learned controller selectively evaluates Top-K route completions at decision points that benefit from additional computation, refining node selections without predicting future events. A cost-to-go signal for each node further improves credit assignment in dynamic training. Experiments on MOTSP and MOCVRP across diverse dynamic profiles and problem sizes demonstrate improved solution quality, efficient inference, and strong cross-scale and large-scale generalization. Adaptive lookahead also improves frozen NCO policies on static instances.

1 INTRODUCTION

Multi-objective routing becomes an online decision problem when requests, nodes, or attributes change after a route has been partially executed(Yang et al., 2008; Li & Feng, 2012). Dynamic vehicle-routing research characterizes such settings by how information evolves and becomes available (Pillac et al., 2013; Ojeda Rios et al., 2021). Established responses insert new requests into service sequences (Wang et al., 2025), optimize over rolling horizons (Tarhan et al., 2024; Song et al., 2022), evaluate solution robustness under uncertain parameters through simulation (Mavrotas et al., 2015), or repair and replan solutions after disruptions with evolutionary or neighborhood-based methods (Wang et al., 2021; Liu et al., 2024). Online reinforcement learning shifts computation toward offline policy training and supports rapid route generation at deployment (Yu et al., 2019; Chen et al., 2022; Li et al., 2025). These methods address environmental change from complementary directions, but they do not provide a unified learning and evaluation protocol for multi-objective neural solvers operating on partially executed routes.

Neural combinatorial optimization (NCO) learns reusable heuristics directly from instance distributions (Bengio et al., 2021). Attention-based encoder–decoder policies and multi-start policy optimization provide effective routing backbones (Kool et al., 2019; Kwon et al., 2020), while preference-conditioned models generate approximate Pareto solutions for multiple trade-offs with a single network (Lin et al., 2022; Wang et al., 2024). The same architecture can naturally re-encode a modified graph after updates to the node set and attributes; the current location, remaining capacity, served-node mask, and objective preference then define the next decision state (Kool et al., 2019; Fan et al., 2025). The key gap is therefore not the absence of a mechanism for processing updated states. Rather, conventional static training (Fan et al., 2026; Zhang et al., 2023) rarely exposes policies to states jointly shaped by an executed route prefix and an evolving graph, limiting their ability to learn how to adapt decisions under dynamic changes (Bengio et al., 2021; Son et al., 2025).

Reviewers: please read the Reviewer Guidelines (iclr.cc/Conferences/2027/ReviewerGuidelines) and the AI Policy for Reviewers (iclr.cc/Conferences/2027/AIPolicyForReviewers). If you used AI to expand, edit, or polish your review, please provide the input text to the LLM. Better still, consider skipping the LLM and submitting your original text: we, and the authors, are much more interested in your unedited thoughts than in what an LLM has to say. AI-assisted or not, you are putting your name and reputation behind your review: LLM-generated falsehoods, hallucinations or misrepresentations are subject to disciplinary action, which may include desk-rejecting all papers you have authored.

We introduce a standardized training and evaluation framework for dynamic multi objective combinatorial optimization. The framework covers five event families: node addition, node removal, attribute drift, addition with attribute drift, and removal with attribute drift, and injects them into partially executed solution trajectories. After each observed event, the solver updates and re encodes the current graph using the same encoder decoder architecture as the static counterpart.

We further propose world model guided adaptive lookahead. World models enable decision making by leveraging imagined future consequences (Ha & Schmidhuber, 2018; Janner et al., 2019; Hansen et al., 2022; Hafner et al., 2025), while test time search improves neural combinatorial solvers by allocating additional computation to candidate solutions (Choo et al., 2022; Hottung et al., 2022; Chalumeau et al., 2023; Verdù et al., 2025; Ye et al., 2025). We introduce a world model formulation for NCO, where the frozen solver is rolled out to the end of the route to estimate the consequences of candidate node selections. The learned controller identifies decision points that benefit from additional computation and selectively activates rollout refinement. For dynamic training, we introduce a node level cost to go signal to improve credit assignment under evolving environments.

Our contributions are summarized as follows: (1) A training and evaluation standard for dynamic multi-objective combinatorial optimization: We formalize five atomic and compound event families and integrate them into partially executed trajectories, establishing controlled training and evaluation protocols for dynamic neural solvers. (2) World model guided adaptive lookahead: We introduce future consequence reasoning into NCO and adaptively refine node selections using imagined route completions at critical decision points. (3) Cost-to-go credit assignment for dynamic training: We introduce a node-level learning signal that attributes downstream costs to current decisions while mitigating interference caused by exogenous environmental changes.

2 Related Work

Dynamic Multi-Objective Routing and Online Replanning. Dynamic vehicle routing studies decision making under evolving information, with taxonomies covering information evolution, applications, and solution methods (Pillac et al., 2013; Ojeda Rios et al., 2021). Approaches address dynamic changes through demand insertion (Wang et al., 2025), rolling-horizon routing and scheduling (Tarhan et al., 2024), stochastic-request adaptation (Wang et al., 2021), and proactive–reactive rescheduling with adaptive large-neighborhood search (Liu et al., 2024). These methods mainly rely on replanning and search-based reconstruction. Learning-based operator control learns adaptation policies from decisions. Deep Q-learning and hierarchical reinforcement learning have been applied to vehicle–drone dispatching (Chen et al., 2022; Li et al., 2025). For dynamic multi-objective scheduling, hierarchical Q-networks and multi-objective reinforcement learning select dispatching objectives or rules under job and machine changes (Luo et al., 2021; Wang et al., 2022), while Q-learning has been combined with NSGA-II for event-driven rescheduling (Chen et al., 2024).

Multi-Objective Neural Combinatorial Optimization Attention-based NCO learns constructive routing heuristics with an encoder–decoder policy (Kool et al., 2019), POMO improves policy optimization by exploiting multiple optima (Kwon et al., 2020), and online NCO combines graph-embedded pointer policies with deep reinforcement learning for rapid route generation (Yu et al., 2019). Multi-objective extensions condition models on preferences or objective information to approximate Pareto trade-offs (Lin et al., 2022; Wang et al., 2024). Meta neural heuristics further improve adaptation across scalarized subproblems (Chen et al., 2023), while conditional neural heuristics incorporate instance context, preference, and problem size into multi-objective routing policies (Fan et al., 2025). Overall, this line of work focuses on neural solver architectures, policy optimization, and preference-conditioned Pareto approximation.

World Models and Model-Based Planning. World models learn decision relevant representations and use imagined consequences to improve behavior (Ha & Schmidhuber, 2018; Hafner et al., 2019a). Model based reinforcement learning extends this idea through uncertainty aware trajectory sampling (Chua et al., 2018), model generated rollouts from real data (Janner et al., 2019), learned model search (Schrittwieser et al., 2020), and task oriented latent dynamics for model predictive control (Hafner et al., 2019b; Hansen et al., 2022). By exposing action dependent consequences, prediction enables planning by allowing decisions to be evaluated based on their potential future outcomes, while rollout horizon and uncertainty limit the reliability of imagined trajectories. For

combinatorial construction, this idea suggests using solution rollouts to estimate the consequences of current decisions rather than modeling future environment transitions.

Test-Time Search and Rollout-Enhanced NCO. Test-time computation improves neural solutions through learned local rewriting (Chen & Tian, 2019), neural large-neighborhood search (Hottung & Tierney, 2020), active search with instance-specific adaptation (Hottung et al., 2022), simulation-guided beam search (Choo et al., 2022), latent-space policy selection (Chalumeau et al., 2023), and rollout beam search with online adaptation (Verdù et al., 2025). These methods apply additional computation to different targets, including solution components, model parameters, search branches, and latent policies. Their effectiveness depends not only on the computation budget but also on whether the additional computation can improve decisions and final solution quality.

Motivation. Prior work provides four relevant foundations: learning based replanning, multi-objective neural construction, world models, and test-time search. However, existing methods do not address how a fixed NCO architecture can learn dynamic adaptation or how world model reasoning can be introduced into NCO to exploit imagined solution consequences for decision making.

3 Problem Formulation

3.1 Dynamic Multi-Objective Routing Problem

Dynamic routing considers sequential decision making under evolving instance information during route execution (Pillac et al., 2013; Ojeda Rios et al., 2021). At decision step t, the observed graph is $\mathcal{G}_t = (\mathcal{V}_t, \mathcal{E}_t, \mathbf{X}_t)$, where $\mathcal{V}_t$, $\mathcal{E}_t$, and $\mathbf{X}_t$ denote the node set, routing edges, and task-specific attributes, respectively. Let $\mathcal{U}_t \subseteq \mathcal{V}_t$ denote active unserved nodes, i_t the current node, q_t the remaining capacity when applicable, and $\mathbf{m}_t$ the feasibility mask. Given preference vector $\boldsymbol{\lambda} \in \Delta^{M-1}$, the state is $s_t = (\mathcal{G}_t, i_t, q_t, \mathbf{m}_t, \boldsymbol{\lambda})$, where q_t is omitted for MOTSP.

The executed prefix $P_t = (a_0, \ldots, a_{t-1})$ is irrevocable and contributes to realized objectives, but is not encoded separately. Its effects are reflected through the current node, remaining resources, and feasibility mask. The feasible action set is $\mathcal{A}(s_t) = \{j \in \mathcal{U}_t : j \text{ is feasible}\} \cup \mathcal{H}_t$, where $\mathcal{H}_t = \varnothing$ for MOTSP and $\mathcal{H}_t = \{0\}$ allows depot returns in CVRP. The operator $\mathcal{R}$ reconstructs current node attributes and feasibility state without revising executed actions. After an action is executed, newly observed events modify the instance before the next decision:

$$\tilde{s}_{t+1} = \mathcal{T}(s_t, a_t), \qquad \mathcal{G}_{t+1} = \Phi_{e_{t+1}}(\tilde{\mathcal{G}}_{t+1}), \qquad s_{t+1} = \mathcal{R}(\tilde{s}_{t+1}, \mathcal{G}_{t+1}). \tag{1}$$

3.2 Dynamic Events and Routing Tasks

At each eligible step, an event is triggered according to $b_t \sim \mathrm{Bernoulli}(p_{\mathrm{evt}})$. When $b_t = 1$, the event type z_t is sampled from the five families $\{\mathrm{Add}, \mathrm{Del}, \mathrm{Drift}, \mathrm{Add{+}Drift}, \mathrm{Del{+}Drift}\}$ according to categorical weights $\boldsymbol{\omega}$. Let $\mathcal{A}_t$ denote the arriving nodes, $\mathcal{D}_t \subseteq \mathcal{U}_t$ the cancelled nodes, and $\mathcal{C}_t \subseteq \mathcal{U}_t$ the nodes with changed attributes. The corresponding atomic updates are defined as

$$\Phi_{\mathrm{Add}}(\mathcal{U}_t) = \mathcal{U}_t \cup \mathcal{A}_t, \qquad \Phi_{\mathrm{Del}}(\mathcal{U}_t) = \mathcal{U}_t \setminus \mathcal{D}_t, \qquad \mathbf{x}_{i,t}^{+} = \Pi_{\mathcal{X}}(\mathbf{x}_{i,t} + \boldsymbol{\delta}_{i,t}),\ i \in \mathcal{C}_t, \tag{2}$$

where $\boldsymbol{\delta}_{i,t} \sim \mathcal{N}(\mathbf{0}, \sigma^2 \mathbf{I})$ and $\Pi_{\mathcal{X}}$ projects attributes to their feasible domain. The two compound events compose addition or deletion with drift. With affected-node ratio ρ and current request count n_t, a positive event changes $\kappa_t = \max\{1, \lceil \rho n_t \rceil\}$ nodes, subject to the number of eligible nodes. Changes act on unserved nodes, while an added node becomes immediately serviceable. Training samples events online; evaluation replays fixed schedules shared by all solvers.

