Dual-Drive-ERL: A Dual-Population Interactive-Driven Evolutionary Reinforcement Learning Algorithm for Dynamic Traffic Assignment

Jing-Yuan Chen
South China University of Technology
China
cscjy202321043934@mail.scut.edu.cn

Feng-Feng Wei
South China University of Technology
China
fengfeng_scut@163.com

Tai-You Chen
South China University of Technology
China
cstaiutan@mail.scut.edu.cn

Wei-Neng Chen
South China University of Technology
China
cschenwn@scut.edu.cn

***Abstract*—With the acceleration of urbanization, the demand for travel in urban areas and the number of private vehicles have risen sharply. However, the structure of the urban transportation network remains largely unchanged, and the road capacity is limited, unable to meet the increasing transportation demands, resulting in traffic congestion and further causing a series of negative impacts, such as time waste, aggravated environmental pollution, and frequent traffic accidents, ultimately leading to a low social benefit. Traditional travel planning usually only provides a few shortest path navigation suggestions, and the feedback of road conditions is often delayed, which easily leads travelers to make wrong traffic decisions, causing the traffic congestion to worsen. Reasonable traffic flow distribution can balance the traffic flow in the network and effectively alleviate traffic congestion. Therefore, the problem of traffic flow distribution has received widespread attention and research. This paper proposes a dual-population information interaction-driven evolutionary reinforcement learning algorithm, Dual-Drive-ERL, for solving the dynamic traffic flow distribution problem. This is the first time that the evolutionary reinforcement learning algorithm has been applied to the dynamic traffic flow distribution problem, and an innovative mechanism of dual-population information interaction-driven is proposed to enhance the exploration ability. The algorithm has shown superior performance to other comparison algorithms in experiments conducted in 45 different scenarios.**

***Index Terms*—traffic flow assignment,evolutionary computation,evolutionary reinforcement learning**

I. Introduction

Rapid urban growth has led to an increase in travel demand and private vehicle ownership in urban areas. However, limited road capacity and existing infrastructure fail to meet the growing travel needs, resulting in traffic congestion, which further gives rise to adverse effects such as energy consumption, environmental pollution and frequent accidents.

Traffic Assignment Problem (TAP) is a popular research direction, where traffic congestion can be alleviated through traffic flow redistribution. TAP investigates traffic flow and allocates optimal routes for travelers with given origins and destinations, aiming to minimize individual travel time or maximize the overall throughput of the road network. As an extremely challenging problem, it has been addressed by various methods proposed in academia, including mathematical programming [1] [2], heuristic algorithms [3] [4] [5] and meta-heuristic algorithms [6] [7]. In addition, some recent studies have abstracted the traffic flow assignment process as a Markov Decision Process (MDP) and attempted to solve it using reinforcement learning techniques [8].

To date, most studies have focused on traffic assignment problems in static environments, where the entry time and location of vehicles into the road network are known. Nevertheless, vehicle travel in the real world is more stochastic, which exerts unpredictable impacts on road network conditions. Only a limited number of studies have attempted to analyze scenarios with dynamically arriving vehicles. This paper focuses on the Uncertain Arrival Traffic Assignment Problem (UATAP), which is more challenging than traditional traffic assignment problems, mainly reflected in the following aspects: First, the dynamic entry of vehicles into the road network renders the deterministic solutions generated by classical optimization methods ineffective. Frequent replanning is thus required, leading to high computational costs [9]. Second, the traffic assignment problem already has a huge search space [8], and uncertain traffic demand further expands this space, increasing the difficulty of finding optimal solutions.

To address the UATAP, this paper proposes a dual-population interaction-driven evolutionary reinforcement

This work was supported by the National Key Research and Development Project No. 2023YFE0206200, the National Natural Science Foundation of China under Grant 62376097, and the Guangdong Basic and Applied Basic Research Foundation under Grant 2025A1515012028. (Corresponding author:Wei-Neng Chen).

learning algorithm named Dual-Drive-ERL, which integrates the efficient exploration capability of reinforcement learning with the powerful global search capability of evolutionary computation [10] [11] [12]. The algorithm designs a reactive policy network, which is used to distribute traffic flow to the next road segment at each node of the road network during the execution of traffic simulation. To cope with the dynamics and complexity of the problem, Dual-Drive-ERL adopts a collaborative optimization approach combining evolutionary algorithms and reinforcement learning to enhance exploration capability. Specifically, it maintains two populations with different optimization objectives, which are independently optimized using genetic algorithms and perform periodic information interaction. Therefore, the main work and contributions of this paper are summarized as follows:

- This paper proposes a dual-population interactive-driven evolutionary reinforcement learning algorithm for solving the traffic flow assignment problem with uncertain arriving travelers, marking the first application of evolutionary reinforcement learning algorithms in the field of traffic assignment.
- This paper presents a reactive policy network, which provides real-time route decision-making for traveling vehicles by perceiving the traffic conditions around road network nodes, thus effectively addressing the dynamics of traveler arrivals.
- This paper designs a dual-population interactive-driven mechanism. By maintaining two evolutionary populations with distinct optimization objectives and periodically exchanging information of elite individuals between them, the exploration capability of the evolutionary reinforcement learning algorithm is significantly improved.

