Coordinated Control of Intelligent Fuzzy Traffic Signal Based on Edge Computing Distribution

Chaodong Yu, Jian Chen and Geming Xia *

School of Computer Science and Engineering, National University of Defense Technology, Changsha 410003, China
* Correspondence: xiageming@163.com

Abstract: With the development of Internet of Things infrastructures and intelligent traffic systems, the traffic congestion that results from the continuous complexity of urban road networks and traffic saturation has a new solution. In this research, we propose a traffic signal control scenario based on edge computing. We also propose a chemical reaction–cooperative particle swarm optimization (CRO-CPSO) algorithm so that flexible traffic control is sunk to the edge. To implement short-term real-time vehicle waiting time prediction as a collaborative judgment of CRO-CPSO, we suggest a traffic flow prediction system based on fuzzy logic. In addition, we introduce a co-factor (collaborative factor) set based on offline learning to take into account the experiential characteristics of intersections in urban road networks for the generation of strategies by the algorithm. Furthermore, the real case of Changsha County is simulated on the SUMO simulation platform. The issue of traffic flow saturation is improved by our method. Compared with other methods, our algorithm enhances the proportions of vehicles that reach their destinations on time by 13.03%, which maximizes the driving experience for drivers. Meanwhile, our algorithm reduces the driving times of vehicles by 25.34%, thus alleviating traffic jams.

Keywords: ITS; edge computing; swarm intelligence; traffic signal control; fuzzy logic; offline learning

1. Introduction

With the rapid increase in the numbers of vehicles and a large and complex road network, delays, traffic accidents and environmental pollution due to vehicle queuing caused by traffic congestion have created an urgent need for traffic control strategies. How to improve traffic efficiency while reducing the traffic accident rate by using efficacious control measures has attracted the attention of both academia and industry [1–4]. Traffic signal control can effectively alleviate the saturated traffic conditions and improve the utilization of the road network. In recent years, many studies have focused on traffic signal control.

The earliest traffic control methods relied on hand signals [2]. In order to alleviate the economic losses caused by traffic congestion in Toronto, Hewton [1] proposed online optimization of traffic control signals through computers, which promoted the application of computer technology in the field of traffic management. For reasons of safety, Inose H [3] coordinated the timing of traffic signals according to the traffic flow on a road. With the continuous expansion of traffic scale, automatic traffic control strategies [4] have gradually formed a system. In the era of cloud computing, artificial intelligence (AI) algorithms continue to develop, and new concepts such as intelligent transportation systems (ITSs) have been put forward [5]. Wang [6] and Wu [7] et al. designed adaptive traffic signal control strategies by using deep reinforcement learning (DRL) of multi-agent cooperation to deal with large and complex road networks. Shao [8] took the weight of special vehicles into extra consideration when setting the state and reward function, and had an appropriate preference for special vehicles in traffic signal control. With the emergence of Internet of
Vehicles (IoV) [9], edge computing [10–12] and 6G [13], vehicle-to-infrastructure (V2I) and traffic management mechanisms will make giant leaps. With the heterogeneity of the vehicle network and the further complexity of road structure, space–air–ground and even underwater vehicles may appear in the future [14], and road structure may tend to be multi-dimensional and multi-forking. Therefore, the proactive exploration of traffic signal control strategies is required. At present, many researches have carried out short-term traffic prediction based on the IoV [15–19], thus providing relatively reliable real-time predictions of changes in traffic conditions. In their study, Yang et al. [16] found that the total queuing time at traffic lights accounted for a large part of total driving time, which can be more than 50%. Thus, an efficient traffic signal control strategy is necessary. Wu et al. [20] thought that an intelligent algorithm was too expensive to be adapted to highly dynamic traffic flow when designing a traffic signal control strategy. The integration of edge computing and ITS [21] can sink the perception, collection and processing of traffic data to the edge. Through the reasonable deployment of an edge server [22], all the data on the traffic network can be interactive with low delay, making intelligent traffic signal control at the edge possible. Chen [23] applied an edge-distributed architecture combined with dynamic route guidance and signal control to effectively improve the traffic efficiency of the road network. However, in general, the above methods did not consider the use of regional traffic lights to achieve local coordinated regulation so as to achieve intelligent and efficient traffic flow scheduling. Moreover, traffic signal control using the architecture of edge computing needs further research.

Due to the existence of certain constraints and dynamic events (such as emergency vehicles and traffic evacuation, etc.), as well as subjective factors relating to drivers and pedestrians that need to be considered, uncertainty characterizes road traffic situations. Therefore, a fuzzy system [24] has been applied to effectively match the uncertainty of vehicles, traffic signals and control systems. Chou [25] simulated a traffic environment which is close to a real road network structure, and he designed a fuzzy logic controller suitable for this environment, which can achieve efficient reasoning with a few control rules. Gokulan [26] took limited environmental information as input and configured different levels of uncertainty in the rule base to realize a fuzzy logic reasoning system responding to traffic changes. Zaied [27] adjusted the time interval of traffic signals based on fuzzy logic decisions based on traffic conditions. He tested the system on real data sets and thereby increased the cycle speed of the system. The implementation of a fuzzy logic system is low cost, and we can achieve better inference effects through well-designed fuzzy rules. In this paper, a fuzzy logic system is used to predict traffic flow, and it is introduced into the fitness function of the traffic signal control algorithm to achieve global coordination evaluation.

The swarm intelligence algorithm [28] has also been studied in the context of solving the multi-dimensional nonlinear programming problem of traffic signal control. Ceylan [29] realized traffic signal timing optimization of balanced networks through genetic algorithms, thus improving computational efficiency and performance. Jose [30] optimized the traffic signal regulation of a large heterogeneous urban road network through a particle swarm optimization algorithm which reflected the potential effect of swarm intelligence algorithms in addressing such problems. Nasser [31] introduced an index scheme to control the iteration time of local optimization, thus accelerating the convergence speed of traffic signal timing strategy. Aleksandar [32] optimized the traffic control system of a large area in regional cities through the cluster principle of a bee colony collecting honey, thus improving the control range of a swarm intelligence algorithm for an urban road network. However, the swarm intelligence algorithm is still unable to cope with large-scale urban road network structures and the continuous complexity of traffic conditions. Moreover, how to construct a cooperative optimization framework to adapt to large-scale transportation network optimization is a problem that also needs further study.

Our study has investigated all of the research gaps mentioned above. We completely utilize the arithmetic power and control of edge scene, and we achieve cooperative traffic signal control based on fuzzy logic and the swarm intelligence algorithm. With this
initially, we propose a traffic signal control scenario based on edge computing, and then we propose a new swarm intelligence algorithm: chemical reaction–cooperative particle swarm optimization (CRO-CPSO). We change the generation method of the traditional swarm intelligence strategy and make full use of the idle computing power at the edge to generate a local strategy so that the flexible traffic control sinks to the edge. The CRO-CPSO algorithm can effectively adapt to large-scale urban road network structures and complex and dynamic traffic conditions.

