

Article

TSS GAZ PTP: Towards Improving Gumbel AlphaZero with Two-Stage Self-Play for Multi-Constrained Electric Vehicle Routing Problems

Hui Wang ^{1,2} , Xufeng Zhang ^{1,2} and Chaoxu Mu ^{1,2,*}

¹ School of Artificial Intelligence, Anhui University, Jiulong Road, Hefei 230601, China; h.wang.13@ahu.edu.cn (H.W.); wa23201006@stu.ahu.edu.cn (X.Z.)

² Anhui Provincial Key Laboratory of Security Artificial Intelligence, Anhui University, Jiulong Road, Hefei 230601, China

* Correspondence: cxmu@tju.edu.cn

Highlights

What are the main findings?

- We propose a Two-Stage Self-Play strategy that fosters more robust and effective policy improvement.
- We successfully address a key limitation of the original GAZ PTP framework, enabling its application to a wider range of real-world routing challenges.

What is the implication of the main finding?

- As a key challenge for smart cities, the Electric Vehicle Routing Problem (EVRP) is addressed using the proposed deep reinforcement learning (DRL) framework with the explicit goal of minimizing energy consumption.

Abstract

Deep reinforcement learning (DRL) with self-play has emerged as a promising paradigm for solving combinatorial optimization (CO) problems. The recently proposed Gumbel AlphaZero Plan-to-Play (GAZ PTP) framework adopts a competitive training setup between a learning agent and an opponent to tackle classical CO tasks such as the Traveling Salesman Problem (TSP). However, in complex and multi-constrained environments like the Electric Vehicle Routing Problem (EVRP), standard self-play often suffers from opponent mismatch: when the opponent is either too weak or too strong, the resulting learning signal becomes ineffective. To address this challenge, we introduce Two-Stage Self-Play GAZ PTP (TSS GAZ PTP), a novel DRL method designed to maintain adaptive and effective learning pressure throughout the training process. In the first stage, the learning agent, guided by Gumbel Monte Carlo Tree Search (MCTS), competes against a greedy opponent that follows the best historical policy. As training progresses, the framework transitions to a second stage in which both agents employ Gumbel MCTS, thereby establishing a dynamically balanced competitive environment that encourages continuous strategy refinement. The primary objective of this work is to develop a robust self-play mechanism capable of handling the high-dimensional constraints inherent in real-world routing problems. We first validate our approach on the TSP, a benchmark used in the original GAZ PTP study, and then extend it to the multi-constrained EVRP, which incorporates practical limitations including battery capacity, time windows, vehicle load limits, and charging infrastructure availability. The experimental results show that TSS GAZ PTP consistently outperforms existing DRL methods, with particularly notable improvements on large-scale instances.



Academic Editor: Pierluigi Siano

Received: 3 December 2025

Revised: 13 January 2026

Accepted: 19 January 2026

Published: 23 January 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.

Keywords: deep reinforcement learning; self-play strategy; electric vehicle routing problem; combinatorial optimization problem; Monte Carlo Tree Search

1. Introduction

The expansion of transportation infrastructure, while vital for socioeconomic growth, has created significant environmental challenges, primarily due to carbon emissions [1–3]. Recent studies have explored time-constrained SAEV scheduling [4], demand-responsive charging control [5], and competitive charging pricing [6]; while these studies provide valuable insights at the electric mobility ecosystem level, path planning for electric vehicle (EV) at the routing layer is also critical. Path-planning models like the Traveling Salesman Problem (TSP) and the Vehicle Routing Problem (VRP) offer foundational strategies for mitigating these emissions by optimizing routes to minimize travel distance [7–9]. However, these classic combinatorial optimization problems and their variants were predominantly designed for traditional fuel vehicles, which remain a major source of carbon pollution [10]. The rapid development of EV technology and its supporting charging infrastructure now presents a promising and sustainable alternative [11–13]. Moreover, the integration of EVs is intrinsically linked to the development of Smart Cities, which fundamentally reshapes the objectives and constraints of path-planning algorithms [14,15].

Despite their environmental benefits, EVs face operational hurdles, most notably their limited driving range (500–700 km) and the relative scarcity of charging stations compared to ubiquitous gas stations [16]. This limitation complicates logistics planning, giving rise to a specific and more complex variant known as the Electric Vehicle Routing Problem (EVRP) [17,18]. Conventional approaches to solving the EVRP, such as adaptive large neighborhood search (ALNS) [19,20], variable neighborhood search (VNS) [21], and ant colony optimization (ACO) [22], have demonstrated success on small-scale problems. However, they often struggle with large-scale instances, require extensive parameter tuning, and are prone to converging on local optima.

The advent of deep reinforcement learning (DRL) has significantly transformed the landscape of solving complex optimization problems. By training deep neural network models, DRL offers new avenues for efficient and innovative solutions. Although DRL can provide desirable solutions for both small-scale and large-scale problems with lower computational costs, potentially outperforming traditional techniques, most current DRL research focuses on problems with fewer features and constraints, such as the TSP and VRP, which involve relatively few features and constraints [23,24]. Since these models typically consider distance as the sole objective function, the underlying problems are inherently linear in nature. In contrast, we adopt a multi-constraint EVRP model that adopts energy consumption as the primary objective function. This model rigorously incorporates realistic physical constraints, including road slope, charging and discharging efficiency, and battery capacity limits. By transitioning from distance-based to energy-based optimization, our approach significantly enhances the operational reliability of logistics within Smart Cities. It provides the precise energy forecasting required for grid load balancing and ensures that autonomous logistics systems can navigate complex urban environments without the risk of range failure.

AlphaZero, renowned for its superhuman performance in complex games like Go, leverages Monte Carlo Tree Search (MCTS) to guide its decision-making. Its performance, however, is highly dependent on a large number of MCTS simulations. Gumbel AlphaZero (GAZ) addresses this by achieving strong performance with far fewer simulations [25]. A subsequent variant, Gumbel AlphaZero Play-to-Plan (GAZ PTP), has proven effective for

fixed-step problems like the TSP and the Job-Shop Scheduling Problem (JSSP) [26]. A critical drawback of GAZ PTP, however, is that its self-play mechanism can become unbalanced; the superior MCTS-guided player can create a large performance gap, preventing the opposing policy network from effective learning. To overcome this, we introduce a two-stage self-play strategy to enhance the GAZ PTP algorithm. We first validate our TSS GAZ PTP method on the TSP and then extend it to solve the more complex, variable-step multi-constrained EVRP. This is a non-trivial extension, as EV routes can include multiple revisits to the depot and charging stations. Our approach is designed to tackle both the Distance Minimization EVRP (DM-EVRP) and Energy Minimization EVRP (EM-EVRP) variants of the problem [27]. A general visualization of the framework of this proposed algorithm is described in Figure 1.

Above all, the main contributions of this paper can be summarized as follows.

- We propose a Two-stage self-play strategy that resolves the unbalanced competition issue in GAZ PTP. This strategy forces the learning agent to consistently compete against an opponent of comparable strength, fostering more robust and effective policy improvement.
- We successfully extend our method from fixed-step problems to complex variable-step problems like the multi-constrained EVRP. This addresses a key limitation of the original GAZ PTP framework, enabling its application to a wider range of real-world routing challenges.
- Our proposed TSS GAZ PTP algorithm achieves state-of-the-art performance not only on fixed-step benchmarks but also on both multi-constrained DM-EVRP and EM-EVRP. It demonstrates significant advantages over traditional heuristics and other learning-based methods, particularly on large-scale instances.