II. Problem Definition

In this paper, we propose the Uncertain Arrival Traffic Assignment Problem (UATAP). In this problem, the road network is defined as $G = (V, E)$, where $V = \{v_1, v_2, ..., v_n\}$ denotes the set of traffic nodes in the network, and $E = \{e_{ij} \mid v_i, v_j \in V\}$ represents the roads in the network. Specifically, e_{ij} refers to the link between nodes v_i and v_j, and each link e_{ij} has attributes such as link length L_{ij} (with $L_{ij} > 0$). We denote the set of vehicles as $C = \{c_1, c_2, ..., c_n\}$. Each vehicle c_i is typically characterized by attributes including its origin O_i, destination D_i, and entry time into the network t_{io}, i.e., (O_i, D_i, t_{io}). In the context of UATAP, O_i, D_i, and t_{io} can all be random variables.

In UATAP, when vehicle c_i travels to an intersection v_i, if v_i is the vehicle's destination D_i, c_i has reached its destination and stops moving. Otherwise, it needs to select a suitable road from the outgoing links connected to the current intersection (denoted as $E_{\text{out}}(v_i) = \{e_{ij} \mid v_j \in V,\ e_{ij} \in E\}$) to travel to the next intersection. This process repeats until the vehicle reaches its destination D_i. The travel time of a vehicle is simulated using the Volume Delay Function (VDF), where the relationship between the actual travel time t and VDF is defined as follows:

$$t = \text{VDF}\left(1 + a\left(\frac{f_{u,v}}{C_{u,v}}\right)^b\right) t_0 \tag{1}$$

where both a and b are variables defined for the physical characteristics of the traffic road network, $f_{u,v}$ and $c_{u,v}$ respectively represent the traffic volume and capacity of the road between two adjacent intersections, namely u and v.

In UATAP, a maximum travel time limit T is given for the traffic network. Vehicles can enter the network at any time and any intersection, and travel according to a certain strategy until they reach their destination or the time limit T is exceeded. On this basis, we aim to find an appropriate travel strategy that decides the vehicle's route in real time, and maximizes the vehicle throughput of the entire network within the given time T. The objective function is expressed as:

$$\begin{cases} \max & |\{c_i \in C \mid t_{id} \le T\}| \\ \text{s.t.} & p_i^{u,v} \in E \\ & t_{io} \ge 0 \end{cases} \tag{2}$$

Here, t_{id} denotes the time when vehicle c_i arrives at its destination D_i, and $p_i^{m,n}$ represents a segment of c_i's journey: the road from intersection u to v.

III. The Proposed Algorithm

This paper proposes a dual-population interaction-driven evolutionary reinforcement learning algorithm, Dual-Drive-ERL, to address the UATAP problem. Fig. 1 presents the overall framework of the Dual-Drive-ERL algorithm, and Alg. 1 provides the pseudocode for the Dual-Drive-ERL algorithm.

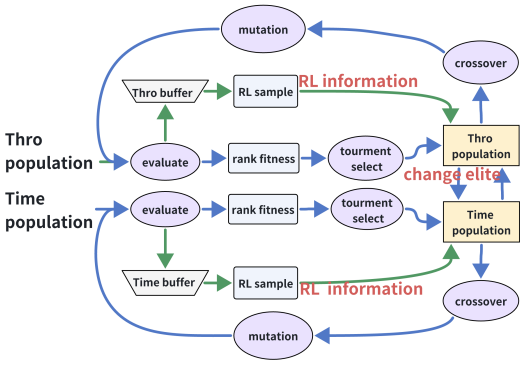

Fig. 1. Dual-Drive-ERL Framework.

A. Policy Network Design

In the UATAP problem, in order to achieve the goal of maximizing throughput, the policy network needs to real-time perceive the surrounding road conditions and plan the appropriate "next intersection" for the vehicles, thereby achieving the maximum overall throughput.

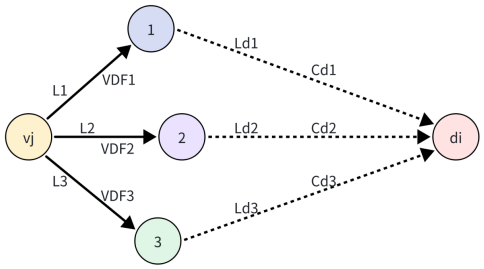

Fig. 2. Illustration of Road Intersection Information Extraction.

Suppose vehicle c arrives at intersection v_j. The policy network extracts input information based on the current intersection of the vehicle, which consists of four components (examples of these components are illustrated in Fig. 2):

L_{jk}: The length of the road between the current intersection v_j and the next optional intersection v_k, where v_k belongs to the outgoing neighbor node set of v_j.