3.3 Decomposition-Based Neural MOCO Methods

Let $\tau_\theta = (a_0, \ldots, a_T)$ denote a feasible route generated by the causal policy $a_t \sim \pi_\theta(\cdot \mid s_t, \boldsymbol{\lambda})$, which conditions only on the current state and observed events. The objective is to minimize the expected objective vector $\mathbb{E}_{\mathbf{e} \sim p_{\mathcal{E}}, \tau_\theta}[\mathbf{F}(\tau_\theta; \mathbf{e})]$ under the event process. A solution dominates another if it is no worse in all objectives and strictly better in at least one. Since the nondominated set

can be large, multi-objective optimization typically seeks a representative approximation (Lust & Teghem, 2010; Herzel et al., 2021). Decomposition transforms the vector optimization problem into preference-specific scalar subproblems (Zhang & Li, 2007). For $\boldsymbol{\lambda} \in \Delta^{M-1}$, we use

$$g_{\boldsymbol{\lambda}}^{\mathrm{WS}}(\mathbf{F}) = \sum_{m=1}^{M} \lambda_m F_m, \qquad g_{\boldsymbol{\lambda}}^{\mathrm{TCH}}(\mathbf{F}) = \max_m \lambda_m |F_m - z_m^\star|. \tag{3}$$

Here $\mathbf{z}^\star$ is a reference point, set to zero in our implementation. The preference-conditioned policy shares parameters across these subproblems and generates solutions for different trade-offs (Lin et al., 2022). When an event occurs, the node set or attributes and feasibility mask are updated and the current instance is re-encoded by the same policy architecture. Dynamic adaptation is therefore defined over the evolving state distribution, while the neural architecture remains unchanged.

4 World-Model-Guided Dynamic Neural Solver

4.1 Framework Overview

As illustrated in Fig. 1, the proposed framework enables adaptive online decision making through neural replanning and selective rollout refinement. Dynamic events update the pending graph while preserving the executed route prefix. The preference-conditioned solver re-encodes the graph and proposes candidate actions, while the learned controller selectively invokes the world model to evaluate their consequences. The best rollout is executed to improve decisions with efficient inference.

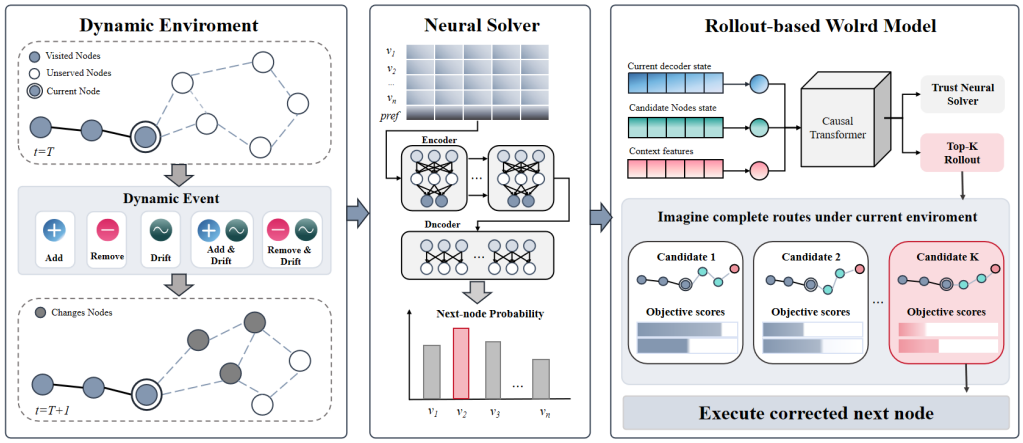


Figure 1: Overview of the proposed world-model-guided adaptive neural routing framework.

4.2 Neural Solver

We instantiate the base solver with the Neat Weight Embedding (NWE) approach to neural multi-objective combinatorial optimization (Chen et al., 2025). In this instantiation, an attention encoder embeds task-specific node features together with the objective preference: two coordinate views for MOTSP, and depot location, customer locations, and demands for MOCVRP. A POMO multi-start decoder (Kwon et al., 2020) conditions each construction trajectory on the encoded graph, its current node, remaining resources, and the feasibility mask:

$$\mathbf{H}_t = \mathrm{Enc}_\theta(\mathcal{G}_t, \boldsymbol{\lambda}), \qquad p_t^{(p)}(a) = \mathrm{Dec}_\theta(a \mid \mathbf{H}_t, i_t^{(p)}, q_t^{(p)}, \mathbf{m}_t^{(p)}, \boldsymbol{\lambda}). \tag{4}$$

Masked attention restricts actions to feasible nodes or valid depot returns. After each event, the updated graph is reprocessed by the same encoder–decoder without revising executed routes. The proposed training and rollout mechanisms only require this action-distribution interface and can extend to other autoregressive NCO solvers.

4.3 ROLLOUT-BASED NCO WORLD MODEL

We construct a rollout-based world model that evaluates candidate consequences under the current graph using the frozen NCO policy. For each feasible candidate a in POMO trajectory p, we fix a as the next node, complete the route, and evaluate its terminal objective under the current preference:

$$\widehat{J}_t^{(p)}(a) = g_{\boldsymbol{\lambda}}\Big[\mathbf{F}\Big(P_t^{(p)} \oplus a \oplus \widehat{\tau}_t^{(p,a)}; \mathcal{G}_t\Big)\Big]. \tag{5}$$

Here, $P_t^{(p)}$ denotes the committed prefix and $\widehat{\tau}_t^{(p,a)}$ denotes the imagined suffix. The rollout keeps $\mathcal{G}_t$ fixed and satisfies feasibility constraints, evaluating the consequences of candidate actions rather than predicting exogenous dynamics. Environmental changes are introduced only through subsequent observed graphs, connecting world-model reasoning with simulation-guided NCO search (Ha & Schmidhuber, 2018; Choo et al., 2022). Since the benefit of lookahead varies across states, a learned rollout allocation policy μ_ψ determines whether additional rollouts are required and how many candidates to evaluate. It takes the preference-conditioned decoder state $\mathbf{d}_t^{(p)}$, candidate competition features $\mathbf{c}_t^{(p)}$, and environment features $\boldsymbol{\xi}_t^{(p)}$ as inputs. The representation aggregates POMO trajectories and causally encodes recent rollout allocation states and actions:

$$\begin{aligned}
&\mathbf{u}_t^{(p)} = \phi_\psi\Big([\mathbf{d}_t^{(p)}, \phi_c(\mathbf{c}_t^{(p)}), \phi_\xi(\boldsymbol{\xi}_t^{(p)})]\Big), \\
&\mathbf{v}_t = \mathrm{AttnPool}_\psi\Big(\{\mathbf{u}_t^{(p)}\}_{p=1}^{P}\Big), \\
&\mathbf{z}_t = \mathrm{CausalTransformer}_\psi\big(\{\mathbf{v}_\tau + \mathrm{Emb}_\psi(K_{\tau-1})\}_{\tau=t-H+1}^{t}\big), \\
&\mu_\psi(\cdot \mid \mathbf{z}_t) = \mathrm{Softmax}(W_\psi \mathbf{z}_t).
\end{aligned} \tag{6}$$

Here, H denotes the causal context length. The rollout allocation policy μ_ψ uses $\mathbf{z}_t$ to select the rollout budget before candidate scores are available, and selected candidates are ranked by $\widehat{J}_t^{(p)}$. The allocation module improves computational efficiency without modifying the NCO backbone.

4.4 TRAINING AND INFERENCE

Training first exposes the base NCO to the five event families in Section 3. Events occur along partially executed routes, update pending obligations, and are revealed before each decision. The objective is terminal scalarized route quality across preferences and event sequences, with the following node-level signal used for policy-gradient training. For POMO lane p, let $c_u^{(p)}$ denote the realized step cost. The suffix cost and POMO advantage are

$$G_t^{(p)} = \sum_{u=t}^{T} c_u^{(p)}, \qquad A_t^{(p)} = \frac{1}{P}\sum_{r=1}^{P} G_t^{(r)} - G_t^{(p)}. \tag{7}$$

The detached advantage weights the selected node's log probability. For MOTSP, the suffix cost provides a stepwise surrogate under tchebycheff or weight scalarization; for MOCVRP, depot returns are aligned with customer stages, while non-additive objectives retain terminal comparison. The cost-to-go signal only trains the base solver and introduces no additional inference module.

After freezing the base NCO, the rollout controller is trained with matched event schedules. Given the current graph state and solver state, the controller predicts whether additional rollout is beneficial and selects the rollout size. Its SKIP/K decisions generate complete routes, and a PPO actor–critic update learns the rollout selection policy from a relative reward measuring the improvement over the standard POMO decision. During inference, the controller first determines whether rollout is required and evaluates candidate suffixes only when activated.

$$K_t = \operatorname*{argmax}_{k \in \mathcal{K}_t} \mu_\psi(k \mid \mathbf{z}_t), \qquad a_t^{(p)} = \begin{cases} \arg\max_a p_t^{(p)}(a), & K_t = 0, \\ \arg\min_{a \in \mathrm{TopK}_{K_t}(p_t^{(p)})} \widehat{J}_t^{(p)}(a), & K_t > 0. \end{cases} \tag{8}$$

Here $K_t = 0$ indicates that rollout is skipped. When rollout is activated, only the next executed node can be modified; subsequent events trigger graph updates and re-encoding before the next decision. The same mechanism applies to static instances with inactive event indicators.

5 EXPERIMENTS

5.1 EXPERIMENTAL SETUP

Benchmarks. We consider two bi-objective routing benchmarks: MOTSP (Lust & Teghem, 2010) and MOCVRP (Zajac & Huber, 2021). Following standard neural combinatorial optimization settings (Kwon et al., 2020), node coordinates are sampled uniformly from $[0, 1]^2$, with training sizes $n \in \{20, \ldots, 100\}$. Dynamic profiles D1-D3 use event probabilities $(0.10, 0.20, 0.30)$, affected-node ratios $(0.05, 0.10, 0.15)$, and drift scales $(0.050, 0.075, 0.100)$. We evaluate controlled settings with $n \in \{20, 50, 100\}$, size generalization with $n \in \{150, 200\}$, and benchmark instances using KroAB100/150/200 and XML100/150/200 for MOTSP and MOCVRP, respectively. The default setting combines all five event types, with 200 synthetic instances generated using event seed 1234 and three fixed event schedules reused for benchmark instances.

Compared Methods. We compare against four groups of methods. (1) The weighted-sum solver **WS-LKH** for MOTSP (Helsgaun, 2000; Tinós et al., 2018), using 101 preference weights, one independent LKH run per weight, and 10 maximum trials at each replanning step. (2) Multi-objective metaheuristics, including **MOGLS** (Jaszkiewicz, 2002), **NSGA-II** (Deb et al., 2002), and **MOEA/D** (Zhang & Li, 2007); NSGA-II and MOEA/D use populations of 101 solutions with 100 initial generations and 30 adaptation generations after each event, while MOGLS uses 101 solutions with 10^5 initial evaluations and 10^{-4} adaptation evaluations after each event. (3) The pretrained static **WECA** neural solver (Chen et al., 2025), which re-encodes the observed graph after each change. (4) Dynamic learning based methods, including **MAPT** (Zou et al., 2026) and **SIT** (Luo et al., 2024). Additionally, to evaluate the effectiveness of world model guided lookahead on static instances, we combine the trained world model with the pretrained static **WECA** solver (Chen et al., 2025). The resulting static WECA-WM is compared with state of the art neural multi-objective solvers, including **MDRL** (Zhang et al., 2023), **EMNH** (Chen et al., 2023), **POCCO-C**, and **POCCO-W** (Fan et al., 2026), as well as heuristic baselines.