Secondly, we propose a traffic flow prediction system based on fuzzy logic and maintain the global traffic flow state table. We implement short-term real-time vehicle waiting time prediction as a collaborative judgment of CRO-CPSO, which effectively responds to the uncertainty of road traffic and achieves global coordination evaluation of the strategy.

Then, we introduce a co-factor (collaborative factor) set based on offline learning to take the experiential characteristics of intersections in an urban road network into account when generating strategies. The co-factor set integrates the potential influence of historical traffic flow of adjacent sections into the strategy generation, which increases the convergence of the algorithm and effectively adapts to the complexity of traffic, thus enhancing coordination in the regulation of large road network structures.

Finally, we examine the real case of Changsha County using the traffic simulator SUMO and compare other optimization algorithms (the evolutionary algorithm GA and the swarm intelligence algorithm PSO) to prove the effectiveness and advantages of CRO-CPSO.

The remainder of this paper is organized as follows. In Section 2, we describe our optimization methodologies, including an overall model diagram. In Section 3, we introduce the experimental methods and results. The conclusions and prospects for future work are detailed in Section 4.

2. Method

In this section, the specific details of the edge swarm intelligence ATSC environment (the scene of traffic signal control), the global traffic flow prediction (involved in collaborative judgment), and the CRO-CPSO model (the core of our solver technique) are introduced.

2.1. Edge Swarm Intelligence ATSC (Adaptive Traffic Signal Control) Environment

This subsection describes the structure and the data flow of the edge swarm intelligence ATSC environment.

Figure 1 shows the edge swarm intelligence ATSC environment, which includes cloud servers, MEC servers, traffic signal controllers, RSUs (roadside units) and vehicle units. The cloud layer carries on the global overall coordination. Cloud servers aggregate and generate the global co-factor set and predict the global traffic flow so as to make a collaborative judgment on the strategy of CRO-CPSO. In the MEC layer, the local strategy generation of CRO-CPSO is carried out, and the local co-factor set is trained off-line. MEC servers are deployed in their distribution area to achieve instantaneous data transmission with the traffic signal controllers. Vehicle units communicate with each other via V2V and exchange information with RSUs via V2I. The information is fed to MEC servers which communicate with each other via I2I. After aggregation and preprocessing by MEC servers, local road pheromone information is generated and summarized for cloud servers. The cloud servers perform fuzzy logic processing on the road pheromone information to obtain short-term real-time vehicle waiting time prediction as a collaborative judgment on the strategy of CRO-CPSO.
waiting time prediction is used as the collaborative judgement on local traffic signal control which is uploaded to the cloud layer. The cloud layer takes the road pheromone information ATSC environment, the underlying data constitute the inherent data set for a road, such as preprocesses road network data to generate local road pheromone information and upload it to the cloud layer. The cloud layer takes the road pheromone information which can be used for traffic flow prediction. Short-term real-time vehicle waiting time prediction. Finally, the MEC layer transmits the optimal strategy of CRO-CPSO to the user layer.

2.2. The Global Traffic Flow Prediction

This subsection describes the pheromones in the global traffic flow prediction, the input and output fuzzy membership of the fuzzy logic system, and finally the update strategy for the states of vehicle units.

We designed a method to aggregate and process edge-distributed road pheromone information which can be used for traffic flow prediction. Short-term real-time vehicle waiting time prediction is used as the collaborative judgement on local traffic signal control strategies and provides support for local coordinated adjustment. In the edge-distributed ATSC environment, the underlying data constitute the inherent data set for a road, such as road structure names, identifications, geographical location and its graph topology. The dynamic data set of the road includes real-time vehicle speed and positional information from vehicle units and RSUs and are then further filtered and processed. The MEC servers obtain real-time speed and positional information from vehicle units and obtains inherent road data sets from RSUs. Then, MEC servers predict short-term real-time vehicle waiting time.
Aggregating and generating global traffic flow predictions is a way to directly reflect the real-time state information of vehicle units in highly dynamic road networks. The pheromones used in global traffic flow prediction mainly include VRA (the current road name of the vehicle unit), VST (the time of vehicle unit switching state), VRN (the number of remaining roads for the vehicle unit) and so on. VRA can be obtained by analyzing the real-time location and speed of a vehicle unit and road map topology via V2I. When a vehicle unit runs on a road section at normal speed, then VRA is the name of this road section. When the real-time speed of the vehicle unit is lower than the normal speed, then VRA is “waiting_state”. The state of the vehicle unit mainly includes two states: moving and waiting. VST is mainly divided into two types: the remaining time from moving state to waiting state and the remaining time from waiting state to moving state. VST is mainly composed of experienced travel time, intersection delay caused by traffic signal control and delay caused by traffic congestion. VST can be calculated from the real-time vehicle unit speed and the road map topology obtained via V2I and the real-time queue at the intersection obtained via I2I. VRN is the prediction of the remaining path of the vehicle unit which can be obtained via V2I.

The first step is to calculate the experienced travel time. The speed information of vehicle units on the road network is updated frequently; thus, we calculate the real-time spatial average speed and use the real-time spatial average speed to calculate the experienced travel time. The real-time spatial average speed is the average speed of k samples within a time interval t, which can be calculated by Equation (1). Then, we divide the length of the road section travelled in time interval t by the real-time spatial average speed to obtain the experienced travel time, as shown in Equation (2).

\[ \bar{v}_{V_{\sigma}} = \frac{\sum_{m=1}^{k} v_{m, V_{\sigma}}}{k} \]  
\[ T_{V_{\sigma}}^{t} = \frac{L_{V_{\sigma}}^{t}}{\bar{v}_{V_{\sigma}}} \]  

where \( v_{m, V_{\sigma}} \) is the instantaneous speed of the mth sample of vehicle unit \( V_{\sigma} \) and \( L_{V_{\sigma}}^{t} \) is the length of the road section travelled by vehicle unit \( V_{\sigma} \) in time interval \( t \).

Then, we calculate the delay caused by traffic congestion. The traffic flow is a continuous cycle, and the subjective factors of drivers are uncontrollable. Any slight change in the traffic cycle may result in a mismatch between the calculated and optimal cycle. Thus we use fuzzy logic [33] in each successive cycle to overcome this mismatch. We construct the fuzzy system for the waiting time of vehicle units by establishing relations between the inputs and outputs of the fuzzy system using if–then rules. The proposed fuzzy logic system consists of four inputs: VQL (the queue length of vehicle units), IWJ (the judgement of whether to wait at the intersection), GPD (the duration of the green phase) and VP (the priority of vehicle units).

The VQL is the remaining queue length of vehicle units in the traffic flow. It contains three membership functions named Zero, Short and Long that range from 0 to 25 vehicle units, as illustrated in Figure 2a.

The IWJ is the judgement of whether a vehicle unit is waiting at the intersection. It contains three membership functions named Ahead Oftime, Ordinary and Delayed that range from 0% to 100%, as illustrated in Figure 2b.