VDF_k: The VDF of the road between the current intersection v_j and the next intersection v_k.

L_{dk}: The estimated shortest path length from the next optional intersection v_k to the destination of vehicle c.

C_{dk}: The average load of roads along the estimated shortest path from the next intersection v_k to the destination of vehicle c. Its calculation formula is given as follows:

$$C_{dk} = \frac{1}{|\{v_i \in \text{Nodes}_{e_{sk}}\}|} \sum_{k}^{d} \frac{\text{Num}_i}{\text{Capacity}_i} \tag{3}$$

Here, $Nodes_{k,d}$ denotes the set of intersections along the estimated shortest path from v_k to the destination of vehicle c. Num_i represents the number of vehicles assigned to intersection v_i. C_{dk} indirectly reflects the congestion status of subsequent travel when choosing intersection v_k.

B. Design of Reinforcement Learning Agent

The UATAP problem can be modeled as a Markov Decision Process (MDP). The reinforcement learning (RL) agent in the Dual-Drive-ERL algorithm is primarily designed based on Double DQN (DDQN) [15], where the state s_t of the RL agent is defined as:

$$s_t = (L_{jk},\ \text{VDF}_k,\ L_{dk},\ C_{dk}) \tag{4}$$

The action a_t corresponds to the directly adjacent intersections of the current intersection. In implementation, Dual-Drive-ERL uses an index-based approach to isolate the influence of intersection numbering: it sorts the adjacent intersections of the current node in clockwise order, resulting in a sorted list of adjacent intersection IDs:

$$\text{nbr\_list} = \{\text{nbr}_1, \text{nbr}_2, \ldots, \text{nbr}_n\} \tag{5}$$

After obtaining the output from the policy network, a greedy or $\epsilon - greedy$ method is used to select an index, and the actual intersection ID is retrieved from nbr_list. In addition, the reward function of the RL agent is designed as follows:

$$r_k = -\alpha \cdot C_{jk} - \beta \cdot \left(\frac{P_{\text{new}}}{P_{\text{pre}}}\right) + F_k \tag{6}$$

where $\alpha > 0$ and $\beta > 0$ are two weight coefficients. C_{jk} denotes the volume delay factor of the selected intersection; P_{new} and P_{pre} represent the subsequent travel distances of the new and old routes, respectively; F_k is a incentive term while the vehicle has reached its destination, in which case a

Algorithm 1 Dual-Drive-ERL

Input: The size of population $thro\_size$ and $time\_size$, the steps of one episode T, the number of iterations Gen, the size of Replay Buffer B, the update frequency $trans\_freq$ and $dual_freq$, the number of elites e

Output: The best strategy

1: Initialize $Thro$ population and its RL agent $Thro\_A$ and target RL-agent $Thro\_A_{tar}$
2: Initialize $Time$ population and its RL agent $Time\_A$ and target RL-agent $Time\_A_{tar}$
3: Initialize two empty replay buffer $Thro\_R$ and $Time\_R$ with size B
4: **for** $generation = 1$ to Gen **do**
5: **for** agent $a \in Thro$ **do**
6: fitness, $Thro\_R$ = Evaluate(a, $Thro\_R$, T)
7: **end for**
8: Rank the population based on fitness
9: Select first e agents as elites, node as E
10: Select $(N-e)$ agents a from $Thro$, to form Set S using tournament selection with replacement
11: **while** $|S| < N - e$ **do**
12: Randomly sample agent a_1, a_2 from E and S
13: a_1, a_2 = Crossover(a_1, a_2)
14: **end while**
15: **for** agent $a \in SetS$ **do**
16: **if** rand() $< p_m$ **then**
17: a = Mutation(a)
18: **end if**
19: **end for**
20: Generate new pop with E and S
21: _, $Thro\_R$ = Evaluate($Thro\_A$, $Thro\_R$, T)
22: Sample a batchsize (s_i, a_i, r_i, s_{i+1}) from $Thro\_R$
23: Update $Thro\_A$
24: Soft update $Thro\_A_{tar}$
25: **if** generation mode $trans\_freq$ = 0 **then**
26: Copy the RL-agent $Thro\_A$ into the $Thro$
27: **end if**
28: same operation like line 5-27 in $Time$ and $Time\_A$
29: **if** generation mode $dual\_freq$ == 0 **then**
30: The elite individuals of $Thro$ and $Time$ exchange information.
31: **end if**
32: **end for**
33: **return** The best strategy

positive reward is applied to ensure the vehicle can reach its destination. F_k is defined as:

$$F_k = \begin{cases} R, & \text{if } k == D \\ 0, & \text{else} \end{cases} \tag{7}$$

where $R > 0$ is a constant.