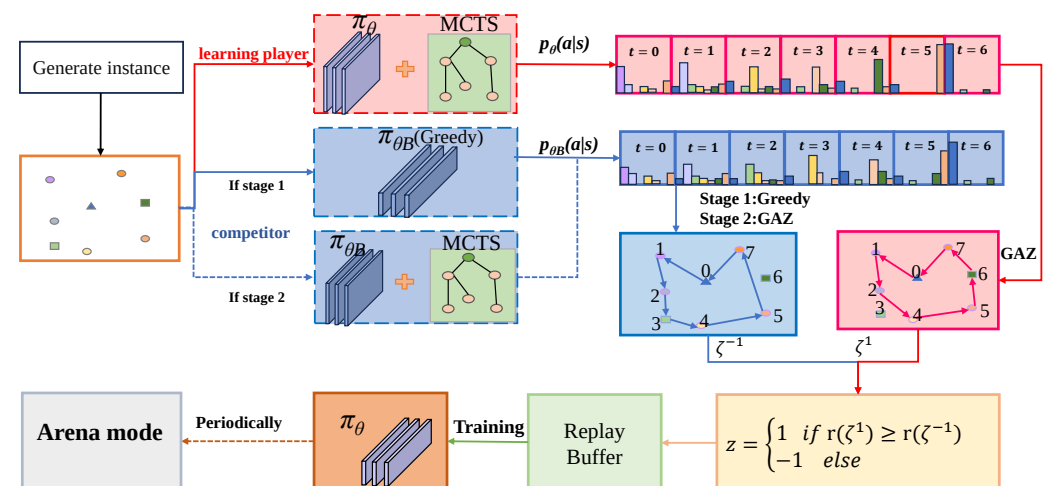


Figure 1. The basic framework of proposed Two-Stage Self-Play Gumbel AlphaZero (TSS GAZ PTP). The red part represents the action selection made by the learning player with Gumbel MCTS, and the blue part represents the action selection made by the competitor player with either greedy strategy in stage 1 or Gumbel MCTS in stage 2.

2. Related Work

2.1. Games with Deep Reinforcement Learning

DeepMind ushered in a new AI era by designing algorithms such as AlphaGo, AlphaGo Zero, and AlphaZero to defeat professional masters in complex games like Go and Chess [28]. These systems rely heavily on MCTS [29] and UCB for decision-making within vast state spaces. Originally, AlphaGo used a two-stage process involving supervised learn-

ing from expert experience followed by iterative RL policy optimization [30]. However, the updated AlphaGo Zero and AlphaZero removed the need for human input; knowing only the rules, they trained via self-play and surpassed the original AlphaGo [31].

While DRL has thrived in single-agent and two-player settings, the real world requires navigating multi-agent environments involving both cooperation and competition. DeepMind attained human-level performance in the Capture-the-Flag variant of the 3D multiplayer game Quake III Arena by means of population-based reinforcement learning [32]. Furthermore, DeepMind's AlphaStar [33] demonstrates that multi-agent RL can master one of the most challenging real-time strategy games with vast action spaces, partial observability, and long-term planning, suggesting broader applicability to complex sequential decision-making problems.

2.2. Combinatorial Optimization with Deep Reinforcement Learning

In recent years, Transformer has shown superior performance in the area of Natural Language Processing (NLP), particularly in machine translation and sequence-to-sequence tasks. COPs are inherently sequence optimization problems, as their solutions can be described through sequential decisions. Consequently, typical COPs like the TSP and the EVRP are increasingly being studied using DRL methods. Therefore, TSP and EVRP, as typical COPs, are also being studied using DRL methods. For instance, Wang [24] proposed a Graph Attention Network (GAT)-based encoder capable of providing high-dimensional node and graph embeddings for downstream EVRP tasks. Tang [27] formulated an energy consumption model (as opposed to a traditional distance model) using a Transformer-based DRL approach. To address the computational inefficiency for large-scale EVRP instances, Zhang [34] designed a two-layer model that identifies near-optimal solutions based on predefined feasibility conditions and reward structures.

2.3. AlphaGo Zero's Inspiration for Combinatorial Optimization

In addition, based on the self-play training strategy, AlphaGo Zero [31] achieved superhuman performance on the game of Go. Wang [35] used MCTS with warm-start enhancements to enhance the quality of the plays produced by self-play. Although AlphaGo Zero was designed for two-player games, many researchers attempted to apply the AlphaZero algorithm to single-player tasks by creating competitive environments. Most of them reconstruct reward mechanisms based on self-competition for different problems [36–38]. However, these methods do not use MCTS during training the policy network or value network; they only apply the MCTS-guided network after training. To address this issue, Wang [39] improved the policy by employing complete information from the MCTS search tree and learning the trajectory produced by MCTS. To save simulation costs, GAZ with Gumbel MCTS was proposed. And in order to improve the GAZ-based policy network, researchers further developed a new framework [25,26].

Through self-play, the agent learns to find strong paths by planning against potential strategies of its previous self and has shown higher performance on classical CO problems such as TSP and JSSP. But GAZ-based methods have not been investigated yet in multi-constrained EVRP with variable planning steps, which still remains a challenge.

3. EVRP

3.1. Problem Formulation

This section introduces the multi-constrained EVRP. Following [27], we define it with a directed graph $G = (V, E)$, $V = C \cup D \cup \hat{F}$, where $C = \{1 + s, \dots, n + s\}$ is a set of n customers, $D = \{0\}$ represents the depot, $\hat{F} = \{1, \dots, s\}$ is a set of recharge stations, and

$E = \{(i, j) \mid i, j \in V\}$ is a set of edges connecting two nodes. Each customer i with demand c_i can only be served once. Our goal is to visit all customers and plan completed routes for electric vehicles to minimize the total distance or total energy consumption while satisfying all relevant constraints described in the following equations. As shown in Figure 2, electric vehicles start from the depot D with a maximum load capacity L and a maximum battery capacity Q , and then return to the depot D after serving all customers. During the whole journey, we are supposed to ensure that the capacity of the electric vehicles is not less than 0, and the electric vehicles must visit the recharge stations at the appropriate time to avoid being stranded while driving. We also take into account the maximum serving time of the driver, T_{max} .