Training details. WECA uses 128-dimensional embeddings, six encoder layers, and eight attention heads, and is trained for 200 epochs with 10^5 instances per epoch and batch size 64. Adam uses learning rate 10^{-4} and weight decay 10^{-6}; dynamic training uses event probability 0.3, affected-node ratio 0.01, and drift scale 0.1. The WM uses a two-layer causal Transformer with hidden size 384, six heads, feed-forward size 768, and context length four, and is trained with PPO for 200 epochs with 10^4 episodes per epoch, batch size 64, learning rate 10^{-5}, weight decay 10^{-5}, and clipping threshold 0.2. Its action space includes SKIP and Top-K with $K \in \{2, 3, 4\}$. MAPT uses 128-dimensional embeddings, two attention heads, five encoder blocks, and three decoder blocks, and is trained with PPO for 200 epochs with 10^5 instances per epoch, batch size 256, and learning rate 10^{-4}. SIT uses 128-dimensional embeddings and is trained for 200 epochs with 10^5 instances per epoch, batch size 64, event probability 0.3, change ratio 0.01, drift scale 0.1, and learning rate 10^{-4}. Experiments are conducted on Ubuntu 22.04 with six 32-GB virtual GPUs and 72 Intel Xeon Platinum 8375C vCPUs using Python 3.12, PyTorch 2.7.0, and CUDA 12.8.

Evaluation protocol. We use 101 uniformly spaced preference vectors to construct an approximate Pareto front and evaluate it by normalized hypervolume (HV), which captures convergence and diversity; higher values indicate better solution sets (While et al., 2006; Audet et al., 2021). Except for KroAB and XML with single instance, all experiments are evaluated on 200 randomly generated test instances to improve the robustness and reliability of the reported results. Ideal and reference points are fixed for each task, suite, profile, scale, and seed. Every method replays the same initial instance, committed prefix, and event sequence, and observes each event only before the subsequent decision. Statistical significance is assessed by a two-sided Wilcoxon signed-rank test at the 1% level. The best and second-best results are shown in **bold** and underline, respectively.

Table 1: Average comparison results over three dynamic configurations (D1–D3) on Bi-TSP and Bi-CVRP benchmarks.

Method	Bi-TSP20			Bi-TSP50			Bi-TSP100		
	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓
WS-LKH	0.6630	7.56%	17.2m	0.5516	31.42%	1.6h	0.6756	17.93%	8.2h
MOGLS	0.6405	10.70%	5.9m	0.4351	45.91%	19.3m	0.3280	60.16%	43.5m
NSGA-II	0.6383	11.01%	22.5m	0.4224	47.49%	37.5m	0.3203	61.09%	45.0m
MOEA/D	0.6491	9.49%	13.9m	0.4320	46.29%	30.0m	0.3279	60.16%	43.0m
MAPT	0.6672	6.98%	13s	0.7460	7.26%	27s	0.7954	3.37%	3.8m
SIT	0.6771	5.59%	35s	0.7508	6.66%	2.7m	0.7961	3.29%	12.8m
Static-WECA	0.7071	1.41%	4s	0.7915	1.60%	6s	0.8030	2.45%	24s
WECA-WM	**0.7172**	**0.00%**	17s	**0.8044**	**0.00%**	40s	**0.8231**	**0.00%**	1.3m
Method	Bi-CVRP20			Bi-CVRP50			Bi-CVRP100		
	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓
MOGLS	0.4183	7.54%	6.4m	0.3148	32.91%	17.2m	0.0926	81.34%	12.0m
NSGA-II	0.3779	16.48%	48.7m	0.2112	54.99%	1.5h	0.0789	84.10%	1.0h
MOEA/D	0.3907	13.65%	31.4m	0.2608	44.42%	1.2h	0.1234	75.14%	2.2h
MAPT	0.4089	9.62%	20s	0.4375	6.77%	47s	0.4614	7.07%	2.4m
SIT	0.3358	25.78%	51s	0.3844	18.09%	3.2m	0.4172	15.97%	8.4m
Static-WECA	0.4405	2.62%	7s	0.4359	7.11%	16s	0.4603	7.28%	45s
WECA-WM	**0.4524**	**0.00%**	13s	**0.4693**	**0.00%**	19s	**0.4965**	**0.00%**	1.2m

5.2 Experimental Results under Dynamic Environments

Comparison Analysis Table 1 reports the average results over three dynamic configurations, with detailed results provided in the Appendix. WECA-WM consistently achieves the highest HV on both Bi-TSP and Bi-CVRP, outperforming static and dynamic learning-based baselines across different problem sizes. The improvements demonstrate that world model guidance effectively enhances neural decision making under dynamic environments, while maintaining efficient inference and a favorable quality efficiency trade-off.

Out-of-Distribution Size Generalization Fig. 2 presents the HV results on large scale Bi-TSP instances under D1-D3 settings. WECA-WM consistently achieves the highest HV across all scales and dynamic profiles, outperforming both traditional optimization and learning based methods. The improvements over Static-WECA, MAPT, and SIT demonstrate the effectiveness of world model guided future consequence reasoning for dynamic decision making. Compared with WS-LKH, the proposed framework maintains strong solution quality while providing efficient neural inference, showing improved scalability and adaptability on large scale dynamic routing problems.

Ablation Study Figure. 3 presents the ablation results of WECA-WM on Bi-TSP and Bi-CVRP benchmarks. The full model achieves the highest HV, demonstrating the effectiveness of combining world model guided lookahead with cost-to-go based credit assignment. Removing the world model (w/o WM) degrades performance, confirming the value of future consequence evaluation, while further removing cost-to-go (w/o CTG) causes additional drops, indicating improved credit assignment under dynamic changes. These results verify the complementary benefits of the two components.

5.3 Experimental Results under Static Environments

Table 2 compares WECA-WM, which combines the pretrained WECA solver (Chen et al., 2025) with a world model trained under dynamic environments, against conventional optimization algorithms, static neural combinatorial optimization methods, and graph augmentation variants on Bi-TSP and Bi-CVRP benchmarks. Graph augmentation provides consistent improvements for static NCO solvers, but the gains become limited as problem size increases. By incorporating future solution consequences through the world model, WECA-WM further improves over Static-WECA

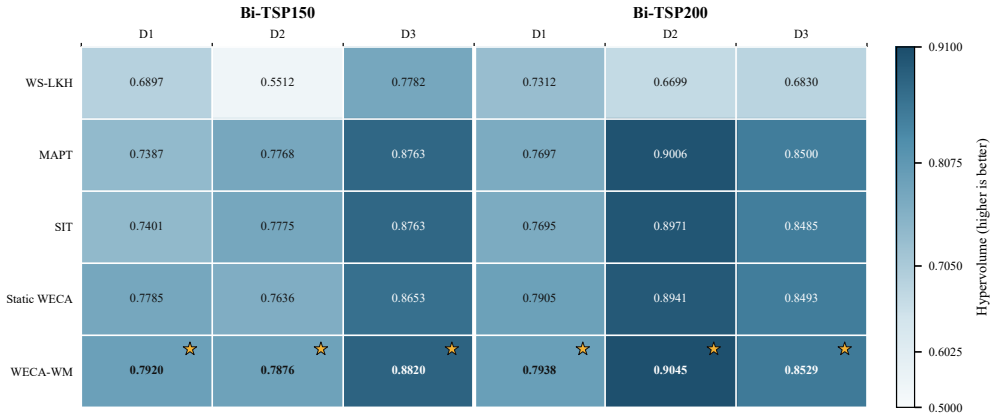


Figure 2: Scalability and dynamic robustness analysis on large-scale Bi-TSP instances.

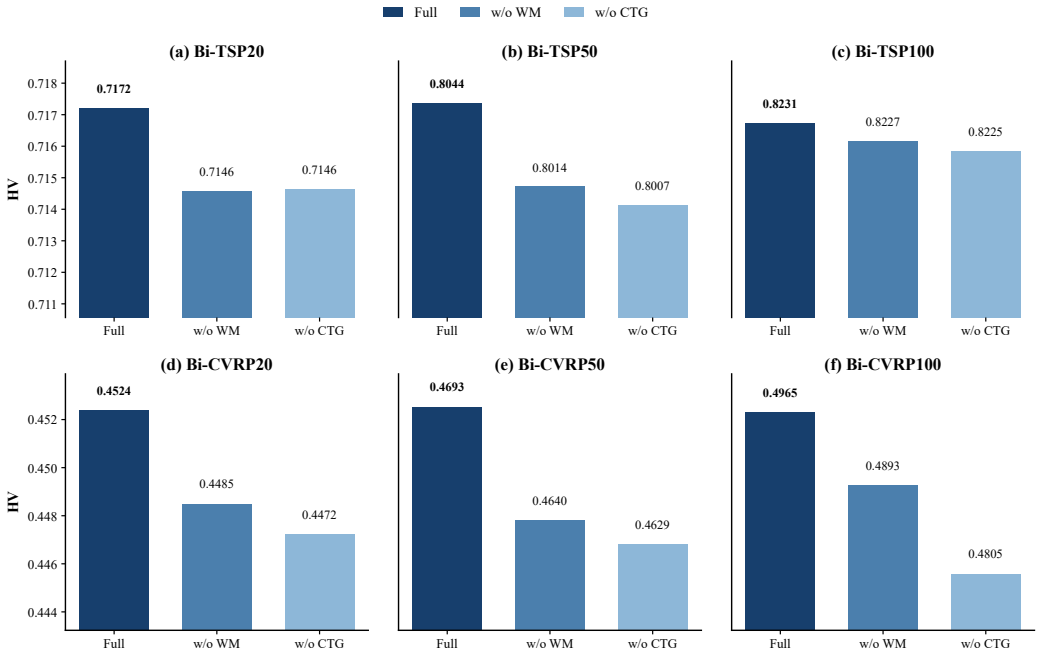


Figure 3: Scalability and dynamic robustness analysis on large-scale Bi-TSP instances.

and augmentation based variants. In particular, WECA-WM-Aug achieves the best HV results Bi-CVRP20/50/100, demonstrating the compatibility between world model guidance and graph augmentation. These results show that the world model provides complementary decision level improvements beyond representation enhancement and strengthens neural combinatorial optimization.

Figure 4 visualizes the Pareto fronts of WECA-WM-Aug on KroAB benchmarks. WECA-WM-Aug achieves competitive solution sets across all problem sizes and consistently improves over neural augmentation variants, obtaining higher HV values on KroAB100, KroAB150, and KroAB200. These results further demonstrate that world model guidance provides complementary decision refinement beyond graph augmentation, with sustained benefits on larger instances.

Table 2: Comparison with existing NCO solvers and graph augmentation strategies on static Bi-TSP and Bi-CVRP benchmarks.

Category	Method	Bi-TSP20			Bi-TSP50			Bi-TSP100		
		HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓
Non-learnable	WS-LKH	0.6270	0.02%	10m	0.6415	-0.05%	1.8h	**0.7090**	**-0.33%**	6.0h
	MOGLS	**0.6279**	-0.12%	1.6h	0.6330	1.28%	3.7h	0.6854	3.01%	11h
	MOEA/D	0.6241	0.48%	1.7h	0.6316	1.49%	1.8h	0.6899	2.37%	2.2h
	NSGA-II	0.6258	0.21%	6.0h	0.6120	4.55%	6.1h	0.6692	5.30%	6.9h
Static NCO	MDRL	0.6271	0.00%	5s	0.6364	0.75%	9s	0.6969	1.38%	17s
	EMNH	0.6271	0.00%	5s	0.6364	0.75%	9s	0.6969	1.38%	16s
	WECA	0.6270	0.02%	7s	0.6392	0.31%	10s	0.7034	0.46%	21s
	POCCO-C	0.6275	-0.06%	14s	0.6409	0.04%	20s	0.7047	0.28%	42s
	POCCO-W	0.6275	-0.06%	7s	0.6411	0.01%	14s	0.7055	0.16%	36s
Graph Aug.	MDRL-Aug	0.6271	0.00%	33s	0.6408	0.06%	1.7m	0.7022	0.63%	14m
	EMNH-Aug	0.6271	0.00%	33s	0.6408	0.06%	1.7m	0.7023	0.62%	14m
	WECA-Aug	0.6271	0.00%	1.3m	0.6413	-0.02%	3.6m	0.7066	0.01%	12m
	POCCO-C-Aug	0.6270	0.02%	2.2m	0.6416	-0.07%	4.0m	0.7071	-0.06%	14m
	POCCO-W-Aug	0.6270	0.02%	2.2m	**0.6418**	-0.10%	4.0m	0.7078	-0.16%	14m
Ours	**WECA-WM**	0.6273	-0.03%	22s	0.6415	0.00%	136s	0.7071	0.00%	1019s
	WECA-WM-Aug	0.6270	0.02%	5.5m	0.6416	-0.07%	1.5h	0.7086	-0.27%	12h