C. Evolution optimization design

Evolutionary algorithms are a type of gradient-free, black-box optimization algorithm with strong exploration capabilities [13] [14]. In this paper, evolutionary algorithms are used in conjunction with reinforcement learning algorithms to jointly solve the UATAP problem. The evolution optimization module in the Dual-Drive-ERL algorithm consists of two populations: the $Thro$ population (aimed at maximizing vehicle throughput) and the $Time$ population (aimed at minimizing the average vehicle travel time). Each population contains $thro\_size$ and $time\_size$ individuals, respectively. During interaction with the environment, each individual in the population independently participates in a simulation environment and decides the vehicle's next travel road based on its own strategy. The $Thro$ and $Time$ populations in Dual-Drive-ERL evolve using genetic algorithms, generating offspring through operators such as tournament selection, crossover, and mutation. Specifically, the $Thro$ population uses the number of vehicles that complete their travel as the individual fitness function, defined as:

$$\text{fitness}_{\text{Thro}}(x_i) = |\{c_k \in C \mid t_{kd} < T\}| \tag{8}$$

where x_i denotes the i-th individual in the population, and t_{kd} is the time when the k-th vehicle reaches its destination. Only vehicles that complete travel within the specified simulation time are counted.

The $Time$ population uses the negative value of the average vehicle travel time as the individual fitness function, defined as:

$$\text{fitness}_{\text{Time}}(x_j) = -\frac{1}{\sum_{k=1}^{N} \text{sign}(k)} \sum_{k=1}^{N} t_k \cdot \text{sign}(k) \tag{9}$$

where x_j is the j-th individual in the population, and the sign function is defined as:

$$\text{sign}(k) = \begin{cases} 1, & \text{if } c_k \text{ finished} \\ 0, & \text{else} \end{cases} \tag{10}$$

For the fitness value of $Time$ individuals, only the average travel time of vehicles that complete their travel is counted. This avoids the bias caused by short travel times of vehicles that have just entered the network and not yet finished their travel.

D. Information interaction-driven design

This section mainly introduces the information interaction and drive-based optimization between different modules in the Dual-Drive-ERL algorithm, which consists of two core parts: First is the information interaction and drive between evolutionary populations and RL agent. The second is the information interaction and drive between the $Thro$ and $Time$ populations.

First, we take the $Thro$ population and its corresponding RL agent as an example to illustrate the information interaction and drive between an evolutionary population and an RL agent.During the interaction between population individuals and the environment, each individual independently perceives the surrounding environment of the vehicle, makes travel decisions, and extracts quadruple experience (s_t, a_t, r_t, s_{t+1}), which is then stored in the $Thro$ experience replay buffer for sampling and learning by the $Thro$ RL agent. Additionally, according to a predefined frequency $trans\_freq$, once every $trans\_freq$ iterations, the $Thro$ RL agent will replace some inferior individuals in the $Thro$ population. These replaced inferior individuals are promoted to new elite individuals, which then generate new offspring through crossover and mutation operations.

Next, the primary objective of the Dual-Drive-ERL algorithm is to maximize vehicle throughput. However, evolutionary populations with a single optimization objective often struggle to find new growth points in the later stages of optimization, leading to a decline in individual diversity within the population. To address this issue, Dual-Drive-ERL introduces the $Time$ population, which aims to minimize average travel time, and maintains interaction with the $Thro$ population during exploration to inject diversity into each other. This forms the second part of the information interaction and drive mechanism. Specifically, an information interaction is performed once every $dual\_freq$ iterations. After individuals from both the $Thro$ and Time populations interact with the simulation environment and are sorted by fitness, the best individuals of each population are identified. Information interaction is then achieved by replacing some inferior individuals in each population with the best individuals from the other population, and promoting these replacements to new elite individuals.

IV. Experiment

This section has designed two experiments to verify the effectiveness of the proposed Dual-Drive-ERL algorithm and to compare its superiority with other algorithms.

A. Experiment Settings

This paper mainly conducts experiments on three types of networks: Regular Network(RG(N, k)), Watts-Strogatz Small World Network(WS(N, k, p)), and Grid Network(GR(M×M)). The explanations of the parameter symbols are show in Tab. I.

TABLE I
NETWORK PARAMETER SYMBOL EXPLANATION TABLE

Symbol	Explanation
N	Number of road network nodes
k	Average degree of road network intersections
p	The probability of edge reconnection during the construction of the WS network
M	The number of intersections in each row/column of the GR network

TABLE II
ANALYSIS OF NECESSARY PARAMETERS AND SETTING OF PARAMETER VALUES

Name	Value
Reward function weight coefficient α	2.0
Reward function weight coefficient β	0.2
Positive motivation R	20.0
Discount factor γ	0.95
Population size $thro\_size$	10
Population size $time\_size$	10
Gradient information frequency $trans\_freq$	1
Interpopulation interaction frequency $dual\_freq$	5

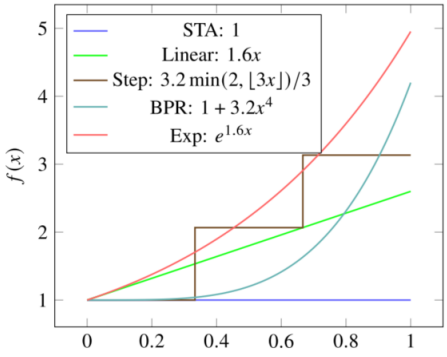

Fig. 3. Details of Five Traffic Delay Functions.