The GPD is the duration of the green phase of the traffic light. It contains three membership functions named Short, Medium and Long that range from 0 to 60 s, as illustrated in Figure 2c.

The VP is the priority of vehicle units on the road network. It contains three membership functions named Optimal, Suboptimal and Ordinary that range from 0% to 100%, as illustrated in Figure 2d.
When VST will continue to drive. Thus, it is necessary to analyze the change of the state of the vehicle. The state of the vehicle unit needs to be updated constantly to ensure real-time and accurate representation of the current traffic status. The update strategy for the state of vehicle units is shown in Algorithm 1. With increase in $T_s$, VST decreases synchronously. When VST is non-zero, the vehicle unit will maintain the current state unchanged. When VST is zero, we need to judge whether the intersection of the current road section is the destination of the vehicle unit. When the vehicle unit does not reach the final destination, it will continue to drive. Thus, it is necessary to analyze the change of the state of the vehicle unit. If the vehicle unit is in “waiting_state”, then it ends the “waiting_state” and continues to drive on the next road section. VST will be reset and the experienced travel time for the next road section and intersection delay caused by traffic signal control will be added, as shown in Equation (3). If the vehicle unit was in the driving state before, then it arrives at the intersection. We need to judge whether the vehicle unit needs to wait for the traffic light. If the vehicle unit does not need to wait for the traffic light, it will pass through the intersection directly and continue driving on the next road section. VST will be reset and

The output of the fuzzy logic system $WT$ (the waiting time of a vehicle unit) is used to identify the delay caused by traffic congestion. It contains five membership functions called $RU$ (Road Unobstructed), $LC$ (Light Congestion), $MC$ (Moderate Congestion), $SC$ (Severe Congestion), $TU$ (Traffic Tie-Up) that range from 0 to 100s, as illustrated in Figure 3. The proposed fuzzy logic system comprises forty-five fuzzy rules. Some of these rules are shown in Table 1, below.

**Table 1.**

| Rule No | The Rule |
|---------|----------|
| Rule 1 | If $VQL$ is Short and $GPD$ is Short and $VP$ is Optimal and $IWJ$ is Delayed, then $WT$ is $LC$ |
| Rule 2 | If $VQL$ is Short and $GPD$ is Short and $VP$ is Optimal and $IWJ$ is Delayed, then $WT$ is $LC$ |
| Rule 3 | If $VQL$ is Short and $GPD$ is Short and $VP$ is Optimal and $IWJ$ is Delayed, then $WT$ is $LC$ |
| Rule 4 | If $VQL$ is Short and $GPD$ is Short and $VP$ is Optimal and $IWJ$ is Delayed, then $WT$ is $LC$ |
| Rule 5 | If $VQL$ is Short and $GPD$ is Short and $VP$ is Optimal and $IWJ$ is Delayed, then $WT$ is $LC$ |

**Figure 2.** The input fuzzy membership of the fuzzy logic system.

**Figure 3.** The output fuzzy membership of the fuzzy logic system.
calculated, as shown in Equation (3). When the vehicle unit needs to wait for the traffic light, its VRA will be “waiting_state”. VST will be reset and WT will be added, as shown in Equation (4). When the vehicle unit arrives at its destination, we will no longer predict its real-time vehicle state, thus saving computing resources.

\[ VST = VST + T_t + t_{\text{delay}} \] (3)

\[ VST = VST + WT \] (4)

where \( t_{\text{delay}} \) is the intersection delay caused by traffic signal control.

\[ \text{Algorithm 1} \text{ The update strategy of the state of vehicle unit} \]

1: \( \text{if } T_t = 0 \& \& VST > 0 \& \& \text{VRA} \neq \text{"waiting_state"} \text{ then} \)
2: \( \text{Analysis of vehicle unit queuing:} \)
3: \( \text{if } WT = 0 \text{ then} \)
4: \( \text{Update VRA to name of next road section;} \)
5: \( \text{Reset VST;} \)
6: \( VST = VST + T_t + t_{\text{delay}}; \)
7: \( VRN = VRN - 1; \)
8: \( \text{else if } WT \neq 0 \text{ then} \)
9: \( \text{VRA = "waiting_state";} \)
10: \( \text{Reset VST;} \)
11: \( VST = VST + WT; \)
12: \( \text{end if} \)
13: \( \text{else if } \text{VRA = "waiting_state" } \& \& WT = 0 \text{ then} \)
14: \( \text{VRA = "waiting_state";} \)
15: \( \text{Reset VST;} \)
16: \( VST = VST + WT; \)
17: \( VRN = VRN - 1; \)
18: \( \text{else if } WT = 0 \& \& VRN = 0 \text{ then} \)
19: \( \text{Remove the vehicle unit from the predicting cycle;} \)
20: \( \text{end if} \)

\[ \text{Table 1. Some rules example of the fuzzy logic.} \]

| Rule No | The Rule |
|---------|----------|
| Rule 1  | If VQL is Short and VP is Optimal and IWJ is Ordinary, then WT is RU |
| Rule 2  | If VQL is Zero and IWJ is Ordinary, then WT is RU |
| Rule 3  | If VQL is Zero and IWJ is Delayed, then WT is SC |
| Rule 4  | If VQL is Short and GPD is Short and VP is Optimal and IWJ is Delayed, then WT is LC |
| Rule 5  | If VQL is Short and GPD is Short and VP is Optimal and IWJ is Delayed, then WT is SC |
| Rule 6  | If VQL is Short and GPD is Short and VP is Optimal and IWJ is Delayed, then WT is SC |
| ...     | ...      |
| Rule 42 | If VQL is Long and GPD is Long and VP is Suboptimal and IWJ is Delayed, then WT is SC |
| Rule 43 | If VQL is Long and GPD is Long and VP is Ordinary and IWJ is Ahead Oftime, then WT is TU |
| Rule 44 | If VQL is Long and GPD is Long and VP is Ordinary and IWJ is Ordinary, then WT is MC |
| Rule 45 | If VQL is Long and GPD is Long and VP is Ordinary and IWJ is Ahead Oftime, then WT is TU |
2.3. CRO-CPSO Model

This subsection describes the fitness function, the co-factor set, the solution encoding and finally the optimization procedure for our proposed CRO-CPSO algorithm.

Research [29–32] on traffic signal control optimization has shown that swarm intelligence algorithms can outperform traditional methods in many cases. However, swarm intelligence algorithms have slow convergence speeds when dealing with multi-constraint optimization problems and cannot be well adapted to the current problem of large-scale and complex road networks. Thus, this paper proposes a distributed adaptive cooperative chemical reaction–cooperative particle swarm optimization (CRO-CPSO) algorithm. CRO-CPSO generates and iterates the local strategy at the edge via a distributed structure. It avoids the exponential growth of the solution space when confronted with large-scale road networks. The distributed structure also promotes cooperative control among traffic signal lights in the surrounding area. We use energy exchange as an indicator scheme to achieve an adaptive combination of local search and global exploration in the solution space. We also use a co-factor set to realize cooperative and coordinated control actions among adjacent edges and offline learning with historical traffic flow data.