Category	Method	Bi-CVRP20			Bi-CVRP50			Bi-CVRP100		
		HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓
Non-learnable	MOEA/D	0.4255	1.07%	2.3h	0.4000	2.53%	2.9h	0.3953	3.16%	5.0h
	NSGA-II	0.4275	0.60%	6.4h	0.3896	5.06%	8.8h	0.3620	11.32%	9.4h
	MOGLS	0.4278	0.54%	9.0h	0.3984	2.92%	20h	0.3875	5.07%	72h
Static NCO	MDRL	0.4291	0.23%	8s	0.4082	0.53%	13s	0.4056	0.64%	32s
	EMNH	0.4299	0.05%	7s	0.4098	0.14%	13s	0.4072	0.24%	31s
	WECA	0.4290	0.26%	7s	0.4089	0.36%	10s	0.4068	0.34%	21s
	POCCO-C	0.4294	0.16%	16s	0.4101	0.07%	25s	0.4079	0.07%	53s
	POCCO-W	0.4294	0.16%	8s	0.4102	0.04%	17s	0.4084	-0.05%	46s
Graph Aug.	MDRL-Aug	0.4294	0.16%	11s	0.4092	0.29%	36s	0.4072	0.24%	2.8m
	EMNH-Aug	0.4302	-0.02%	11s	0.4106	-0.05%	35s	0.4079	0.07%	2.8m
	WECA-Aug	0.4300	0.02%	15s	0.4103	0.02%	36s	0.4081	0.02%	1.8m
	POCCO-C-Aug	0.4302	-0.02%	31s	0.4108	-0.10%	1.4m	0.4086	-0.10%	2.4m
	POCCO-W-Aug	0.4301	0.00%	24s	0.4108	-0.10%	1.2m	0.4089	-0.17%	2.3m
Ours	**WECA-WM**	0.4301	0.00%	55s	0.4107	-0.07%	7.8m	0.4088	-0.15%	1.5h
	WECA-WM-Aug	**0.4305**	-0.05%	3.0m	**0.4112**	-0.19%	1.1h	**0.4093**	-0.27%	13h

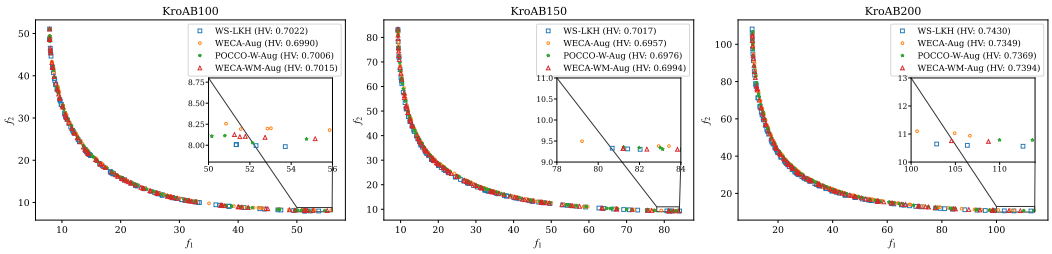


Figure 4: Pareto front comparison on Static KroAB benchmarks.

6 CONCLUSION

This work establishes a training and evaluation framework for dynamic multi objective neural combinatorial optimization. We formalize five event families and inject them into partially executed trajectories, enabling unchanged preference conditioned NCO architectures to learn dynamic adaptation. We further introduce a world model formulation for NCO that uses imagined route completions to estimate decision consequences and guide online node selection. A node level cost to go signal improves dynamic training through decision aware credit assignment. Experiments on MOTSP and MOCVRP across diverse dynamic profiles and problem sizes demonstrate improved solution quality, efficient inference, and strong cross scale generalization. The adaptive lookahead also improves frozen NCO policies on static instances.

REFERENCES

Charles Audet, Jean Bigeon, Dominique Cartier, Sébastien Le Digabel, and Ludovic Salomon. Performance indicators in multiobjective optimization. *European Journal of Operational Research*, 292(2):397–422, July 2021. ISSN 0377-2217. doi: 10.1016/j.ejor.2020.11.016.

Yoshua Bengio, Andrea Lodi, and Antoine Prouvost. Machine learning for combinatorial optimization: A methodological tour d'horizon. *European Journal of Operational Research*, 290(2): 405–421, April 2021. ISSN 0377-2217. doi: 10.1016/j.ejor.2020.07.063.

Felix Chalumeau, Shikha Surana, Clement Bonnet, Nathan Grinsztajn, Arnu Pretorius, Alexandre Laterre, and Thomas D. Barrett. Combinatorial optimization with policy adaptation using latent space search. In *Advances in Neural Information Processing Systems*, volume 36, pp. 7947–7959, May 2023. doi: 10.48550/arXiv.2311.13569.

Jinbiao Chen, Jiahai Wang, Zizhen Zhang, Zhiguang Cao, Te Ye, and Siyuan Chen. Efficient meta neural heuristic for multi-objective combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 36, pp. 56825–56837, October 2023.

Jinbiao Chen, Zhiguang Cao, Jiahai Wang, Yaoxin Wu, Hanzhang Qin, Zizhen Zhang, and Yue-Jiao Gong. Rethinking neural multi-objective combinatorial optimization via neat weight embedding. In *International Conference on Learning Representations*, volume 2025, pp. 48510–48529, May 2025.

Rensheng Chen, Bin Wu, Hua Wang, Huagang Tong, and Feiyi Yan. A q -learning based nsga-ii for dynamic flexible job shop scheduling with limited transportation resources. *Swarm and Evolutionary Computation*, 90:101658, October 2024. ISSN 22106502. doi: 10.1016/j.swevo.2024.101658.

Xinwei Chen, Marlin W. Ulmer, and Barrett W. Thomas. Deep q-learning for same-day delivery with vehicles and drones. *European Journal of Operational Research*, 298(3):939–952, May 2022. ISSN 0377-2217. doi: 10.1016/j.ejor.2021.06.021.

Xinyun Chen and Yuandong Tian. Learning to perform local rewriting for combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 32, October 2019.

Jinho Choo, Yeong-Dae Kwon, Jihoon Kim, Jeongwoo Jae, André Hottung, Kevin Tierney, and Youngjune Gwon. Simulation-guided beam search for neural combinatorial optimization. In *Advances in Neural Information Processing Systems*, volume 35, pp. 8760–8772. Curran Associates, Inc., 2022. doi: 10.52202/068431-0637.

Kurtland Chua, Roberto Calandra, Rowan McAllister, and Sergey Levine. Deep reinforcement learning in a handful of trials using probabilistic dynamics models. In *Advances in Neural Information Processing Systems*, volume 31. Curran Associates, Inc., 2018.

K. Deb, A. Pratap, S. Agarwal, and T. Meyarivan. A fast and elitist multiobjective genetic algorithm: Nsga-ii. *IEEE Transactions on Evolutionary Computation*, 6(2):182–197, April 2002. ISSN 1941-0026. doi: 10.1109/4235.996017.

Mingfeng Fan, Yaoxin Wu, Zhiguang Cao, Wen Song, Guillaume Sartoretti, Huan Liu, and Guohua Wu. Conditional neural heuristic for multiobjective vehicle routing problems. *IEEE Transactions on Neural Networks and Learning Systems*, 36(3):4677–4689, March 2025. ISSN 2162-2388. doi: 10.1109/TNNLS.2024.3371706.

Mingfeng Fan, Jianan Zhou, Yifeng Zhang, Yaoxin Wu, Jinbiao Chen, and Guillaume Adrien Sartoretti. Preference-driven multi-objective combinatorial optimization with conditional computation. In *Advances in Neural Information Processing Systems*, volume 38, pp. 11594–11627, June 2026. doi: 10.48550/arXiv.2506.08898.

David Ha and Jürgen Schmidhuber. World models. In *Advances in Neural Information Processing Systems*, March 2018. doi: 10.5281/zenodo.1207631.

Danijar Hafner, Timothy Lillicrap, Jimmy Ba, and Mohammad Norouzi. Dream to control: Learning behaviors by latent imagination. In *International Conference on Learning Representations*, September 2019a.

Danijar Hafner, Timothy Lillicrap, Ian Fischer, Ruben Villegas, David Ha, Honglak Lee, and James Davidson. Learning latent dynamics for planning from pixels. In *Proceedings of the 36th International Conference on Machine Learning*, pp. 2555–2565. PMLR, May 2019b.

Danijar Hafner, Jurgis Pasukonis, Jimmy Ba, and Timothy Lillicrap. Mastering diverse control tasks through world models. *Nature*, 640(8059):647–653, April 2025. ISSN 1476-4687. doi: 10.1038/s41586-025-08744-2.

Nicklas A. Hansen, Hao Su, and Xiaolong Wang. Temporal difference learning for model predictive control. In *Proceedings of the 39th International Conference on Machine Learning*, pp. 8387–8406. PMLR, June 2022.

Keld Helsgaun. An effective implementation of the lin–kernighan traveling salesman heuristic. *European Journal of Operational Research*, 126(1):106–130, October 2000. ISSN 0377-2217. doi: 10.1016/S0377-2217(99)00284-2.

Arne Herzel, Stefan Ruzika, and Clemens Thielen. Approximation methods for multiobjective optimization problems: A survey. *INFORMS Journal on Computing*, 33(4):1284–1299, October 2021. ISSN 1091-9856. doi: 10.1287/ijoc.2020.1028.

André Hottung and Kevin Tierney. Neural large neighborhood search for the capacitated vehicle routing problem. In *24th European Conference on Artificial Intelligence*, 2020. doi: 10.3233/FAIA200124.

André Hottung, Yeong-Dae Kwon, and Kevin Tierney. Efficient active search for combinatorial optimization problems. In *International Conference on Learning Representations*, March 2022. doi: 10.48550/arXiv.2106.05126.

Michael Janner, Justin Fu, Marvin Zhang, and Sergey Levine. When to trust your model: Model-based policy optimization. In *Advances in Neural Information Processing Systems*, volume 32. Curran Associates, Inc., 2019.

Andrzej Jaszkiewicz. Genetic local search for multi-objective combinatorial optimization. *European Journal of Operational Research*, 137(1):50–71, February 2002. ISSN 0377-2217. doi: 10.1016/S0377-2217(01)00104-7.

Wouter Kool, Herke van Hoof, and Max Welling. Attention, learn to solve routing problems! In *International Conference on Machine Learning*. arXiv, February 2019. doi: 10.48550/arXiv.1803.08475.

Yeong-Dae Kwon, Jinho Choo, Byoungjip Kim, Iljoo Yoon, Youngjune Gwon, and Seungjai Min. Pomo: Policy optimization with multiple optima for reinforcement learning. In *Advances in Neural Information Processing Systems*, volume 33, pp. 21188–21198. Curran Associates, Inc., 2020.

Meng Li, Kaiquan Cai, and Peng Zhao. Optimizing same-day delivery with vehicles and drones: A hierarchical deep reinforcement learning approach. *Transportation Research Part E: Logistics and Transportation Review*, 193:103878, January 2025. ISSN 1366-5545. doi: 10.1016/j.tre.2024.103878.

Weiqi Li and Mingyuan Feng. A parallel procedure for dynamic multi-objective tsp. In *2012 IEEE 10th International Symposium on Parallel and Distributed Processing with Applications*, pp. 1–8, July 2012. doi: 10.1109/ISPA.2012.10.

Xi Lin, Zhiyuan Yang, and Qingfu Zhang. Pareto set learning for neural multi-objective combinatorial optimization. In *International Conference on Learning Representations*. arXiv, May 2022. doi: 10.48550/arXiv.2203.15386.