In this chapter's experiment, RG and WS networks with 100, 200, 500 and 1000 intersections were provided for the experiments, and the average degree of intersections in the RG and WS networks was set to 4. For the GR network, a road network of 10×10 size was provided. In the above settings, considering the training cost and the issue of algorithm generalization testing, the road network used for training was uniformly limited to 100 intersections.

In addition, 5 types of volume delay functions were set up for the experiment to comprehensively test the performance of the Dual-Drive-ERL algorithm, including the static function (STA), the Bureau of Public Roads function (BPR), the linear function (Linear), the step function (Step), and the exponential function (Exp) [16]. The specific function information and function curves are shown in Fig. 3.

For convenience, in the experimental data section below, the specific simulation environment is referred to using the format "{network type}_{volume delay function}_{network size}", such as "RG_BPR_1000" which refers to the simulation environment of a RG network with 1000 intersections, using BPR as the basis for vehicle operation.

In the experiments of this chapter, vehicle throughput is taken as the core evaluation metric. After training the optimal strategy network, 10 independent tests are conducted and the average value and standard deviation of the throughput are compared. The necessary parameters are shown in Table II.

B. Performance comparison experiment

The experiments in this section mainly focus on the performance comparison of the Dual-Drive-ERL algorithm, comparing it with five other algorithms on different traffic network topologies. The comparison algorithms used include Greedy, Dijkstra, CCRP, CASPER, and GEPHH. The following provides a brief introduction to the involved comparison algorithms.

(1) Greedy is an unconstrained, greedy approach that selects the shortest path to travel.

(2) Dijkstra operates under the constraints of the starting and ending points of the vehicles and follows the shortest route, with the route known.

(3) CCRP [17] is a capacity-constrained path evacuation algorithm, mainly applied to large-scale path evacuations.

(4) CASPER [18] is a traffic assignment algorithm with capacity and flow awareness, capable of reducing congestion during dynamic traffic assignment.

(5) GEPHH [19] is a traffic flow allocation solution based on genetic programming, encoding the driving strategies with genes and making decisions based on the perception of the surrounding environment at intersections.

This experiment stipulates that 1/5 of the road network is used to generate the starting-point-destination pairs of vehicles. To provide a challenging experimental environment, these generated starting-destination pairs are non-repetitive and are selected from the largest to the smallest based on the shortest path. At the same time, 200 vehicles are added as initial loads for each starting point at the beginning of the simulation.

Tab. III, IV and V report the average throughput and standard deviation of the Dual-Drive-ERL algorithm and five comparative algorithms on the RG, WS and GR networks, respectively. The Greedy algorithm performs the worst in most cases, as its greedy selection of the shortest road segments fails to ensure vehicles reach their destinations. The Dijkstra algorithm shows relatively better performance: bound by the shortest path constraint, vehicles can always arrive at destinations but follow fixed routes without real-time traffic adjustment, which increases travel time under congestion and thus reduces overall throughput. CCRP performs almost the same as Dijkstra, thanks to its capacity constraint mechanism that enables the selection of suboptimal shortest paths during congestion. CASPER outperforms CCRP and Dijkstra by dynamically adjusting travel routes based on the traffic conditions of subsequent segments, due to its traffic flow and capacity perception capability. GEPHH exhibits competitive performance in all experiments except those with Linear and Exp volume delay functions.

The proposed Dual-Drive-ERL algorithm outperforms all comparative algorithms in overall performance. It achieves the optimal throughput in 12 out of 20 experiments on the RG network. On the WS network, it yields the best throughput in 16 out of 20 experiments, and on the GR network, in 4 out of 5 experiments. Dual-Drive-ERL demonstrates dominant performance on small and medium-scale networks, surpassing all other algorithms; its only drawback is slightly weaker