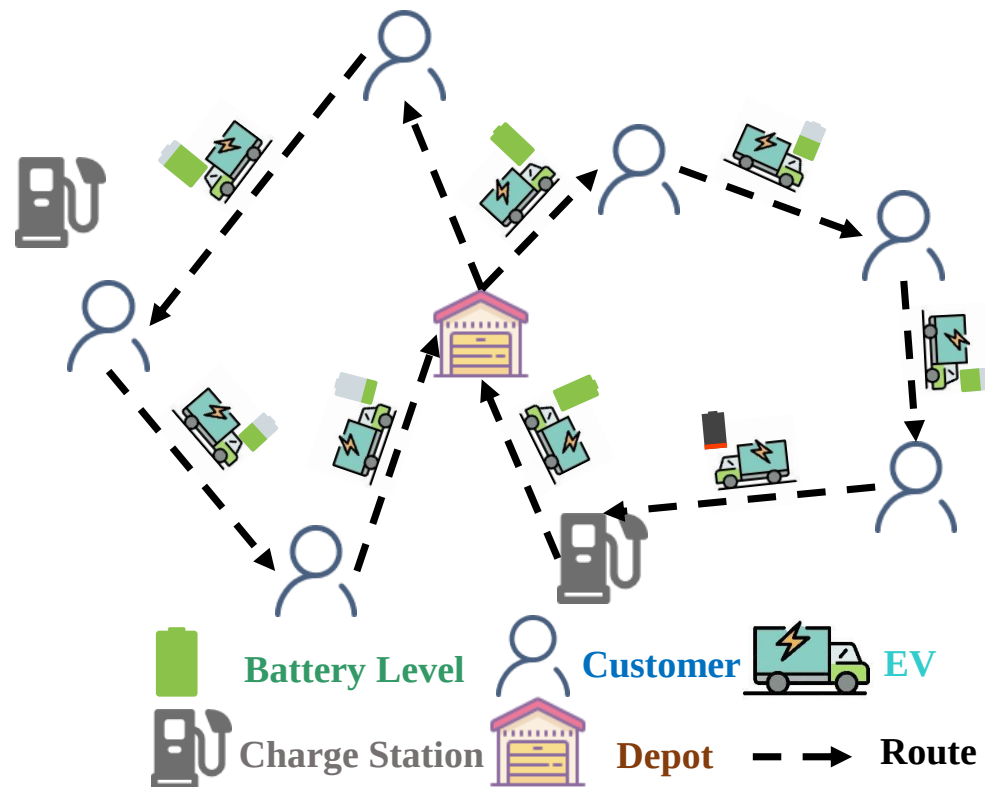


Figure 2. A simple example of the multi-constrained EVRP path planning.

$$\min f(\mathbf{x}) = \sum_{i \in V, j \in V, i \neq j} E_{ij} x_{ij} \quad (1)$$

s.t.

$$\sum_{j \in V, i \neq j} x_{ij} = 1 \quad \forall i \in C \quad (2)$$

$$\sum_{j \in V, i \neq j} x_{ij} \leq 1 \quad \forall i \in \hat{F} \quad (3)$$

$$\sum_{j \in V, i \neq j} x_{ij} - \sum_{j \in V, i \neq j} x_{ji} = 0 \quad \forall i \in C \cup \hat{F} \quad (4)$$

$$\tau_0 = 0 \quad (5)$$

$$0 \leq \tau_i \leq T_{max}, \forall i \in C \cup D \cup \hat{F} \quad (6)$$

$$\tau_i + (g_i + t_{ij})x_{ij} - T_{max}(1 - x_{ij}) \leq \tau_j, \forall i, j \in V, i \neq j \quad (7)$$

$$l_j \leq l_i - c_i x_{ij} + L(1 - x_{ij}) \quad \forall i, j \in V, i \neq j \quad (8)$$

$$0 \leq l_i \leq L \quad \forall i \in C \cup D \cup \hat{F} \quad (9)$$

$$u_0 = L \quad (10)$$

$$e_j \leq e_i - E_{ij} + Q(1 - x_{ij}) \quad \forall i \in C \quad \forall j \in V, i \neq j \quad (11)$$

$$e_i = Q \quad \forall i \in D \cup \hat{F} \quad (12)$$

$$0 \leq e_i \leq Q \quad \forall i \in C \cup D \cup \hat{F} \quad (13)$$

$$x_{ij} \in \{0, 1\} \quad \forall i, j \in V, i \neq j \quad (14)$$

The objective function of the multi-constrained EM-EVRP is depicted in Equation (1). Constraints (2) and (3) stipulate that each customer should only be served once, and the charging stations can be visited multiple times separately. This ensures that each customer is served only once to avoid resource waste and redundant deliveries. Constraint (4) indicates the route's continuity, while Constraints (5) and (6) ensure that the driver's serving time is no longer than T_{max} and recounted after returning to the depot. This ensures driver safety and health and prevents fatigued driving while complying with practical operational management standards. Constraint (7) states that the driver's serving time is constantly updated when arriving at a site. Constraint (8) tracks electric vehicle cargo, while Constraints (9) and (10) ensure that electric vehicles leave the depot fully loaded and have a cargo capacity of L . Constraint (11) states that the battery capacity of electric vehicles is constantly updated when they arrive at a site, while Constraint (12) and (13) ensure that the remaining battery capacity is not greater than Q and is fully charged when arriving at the depot or recharge stations. This is the core feature that distinguishes EVRP from traditional VRP. Accurate modeling of battery behavior is crucial for preventing "mid-route power depletion," directly determining whether a path is feasible. Constraint (14) defines the decision variables. The parameters used in these equations and their corresponding explanations are described in detail in Table 1.

The multi-constrained DM-EVRP model shares the same constraints (Equations (2)–(14)), but the objective function for multi-constrained DM-EVRP is defined as (15).

$$\min f(\mathbf{x}) = \sum_{i \in V, j \in V, i \neq j} d_{ij} x_{ij} \quad (15)$$

Table 1. The parameters and descriptions of the multi-constrained EVRP.

Parameter	Description
L	Capacity of the vehicle
T_{max}	Driver's maximum serving time
Q	Battery capacity of the vehicle
c_i	Demand of the customer i
l_i	Remaining capacity while reaching node i
τ_i	Travel time while reaching node i
g_i	Serving time while reaching node i

Table 1. Cont.

Parameter	Description
e_i	Remaining battery capacity while reaching node i
d_{ij}	Distance between node i and node j
t_{ij}	Travel time of the node i from the node j
E_{ij}	Energy consumption from the node i to j
u_{ij}	Cargo load from the node i to the node j
v_{ij}	Average speed from the node i to the node j

3.2. Energy Consumption

We calculate the energy consumption of the electric vehicles between node i and node j as follows:

$$P_{ij} = (m_{ij}(a + g \sin(\alpha_{ij}) + C_r \cos(\alpha_{ij})) + S)v_{ij} \quad (16)$$

$$S = 0.5 \cdot C_d \cdot \rho \cdot A \cdot v_{ij}^2 \quad (17)$$

where m_{ij} represents the capacity of the electric vehicles between the node i and node j ; a is the acceleration of the vehicle and is set to 0 since we assume the speed is constant in our experiments. Gravity is shown by g , air density is indicated by ρ , the resistance coefficient is indicated by C_r , the aerodynamic drag coefficient is indicated by C_d , and the slope between nodes i and j is indicated by α_{ij} . After obtaining the mechanical power P_{ij} , the energy consumption is calculated as follows:

$$E_{ij} = \begin{cases} \phi^d \cdot \phi^d \cdot P_{ij} \cdot t_{ij} & P_{ij}(m_{ij}) \geq 0 \text{kw} \\ \phi^r \cdot \phi^r \cdot P_{ij} \cdot t_{ij} & P_{ij}(m_{ij}) < 0 \text{kw} \end{cases} \quad (18)$$

Since the effect of slope is a factor, the formula for energy consumption is divided into two cases: electric vehicles need more energy when traveling uphill and are allowed to charge for power recovery when traveling downhill. While t_{ij} represents the travel time between node i and node j , ϕ^d and ϕ^r denote the charging and discharging efficiency of the battery, respectively. Our current EVRP formulation assumes uniform vehicle speed and omits transient energy effects such as those from acceleration and traffic-induced speed variations. While this assumption enhances the model's manageability and aligns with common practices in existing path planning research based on deep reinforcement learning, it inevitably overlooks several critical real-world factors inherent to urban driving.