Jiaojiao Liu, Baofeng Sun, Gendao Li, and Yuqi Chen. Multi-objective adaptive large neighbourhood search algorithm for dynamic flexible job shop schedule problem with transportation resource. *Engineering Applications of Artificial Intelligence*, 132:107917, June 2024. ISSN 0952-1976. doi: 10.1016/j.engappai.2024.107917.

Fu Luo, Xi Lin, Yaoxin Wu, Zhenkun Wang, Tong Xialiang, Mingxuan Yuan, and Qingfu Zhang. Boosting neural combinatorial optimization for large-scale vehicle routing problems. In *The Thirteenth International Conference on Learning Representations*, October 2024.

Shu Luo, Linxuan Zhang, and Yushun Fan. Dynamic multi-objective scheduling for flexible job shop by deep reinforcement learning. *Computers & Industrial Engineering*, 159:107489, September 2021. ISSN 0360-8352. doi: 10.1016/j.cie.2021.107489.

Thibaut Lust and Jacques Teghem. The multiobjective traveling salesman problem: A survey and a new approach. In Carlos A. Coello Coello, Clarisse Dhaenens, and Laetitia Jourdan (eds.), *Advances in Multi-Objective Nature Inspired Computing*, pp. 119–141. Springer, Berlin, Heidelberg, 2010. ISBN 978-3-642-11218-8. doi: 10.1007/978-3-642-11218-8_6.

George Mavrotas, José Rui Figueira, and Eleftherios Siskos. Robustness analysis methodology for multi-objective combinatorial optimization problems and application to project selection. *Omega*, 52:142–155, April 2015. ISSN 0305-0483. doi: 10.1016/j.omega.2014.11.005.

Brenner Humberto Ojeda Rios, Eduardo C. Xavier, Flávio K. Miyazawa, Pedro Amorim, Eduardo Curcio, and Maria João Santos. Recent dynamic vehicle routing problems: A survey. *Computers & Industrial Engineering*, 160:107604, October 2021. ISSN 0360-8352. doi: 10.1016/j.cie.2021.107604.

Victor Pillac, Michel Gendreau, Christelle Guéret, and Andrés L. Medaglia. A review of dynamic vehicle routing problems. *European Journal of Operational Research*, 225(1):1–11, February 2013. ISSN 0377-2217. doi: 10.1016/j.ejor.2012.08.015.

Julian Schrittwieser, Ioannis Antonoglou, Thomas Hubert, Karen Simonyan, Laurent Sifre, Simon Schmitt, Arthur Guez, Edward Lockhart, Demis Hassabis, Thore Graepel, Timothy Lillicrap, and David Silver. Mastering atari, go, chess and shogi by planning with a learned model. *Nature*, 588 (7839):604–609, December 2020. ISSN 1476-4687. doi: 10.1038/s41586-020-03051-4.

Jiwoo Son, Zhikai Zhao, Federico Berto, Chuanbo Hua, Changhyun Kwon, and Jinkyoo Park. Neural combinatorial optimization for real-world routing. In *International Conference on Learning Representations*. arXiv, March 2025. doi: 10.48550/arXiv.2503.16159.

Byung Duk Song, Hyorin Park, and Kyungsu Park. Toward flexible and persistent uav service: Multi-period and multi-objective system design with task assignment for disaster management. *Expert Systems with Applications*, 206:117855, November 2022. ISSN 0957-4174. doi: 10.1016/j.eswa.2022.117855.

İstenç Tarhan, Konstantinos G. Zografos, Juliana Sutanto, and Ahmed Kheiri. A quadrant shrinking heuristic for solving the dynamic multi-objective disaster response personnel routing and scheduling problem. *European Journal of Operational Research*, 314(2):776–791, April 2024. ISSN 0377-2217. doi: 10.1016/j.ejor.2023.09.002.

Renato Tinós, Keld Helsgaun, and Darrell Whitley. Efficient recombination in the lin-kernighan-helsgaun traveling salesman heuristic. In Anne Auger, Carlos M. Fonseca, Nuno Lourenço, Penousal Machado, Luís Paquete, and Darrell Whitley (eds.), *Parallel Problem Solving from Nature – PPSN XV*, pp. 95–107, Cham, 2018. Springer International Publishing. ISBN 978-3-319-99253-2. doi: 10.1007/978-3-319-99253-2_8.

Federico Julian Camerota Verdù, Lorenzo Castelli, and Luca Bortolussi. Scaling combinatorial optimization neural improvement heuristics with online search and adaptation. *Proceedings of the AAAI Conference on Artificial Intelligence*, 39(25):27135–27143, April 2025. ISSN 2374-3468. doi: 10.1609/aaai.v39i25.34921.

Feng Wang, Fanshu Liao, Yixuan Li, Xuesong Yan, and Xu Chen. An ensemble learning based multi-objective evolutionary algorithm for the dynamic vehicle routing problem with time windows. *Computers & Industrial Engineering*, 154:107131, April 2021. ISSN 03608352. doi: 10.1016/j.cie.2021.107131.

Hao Wang, Junfu Cheng, Chang Liu, Yuanyuan Zhang, Shunfang Hu, and Liangyin Chen. Multi-objective reinforcement learning framework for dynamic flexible job shop scheduling problem with uncertain events. *Applied Soft Computing*, 131:109717, December 2022. ISSN 15684946. doi: 10.1016/j.asoc.2022.109717.

Yong Wang, Mengyuan Gou, Siyu Luo, Jianxin Fan, and Haizhong Wang. The multi-depot pickup and delivery vehicle routing problem with time windows and dynamic demands. *Engineering Applications of Artificial Intelligence*, 139:109700, January 2025. ISSN 0952-1976. doi: 10.1016/j.engappai.2024.109700.

Zhenkun Wang, Shunyu Yao, Genghui Li, and Qingfu Zhang. Multiobjective combinatorial optimization using a single deep reinforcement learning model. *IEEE Transactions on Cybernetics*, 54(3):1984–1996, March 2024. ISSN 2168-2275. doi: 10.1109/TCYB.2023.3312476.

L. While, P. Hingston, L. Barone, and S. Huband. A faster algorithm for calculating hypervolume. *IEEE Transactions on Evolutionary Computation*, 10(1):29–38, February 2006. ISSN 1941-0026. doi: 10.1109/TEVC.2005.851275.

Ming Yang, Lishan Kang, and Jing Guan. Multi-algorithm co-evolution strategy for dynamic multi-objective tsp. In *2008 IEEE Congress on Evolutionary Computation (IEEE World Congress on Computational Intelligence)*, pp. 466–471, June 2008. doi: 10.1109/CEC.2008.4630839.

Te Ye, Zizhen Zhang, Qingfu Zhang, Jinbiao Chen, and Jiahai Wang. Solving multiobjective combinatorial optimization via learning to improve method. *IEEE Transactions on Emerging Topics in Computational Intelligence*, 9(3):2122–2136, June 2025. ISSN 2471-285X. doi: 10.1109/TETCI.2025.3540424.

James J. Q. Yu, Wen Yu, and Jiatao Gu. Online vehicle routing with neural combinatorial optimization and deep reinforcement learning. *IEEE Transactions on Intelligent Transportation Systems*, 20(10):3806–3817, October 2019. ISSN 1558-0016. doi: 10.1109/TITS.2019.2909109.

Sandra Zajac and Sandra Huber. Objectives and methods in multi-objective routing problems: A survey and classification scheme. *European Journal of Operational Research*, 290(1):1–25, April 2021. ISSN 0377-2217. doi: 10.1016/j.ejor.2020.07.005.

Qingfu Zhang and Hui Li. Moea/d: A multiobjective evolutionary algorithm based on decomposition. *IEEE Transactions on Evolutionary Computation*, 11(6):712–731, December 2007. ISSN 1941-0026. doi: 10.1109/TEVC.2007.892759.

Zizhen Zhang, Zhiyuan Wu, Hang Zhang, and Jiahai Wang. Meta-learning-based deep reinforcement learning for multiobjective optimization problems. *IEEE Transactions on Neural Networks and Learning Systems*, 34(10):7978–7991, October 2023. ISSN 2162-2388. doi: 10.1109/TNNLS.2022.3148435.

Zengyu Zou, Jingyuan Wang, Yixuan Huang, and Junjie Wu. Multi-agent pointer transformer: Seq-to-seq reinforcement learning for multi-vehicle dynamic pickup-delivery problems. *Proceedings of the AAAI Conference on Artificial Intelligence*, 40(19):16593–16601, March 2026. ISSN 2374-3468. doi: 10.1609/aaai.v40i19.38700.

A Dynamic Multi-Objective Routing Problems

A.1 Dynamic Multi-Objective Traveling Salesman Problem

An instance contains an evolving set of nodes and M cost matrices $\mathbf{C}_t^{(m)} = (c_{ij}^{(m)}(t))$, one per objective. In the Euclidean setting, node i has a coordinate $\mathbf{x}_{i,t}^{(m)} \in [0,1]^2$ for each objective, and

$$c_{ij}^{(m)}(t) = \left\| \mathbf{x}_{i,t}^{(m)} - \mathbf{x}_{j,t}^{(m)} \right\|_2, \qquad m \in \{1, \dots, M\}. \tag{9}$$

Let $\pi = (\pi_1, \dots, \pi_N)$ be the realized tour, with $\pi_{N+1} = \pi_1$. Each request that remains active is visited exactly once; N can change as requests arrive or are cancelled before service. If τ_k is the step at which edge (π_k, π_{k+1}) is traversed, the objective vector $\mathbf{F}^{\mathrm{MOTSP}} = (F_1^{\mathrm{MOTSP}}, \dots, F_M^{\mathrm{MOTSP}})$ has components

$$F_m^{\mathrm{MOTSP}}(\pi; \mathbf{e}) = \sum_{k=1}^{N} c_{\pi_k \pi_{k+1}}^{(m)}(\tau_k), \qquad m \in \{1, \dots, M\}, \tag{10}$$

where $\mathbf{e}$ is the observed event sequence. Addition inserts a pending node, deletion removes an unvisited node, and coordinate drift changes future edge costs. The executed part of π and its realized costs remain fixed. With no events, the expression reduces to the standard multi-objective tour length.

A.2 Dynamic Multi-Objective Capacitated Vehicle Routing Problem

An instance contains depot 0 and an evolving set of customers. Customer i has location $\mathbf{y}_{i,t} \in [0,1]^2$ and normalized demand $d_{i,t} \in (0,1]$; depot demand is zero and vehicle capacity is 1. A feasible solution $\mathcal{Q}$ consists of depot-to-depot routes that serve each active customer exactly once. For a route $r = (0, v_1, \dots, v_\ell, 0)$, let σ_k denote the service step of customer v_k and τ_k the traversal step of edge (v_k, v_{k+1}). Its accepted demand and realized length are

$$D(r; \mathbf{e}) = \sum_{k=1}^{\ell} d_{v_k, \sigma_k} \le 1, \qquad L(r; \mathbf{e}) = \sum_{k=0}^{\ell} \left\| \mathbf{y}_{v_k, \tau_k} - \mathbf{y}_{v_{k+1}, \tau_k} \right\|_2, \qquad v_0 = v_{\ell+1} = 0. \tag{11}$$

At step t, customer j is selectable only if $d_{j,t} \le q_t$; returning to the depot replenishes capacity for a subsequent route. The objective vector used in this study is

$$\mathbf{F}^{\mathrm{MOCVRP}}(\mathcal{Q}; \mathbf{e}) = \left(\sum_{r \in \mathcal{Q}} L(r; \mathbf{e}),\ \max_{r \in \mathcal{Q}} L(r; \mathbf{e}) \right). \tag{12}$$

Its components measure total travel distance and the longest vehicle route. Customer additions and cancellations update pending service obligations, while demand drift updates the capacity mask. Completed services, traversed edges, and their realized costs remain fixed.