TABLE III
COMPARISON OF THROUGHPUT FOR DIFFERENT ALGORITHMS IN RG NETWORK

Network	Greedy	Dijkstra	CCRP	CASPER	GEPHH	Dual-Drive-ERL
RG_BPR_100	2879.2(20.371)	4200.0(0.0)	4200.0(0.0)	4408.4(0.49)	4311.7(202.545)	**4424.2(116.077)**
RG_BPR_200	3406.6(13.2)	9000.0(0.0)	8600.0(0.0)	8552.4(2.332)	9099.5(231.871)	**9339.8(96.833)**
RG_BPR_500	1800.0(0.0)	22200.0(0.0)	21600.0(0.0)	22783.4(8.309)	23209.3(384.936)	**23774.2(264.101)**
RG_BPR_1000	1200.0(0.0)	45392.4(4.079)	47120.6(3.878)	46848.2(19.156)	**47046.8(1751.301)**	46480.6(3513.479)
RG_STA_100	4398.6(1.497)	4000.0(0.0)	4200.0(0.0)	4200.0(0.0)	4144.0(132.086)	**4607.2(155.311)**
RG_STA_200	4232.6(6.344)	8800.0(0.0)	8600.0(0.0)	8600.0(0.0)	8754.0(453.442)	**9271.4(157.519)**
RG_STA_500	4089.4(4.587)	22558.4(10.385)	21800.0(0.0)	21800.0(0.0)	22758.2(454.364)	**23084.8(1073.158)**
RG_STA_1000	3202.4(1.625)	46078.8(6.853)	45800.0(0.0)	45800.0(0.0)	**47068.4(1172.927)**	44375.2(5371.814)
RG_Linear_100	3235.0(14.029)	4400.0(0.0)	4400.0(0.0)	4219.0(0.0)	4123.6(160.43)	**4720.2(108.354)**
RG_Linear_200	7876.0(53.74)	8800.0(0.0)	8200.0(0.0)	8789.6(5.499)	7812.0(1735.618)	**9013.4(990.499)**
RG_Linear_500	9824.0(145.516)	22649.0(5.404)	21685.8(5.741)	**21979.6(4.128)**	18029.8(8024.636)	16506.8(5031.276)
RG_Linear_1000	4537.6(95.414)	44431.8(7.414)	44952.8(11.426)	**45863.8(19.395)**	36938.0(17283.345)	23877.6(14034.232)
RG_Step_100	3080.8(32.523)	4400.0(0.0)	4200.0(0.0)	4309.6(4.923)	4342.2(176.418)	**4712.8(99.775)**
RG_Step_200	7889.8(100.083)	8800.0(0.0)	8751.6(16.776)	8734.0(0.0)	8781.0(163.458)	**9081.0(605.624)**
RG_Step_500	6713.4(80.094)	22592.6(8.868)	21800.0(0.0)	**22170.0(4.0)**	21858.2(451.224)	18182.2(4546.369)
RG_Step_1000	2600.0(0.0)	44971.4(7.94)	**46231.0(73.965)**	44833.4(8.593)	45770.2(1147.978)	27732.8(12863.726)
RG_Exp_100	3912.8(24.815)	4400.0(0.0)	4400.0(0.0)	4250.2(0.748)	4296.4(95.234)	**4482.75(165.889)**
RG_Exp_200	2044.8(12.352)	9000.0(0.0)	8600.0(0.0)	8651.8(1.6)	8515.6(511.63)	**9020.0(492.046)**
RG_Exp_500	1000.0(0.0)	22399.6(0.49)	21600.0(0.0)	**23151.4(6.086)**	20989.0(2195.882)	21152.75(3490.287)
RG_Exp_1000	800.0(0.0)	45196.2(6.079)	44422.8(12.27)	**47051.2(6.969)**	41600.2(6111.135)	40539.75(9981.256)