3.3. Markov Decision Process of Multi-Constrained EVRP

In this paper, we model the multi-constrained EVRP as a two-player game; then, the Markov decision process (MDP) can be defined as a tuple $(\mathcal{S}, \mathcal{A}, \mathcal{R}, \mathcal{P})$, consisting of state $\mathcal{S}$, action $\mathcal{A}$, reward $\mathcal{R}$ and state transition $\mathcal{P}$.

- **State:** $\mathcal{S}$ represents the state space. In a two-player game, $\mathcal{S} = (s_t^1, s_t^{-1})$, each player starts from the depot with the initial state s_0^1, s_0^{-1} , respectively, and s_t^1, s_t^{-1} represent the states of two players at time step t . The graph node state and the electric vehicle state are represented as $\{s_t^1, s_t^{-1}\} = \{x_t^1, v_t^1, x_t^{-1}, v_t^{-1}\}$. For each node i , $x_t^i = (x_s^i, c_t^i)$, x_s^i, c_t^i are the static and dynamic information of the node i , respectively. The static information is composed of the two-dimensional coordinates of the node x_s^i and the demand of each customer c_t^i . For the vehicle state $\{v_t^1, v_t^{-1}\} = \{e_t^1, \tau_t^1, u_t^1, e_t^{-1}, \tau_t^{-1}, u_t^{-1}\}$, e_t is the remaining battery of the electric vehicles; τ_t is the current travel time; and u_t is the remaining capacity of the electric vehicles.
- **Action:** $\mathcal{A}$ represents the action space. In the two-player game $\mathcal{A} = \{a_0^1, a_1^1, \dots, a_T^1, a_0^{-1}, a_1^{-1}, \dots, a_T^{-1}\}$, the action a_t represents the action that has been chosen at the time step t .

- **Reward:** Unlike a single task that sets the reward as minimization or maximization of the objective function, the reward $\mathcal{R}$ is reshaped into a binary ± 1 based on self-competition, to which we compare the trajectory $\zeta_0^p = (s_0^p, a_0^p, \dots, s_{T-1}^p, a_{T-1}^p, s_T^p)$ for the player $p \in \{1, -1\}$ at the time step t , $\mathcal{R} = 0$ if $t < T - 1$.
- **Transition:** The state transitions deterministically to $\mathcal{P} = F(s_t, a_t)$ due to the deterministic state transition function $F: \mathcal{S} \times \mathcal{A} \rightarrow \mathcal{S}$.

4. Methodology

4.1. Two-Stage Self-Play

This section presents our new GAZ PTP method with the Two-Stage Self-Play strategy. As shown in Figure 3, at stage 1, the learning player uses Gumbel MCTS to choose actions, while the competitor chooses the action from the best historical policy network. In this stage, only the learning player can take the state of the competitor into consideration when in the expansion phase and update the node information through backpropagation. After a period of training episodes, the learning player is unable to find a better trajectory because it has a great advantage to employ Gumbel MCTS in complex tasks, leading to unbalanced competition. Therefore, we introduce the second stage. Both players use Gumbel MCTS, which ensures that they try to find a better trajectory during the competition. It also increases the depth of MCTS because both players can take the other player's state into account and update the node information through backpropagation.

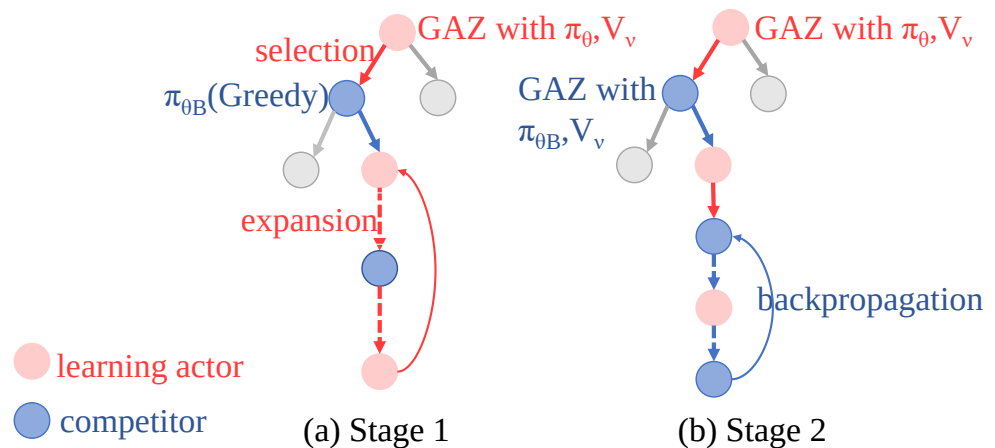


Figure 3. The comparison of Gumbel MCTS for Stage 1 and 2.

4.2. Algorithm

Algorithm 1 illustrates our proposed training framework. In each episode, we divide the two players into the learning player and the competitor, while we also divide the training into two stages. In the first stage, the learning player chooses the action according to the policy network π_θ based on GAZ PTP [25], while the competitor chooses the action greedily from the policy network π_{θ^B} , which is the best historical policy network of the learning player. The π_{θ^B} only updates periodically in the arena mode when the outcome of greedily rolling out π_θ is better than π_{θ^B} . Our aim is to train the model to converge quickly and save computational resources simultaneously as the competitor does not employ Monte Carlo simulations and Gumbel MCTS only needs few simulations. After n episodes of training, the learning player performs much better than the competitor.

Therefore, to ensure that the learning player learns stronger trajectories, in the second stage, each player uses the Gumbel MCTS-based policy network π_θ and π_{θ^B} to select actions. Our proposed Two-Stage Self-Play strategy can make the learning player constantly compete with an opponent with similar playing strength so that the learning player can

learn smarter trajectories from games. In each episode, there is a probability P of performing self-play that helps the learning player learn the trajectories of its own policy and also maintains the diversity of the training data.

Algorithm 1 Gumbel AlphaZero Play-to-Plan with Two-Stage Self-Play

Input: ρ_0 : initial state distribution; $\mathcal{J}_{\text{arena}}$: set of initial states sampled from ρ_0

Input: $0 \leq P < 1$: self-play parameter

Init policy replay buffer $B_\pi = \{\}$ and value replay buffer $B_V = \{\}$

Init parameters θ, v for policy net $\pi_\theta: \mathcal{S} \rightarrow \Delta\mathcal{A}$ and value net $V_v: \mathcal{S} \times \mathcal{S} \rightarrow [-1, 1]$