When using the swarm intelligence algorithm to solve traffic signal control optimization, the scale of solution space will increase exponentially with the continuous expansion of a road network and the increasing complexity of intersections. Using the traditional centralized method, even if there is a powerful centralized cloud computing server, there will be a high computing cost and overhead. Thus, it cannot adapt to large-scale road networks. With the development of edge computing, the computing resources deployed on the edge have a wide application prospect. In our proposed CRO-CPSO, the local strategy for each intersection is generated using the computing resources of edge servers deployed at the intersections. It realizes the parallel utilization of distributed computing resources and reduces the computing load of the cloud server. The co-factor set is maintained on the adjacent edge server to realize offline learning of historical traffic data and cooperative control of adjacent traffic signals.

The traffic signal control optimization problem in this paper is a typical objective optimization problem. The solution space of the signal control set is divided into two parts: \( S' = \{s'_1, s'_2, \ldots, s'_{N_{ES}}\} \) and \( S'' = \{s''_1, s''_2, \ldots, s''_{N_{ES}}\} \), where \( S' \) is the green phase sequence of each traffic light at the intersection, \( S'' \) is the duration of the green phase of the traffic light at the intersection and \( N_{ES} \) is the number of traffic lights at the intersection.

Considering information on various events during the simulation, the fitness function of CRO-CPSO is shown as Equation (5). Equations (8)–(10) are the constraints on it. The main objective of Equation (5) is to enhance the driver’s driving experience and reduce driving time. We achieve the incentive effect by increasing the number of vehicles that arrive at the destination within the reward time \( T_{II} \) and giving the reward score \( D_{score} \).

We set the decision factor for the reward score \( \chi \) so that the vehicle that runs out of time will not receive \( D_{score} \). Equation (8) ensures that each vehicle unit only passes through any intersection at most once. Equation (9) ensures that vehicle units with higher priority pass first while waiting at the intersection. Equation (10) ensures that only one traffic light is green at each intersection at any given time.

\[
\max Z(T) = \sum_{V_{a} \in N_V} \chi \cdot \left[ D_{score} + T_{II} - T_{V_{a}}^D \right] \tag{5}
\]

\[
T_{V_{a}}^D = \int \sum_{d \in \Theta_{V_{a}}} T_{V_{a},d}^r \, dt + \sum_{b \in Z_{V_{a}}} W T_{V_{a},s} + (\Theta_{V_{a}} - 1) \cdot t_{delay} \tag{6}
\]

\[
\chi = \begin{cases} 
0, & T_{V_{a}}^D \geq T_{II} \\
1, & T_{V_{a}}^D < T_{II}
\end{cases} \tag{7}
\]
where \( D_{\text{score}} \) is the reward score, \( T_{\text{II}} \) is the maximum time to ensure the drivers’ driving experience, \( N_V \) is the set of all vehicle units, \( \Theta_{V_{ij}} \) is the set of road sections of vehicle unit \( V_\sigma \) and \( \Xi_{V_\sigma} \) is the set of wating state of vehicle unit \( V_\sigma \).

\[
\begin{align*}
\begin{cases}
n_{V_\sigma}^{(i)} = 1, & \forall i, j \in \Phi_{V_\sigma} \land \varphi_{ij} \in \Psi_{V_\sigma} \\
n_{V_\sigma}^{(i)} = 0, & \forall i, j \in \Phi - \Phi_{V_\sigma} \land \varphi_{ij} \in \Psi - \Psi_{V_\sigma}
\end{cases}
\end{align*}
\]  

(8)

where \( n_{V_\sigma}^{(i)} \) is the number of times vehicle unit \( V_\sigma \) arrives at intersection \( i \), \( \Phi \) is the set of all intersections, \( \varphi_{ij} \) is the road section starting from intersection \( i \) and ending at intersection \( j \), and \( \Psi \) is the set of all road sections.

\[
\omega_{V_\sigma}^{\varphi_{ij}} < \omega_{V_\sigma}^{\varphi_{ij}'} \wedge \forall V_\sigma \leq V_\sigma', \forall V_\sigma, V_\sigma' \in V_{t,\varphi_{ij}}
\]  

(9)

where \( \omega_{V_\sigma}^{\varphi_{ij}} \) is the queuing sequence of vehicle unit \( V_\sigma \) waiting for the green light in road section \( \varphi_{ij} \) when it arrives at intersection \( j \) and \( V_{t,\varphi_{ij}} \) is the set \( V_\sigma \) of vehicle units arriving at intersection \( j \) in road section \( \varphi_{ij} \) in time interval \( t \).

\[
\begin{align*}
\begin{cases}
\sum_{j} \Omega_{\varphi_{ij}} = 1, & \forall \Omega \cap \forall j \in \Phi_{\rightarrow j} \\
\Omega_{\varphi_{ij}} \in \{0, 1\}
\end{cases}
\end{align*}
\]  

(10)

where \( \Omega_{\varphi_{ij}} \) is the identifier of the state of the green traffic light in road section \( \varphi_{ij} \) and \( \Phi_{\rightarrow j} \) is the set of all intersections leading to intersection \( j \).

In this paper, we use a co-factor set to realize the cooperative and coordinated control actions among adjacent edges and offline learning with historical traffic flow data. The co-factor set objectively reflects the congestion of each road section and the potential "Attacking Traffic Flow" (after passing the intersection, the vehicle units will enter the next road section, thus increasing the congestion level and attacking the traffic density of the road section) among road sections. As the experienced set of global road sections, the co-factor set reflects both the past traffic flow information of each intersection and the associated influence among road sections. Thus, we can deepen the degree of cooperation among edge servers in the global control of traffic lights by introducing the co-factor set. It enables edge servers to consider the cooperation among servers in generating local strategies. The joint reward feedback for each edge server will further iterate the co-factor set, so as to take experience and timeliness into account.