TABLE IV
COMPARISON OF THROUGHPUT FOR DIFFERENT ALGORITHMS IN WS NETWORK

Network	Greedy	Dijkstra	CCRP	CASPER	GEPHH	Dual-Drive-ERL
WS_BPR_100	2148.4(62.81)	4951.4(24.278)	4600.0(0.0)	5248.2(23.592)	5131.2(96.961)	**5340.2(147.378)**
WS_BPR_200	2318.8(33.175)	8200.0(0.0)	8600.0(0.0)	8792.8(3.544)	8562.6(144.179)	**9069.8(176.174)**
WS_BPR_500	600.0(0.0)	22076.8(2.638)	23679.2(2.4)	23410.4(5.004)	23029.8(216.863)	**24935.2(370.871)**
WS_BPR_1000	5022.4(3.611)	45038.0(10.139)	45600.0(0.0)	45896.2(14.134)	45699.8(263.158)	**46567.8(976.281)**
WS_STA_100	2762.6(8.333)	4600.0(0.0)	5000.0(0.0)	5000.0(0.0)	5184.4(145.124)	**5482.4(88.838)**
WS_STA_200	3777.6(3.611)	8400.0(0.0)	8600.0(0.0)	8600.0(0.0)	8355.4(114.224)	**9153.6(94.764)**
WS_STA_500	1200.0(0.0)	23000.0(0.0)	23600.0(0.0)	23600.0(0.0)	22002.4(2166.629)	**24856.0(489.224)**
WS_STA_1000	7349.8(7.859)	43800.0(0.0)	44400.0(0.0)	44800.0(0.0)	41452.4(8116.637)	**45887.4(3231.846)**
WS_Linear_100	2815.0(33.148)	4888.0(5.692)	5543.4(4.964)	5047.2(20.798)	5187.8(86.4)	**5788.6(214.375)**
WS_Linear_200	7318.4(64.803)	8200.0(0.0)	8200.0(0.0)	8360.0(0.0)	8400.0(0.0)	**9081.8(264.327)**
WS_Linear_500	5385.2(71.976)	22486.8(3.655)	22176.8(4.956)	22776.6(7.392)	22180.0(250.0)	**23873.6(2892.649)**
WS_Linear_1000	13224.2(159.793)	44800.0(0.0)	44800.0(0.0)	**46595.8(11.957)**	45125.0(107.517)	35509.0(8895.413)
WS_Step_100	2575.8(38.306)	4932.4(6.621)	5050.2(6.997)	4827.4(5.643)	4750.4(124.175)	**5638.6(173.972)**
WS_Step_200	7076.2(80.086)	8200.0(0.0)	8200.0(0.0)	8666.0(0.0)	8233.8(67.6)	**9015.4(165.004)**
WS_Step_500	4009.8(61.917)	22758.8(11.089)	22872.4(10.21)	22643.0(3.347)	22415.6(245.08)	**25640.4(928.749)**
WS_Step_1000	9776.6(83.521)	45083.8(12.844)	44703.6(20.304)	**46333.6(9.749)**	45901.6(260.887)	42120.6(6877.776)
WS_Exp_100	1740.2(31.403)	4976.6(9.265)	5000.8(7.626)	5416.0(4.29)	4987.4(143.305)	**5420.75(40.69)**
WS_Exp_200	1612.6(10.307)	8200.0(0.0)	8200.0(0.0)	8731.6(3.611)	8433.6(132.276)	**8991.25(83.248)**
WS_Exp_500	0.0(0.0)	22247.4(4.841)	22800.0(0.0)	**23880.4(8.212)**	23245.6(362.638)	23804.5(459.028)
WS_Exp_1000	3922.4(22.473)	45344.8(18.104)	44550.6(5.607)	**47136.2(5.636)**	45544.2(241.806)	39797.75(664.066)

TABLE V
COMPARISON OF THROUGHPUT FOR DIFFERENT ALGORITHMS IN GR NETWORK

Network	Greedy	Dijkstra	CCRP	CASPER	GEPHH	Dual-Drive-ERL
GR_BPR_10_10	3837.0(30.757)	4600.0(0.0)	4600.0(0.0)	5052.8(1.166)	5010.2(358.053)	**5379.5(100.126)**
GR_STA_10_10	4262.8(7.756)	5200.0(0.0)	4800.0(0.0)	4800.0(0.0)	**5280.0(348.712)**	5257.8(128.229)
GR_Linear_10_10	4906.0(61.384)	4358.6(5.314)	4534.4(3.441)	5064.8(12.703)	4560.6(356.175)	**5593.0(208.811)**
GR_Step_10_10	5246.8(11.268)	4255.8(6.431)	4400.0(0.0)	4678.6(3.98)	4816.4(378.207)	**5354.4(254.961)**
GR_Exp_10_10	3005.8(3.709)	4400.0(0.0)	4400.0(0.0)	4961.0(2.828)	4888.0(118.152)	**5106.4(292.853)**

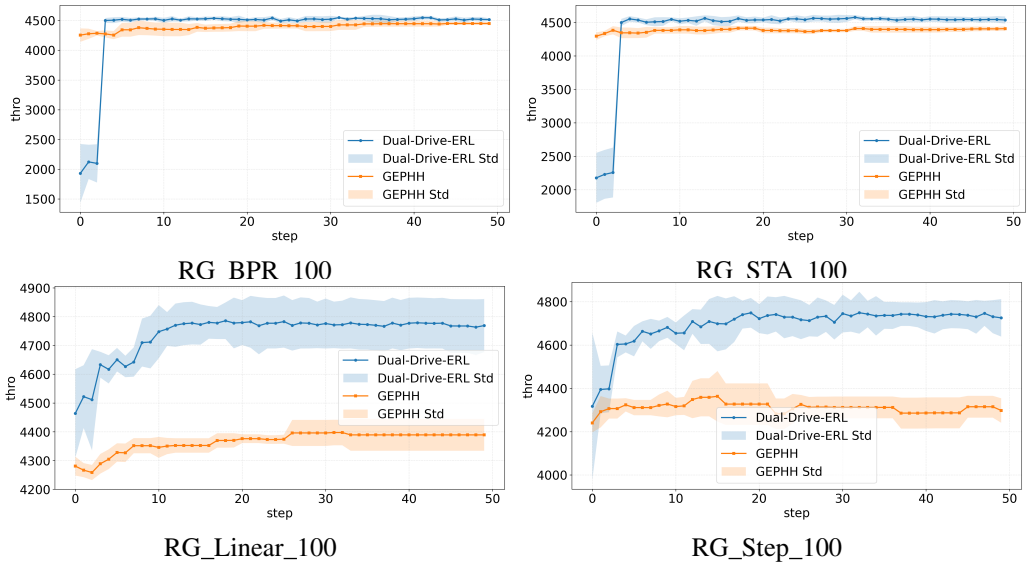

Fig. 4. The average curve of road distance changes during the training process of the Dual-Drive-ERL algorithm and the GEPHH algorithm

generalization ability on large-scale networks. In addition, Fig. 4 presents the partial convergence curves of Dual-Drive-ERL and GEPHH on the three networks, where Dual-Drive-ERL converges faster and more stably with better performance than GEPHH. Based on the above analysis, the Dual-Drive-ERL algorithm outperforms other algorithms in throughput for the UATAP problem, serving as a competitive solution to this problem.