Init 'best' parameters $\theta^B \leftarrow \theta$

Init stage $g = 1$

```

1: for episode = 1, ..., N do
2:   Assign learning player:  $l \leftarrow \text{random}(\{1, -1\})$ 
3:   if  $P \leq \text{random}(0,1)$  then
4:      $\mu \leftarrow \pi_\theta$ 
5:   else
6:      $\mu \leftarrow \pi_{\theta^B}$ 
7:   for step  $t = 0, \dots, T-1$  do
8:     for player  $p = -1, 1$ , do
9:       if player  $p \neq l$  and stage  $g = 1$  then
10:        Player choose action  $a_t^p$  according to policy  $\mu$ 
11:        and update state  $s_{t+1}^p$ 
12:      else
13:        Performing policy improvement  $\mathcal{I}\pi(s_t^p)$ 
14:        and using  $V_v$  and  $\pi_\theta$  based on MCTS to
15:        choose action  $a_t^p$  and update state  $s_{t+1}^p$ 
16:        Store  $(s_t^p, \mathcal{I}\pi(s_t^p))$  in policy replay buffer
17:         $\mathcal{M}_V$ 
18:      Trajectories  $\zeta^p \leftarrow (s_0^p, a_0^p, \dots, s_{T-1}^p, a_{T-1}^p, s_T^p)$  for player  $p \in \{1, -1\}$ 
19:      if  $r(\zeta^1) \geq r(\zeta^{-1})$  then
20:        Game outcome  $z = 1$ 
21:      else
22:        Game outcome  $z = -1$ 
23:      Store tuples  $(s_t^1, s_t^{-1}, z)$  and  $(s_t^{-1}, s_{t+1}^1, -z)$  in value replay buffer  $\mathcal{M}_V$ 
24:      if  $\sum_{s_0 \in \mathcal{J}_{\text{arena}}} (r(\zeta_{0,\pi_\theta}^{\text{greedy}}) - r(\zeta_{0,\pi_{\theta^B}}^{\text{greedy}})) > 0$  then
25:        Update  $\theta^B \leftarrow \theta$ 
26:      if episode  $E \geq n$  then
27:        Set stage  $g = 2$ 

```

4.3. Network Architecture

The policy and value networks are based on the Transformer architecture, and the Transformer block is a little different from the Vanilla Transformer block, as shown in Figure 4. We employ batch normalization (BN) before the Multi-Head Attention (MHA) and add gate aggregation [7] after the MHA and feedforward network (FFN) to replace the additive aggregation. For the policy network, we use a pointing mechanism based on state attention to compute the probability of each legal action, which is similar to the method used in [23].

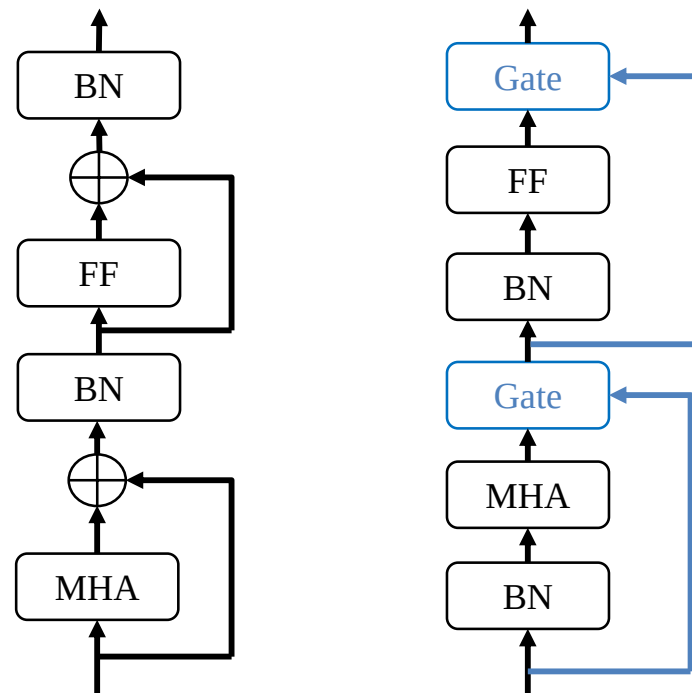


Figure 4. Vanilla Transformer block (left) and our Transformer block (right) that adds gate aggregation.

5. Experiments

5.1. Validation on TSP

In order to compare the performance of our proposed TSS GAZ PTP and the original GAZ PTP method, we first performed experiments on TSP instances with 20, 50 and 100 nodes, which are also tested by [23,26]. The coordinates for each instance are sampled from $[0, 1]^2$. For problems of different scales, we use the same hyper-parameter settings based on GAZ PTP [26]. As shown in Table 2, the experimental results demonstrate that our method achieves the best performance compared to the other historically best learning-based methods on TSP problems.

Table 2. TSS GAZ PTP VS Baselines for TSP, where each size of the test set consists of 10,000 instances.

Method	$n = 20$		$n = 50$		$n = 100$	
	Obj.	Gap	Obj.	Gap	Obj.	Gap
Optimal Solver (Concorde)	3.84	0.00%	5.70	0.00%	7.76	0.00%
Kool (Attention)	3.85	0.34%	5.80	1.76%	8.12	4.53%
GAZ PTP	3.84	0.17%	5.78	1.55 %	8.01	3.16%
TSS GAZ PTP (ours)	3.84	0.15%	5.76	1.23%	7.97	2.71%

5.2. Extension to EVRP

Since the advantage of our TSS GAZ PTP method on TSP has been validated, our aim is to extend the method to variable-step problems. Therefore, we further conducted experiments on multi-constrained EVRP instances with $n = 10, 20$ and 50 nodes, where each category consists of 512 different instances, which is the same as in [27]. For example, instances with 10 and 20 customers have four recharge stations, while instances with 50 customers have eight. Both the customer sites and recharge stations are uniformly distributed in the $[0, 100]^2$ km area, and the depot is randomly distributed in the $[25, 75]^2$ km area. The demand of each customer is uniformly distributed between 0.25, 0.5, 0.75, 1. The specific parameters and their descriptions of the vehicle are shown in Table 3. The current model employs a simplified energy consumption function based on average speed

and distance traveled. While this approach helps maintain computational feasibility, it fails to adequately reflect the dynamic energy consumption variations caused by frequent starts and stops, traffic lights, congestion, and other factors encountered on actual urban roads. For different scales of problems, we use the same hyperparameter settings based on GAZ PTP [26]. The node embedding dimension is 128 and the batch size is 256. We employ the Adam optimizer with a constant learning rate of 10^{-4} .

Table 3. The parameter values and description for multi-constrained EVRP setup.

Parameter	Description	Value
L	Capacity of the vehicle	4000 kg
m_c	Unladen load	4100 kg
Q	Battery capacity of the vehicle	80 kwh
A	Frontal surface area	3.912 m^2
ρ	Atmospheric density	1.2 kg/m^3
g	Gravitational constant	9.81 m/s^2
c_r	Resistance coefficient	0.01
c_d	Aerodynamic drag coefficient	0.7
ϕ^d	Propulsion efficiency	1.18
ϕ^r	Regenerative braking efficiency	0.85
φ^d	Charging efficiency	1.11
φ^r	Discharging efficiency	0.93

5.3. Baselines

We compare the proposed TSS GAZ PTP framework with the following methods:

- Gurobi: A commercial optimization solver.
- ACO: An improved ant colony algorithm based meta-heuristics to solve EVRP [22].
- ALNS: Adaptive large neighborhood search algorithm, which is enhanced by a local search for intensification to solve EVRP [19].
- AM: A Reinforcement Learning method based on attention mechanism [23].
- DRL: A DRL method with Transformer specifically for EVRP [27].
- GAZ PTP: A Reinforcement Learning method based on self-competition [26].
- GAZ PTP (fine-tuned): The framework is the same as GAZ PTP, but we have fine-tuned parameters for multi-constrained EVRP.