As the first step, we calculate the joint reward feedback for each edge server. We take the average waiting time of vehicle units within the coverage area of edge servers as the evaluation index. \( \bar{W}T_\kappa \) is the average waiting time of vehicle units within the coverage area of edge server \( \kappa \) in time interval \( t \), as shown in Equation (11). \( r_{\kappa,\text{local}} \) is the local reward feedback for edge server \( \kappa \), as shown in Equation (12). The adjacent edge servers have traffic flow correlation. Thus, a spatial attenuation factor is introduced so that the joint reward feedback can reflect the gains of the adjacent environment. \( r_{\kappa,\text{joint}} \) is the joint reward feedback for edge server \( \kappa \), as shown in Equation (13). \( \rho_\kappa^t \) is the traffic density within the coverage area of edge server \( \kappa \) in time interval \( t \), as shown in Equation (14).

\[
\bar{W}T_\kappa = \frac{\sum_{N_0^\kappa} N_0^\kappa}{N_0^\kappa}, \forall \kappa \in N_{\text{ES}}
\]  

(11)

where \( N_0^\kappa \) is the set of all vehicle units within the coverage area of edge server \( \kappa \), \( N_0^\kappa \) is the set of all vehicle units within the coverage area of edge server \( \kappa \) and \( N_{\text{ES}} \) is the set of all the edge servers.

\[
r_{\kappa,\text{local}} = \begin{cases} 
1 & \text{if } \bar{W}T_\kappa^t > \bar{W}T_{\kappa-1}^t \\
0 & \text{if } \bar{W}T_\kappa^t = \bar{W}T_{\kappa-1}^t \\
-1 & \text{if } \bar{W}T_\kappa^t < \bar{W}T_{\kappa-1}^t
\end{cases}
\]  

(12)
where $r_{\kappa,\text{joint}} = r_{\kappa,\text{local}} + \sum_{\kappa} \frac{1}{|d_{\kappa \rightarrow \kappa}|} \times r_{\kappa,\text{local}}$ is the spatial attenuation factor and $K_{\kappa}$ is the set of the adjacent edge servers of edge server $\kappa$. Due to the fast attenuation of space, the set of adjacent edge servers only consider a two-layer road network structure.

$$r_{\kappa,\text{local}} = \sum_{\phi \in \Psi_{\kappa}} N_{\kappa,\phi,t}$$

(13)

$\rho_{\kappa,i} = \frac{\sum_{\phi \in \Psi_{\kappa}} N_{\kappa,\phi,t}}{\sum_{\phi \in \Psi_{\kappa}} L_{\kappa,\phi}}$ (14)

where $\Psi_{\kappa}$ is the set of all the road sections within the coverage area of edge server $\kappa$, $N_{\kappa,\phi,t}$ is the number of vehicle units on the road section $\phi$ within the coverage area of edge server $\kappa$ in time interval $t$ and $L_{\kappa,\phi}$ is the length of the road section $\phi$ within the coverage area of edge server $\kappa$.

Then we construct the co-factor set $A$ as a matrix of $K \times K$, as shown in Equation (15).

$$A_{ij} = A_{ij} + \alpha(r_{ij,\text{joint}} + \gamma \rho_{ij,t} + \frac{d_{\kappa,i} \rho_{i,t} - \rho_{j,t}}{d_{ij}}} \forall i, j \in K$$

(16)

where $d_{\kappa,i}$ is the benchmark distance, $|d_{ij}|$ is the distance between edge server $i$ and edge server $j$, and $\alpha$ and $\gamma$ are the attenuation factors.

The swarm intelligence algorithm can be used to solve the multi-objective optimization problem of traffic signal control. In the traditional swarm intelligence algorithm, each iteration of the solution space applies a local search algorithm, which requires a lot of computing resources and reduces the convergence efficiency. Edge servers only have limited computing resources and storage capacities. Thus, we adopt the optimization framework of the chemical reaction optimization algorithm [34] to accelerate the convergence of traffic signal control optimization strategies. We adopt the PSO [35] algorithm with excellent inter-individual coordination and introduce the co-factor set under the CRO framework to realize local cooperative scheduling and global control among traffic lights in the multi-dimensional control problem of large-scale traffic light control. By configuring two necessary attributes, molecular potential energy (PE) and molecular kinetic energy (KE), the algorithm can avoid falling into local optima too early and converge to optima faster. PE represents the stability of the solution space and is defined as the reciprocal of vehicle travel time, as shown in Equation (17). When PE increases, the solution space tends to be stable. Thus, the global exploration will be stopped and local searching will be performed. KE makes the solution space tend to be dynamic. When KE is high, the global exploration will be continued to avoid falling into local optima too early. Therefore, only when KE attenuates to a threshold value and PE tends to be stable, will local searching be carried out, thus greatly improving the convergence efficiency. Thus, the algorithm can be deployed on edge servers with limited computing resources.

$$PE = \frac{1}{\sum_{V \in N_{V}} T_{V}}$$

(17)

The pseudocode of CRO-CPSO is shown in Algorithm 2. The input of Algorithm 2 is road pheromone information and the global co-factor set; its output is the local traffic light regulation strategy. The solution space $S = \{s_1, s_2, \ldots, s_{N_{ES}}\}$ is the green phase sequence of each traffic light at the intersection, where $\{s'_1, s'_2, \ldots, s'_{N_{ES}}\}$ is an array of different...
integers ranging from 1 to $N_{ES}$. In this paper, we use Monomolecular Decomposition and Polymolecular Synthesis to run the global exploration of the solution space. When the solution space tends to be stable, we use Monomolecular Collision and Polymolecular Collision to run the local search in the solution space. Figure 4 shows the updating scheme for the solution space $S'$ of dimension 6.

**Algorithm 2 Pseudocode of CRO-CPSO**

1: InitialSwarm()
2: $m \leftarrow 1$
3: while $m < \text{MaxIter}$ do
4:   for each particle $x_{ij}^m$ do
5:     if $\text{Self - Collision}(S^r)$ then
6:       $\text{status} \leftarrow \text{Monomolecular Collision}(S^r)$
7:     else if $\text{Self - Decomposition}(S^r)$ then
8:       $\text{status} \leftarrow \text{Monomolecular Decomposition}(S^r)$
9:     else if Intergroup – $\text{Collision}(S^r)$ then
10:       $\text{status} \leftarrow \text{Polymolecular Collision}(S^r)$
11:     else if Intergroup – $\text{Synthesis}(S^r)$ then
12:       $\text{status} \leftarrow \text{Polymolecular Synthesis}(S^r)$
13:     end if
14:     $v_{ij}^{m+1} = \text{updateVelocity}\left(\omega, v_{ij}^m, c_1, c_2, E_{ij}^m, Q_{ij}^m, x_{ij}^m\right)$
15:     $x_{ij}^{m+1} = \text{updatePosition}\left(v_{ij}^{m+1}, x_{ij}^m, A_k\right)$
16:     $\text{fitness}(S^r, S'')$
17:     Update $E_{ij}^m$
18:     Update $Q_{ij}^m$
19:   end for
20: end while

When the solution space satisfies the condition of $\text{Self-Collision}$, Monomolecular Collision will occur. As shown in Figure 4a, the solution space is slightly changed. We use the method of Swapping Two-Domain Spaces to ensure that the spatial arrangement of $S'$ is not repetitive. After the Collision, the KE of $S'$ decays and the PE of $S'$ is updated. When the solution space satisfies the condition of $\text{Self-Decomposition}$, Monomolecular Decomposition will occur. As shown in Figure 4b, the solution space is changed considerably. Select a breakpoint for $S$ and divide it into two parts $[\text{first\_half}, \text{second\_half}]$. $S1$ retains $[\text{first\_half}]$ and $S2$ retains $[\text{second\_half}]$. Then, the remaining parts of $S1$ and $S2$ are generated while ensuring that the arrangement of solution spaces is not repetitive. The KE between the solution spaces is redistributed and the PE of the solution spaces is updated.