B Details of Weight Embedding with Conditional Attention Neural Solver

B.1 Task Inputs and Initial Embeddings

We instantiate the preference-conditioned neural solver with weight embedding and conditional attention (WE-CA) from Neat Weight Embedding; the experimental tables abbreviate this backbone as WECA (Chen et al., 2025). At decision step t, its encoder receives node features from the currently observed graph $\mathcal{G}_t$ and a preference vector $\boldsymbol{\lambda} \in \Delta^{M-1}$. For MOTSP, each node concatenates

one two-dimensional coordinate per objective into a $2M$-dimensional feature. For MOCVRP, each customer has a three-dimensional feature comprising its location and normalized demand, whereas the depot has a separate two-dimensional location feature.

The feasibility mask excludes visited or cancelled nodes and, in MOCVRP, customers whose demand exceeds the remaining capacity. WECA maps the preference and node features into a common d-dimensional space. The initial preference and node tokens are

$$\mathbf{h}_{\lambda,t}^{(0)} = W_\lambda \boldsymbol{\lambda} + \mathbf{b}_\lambda, \qquad \mathbf{h}_{i,t}^{(0)} = W_{\mathrm{type}(i)}^{v} \mathbf{v}_{i,t} + \mathbf{b}_{\mathrm{type}(i)}^{v}. \tag{13}$$

All MOTSP nodes share one feature projection; MOCVRP uses separate depot and customer projections. One parameter set processes all preference vectors through the preference token. In our experiments, $d = 128$.

B.2 Conditional-Attention Encoder and POMO Decoder

Each encoder layer uses the current preference representation to modulate node representations before attention. At layer l, the feature-wise scale and shift are $\boldsymbol{\gamma}_t^{(l)} = W_\gamma^{(l)} \mathbf{h}_{\lambda,t}^{(l-1)}$ and $\boldsymbol{\beta}_t^{(l)} = W_\beta^{(l)} \mathbf{h}_{\lambda,t}^{(l-1)}$. They produce the conditional node token

$$\mathbf{h}_{i,t}^{\prime(l-1)} = \boldsymbol{\gamma}_t^{(l)} \odot \mathbf{h}_{i,t}^{(l-1)} + \boldsymbol{\beta}_t^{(l)}. \tag{14}$$

The preference token and modulated node tokens interact through multi-head attention, followed by residual connections, instance normalization, and a feed-forward sublayer. After $L = 6$ layers with eight attention heads, the encoder outputs $\mathbf{h}_{\lambda,t}^{(L)}$ and $\mathbf{H}_t = \{\mathbf{h}_{i,t}^{(L)}\}_{i \in \mathcal{V}_t}$. The POMO decoder maintains P trajectories with different starting choices (Kwon et al., 2020). Its context combines the first and current node embeddings for MOTSP, or the current node embedding and remaining capacity for MOCVRP.

For trajectory p, the decoder attends from its context to the encoded preference and nodes to obtain a glimpse vector $\mathbf{g}_t^{(p)}$. It scores feasible actions with a clipped dot product and applies a softmax:

$$p_t^{(p)}(a) = \mathrm{softmax}_{a \in \mathcal{A}(s_t^{(p)})} \left[C \tanh\left(\frac{(\mathbf{g}_t^{(p)})^\top W^K \mathbf{h}_{a,t}^{(L)}}{\sqrt{d}} \right) \right]. \tag{15}$$

The clipping constant is $C = 10$. The feasible-action set excludes visited or cancelled nodes and capacity-infeasible customers, while permitting an admissible depot return in MOCVRP. The resulting distribution supports node selection and the candidate ranking used by the rollout module.

B.3 Dynamic Re-Encoding and Solver Training

When an event changes the pending graph, the solver updates node features and feasibility masks, then runs the same WECA encoder on the new current graph. The decoder resumes from each trajectory's current node and remaining capacity. Executed actions remain committed; their effects enter through the current routing state and realized objective bookkeeping.

Static-WECA and Dynamic-WECA use the same conditional-attention encoder, POMO decoder, parameterization, masking rule, and inference procedure. Their training trajectories differ: Dynamic-WECA encounters observed changes during route execution, whereas Static-WECA is trained on unchanged instances. Algorithm 1 trains the base solver on dynamically changing routes. After each action, it applies $\mathcal{T}$ to advance the route, updates the observed graph through $\Phi_{e_{t+1}}$, and uses $\mathcal{R}$ to construct the next state. For each POMO lane p, $G_t^{(p)}$ is the remaining realized cost, and $A_t^{(p)}$ is the mean suffix cost across lanes minus that lane's suffix cost. The advantage is treated as a constant when weighting the selected action's log probability. Non-additive MOCVRP objectives retain terminal comparison. The rollout controller is trained separately after the base solver is frozen.

Algorithm 1 Dynamic-WECA training

Input: instance distribution $\mathcal{S}$, preference distribution Λ, event process $p_{\mathcal{E}}$, scalarization $g_{\boldsymbol{\lambda}}$, training updates E, batch size B, POMO lanes P
Initialize WECA parameters θ
for $k = 1, \ldots, E$ **do**
  Draw $\boldsymbol{\lambda} \sim \Lambda$ and $\mathcal{G}_0^{(b)} \sim \mathcal{S}$, $b = 1, \ldots, B$
  for $b = 1, \ldots, B$ **do**
    $\mathbf{H}_0 \leftarrow \mathrm{Enc}\,\theta(\mathcal{G}_0^{(b)}, \boldsymbol{\lambda})$; initialize $s_0^{(p)}$ and set $\mathbf{H}_0^{(p)} \leftarrow \mathbf{H}_0$, $p = 1, \ldots, P$
    while unfinished trajectories remain **do**
      for each active lane p at step t **do**
        $p_t^{(p)}(\cdot) \leftarrow \mathrm{Dec}\,\theta(\cdot \mid \mathbf{H}_t^{(p)}, s_t^{(p)})$; draw $a_t^{(p)} \sim p_t^{(p)}$ and record $\log p_t^{(p)}(a_t^{(p)})$
        $\tilde{s}_{t+1}^{(p)} \leftarrow \mathcal{T}(s_t^{(p)}, a_t^{(p)})$; record $c_t^{(p)}$
        if lane p remains unfinished **then**
          Draw $e_{t+1} \sim p_{\mathcal{E}}$; set $e_{t+1} = \varnothing$ at ineligible steps
          $\mathcal{G}_{t+1}^{(p)} \leftarrow \Phi_{e_{t+1}}(\tilde{\mathcal{G}}t + 1^{(p)})$; $st + 1^{(p)} \leftarrow \mathcal{R}(\tilde{s}_{t+1}^{(p)}, \mathcal{G}_{t+1}^{(p)})$
          if $e_{t+1} \neq \varnothing$ **then**
            $\mathbf{H}_{t+1}^{(p)} \leftarrow \mathrm{Enc}\,\theta(\mathcal{G}t + 1^{(p)}, \boldsymbol{\lambda})$
          else
            $\mathbf{H}_{t+1}^{(p)} \leftarrow \mathbf{H}_t^{(p)}$
          end if
        end if
      end for
    end while
    Compute $G_t^{(p)}$ and $A_t^{(p)}$ by Eq. 7; accumulate gradients weighted by detached $A_t^{(p)}$
  end for
  Update θ with Adam using the batch-averaged gradient
end for
Output: trained WECA parameters θ

C Details of Rollout-Based NCO World Model

C.1 Fixed-Snapshot Candidate Imagination

When the world model decides to imagine at step t, it branches from the current routing state for each selected candidate a. The candidate is fixed as the next action, and frozen WECA completes the remaining route. Within a branch, visited-node masks and remaining capacity evolve normally, while the observed graph $\mathcal{G}_t$ and preference $\boldsymbol{\lambda}$ stay fixed. The committed prefix retains its realized costs; only the imagined suffix contributes new costs to the terminal score in Eq. 5.

Candidate branches never sample future events. The branch with the lowest scalarized terminal score supplies the next action; only that action is executed, and every imagined suffix is discarded. Any later real event is incorporated into a new observed graph before another imagination decision.

C.2 Deciding When to Imagine

The learned component of the world model decides whether the current node warrants imagination. Before any candidate is rolled out, μ_ψ receives the WECA decoder state $\mathbf{d}_t^{(p)}$, features $\boldsymbol{\xi}_t^{(p)}$ of the observed graph, and candidate-solution features $\mathbf{c}_t^{(p)}$ formed from the base decoder's feasible proposals. These features contain no rollout scores or future-event information. Equation 6 combines POMO lanes and recent imagination decisions into $\mathbf{z}_t$; this context is used only by the world-model decision, not by the WECA route encoder.

The decision $K_t = 0$ means that this node needs no imagination and WECA acts directly. A positive $K_t \in \{2, 3, 4\}$ triggers candidate completion; its value only sets how many feasible proposals are examined. Thus, choosing a budget is part of the trigger's action, while the world model itself consists of the imagination decision together with the fixed-snapshot candidate rollouts.

Algorithm 2 Training the rollout-based NCO world model

Input: frozen WECA policy π_θ, instance distribution $\mathcal{S}$, preference distribution Λ, event process $p_{\mathcal{E}}$, scalarization $g_{\boldsymbol{\lambda}}$, budgets $\mathcal{K} = 0, 2, 3, 4$, training iterations E, batch size B, POMO lanes P, PPO clip ϵ_{PPO}
Initialize the imagination policy μ_ψ and value function V_φ
for $k = 1, \ldots, E$ **do**
  Sample B paired instances, preferences, and event schedules from $\mathcal{S}$, Λ, and $p_{\mathcal{E}}$
  for $b = 1, \ldots, B$ **do**
    Execute frozen π_θ without imagination and record the baseline terminal cost $J_0^{(b)}$
    Reset to the same instance and event schedule and initialize P world-model lanes
    while at least one lane remains unfinished **do**
      Construct pre-rollout features $(\mathbf{d}_t^{(p)}, \boldsymbol{\xi}_t^{(p)}, \mathbf{c}_t^{(p)})$ and $\mathbf{z}_t$ using Eq. 6
      Sample $K_t \sim \mu_\psi(\cdot \mid \mathbf{z}_t)$ from the feasible set $\mathcal{K}_t \subseteq \mathcal{K}$
      if $K_t > 0$ **then**
        Complete the Top-K_t candidates on fixed $\mathcal{G}_t$ and select the action using Eq. 5
      else
        Select the action using WECA
      end if
      Execute the selected action, reveal the next event, and re-encode the state if needed
      Store the PPO transition $(\mathbf{z}_t, K_t, \log \mu_\psi(K_t \mid \mathbf{z}_t), V_\varphi(\mathbf{z}_t))$
    end while
    Compute the relative reward $r_{\mathrm{rel}}^{(b)}$ by comparing $J_{\mathrm{WM}}^{(b)}$ with $J_0^{(b)}$
  end for
  Update ψ and φ with PPO using $r_{\mathrm{rel}}^{(b)} * b = 1^B$ and $\epsilon * \mathrm{PPO}$
end for
Output: rollout-based world model (μ_ψ, π_θ)

C.3 Training the World Model

WECA is frozen before world-model training. A no-imagination POMO run and a world-model-guided run start from the same instance and preference under matched event schedules; neither sees an event before it occurs. Their paired terminal scalarized costs supply the relative improvement signal for PPO. Only the imagination policy μ_ψ and its value function V_φ are updated; candidate completions continue to use frozen WECA.

D Hypervolume Indicator

Hypervolume (HV) measures the convergence and coverage of a nondominated solution set. For each evaluation condition c, we evaluate the solver using the 101 preference vectors specified in Section 5.1 and retain the nondominated objective vectors as $\mathcal{F}_c$. Since all objectives are minimized, given a reference point $\mathbf{r}_c = (r_{c,1}, \ldots, r_{c,M})$ that is worse than all vectors in $\mathcal{F}_c$, HV is defined as

$$\mathrm{HV}_{\mathbf{r}_c}(\mathcal{F}_c) = \mu\left(\bigcup_{\mathbf{f}\in\mathcal{F}_c} \prod_{m=1}^{M} [f_m, r_{c,m}]\right), \tag{16}$$

where μ denotes the M-dimensional Lebesgue measure. Figure.5 illustrates the two-objective case, where the shaded region represents the hypervolume. For comparison across different objective scales, we report the normalized hypervolume $\mathrm{nHV}_c(\mathcal{F}_c) = \mathrm{HV}_{\mathbf{r}_c}(\mathcal{F}_c) / \prod_{m=1}^{M}(r_{c,m} - z_{c,m})$. For each synthetic task and problem size, all dynamic profiles, methods, and event share the same ideal point $\mathbf{z}_c$ and reference point $\mathbf{r}_c$ listed in Table.3.