5.4. Results on EVRP

Our techniques are implemented on two NVIDIA GeForce RTX 4090 GPUs and an Intel i9-14900K CPU running at 6.00 GHz. With a total of 50 k training episodes, after the light-weight test, we divided our training into two stages—20 k and 30 k—with a simulation budget of 100 since [23,27] train on 128M trajectories. The following are the training durations for certain problems: 15 h for C10-S4, 40 h for C20-S4, and 120 h for C50-S8.

To present the training curve, we select the category C10-S4 EM-EVRP as an example and present the results in Figure 5. We see that along with training, total energy consumption gradually decreases. For our TSS GAZ PTP method, SOC consumption drastically decreased after 20 K episodes, demonstrating the efficacy of the second training phase. The training curve eventually converges, indicating that, despite sporadic oscillations during the training phase, our approach has learned the best consistent policy.

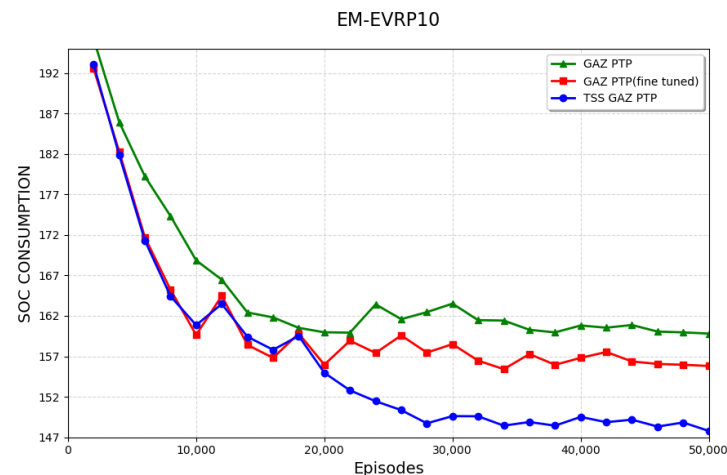


Figure 5. Results of comparison experiments among GAZ PTP, GAZ PTP (fine-tuned) and TSS GAZ PTP on category C10-S4 for EVRP.

In addition, we conducted convergence analysis experiments on the complete learning process of the TST GAZ PTP, visualizing the reduction in SOC consumption as the number of training iterations increases. For comparison, we divide the SOC consumption of EVRP for each category by the number of customers to obtain the average battery consumption per customer. The results indicate the significant improvement of the proposed method, especially for larger problems. In Figure 6, the x-axis represents the number of iterations, and the y-axis represents the average SOC consumption for visiting each customer. The decreasing SOC consumption with increasing training iterations demonstrates the effectiveness of the TST GAZ PTP algorithm on all types of EVRP instances. Importantly, we see that for larger instances, our TSS GAZ PTP achieves more significant improvements, indicating its potential to handle larger-sized problems.

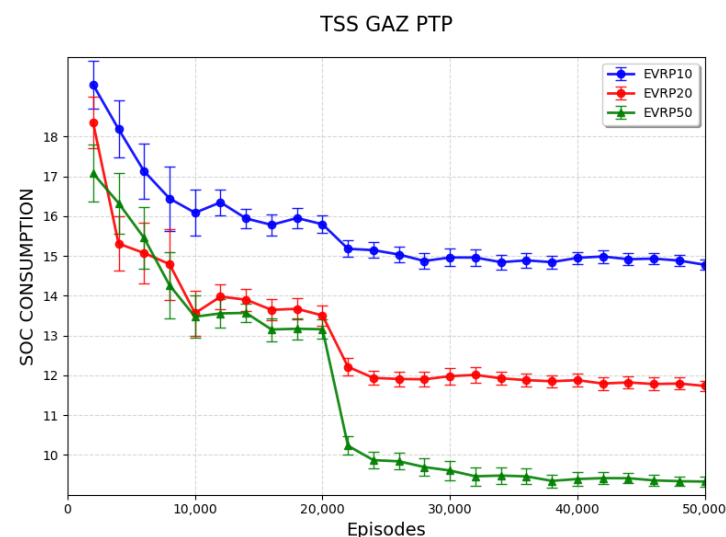


Figure 6. TSS GAZ PTP training curves on instances of category C10-S4, category C20-S4, and category C50-S8.

The complete results of the comparison experiments for instances of different sizes are shown in Table 4. Similar to [27], we measure performance through the total energy consumption E and the normalized energy consumption between the total energy consumption E and the best objective value E_{best} across all methods. For Gurobi, the time limit was set to 1 h per instance, consistent with common practice in large-scale combinatorial optimization

benchmarks. We see that in category C10-S4 and C20-S4, the Gurobi solver performs better than other methods, but fails to find optimal solutions to large-scale problems; therefore, we use “T/O” in the Table 4 to denote that the time limit was exceeded. For Gurobi, the time limit was set to 1 h per instance, consistent with common practice in large-scale combinatorial optimization benchmarks. The proposed TSS GAZ PTP performs better than other learning-based models both in multi-constrained EM-EVRP and DM-EVRP on all different-sized instances. As the number of nodes increases, its advantage becomes increasingly notable. We can see that TSS GAZ PTP significantly outperforms the previous state-of-the-art methods on all C50-S8 instances, indicating the potential of dealing with large-scale multi-constrained problems.

Table 4. TSS GAZ PTP VS Baselines for multi-constrained EM-EVRP and DM-EVRP, where each category of the test set has 512 instances.

Method	EM-EVRP10		EM-EVRP20		EM-EVRP50	
	Obj. (kwh)	Gap	Obj. (kwh)	Gap	Obj. (kwh)	Gap
Gurobi	145.25	0%	225.52	0%	T/O	T/O
ALNS	151.15	4.04%	241.54	7.19%	480.64	3.61%
ACO	151.98	4.61%	240.15	6.49%	476.46	2.71%
AM (Greedy)	152.87	5.22%	247.65	9.81%	486.54	4.88%
AM (Sample1280)	149.62	2.99%	238.63	5.81%	471.54	1.65%
AM (Sample12800)	149.28	2.75%	237.56	5.34%	468.46	0.99%
DRL (Greedy)	151.35	4.18%	244.54	8.43%	482.27	3.96%
DRL (Sample1280)	148.77	2.40%	237.43	5.28%	466.70	0.61%
DRL (Sample12800)	148.43	2.17%	236.33	4.79%	464.69	0.18%
GAZ PTP	160.14	10.25%	252.27	11.86%	570.18	22.92%
GAZ PTP (fine-tuned)	155.23	6.84%	245.85	9.01%	550.75	18.73%
TSS GAZ PTP	147.79	1.72%	234.68	4.06%	463.85	0.00%
Gurobi	346.59	0%	542.55	0%	T/O	T/O
ALNS	355.01	2.43%	572.21	5.47%	1147.31	4.26%
ACO	353.85	2.09%	568.15	4.72%	1140.74	3.66%
AM (Greedy)	361.76	4.37%	584.14	7.67%	1153.05	4.78%
AM (Sample1280)	357.23	3.07%	571.45	5.33%	1118.94	1.68%
AM (Sample12800)	355.01	2.43%	568.34	4.75%	1112.04	1.05%
DRL (Greedy)	357.24	3.07%	577.38	6.42%	1139.56	3.55%
DRL (Sample1280)	352.72	1.77%	565.17	4.17%	1108.74	0.75%
DRL (Sample12800)	352.35	1.66%	562.16	3.61%	1104.03	0.32%
GAZ PTP	380.33	9.83%	599.84	10.56%	1328.14	20.69%
GAZ PTP (fine-tuned)	366.07	5.62%	582.05	7.28%	1283.49	16.63%
TSS GAZ PTP	351.84	1.51%	560.49	3.30%	1100.47	0.00%

5.5. Visualization Analysis on EVRP

To visually compare the performance of the GAZ PTP, GAZ PTP (fine-tuned), and TSS GAZ PTP algorithm models, we conducted a visual analysis of the solutions they generate. As shown in Figure 7, we illustrate the performance of the three algorithms under the EVRP50 instance. Compared with the other two algorithms, TSS GAZ PTP has minimal path crossovers. It also has fewer total paths than the other two algorithms. The visualization demonstrates that our method has strong path planning capabilities and that our training strategy is effective.