When the solution space satisfies the condition of $\text{Intergroup-Collision}$, Polymolecular Collision will occur. As shown in Figure 4c, the solution space is slightly changed. In the Polymolecular Collision, we use Conflict-Detection to map the conflicting values so as to ensure the non-repeatability of the arrangement between $S1'$ and $S2'$. In the example of Conflict-Detection in Figure 4c, we can see two sets of mappings $2 \rightarrow 5 \rightarrow 4$ and $3 \rightarrow 1$. Thus, if there are two 2 s in the solution space after the Polymolecular Collision, 2 will be converted to 4, and so on, until there is no conflict. After the Collision, the KE between the solution spaces is redistributed and the PE of the solution spaces is updated. When the solution space satisfies the condition of $\text{Intergroup-Synthesis}$, Polymolecular Synthesis will occur. As shown in Figure 4d, the solution space is changed considerably. Choose a synthesis point for the two solution spaces. Then, the $[\text{first\_half}]$ of $S1$ and the $[\text{second\_half}]$ of $S2$ are combined to produce a new solution space $S$ with great diversity. In the Polymolecular Synthesis, we also use Conflict-Detection to map the conflicting values so as to ensure the non-repeatability of the arrangement of $S$. After the synthesis, the KE of the solution spaces is aggregated, and the PE of the solution space is updated.
The solution space \( S'' = \{s''_1, s''_2, \ldots, s''_{N_{N_{N}}} \} \) is the duration of the green phase of a traffic light at an intersection, where \( \{s''_1, s''_2, \ldots, s''_{N_{N_{N}}} \} \) is an array of different integers ranging from 0 to 60 s. When edge server \( i \) generates the duration of the green phase of the traffic light, we introduce co-factor set \( A \) into the solution space. We focus on the k-neighbor road sections that are directly related to the current intersection \( \Phi_i \) in updating the solution space. The co-factor \( A_{ij} \) reflects the potential “Attacking Traffic Flow” (including the potential traffic flow of other road sections extending from intersection \( \Phi_j \) at intersection \( \Phi_j \) in the direction of road section \( q_{j_{i_{j}}} \)). Thus, when \( A_{ij} \) accounts for a large proportion of \( A_k \), it indicates that the road section \( q_{j_{i_{j}}} \) may become congested. Thus, experience-based coordinated regulation can be implemented to alleviate the potential congestion of road sections. We need to extend the green phase of road section \( q_{j_{i_{j}}} \) in an appropriate proportion within the green cycle. The green phases of other road sections with smaller proportions in \( A_k \) are appropriately compressed. The ratio of extension to compression is determined by the weight ratio. The extension–compression ratio is also affected by the experiential factor due to the empirical lag of the co-factor set, as shown in Equation (18).

We introduce the idea of particle swarm optimization in the iteration of \( S'' = \{s''_1, s''_2, \ldots, s''_{N_{N_{N}}} \} \). Each potential solution to the problem is the position of the particle, and the particles are updated iteratively on a population scale. The particles are initialized before the iteration begins. The fitness of the particles is calculated using Equation (5) as the initial \( E^n_{ij} \) and \( Q^n_{ij} \) of the particles. Then, the circular heuristic search process is initiated: \( v^n_{ij+1} \) and \( x^n_{ij+1} \) (\( N^n_{ij+1} \) for the particle is rounded in the updating process).
for the particle are updated iteratively, and its fitness is calculated. If the fitness is better than \( E_{mij} \), update \( E_{mij} \). If the fitness is better than \( Q_{mij} \), update \( Q_{mij} \). In each iteration update, the particle updates its position according to Equation (18):

\[
x_{mij+1} = \begin{cases} 
\lceil x_{mij} + v_{mij+1} + \beta \cdot \left( \sum X x_{mij} \cdot A_{ij} \sum A_{ik} A_{ij} - x_{mij} \right) \rceil & \text{if } \sum X x_{mij} \cdot A_{ij} \sum A_{ik} A_{ij} \geq x_{mij} \\
\lfloor x_{mij} + v_{mij+1} + \beta \cdot \left( \sum X x_{mij} \cdot A_{ij} \sum A_{ik} A_{ij} - x_{mij} \right) \rfloor & \text{otherwise.}
\end{cases}
\]

(18)

where \( \beta \) is the experiential factor, \( \lceil \rfloor \) represent rounding processing and \( v_{mij+1} \) is the velocity of the particle, as shown in Equation (19):

\[
v_{mij+1} = \omega \cdot v_{mij} + c_1 \cdot U(0, 1) \cdot \left( E_{mij} - x_{mij} \right) + c_2 \cdot U(0, 1) \cdot \left( Q_{mij} - x_{mij} \right)
\]

(19)

where \( E_{mij} \) is the individual extremum, \( Q_{mij} \) is the global extremum, \( c_1 \) and \( c_2 \) are learning factors, \( U(0, 1) \) is a uniform random value in \([0, 1]\) and \( \omega \) is the inertia weight, as shown in Equation (20):

\[
\omega = \omega_{\text{max}} - \frac{(\omega_{\text{max}} - \omega_{\text{min}}) \cdot P}{P_{\text{max}}}
\]

(20)

where \( \omega_{\text{max}} \) is the maximum of the inertia weight, \( \omega_{\text{min}} \) is the minimum of the inertia weight and \( P_{\text{max}} \) is maximum iteration.

### 3. Experimental Setup

This section presents the experimental framework followed to assess the performance of our method. We first describe the specific road network scenario generated for this paper. Then, we present the detailed simulation parameters.

We used MATLAB to achieve the CRO-CPSO algorithm. The simulation phase was carried out by executing the traffic simulator SUMO v1.12.0 ("Microscopic Traffic Simulation using SUMO"; Pablo Alvarez Lopez, Michael Behrisch, Laura Bieker-Walz, Jakob Erdmann, Yun-Pang Flötteröd, Robert Hilbrich, Leonhard Lücken, Johannes Rummel, Peter Wagner, and Evamarie Wießner. IEEE Intelligent Transportation Systems Conference (ITSC), 2018.) for Windows.

SUMO [36] is a well-known traffic simulator. It provides an open source, microscopic, multi-modal traffic simulation environment so as to realize the simulation of traffic signal scheduling of large road network structures. We imported digital maps from OpenStreetMap (OSM) [37]. The digital map was then transformed and combed through to provide a valid SUMO network using the netconvert script provided in the SUMO package.

We generated an actual road network scenario from the real digital map, as shown in Figure 5. The physical location is Changsha County, Changsha City, Hunan Province, China, which has 64 road intersections. We choose this area because Changsha County has a regular road network structure which can represent the general road network structure found in the central urban area of China and because examining a real case has more practical significance in the context of solving road congestion problems. As for traffic density, this paper tested three different density levels (low density: 300 vehicles, medium density: 500 vehicles, high density: 1000 vehicles) to consider the road traffic flow situations in different periods of time.