E Complete Dynamic Benchmark Results

Tables 4 and 5 report the complete results for the dynamic Bi-TSP and Bi-CVRP benchmarks, respectively. Across Bi-TSP, WECA-WM achieves the highest HV in all nine size–profile combina-

Table 3: Reference points and ideal points.

Problem	Size n	ideal point $\mathbf{z}$	reference point $\mathbf{r}$
MOTSP	20	$(0, 0)$	$(20, 20)$
	50	$(0, 0)$	$(35, 35)$
	100	$(0, 0)$	$(65, 65)$
	150	$(0, 0)$	$(85, 85)$
	200	$(0, 0)$	$(115, 115)$
MOCVRP	20	$(0, 0)$	$(30, 4)$
	50	$(0, 0)$	$(45, 4)$
	100	$(0, 0)$	$(80, 4)$
	150	$(0, 0)$	$(115, 4)$
	200	$(0, 0)$	$(150, 4)$

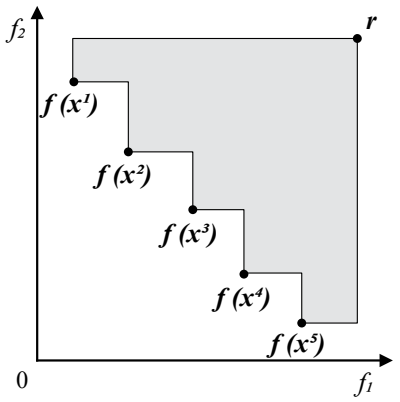


Figure 5: Hypervolume illustration.

Table 4: Complete dynamic Bi-TSP results underlying Table 1.

	D1			D2			D3		
Method	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓
Bi-TSP20									
WS-LKH	0.6254	6.52%	16.1m	0.6750	8.26%	14.9m	0.6886	7.79%	20.5m
MOGLS	0.6006	10.22%	5.7m	0.6638	9.79%	5.6m	0.6570	12.02%	6.3m
NSGA-II	0.5916	11.57%	24.6m	0.6624	9.98%	21.7m	0.6608	11.52%	21.3m
MOEA/D	0.6062	9.39%	10.4m	0.6714	8.75%	12.6m	0.6698	10.31%	18.8m
MAPT	0.6170	7.77%	15s	0.6835	7.11%	12s	0.7010	6.13%	11s
SIT	0.6259	6.44%	32s	0.6942	5.65%	33s	0.7112	4.77%	40s
Static-WECA	0.6597	1.39%	3s	0.7247	1.51%	4s	0.7368	1.34%	5s
WECA-WM	**0.6690**	0.00%	17s	**0.7358**	0.00%	17s	**0.7468**	0.00%	17s
Bi-TSP50									
WS-LKH	0.6547	9.53%	1.5h	0.4747	41.82%	1.7h	0.5255	39.84%	1.7h
MOGLS	0.4697	35.10%	15.4m	0.3947	51.62%	21.8m	0.4408	49.54%	20.7m
NSGA-II	0.4506	37.74%	34.0m	0.3799	53.44%	38.7m	0.4367	50.01%	39.9m
MOEA/D	0.4614	36.24%	23.3m	0.3855	52.75%	30.8m	0.4491	48.59%	35.8m
MAPT	0.6635	8.32%	37s	0.7619	6.62%	24s	0.8125	6.98%	20s
SIT	0.6674	7.78%	2.5m	0.7663	6.08%	2.8m	0.8188	6.26%	2.9m
Static-WECA	0.7113	1.71%	6s	0.8033	1.54%	6s	0.8598	1.57%	6s
WECA-WM	**0.7237**	0.00%	42s	**0.8159**	0.00%	30s	**0.8735**	0.00%	48s
Bi-TSP100									
WS-LKH	0.6485	16.90%	6.2h	0.7392	8.88%	8.4h	0.6390	27.20%	10.2h
MOGLS	0.3170	59.38%	1.1h	0.3707	54.30%	26.1m	0.2962	66.26%	40.3m
NSGA-II	0.2875	63.16%	34.8m	0.3799	53.17%	35.7m	0.2934	66.58%	1.1h
MOEA/D	0.2961	62.06%	43.8m	0.3762	53.62%	37.7m	0.3115	64.51%	47.6m
MAPT	0.7326	6.13%	3.2m	0.7882	2.84%	5.4m	0.8653	1.42%	2.8m
SIT	0.7343	5.91%	8.8m	0.7865	3.04%	16.3m	0.8674	1.18%	13.3m
Static-WECA	0.7724	1.03%	12s	0.7830	3.48%	36s	0.8535	2.77%	25s
WECA-WM	**0.7804**	0.00%	48s	**0.8112**	0.00%	1.7m	**0.8778**	0.00%	1.3m

tions. Static-WECA remains highly competitive, with gaps of only 1.03–3.48% from WECA-WM, while the gaps of the conventional search and evolutionary baselines generally increase on larger instances. MAPT and SIT provide stronger alternatives, particularly on Bi-TSP100, but remain below WECA-WM across all dynamic profiles.

For Bi-CVRP, WECA-WM obtains the highest HV in eight of the nine settings. The only exception is Bi-CVRP50 under D3, where SIT achieves an HV of 0.4411, corresponding to a -0.78% gap relative to WECA-WM. The advantage of WECA-WM becomes more pronounced on Bi-CVRP100, especially relative to MOGLS, NSGA-II, and MOEA/D. Compared with Static-WECA, the additional rollout mechanism consistently improves HV except for no reversal in ranking, with relative gains ranging from 0.30% to 9.31%. This improvement introduces additional runtime over Static-

Table 5: Complete dynamic Bi-CVRP results underlying Table 1.

	D1			D2			D3		
Method	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓	HV↑	Gap↓	Time↓
Bi-CVRP20									
MOGLS	0.4172	4.73%	6.5m	0.4294	3.24%	6.6m	0.4083	14.13%	6.2m
NSGA-II	0.3777	13.75%	42.7m	0.3926	11.54%	37.2m	0.3633	23.60%	1.1h
MOEA/D	0.3907	10.78%	31.8m	0.4036	9.06%	23.2m	0.3777	20.57%	39.2m
MAPT	0.3987	8.95%	20s	0.4064	8.43%	20s	0.4215	11.36%	20s
SIT	0.3211	26.67%	34s	0.3317	25.26%	33s	0.3545	25.45%	1.5m
Static-WECA	0.4366	0.30%	5s	0.4423	0.34%	6s	0.4427	6.90%	9s
WECA-WM	**0.4379**	0.00%	10s	**0.4438**	0.00%	12s	**0.4755**	0.00%	16s
Bi-CVRP50									
MOGLS	0.2890	35.78%	18.6m	0.3552	31.71%	15.6m	0.3003	31.39%	17.3m
NSGA-II	0.1594	64.58%	1.1h	0.2468	52.55%	1.4h	0.2274	48.05%	2.0h
MOEA/D	0.2089	53.58%	52.3m	0.3021	41.92%	1.0h	0.2714	37.99%	1.6h
MAPT	0.3951	12.20%	58s	0.4942	4.98%	34s	0.4232	3.31%	48s
SIT	0.3098	31.16%	2.0m	0.4022	22.67%	2.2m	**0.4411**	-0.78%	5.2m
Static-WECA	0.4227	6.07%	12s	0.4790	7.90%	13s	0.4060	7.24%	24s
WECA-WM	**0.4500**	0.00%	16s	**0.5201**	0.00%	16s	0.4377	0.00%	26s
Bi-CVRP100									
MOGLS	0.0155	96.68%	11.2m	0.0525	89.17%	11.5m	0.2099	60.99%	13.3m
NSGA-II	0.0027	99.42%	55.5m	0.0355	92.68%	1.0h	0.1986	63.09%	1.1h
MOEA/D	0.0214	95.41%	1.7h	0.0869	82.08%	2.5h	0.2619	51.33%	2.3h
MAPT	0.4316	7.46%	3.8m	0.4344	10.41%	2.1m	0.5181	3.72%	1.5m
SIT	0.3513	24.68%	6.8m	0.3977	17.98%	7.3m	0.5025	6.62%	11.1m
Static-WECA	0.4359	6.54%	30s	0.4571	5.73%	40s	0.4880	9.31%	1.1m
WECA-WM	**0.4664**	0.00%	50s	**0.4849**	0.00%	1.7m	**0.5381**	0.00%	1.1m

WECA, but the resulting runtime remains substantially lower than that of the conventional search and evolutionary baselines in most settings.

F Dynamic Benchmark Results on KroAB and XML

The KroAB and XML suites evaluate dynamic routing performance of three different sizes. Figures 6 and 7 report the normalized HV achieved by each method under the D1–D3 dynamic profiles.

On KroAB, WECA-WM achieves the highest HV in all nine size–profile settings. Compared with Static-WECA, the improvement ranges from 0.0037 to 0.0387, with an average gain of 0.0205. This advantage is consistently observed across all three instance sizes and dynamic profiles, indicating that the performance gain is not limited to a particular problem scale or event setting. MAPT and SIT are competitive in several cases, but neither consistently matches the performance of WECA-WM.

A similar trend is observed on XML, where WECA-WM again achieves the highest HV across all nine settings. Relative to Static-WECA, the improvement ranges from 0.0037 to 0.1142, with an average gain of 0.0523, suggesting a more pronounced benefit on this benchmark suite. In addition, the gaps of the other methods vary noticeably across instance sizes and dynamic profiles, whereas WECA-WM remains consistently best in all settings. Overall, the results on both benchmark suites demonstrate that the complete dynamic WECA-WM framework maintains a stable HV advantage under different instance scales and dynamic conditions, supporting the effectiveness of the proposed dynamic adaptation mechanism.

G Sensitivity Analysis of Rollout Strategies

We examine how rollout timing and candidate count affect solution quality on dynamic Bi-TSP and KroAB instances. Figure. 8 shows the paired HV difference from the same WECA solver without lookahead. Every event and every step invoke Top-2 rollout at fixed decision times. The two WM gate variants instead use the learned decision to determine whether the current node warrants imagination; when activated, they evaluate the top two or four candidates, respectively.

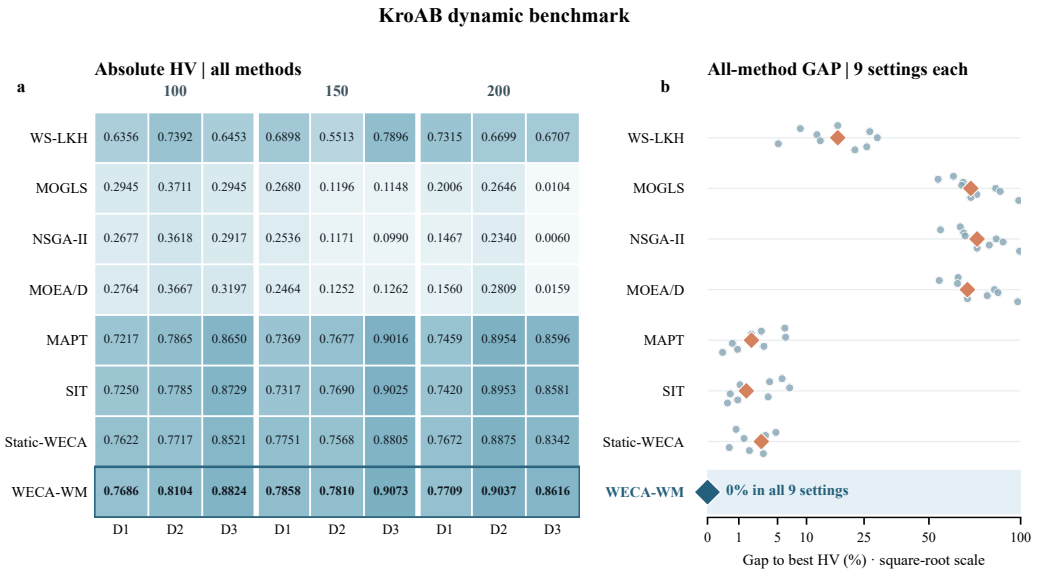


Figure 6: Dynamic KroAB results for instances with 100, 150, and 200 nodes. (a) Normalized HV of all methods under D1–D3. (b) Percentage gap from the best HV in each of the nine settings; each dot represents one setting, and each diamond indicates the median.