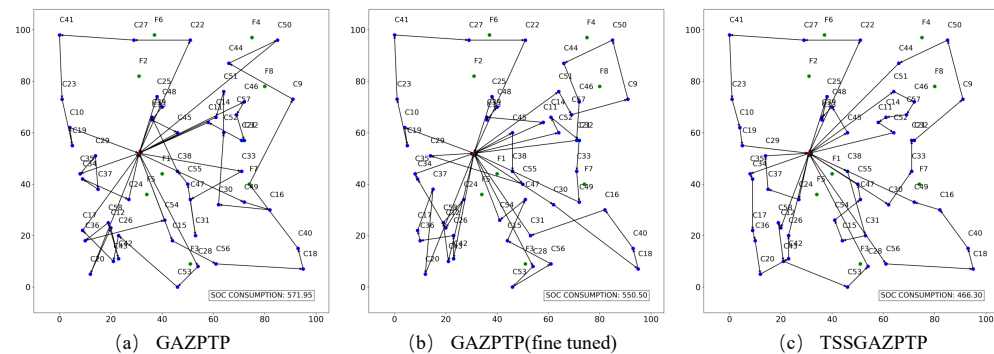


Figure 7. Visualization of solutions generated by GAZ PTP, GAZ PTP (fine-tuned) and TSS GAZ PTP on EVRP50. (a) GAZ PTP. (b) GAZ PTP (fine-tuned). (c) TSS GAZ PTP.

6. Conclusions

In this paper, we proposed TSS GAZ PTP, a novel DRL framework inspired by GAZ, for solving multi-constrained EVRP. Unlike reconstructing the network architecture or adding state feature modules for specific tasks, our aim is to design a new training strategy based on self-play that can be applied to more complex and general tasks and enhance the diversity of training data. Our training strategy ensures that the player continues to explore a smarter trajectory based on self-play and provides an efficient way to eliminate the local optimal. The experimental results show that TSS GAZ PTP consistently improves the solution quality compared to the original GAZ PTP and other deep reinforcement learning baselines, with particularly notable gains on large-scale instances.

For future work, it is also still promising to further explore more efficient self-play strategies for GAZ-type methods and apply these methods to solve more multi-constrained tasks, especially for their large-scale instances. Another direction is to explore alternative ways to reduce the computational cost caused by Monte Carlo simulations in large-scale problems. In addition, combining the proposed approach with the multi-agent methods [40] for more complex tasks requires more investigation.

Author Contributions: X.Z. and H.W.: methodology, experiments, writing original draft; C.M.: conceptualization, writing, review and editing. All authors have read and agreed to the published version of the manuscript.

Funding: Hefei Key Science and Technology Special Projects under Grant 2024SZD006.

Data Availability Statement: In the Experimental Section, we describe how the dataset was generated and provide the fixed parameter seeds. Further inquiries can be directed to the corresponding author.

Conflicts of Interest: The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence this work.

References

1. Bonfiglio, A.; Minetti, M.; Procopio, R. Vehicle-to-home service via electric vehicle energy storage virtual partitioning. *IEEE Trans. Ind. Appl.* **2025**, *61*, 7790–7802. [\[CrossRef\]](#)
2. Van Fan, Y.; Perry, S.; Klemeš, J.J.; Lee, C.T. A review on air emissions assessment: Transportation. *J. Clean. Prod.* **2018**, *194*, 673–684. [\[CrossRef\]](#)
3. Kucukoglu, I.; Dewil, R.; Cattrysse, D. The electric vehicle routing problem and its variations: A literature review. *Comput. Ind. Eng.* **2021**, *161*, 107650. [\[CrossRef\]](#)
4. Wang, G.; Qin, Z.; Wang, S.; Sun, H.; Dong, Z.; Zhang, D. Towards accessible shared autonomous electric mobility with dynamic deadlines. *IEEE Trans. Mob. Comput.* **2024**, *23*, 925–940. [\[CrossRef\]](#)
5. Fan, G.; Yang, Z.; Jin, H.; Gan, X.; Wang, X. Enabling optimal control under demand elasticity for electric vehicle charging systems. *IEEE Trans. Mob. Comput.* **2022**, *21*, 955–970. [\[CrossRef\]](#)