In the CRO-CPSO algorithm, we set the swarm (population) size to 30 particles and the number of iterations to 100 steps. We set other parameters of the algorithm based on a small area of Changsha County (with 64 traffic lights and 100 vehicles). The detailed simulation parameters are shown in Table 2.
Figure 5. Process of creation of actual road network scenario in the real world. Views of Changsha County (113°6′34.2″ E, 28°15′28.7″ N). After selecting the actual road network scenario (Baidu Map view), we transformed it by the means of the OSM tool and then exported it to SUMO by means of the netconvert script (a script that can convert various third-party road network files into sumo readable files).

(a) Baidu Map view

(b) OpenStreetMap view

(c) SUMO view
Table 2. The detailed simulation parameters.

| Simulation Parameters       | Value              |
|----------------------------|--------------------|
| Simulation area            | 5.14 km²           |
| Number of traffic lights   | 64                 |
| Number of vehicles         | 100/300/500        |
| Vehicle speed              | 0–20 km/h          |
| Reward score (D.score)     | 250                |
| Maximum time (T₁₁)         | 858 s              |
| Benchmark distance (dₑ)    | 1                  |
| Number of iterations (P)   | 30                 |
| Swarm size                 | 100                |
| Attenuation factor (α, γ)  | 0.95               |
| Experiential factor (β)    | 0.9                |
| Learning factors (c₁, c₂)  | 2                  |
| Inertia weight (ω)         | 0.4                |

Additionally, we implemented two algorithms, GA [29] and PSO [35], in order to establish comparisons against our CRO-CPSO algorithm. The GA algorithm is a classical algorithm used in evolutionary computation, which mainly imitates the survival of the fittest in nature to carry out natural evolutions. The PSO algorithm is a classical algorithm used in swarm intelligence, which mainly searches for optimizations through a heuristic search process. GA and PSO have good applications in solving nonlinear programming problems, so we selected them as comparison algorithms.

4. Experiment and Analysis

This section presents the results and analyses from several viewpoints. First, we studied and analyzed the influence of parameters on the performance of the algorithm. Then we conducted comparative experiments. Finally, we analyzed the scalability of the algorithm.

4.1. Performance Analysis of Algorithms

Before carrying out the comparative experiment, we first analyzed the performance of the CRO-CPSO algorithm with different swarm sizes and maximum numbers of iterations through a series of experiments. We set the parameter values in the following experiment by this investigation. Considering a small-scale case of Changsha County (with 64 traffic lights and 100 vehicles), we tried different configuration combinations and plotted the traces of the progress of the best fitness values, as shown in Figure 6. These traces corresponded to the configuration combinations of swarm sizes with 10, 20 and 30 particles and maximum iterations of 50, 100 and 200 steps. The swarm size will affect the diversity of the population and the maximum number of iterations will affect the inertia weight; thus, we mainly studied the parameter values through their different configuration combinations.

As shown in Figure 6, for all configuration combinations of swarm sizes and maximum numbers of iterations, our CRO-CPSO algorithm can converge within the interval of 50 to 100 iterations. The algorithm achieved the best performance results under the configuration combination of a swarm size of 30 particles and a maximum iteration of 100 steps. We found that when the maximum iterations was 50 steps, due to the small number of iterations, even if the swarm size was large, it was easier to converge to the local optimal value (between $7.2 \times 10^4$ and $7.25 \times 10^4$). When the swarm size was 10 particles, the small swarm size limited the evolutionary diversity of the population. Thus, even when the maximum iterations was 200 steps, the population could only converge to a low local optimal value ($7.37 \times 10^4$). Therefore, when choosing a configuration combination, one needs to consider both computational cost and optimal fitness value. We found that when the swarm size was 30 particles and the maximum iterations was 100 steps, the population could converge to an approximately optimal value. Although the population achieved a higher fitness value under the configuration combination of a swarm size with 30 particles and maximum
iterations of 200 steps, the small increase in fitness value \((0.015 \times 10^4)\) was not matched by the expensive computation cost (3000 function evaluations). Therefore, we opted to set the swarm size as 30 particles and the maximum iterations as 100 steps in our experimentation.

**Figure 6.** Traces of progress track of best fitness values of CRO-CPSO in the case of Changsha County. SS = swarm size, MI = maximum iteration.

In Figure 7, we plotted the fitness distribution of the whole population in the optimization process of the CRO-CPSO algorithm. Specifically, the plot mainly illustrates the operation of the algorithm in the case of Changsha County (with 64 traffic lights and 300 vehicles). We can see that in the early stages the particles were diverse and with low and fluctuating fitness regions (between 1 and 3). Then they gradually converged to a higher fitness with constant iterations. Thus, all the particles showed ideal convergence and robustness in spite of the differences among them. In this optimization process, 277 vehicles out of 300 arrived at the destination within maximum time \(T_{II}\) (92.3% of the vehicles have high-quality driving experience), as shown in Figure 8. We found that under the baseline control strategy, 37 vehicles did not have high-quality driving experience (i.e., they were unable to reach the destination within \(T_{II}\)), and the journey time of all vehicles was generally high. CRO-CPSO introduces the co-factor set; thus, the influence of potential traffic flow on road conditions is considered. The collaborative effect will not only ensure the driving experience of vehicles but also appropriately extend the waiting time in unobstructed road sections and shorten the waiting time in crowded road sections so as to achieve the effect of alleviating traffic congestion. We introduced the reward score \(D_{score}\) and \(T_{II}\) as the incentive mechanism in the calculation of the fitness function so that the convergence of the algorithm takes into account the driving experience of vehicles. It can be seen from Figure 8 that under the CRO-CPSO control strategy, the overall driving time of 300 vehicles is reduced by 11,937 s (about 199 min) compared with the baseline control strategy, thus reflecting the effectiveness of this algorithm in improving the traffic efficiency of the urban road network. We found that due to traffic light control, the driving times of a few vehicles with low driving times increased slightly (i.e., waiting time in unobstructed road sections was extended), and the driving times of some vehicles with high driving times was sharply reduced (i.e., waiting time in crowded road sections was shortened), which verifies the coordinated control of the co-factor set for each intersection of the urban road network. We found that the number of vehicles with low-quality driving experience decreased by 14 (i.e., a 37.8% reduction), and no single vehicle had too excessive a driving time (the longest vehicle driving time under the baseline strategy was also reduced from 1698 s to 1384 s after optimization). As each iteration of the algorithm will take the driving experience of the vehicles as the incentive factor, it can well avoid the extreme phenomenon of shortening
the waiting time of the majority of vehicles by seriously affecting the driving experience of a very few vehicles. Thus, the CRO-CPSO algorithm takes into account the driving experience of almost all vehicles while ensuring the optimum efficiency.

Figure 7. Absolute frequency histogram of swarm fitness.

Figure 8. Comparison performance diagram of the driving time of 300 vehicles after CRO-CPSO strategy optimization (the green-dotted line is $T_1$).