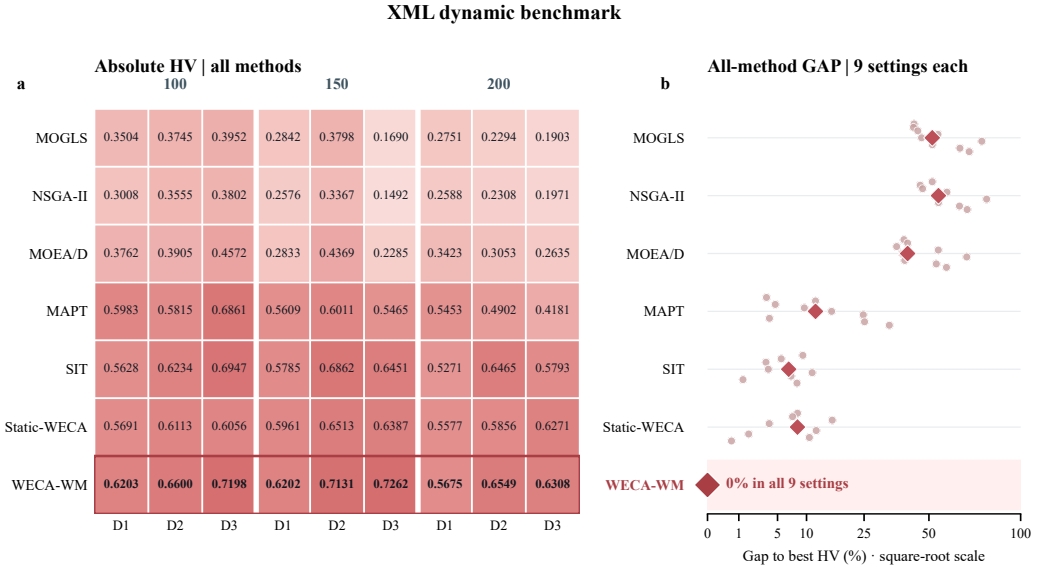


Figure 7: Dynamic XML results for instances with 100, 150, and 200 customers. (a) Normalized HV of all methods under D1–D3. (b) Percentage gap from the best HV in each of the nine settings; each dot represents one setting, and each diamond indicates the median.

Fixed-frequency rollout can improve HV, but its effect varies across settings. On Bi-TSP, every-step Top-2 achieves the largest gains in several D1 settings, while its gains reverse on Bi-TSP100 under D2 and D3. The variation is more pronounced on KroAB: every-step Top-2 reduces HV in six of nine settings, and every-event Top-2 does so in three. For example, both fixed strategies reduce HV on KroAB150 under D3, whereas the WM-gated variants match the WECA baseline.

The learned gate with four candidates gives the most consistent paired outcome across the 18 Bi-TSP and KroAB settings: HV increases in 13, is unchanged in four, and decreases in one. On KroAB,

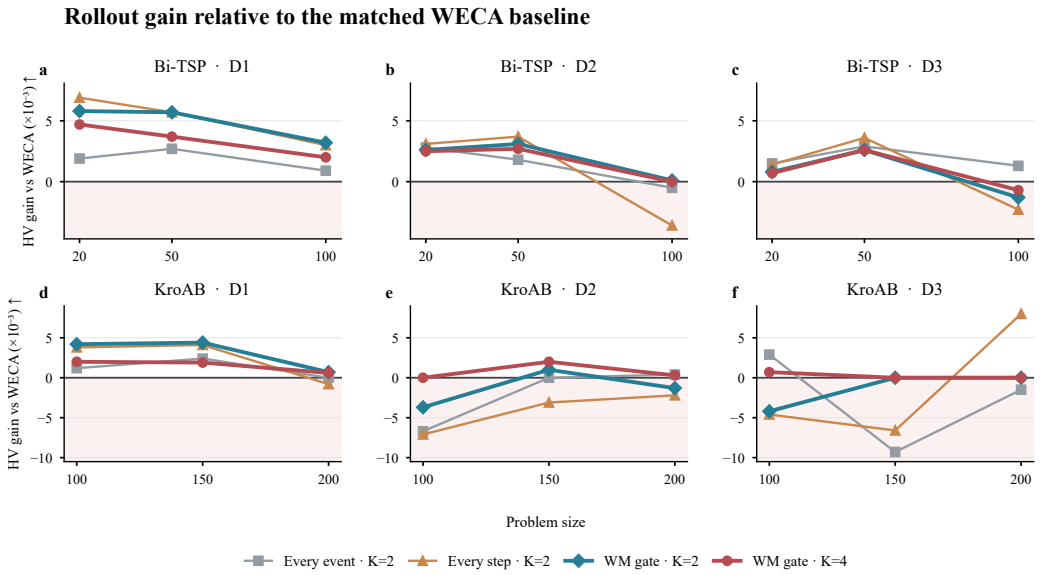


Figure 8: Sensitivity to rollout timing and candidate count on dynamic Bi-TSP and KroAB. Each panel plots the paired HV difference from WECA without lookahead.

it improves six settings and matches the baseline in the remaining three. The same-K comparison also clarifies the role of the trigger: among the Top-2 variants, the learned gate limits the largest observed HV decrease to 0.0042, compared with 0.0093 for every-event rollout and 0.0071 for every-step rollout. These results show the value of deciding when to imagine, alongside the number of candidates examined when imagination is triggered.

H Complete Results of Static Instance

Table 6 reports additional experiments on static Bi-TSP benchmarks to examine whether the benefit of the proposed world model persists when no dynamic events occur. This setting removes the need to react to environmental changes and therefore isolates the contribution of rollout-based lookahead to action selection. On the KroAB instances, WECA-WM consistently improves over the corresponding static NCO baselines. It achieves HV values of 0.6997, 0.6967, and 0.7374 on Kro100, Kro150, and Kro200, respectively, exceeding all methods in the Static NCO group. With graph augmentation, WECA-WM-Aug further increases these values to 0.7015, 0.6994, and 0.7394. In particular, WECA-WM-Aug also achieves higher HV than the other graph-augmentation variants across all three instance sizes. The consistent improvement from WECA-WM to WECA-WM-Aug further indicates that the world-model mechanism and graph augmentation provide complementary benefits rather than relying on the same source of improvement.

The comparison with conventional optimization methods provides additional context. WS-LKH remains highly competitive on the KroAB instances, but requires substantially longer runtime than the NCO-based methods. For example, on Kro100, WECA-WM-Aug obtains an HV of 0.7015 in 10 minutes, compared with 0.7022 in 2.3 hours for WS-LKH. A similar pattern is observed on Kro150, where WECA-WM-Aug reaches 0.6994 in 40 minutes, while WS-LKH obtains 0.7017 in 4.0 hours. On Kro200, the performance difference remains small, with HV values of 0.7394 and 0.7430, respectively. These results show that the world-model extension can improve the solution quality of NCO while retaining substantially lower computational cost than the conventional solver.

The Bi-TSP results exhibit the same overall trend. WECA-K2 achieves HV values of 0.7059 and 0.7402 on Bi-TSP150 and Bi-TSP200, respectively, outperforming all compared static NCO baselines. For instance, the strongest static NCO baseline reaches 0.7033 on Bi-TSP150 and 0.7371 on Bi-TSP200, whereas WECA-K2 further improves both values.

Table 6: Complete comparison results on static Bi-TSP benchmark instances.

(a) KroAB instances

		Kro100			Kro150			Kro200		
Category	Method	HV	Gap	Time	HV	Gap	Time	HV	Gap	Time
Non-learnable	WS-LKH	**0.7022**	-0.36%	2.3m	**0.7017**	-1.05%	4.0m	**0.7430**	-2.07%	5.6m
	MOEA/D	0.6836	2.30%	5.8m	0.6710	3.37%	7.1m	0.7106	2.38%	7.3m
	NSGA-II	0.6676	4.59%	7.0m	0.6552	5.65%	7.9m	0.7011	3.69%	8.4m
	MOGLS	0.6817	2.57%	52m	0.6671	3.93%	1.3h	0.7083	2.70%	1.6h
Static NCO	MDRL	0.6881	1.66%	10s	0.6831	1.63%	17s	0.7209	0.97%	23s
	EMNH	0.6900	1.38%	9s	0.6832	1.61%	16s	0.7217	0.86%	23s
	WE-CA	0.6948	0.70%	9s	0.6924	0.29%	19s	0.7317	-0.52%	26s
	POCCO-C	0.6965	0.45%	30s	0.6925	0.27%	40s	0.7302	-0.31%	50s
	POCCO-W	0.6981	0.23%	20s	0.6946	-0.03%	31s	0.7345	-0.90%	40s
Graph Aug.	MDRL-Aug	0.6950	0.67%	13s	0.6890	0.78%	19s	0.7261	0.25%	28s
	EMNH-Aug	0.6958	0.55%	12s	0.6892	0.75%	18s	0.7270	0.13%	27s
	WE-CA-Aug	0.6990	0.10%	14s	0.6957	-0.19%	23s	0.7349	-0.96%	39s
	POCCO-C-aug	0.6999	-0.03%	32s	0.6959	-0.21%	48s	0.7334	-0.75%	1.1m
	POCCO-W-aug	0.7006	-0.13%	22s	0.6976	-0.46%	39s	0.7369	-1.23%	1.1m
Ours	**WECA-WM**	0.6997	0.00%	5m	0.6967	-0.33%	12m	0.7374	-1.30%	20m
	WECA-WM-Aug	<u>0.7015</u>	-0.26%	10m	<u>0.6994</u>	-0.72%	40m	<u>0.7394</u>	-1.58%	2.3h

(b) Bi-TSP instances

		Bi-TSP150			Bi-TSP200		
Category	Method	HV	Gap	Time	HV	Gap	Time
Non-learnable	WS-LKH	**0.7149**	-1.71%	13h	**0.7490**	-2.50%	22h
	MOEA/D	0.6809	3.13%	2.4h	0.7139	2.30%	2.7h
	NSGA-II	0.6659	5.26%	6.8h	0.7045	3.59%	6.9h
	MOGLS	0.6768	3.71%	22h	0.7114	2.65%	38h
Static NCO	MDRL	0.6922	1.52%	40s	0.7251	0.77%	1.4m
	EMNH	0.6930	1.41%	40s	0.7260	0.65%	1.4m
	WE-CA	0.7008	0.30%	57s	0.7346	-0.53%	1.6m
	POCCO-C	0.7011	0.25%	1.5m	0.7333	-0.35%	2.5m
	POCCO-W	0.7033	-0.06%	1.4m	0.7371	-0.87%	2.4m
Ours	**WECA-K2**	<u>0.7059</u>	-0.43%	1.7h	<u>0.7402</u>	-1.30%	3.1h

Overall, the static experiments confirm that the effectiveness of the world model is not limited to dynamic adaptation. Even when the problem remains unchanged throughout inference, rollout-based evaluation provides useful lookahead for distinguishing candidate actions and improves the quality of the resulting Pareto set. The consistent gains over the static NCO baselines across both KroAB and Bi-TSP instances therefore support the use of the world model as a general decision-enhancement mechanism, rather than only as a component for handling dynamic events.

AI USE STATEMENT

Generative AI tools were used solely to improve the clarity, grammar, and readability of the manuscript. All technical content, experimental design, analysis, results, and conclusions were produced and verified by the authors. The use of AI did not involve generating experimental data, deriving scientific claims, or making research decisions.