6. Yuan, W.; Huang, J.; Jun, Y. Competitive charging station pricing for plug-in electric vehicles. *IEEE Trans. Smart Grid* **2017**, *8*, 627–639.
7. 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. [[CrossRef](#)]
8. Li, X.; Luo, W.; Yuan, M.; Wang, J.; Lu, J.; Wang, J.; Lü, J.; Zeng, J. Learning to optimize industry-scale dynamic pickup and delivery problems. In Proceedings of the 2021 IEEE 37th International Conference on Data Engineering (ICDE), Chania, Greece, 19–22 April 2021; pp. 2511–2522.
9. Imran, N.M.; Won, M. Smartpathfinder: Pushing the limits of heuristic solutions for vehicle routing problem with drones using reinforcement learning. In Proceedings of the 2024 IEEE/RSJ International Conference on Intelligent Robots and Systems (IROS), Abu Dhabi, United Arab Emirates, 14–18 October 2024.
10. Li, J.; Wang, P.; Ma, S. The impact of different transportation infrastructures on urban carbon emissions: Evidence from china. *Energy* **2024**, *295*, 131041. [[CrossRef](#)]
11. Roselli, S.F.; Götvall, P.-L.; Fabian, M.; Åkesson, K. A compositional algorithm for the conflict-free electric vehicle routing problem. *IEEE Trans. Autom. Sci. Eng.* **2022**, *19*, 1405–1421. [[CrossRef](#)]
12. Ding, Z.; Teng, F.; Sarikprueck, P.; Hu, Z. Technical review on advanced approaches for electric vehicle charging demand management, part ii: Applications in transportation system coordination and infrastructure planning. *IEEE Trans. Ind. Appl.* **2020**, *56*, 5695–5703. [[CrossRef](#)]
13. Pan, Y.A.; Song, Y.; Yang, T.; Ding, Y.; Hu, X. Equitable urban electric vehicle charging: Feasibility and benefits of streetlight charging in kansas city right-of-way. *J. Urban Plan. Dev.* **2025**, *151*, 04025066. [[CrossRef](#)]
14. Li, J.; Tian, S.; Zhang, N.; Liu, G.; Wu, Z.; Li, W. Optimization strategy for electric vehicle routing under traffic impedance guidance. *Appl. Sci.* **2023**, *13*, 11474. [[CrossRef](#)]
15. Verma, A. Electric vehicle routing problem with time windows, recharging stations and battery swapping stations. *EURO J. Transp. Logist.* **2018**, *7*, 415–451. [[CrossRef](#)]
16. Zhang, W.; Fang, X.; Sun, C. The alternative path for fossil oil: Electric vehicles or hydrogen fuel cell vehicles? *J. Environ. Manag.* **2023**, *341*, 118019. [[CrossRef](#)] [[PubMed](#)]
17. Yang, B.; Ren, T.; Yu, H.; Chen, J.; Wang, Y. An evolutionary algorithm driving by dimensionality reduction operator and knowledge model for the electric vehicle routing problem with flexible charging strategy. *Swarm Evol. Comput.* **2025**, *92*, 101814. [[CrossRef](#)]
18. Moradi, N.; Boroujeni, N.M. Prize-collecting electric vehicle routing model for parcel delivery problem. *Expert Syst. Appl.* **2025**, *259*, 125183. [[CrossRef](#)]
19. Goeke, D.; Schneider, M. Routing a mixed fleet of electric and conventional vehicles. *Eur. J. Oper. Res.* **2015**, *245*, 81–99. [[CrossRef](#)]
20. Sistig, H.M.; Sauer, D.U. Metaheuristic for the integrated electric vehicle and crew scheduling problem. *Appl. Energy* **2023**, *339*, 120915. [[CrossRef](#)]
21. Mao, H.; Shi, J.; Zhou, Y.; Zhang, G. The electric vehicle routing problem with time windows and multiple recharging options. *IEEE Access* **2020**, *8*, 114864–114875. [[CrossRef](#)]
22. Zhang, S.; Gajpal, Y.; Appadoo, S.S.; Abdulkader, M.M.S. Electric vehicle routing problem with recharging stations for minimizing energy consumption. *Int. J. Prod. Econ.* **2018**, *203*, 404–413. [[CrossRef](#)]
23. Kool, W.; van Hoof, H.; Welling, M. Attention, learn to solve routing problems! In Proceedings of the International Conference on Learning Representations, Vancouver, BC, Canada, 30 April–3 May 2018.
24. Wang, M.; Wei, Y.; Huang, X.; Gao, S. An end-to-end deep reinforcement learning framework for electric vehicle routing problem. *IEEE Internet Things J.* **2024**, *11*, 33671–33682. [[CrossRef](#)]
25. Danihelka, I.; Guez, A.; Schrittwieser, J.; Silver, D. Policy improvement by planning with gumbel. In Proceedings of the International Conference on Learning Representations, Virtual, 25–29 April 2022.
26. Pirnay, J.; Göttl, Q.; Burger, J.; Grimm, D.G. Policy-based self-competition for planning problems. In Proceedings of the Eleventh International Conference on Learning Representations (ICLR), Kigali, Rwanda, 1–5 May 2023.
27. Tang, M.; Zhuang, W.; Li, B.; Liu, H.; Song, Z.; Yin, G. Energy-optimal routing for electric vehicles using deep reinforcement learning with transformer. *Appl. Energy* **2023**, *350*, 121711. [[CrossRef](#)]
28. Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; et al. Mastering chess and shogi by self-play with a general reinforcement learning algorithm. *arXiv* **2017**, arXiv:1712.01815. [[CrossRef](#)]
29. Guez, A.; Weber, T.; Antonoglou, I.; Simonyan, K.; Vinyals, O.; Wierstra, D.; Munos, R.; Silver, D. Learning to search with mctsnet. In Proceedings of the International Conference on Machine Learning, Stockholm Sweden, 10–15 July 2018.
30. Drori, I.; Kharkar, A.; Sickinger, R.; Kates, B.; Ma, Q.; Ge, S.; Dolev, E.; Dietrich, B.; Williamson, D.P.; Udell, M. Learning to solve combinatorial optimization problems on real-world graphs in linear time. In Proceedings of the 2020 19th IEEE International Conference on Machine Learning and Applications (ICMLA), Miami, FL, USA, 14–17 December 2020.

31. Silver, D.; Hubert, T.; Schrittwieser, J.; Antonoglou, I.; Lai, M.; Guez, A.; Lanctot, M.; Sifre, L.; Kumaran, D.; Graepel, T.; et al. A general reinforcement learning algorithm that masters chess, shogi, and go through self-play. *Science* **2018**, *362*, 1140–1144. [[CrossRef](#)]
32. Jaderberg, M.; Czarnecki, W.M.; Dunning, I.; Marris, L.; Lever, G.; Castañeda, A.G.; Beattie, C.; Rabinowitz, N.C.; Morcos, A.S.; Ruderman, A.; et al. Human-level performance in 3d multiplayer games with population-based reinforcement learning. *Science* **2019**, *364*, 859–865. [[CrossRef](#)]
33. Vinyals, O.; Babuschkin, I.; Czarnecki, W.M.; Mathieu, M.; Dudzik, A.; Chung, J.; Choi, D.H.; Powell, R.; Ewalds, T.; Georgiev, P.; et al. Grandmaster level in starcraft ii using multi-agent reinforcement learning. *Nature* **2019**, *575*, 350–354. [[CrossRef](#)]
34. Zhang, Y.; Li, M.; Chen, Y.; Chiang, Y.-Y.; Hua, Y. A constraint-based routing and charging methodology for battery electric vehicles with deep reinforcement learning. *IEEE Trans. Smart Grid* **2023**, *14*, 2446–2459. [[CrossRef](#)]
35. Wang, H.; Preuss, M.; Plaat, A. Adaptive warm-start mcts in alphazero-like deep reinforcement learning. In Proceedings of the Pacific Rim International Conference on Artificial Intelligence, Hanoi, Vietnam, 8–12 November 2021.
36. Bansal, T.; Pachocki, W.; Sidor, S.; Sutskever, I.; Mordatch, I. Emergent complexity via multi-agent competition. *arXiv* **2017**, arXiv:1710.03748.
37. Laterre, A.; Fu, Y.; Jabri, M.K.; Cohen, A.-S.; Kas, D.; Hajjar, K.; Dahl, T.S.; Kerkeni, A.; Beguir, K. Ranked reward: Enabling self-play reinforcement learning for combinatorial optimization. In Proceedings of the AAAI Conference on Artificial Intelligence, New Orleans, LA, USA, 2–7 February 2018.
38. Wang, H.; Preuss, M.; Emmerich, M.; Plaat, A. Tackling morpion solitaire with alphazero-like ranked reward reinforcement learning. In Proceedings of the 2020 22nd International Symposium on Symbolic and Numeric Algorithms for Scientific Computing (SYNASC), Timisoara, Romania, 1–4 September 2020; pp. 149–152.
39. Wang, Q.; Hao, Y.; Cao, J. Learning to traverse over graphs with a monte carlo tree search-based self-play framework. *Eng. Appl. Artif. Intell.* **2021**, *105*, 104422. [[CrossRef](#)]
40. Hao, X.; Hao, J.; Xiao, C.; Li, K.; Li, D.; Zheng, Y. Multiagent gumbel muzero: Efficient planning in combinatorial action spaces. In Proceedings of the AAAI Conference on Artificial Intelligence, Vancouver, BC, Canada, 20–27 February 2024.

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.