V. Conclusion

This paper proposes a dual-population interaction-driven evolutionary reinforcement learning algorithm named Dual-Drive-ERL for solving the traffic flow assignment problem with uncertain arrival passengers. This algorithm combines the advantages of evolutionary algorithms and reinforcement learning and innovatively proposes a dual-population interaction-driven mechanism, enabling it to obtain high-quality solutions during the exploration process. Simulation experiments in 45 different scenarios show that Dual-Drive-ERL has strong performance and adaptability. However, in some scenarios, Dual-Drive-ERL is slightly inferior to other algorithms; moreover, it also incurs relatively high training costs, and these issues need to be addressed in future research.

References

[1] Akamatsu T, Wada K, Iryo T, et al. A new look at departure time choice equilibrium models with heterogeneous users[J]. Transportation Research Part B: Methodological, 2021, 148: 152-182.

[2] Osawa M, Fu H, Akamatsu T. First-best dynamic assignment of commuters with endogenous heterogeneities in a corridor network[J]. Transportation Research Part B: Methodological, 2018, 117: 811-831.

[3] Ameli M. Heuristic methods for calculating dynamic traffic assignment[D]. Université de Lyon, 2019.

[4] Tong C O, Wong S C. Heuristic algorithms for simulation-based dynamic traffic assignment[J]. Transportmetrica, 2010, 6(2): 97-120.

[5] Lu Q, George B, Shekhar S. Capacity constrained routing algorithms for evacuation planning: A summary of results[C]//International symposium on spatial and temporal databases. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005: 291-307.

[6] Ameli M, Lebacque J P, Leclercq L. Simulation-based dynamic traffic assignment: Meta-heuristic solution methods with parallel computing[J]. Computer-Aided Civil and Infrastructure Engineering, 2020, 35(10): 1047-1062.

[7] Huang Z M, Chen W N, Li Q, et al. Ant colony evacuation planner: An ant colony system with incremental flow assignment for multipath crowd evacuation[J]. IEEE Transactions on Cybernetics, 2020, 51(11): 5559-5572.

[8] Shou Z, Chen X, Fu Y, et al. Multi-agent reinforcement learning for Markov routing games: A new modeling paradigm for dynamic traffic assignment[J]. Transportation Research Part C: Emerging Technologies, 2022, 137: 103560.

[9] Grunitzki R, Bazzan A L C. Comparing two multiagent reinforcement learning approaches for the traffic assignment problem[C]//2017 Brazilian Conference on Intelligent Systems (BRACIS). IEEE, 2017: 139-144.

[10] Chen W N, Wei F F, Zhao T F, et al. A survey on distributed evolutionary computation[J]. arXiv preprint arXiv:2304.05811, 2023.

[11] Chen T Y, Chen W N, Wei F F, et al. The Confluence of Evolutionary Computation and Multi-Agent Systems: A Survey[J]. IEEE/CAA Journal of Automatica Sinica, 2025, 12: 1-19.

[12] Wei F F, Chen W N, Guo X Q, et al. Crowdec: Crowdsourcing-based evolutionary computation for distributed optimization[J]. IEEE Transactions on Services Computing, 2024.

[13] Wei F F, Chen W N, Zhang J. AIEA: An asynchronous influence-based evolutionary algorithm for expensive many-objective optimization[J]. IEEE Transactions on Cybernetics, 2024.

[14] Chen T Y, Chen W N, Wei F F, et al. Multi-agent swarm optimization with adaptive internal and external learning for complex consensus-based distributed optimization[J]. IEEE Transactions on Evolutionary Computation, 2024.

[15] Van Hasselt H, Guez A, Silver D. Deep reinforcement learning with double q-learning[C]//Proceedings of the AAAI conference on artificial intelligence. 2016, 30(1).

[16] Spiess H. Conical volume-delay functions[J]. Transportation Science, 1990, 24(2): 153-158.

[17] Yin D. A scalable heuristic for evacuation planning in large road network[C]//Proceedings of the Second International Workshop on Computational Transportation Science. 2009: 19-24.

[18] Shahabi K, Wilson J P. CASPER: Intelligent capacity-aware evacuation routing[J]. Computers, Environment and Urban Systems, 2014, 46: 12-24.

[19] Liao X C, Jia Y H, Hu X M, et al. Uncertain commuters assignment through genetic programming hyper-heuristic[J]. IEEE Transactions on Computational Social Systems, 2023, 11(2): 2606-2619.