4.2. Comparative Experimental Analysis

In this section, we present a test and comparison of three algorithms, our proposed CRO-CPSO, GA and PSO, in the case of Changsha County. Compared with GA, we expected to prove the advantages of using particle swarm optimization (PSO) over evolutionary algorithms in urban traffic control problems. By comparison with the traditional PSO algorithm, we expected to prove that the introduction of prior knowledge (short-term traffic prediction and co-factor set) in the swarm intelligence algorithm could improve the performance of road cooperative regulation. In this case, we set three traffic flow densities...
(100 vehicles, 300 vehicles and 500 vehicles) to simulate different congestion conditions on the urban road network.

Table 3 contains the best fitness values for CRO-CPSO, GA and PSO for different traffic flow densities in Changsha County. We found that with the increase in traffic flow density, the best fitness values of the three algorithms all showed a nonlinear increase in different proportions. Through an analysis of the traffic flow data set, we found that although the number of vehicles increased proportionally, the track of each vehicle was randomly generated, which affected the waiting time of each vehicle. The number of vehicles increased greatly; thus, the best fitness values for the algorithm showed an overall upward trend. We found that the best fitness values for CRO-CPSO were better than those for GA and PSO under different traffic flow densities. We also found an interesting feature. The best fitness values for GA were better than those for PSO at low traffic flow densities (100 vehicles, 300 vehicles). However, when the urban road network entered a high-congestion state (500 vehicles), the best fitness value for PSO exceeded that for GA. Therefore, the PSO algorithm is more suitable than the GA algorithm when dealing with the congestion typical of a complex urban road network. We also found that with the increase in traffic flow density, the performance improvements of CRO-CPSO with regard to traffic control were 2.70% (100 vehicles), 4.10% (300 vehicles) and 13.03% (500 vehicles), respectively. CRO-CPSO introduces co-factor sets and makes collaborative decisions based on fuzzy logic. When congestion occurs in urban road networks, the coordinated regulation can evacuate potentially congested road sections in a targeted way (that is, the green time of traffic lights can be appropriately extended/compressed in a coordinated way). Therefore, when regulating urban road networks with high congestion, our proposed CRO-CPSO shows a better performance than the other algorithms.

### Table 3. Best Fitness Values Obtained by CRO-CPSO, GA, and PSO.

| Instance       | Number of Vehicles | Method        |
|----------------|--------------------|---------------|
| Changsha County| 100                | 7.05 x 10³    | 6.95 x 10³ | 6.86 x 10³ |
|                | 300                | 2.20 x 10³    | 2.14 x 10³ | 2.11 x 10³ |
|                | 500                | 3.76 x 10³    | 3.27 x 10³ | 3.35 x 10³ |

The bold is the best fitness value of our CRO-CPSO.

In Figure 9, we can see the distribution of the journey times of vehicles under the control of the three algorithms with different traffic flow densities on the urban road network. We found that under the three traffic flow densities, the median journey time of vehicles regulated by CRO-CPSO was the lowest, which reflects the excellent performance of our proposed CRO-CPSO in improving the traffic efficiency of the urban road network. The box height of CRO-CPSO was the lowest, which reflects the low fluctuation in the journeys of vehicles. As the traffic flow density reached 500 vehicles, congestion began to occur on the road network (both the maximums for the journey times of vehicles and the numbers of outliers in GA and PSO increase greatly). However, CRO-CPSO can still significantly reduce the journey time of vehicles (25.34%) while maintaining fewer outliers. Therefore, our proposed CRO-CPSO gives full play to the synergy, which not only improves the traffic efficiency of urban road networks, but also fully considers the driving experience of vehicles during road network regulation.
4.3. Scalability Analysis

This section is mainly concerned with the scalability of our proposed CRO-CPSO. We focus on the scale of the road network structure and the scalability of CRO-CPSO synergies in a large-scale road network. In this analysis, there were 2577 intersections and 3497 sections in the simulation of a large-scale urban road network. As shown in Figure 10, when the road network scale is greatly expanded, CRO-CPSO can still achieve good convergence and excellent regulation performance. We found that in the last two iterations, the best fitness value was greatly improved. Through analysis, we found that only 3.50% of the vehicles did not arrive at their destinations on time (failing to obtain $D_{\text{score}}$). That is, the regulation strategy optimized by CRO-CPSO was almost able to guarantee the driving experience of all vehicles, which result was close to the optimal regulation strategy. Thus, our proposed CRO-CPSO has very good scalability and can well adapt to super-large-scale road network structures.
5. Conclusions

In this study, we designed a traffic signal regulation model based on edge computing for traffic congestion in large urban road networks. We used a traffic flow prediction system based on fuzzy logic to predict short-term real-time vehicle waiting times. The co-factor set was generated by offline learning, and the potential influence of historical traffic flow in adjacent sections was fully considered. We proposed the CRO-CPSO algorithm to effectively adapt to large-scale road network structures and complex traffic conditions. We used SUMO, a well-known microscopic traffic simulator, to evaluate our solution. We tested CRO-CPSO in a modern urban road network scenario of Changsha County and compared it with the classical algorithms GA and PSO. A series of analyses were carried out from different viewpoints, from which the following conclusions can be extracted. Our CRO-CPSO algorithm performs successfully in the generation of optimized scheduling strategies for large-scale realistic traffic scenarios. For all the instances, our proposal obtained robust results which were better than those of the other two algorithms compared: the GA and PSO algorithms. Our suggested algorithm can enhance driving experience and shorten journey times. The performance of CRO-CPSO improved by 2.70% (100 vehicles), 4.10% (300 vehicles) and 13.03% (500 vehicles) with different traffic flow densities. CRO-CPSO can also shorten the journey times of vehicles by 25.34% under conditions of high road congestion. All of this means real improvement in city traffic conditions. Additionally, CRO-CPSO has good scalability. When the scenario was extended to 2577 intersections and 3497 sections, the scheduling strategy could still ensure the driving experience of all drivers and was close to the optimal solution for scheduling. Thus, our method can well adapt to different road network sizes and traffic densities.

In future work, we will further improve the training parameters of the co-factor set to expand the coordination range for road scheduling. We will further optimize the rule setting according to fuzzy logic. We will also consider introducing machine learning methods to predict traffic flow in real time and consider pedestrians and other multi-objective factors in traffic control. A future study will also focus on the emergency treatment of traffic emergency faults so as to further fit real urban road scenarios.

Author Contributions: Conceptualization, C.Y. and J.C.; Formal analysis, G.X.; Methodology, C.Y.; Software, C.Y.; Validation, C.Y. and G.X. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Institutional Review Board Statement: Not applicable.

Informed Consent Statement: Not applicable.

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

Nomenclature

| Symbol | Description |
|--------|-------------|
| $V_\sigma$ | Vehicle unit |
| $v_\sigma$ | Vector of real-time spatial average speed of $k$ samples within a time interval $t$ |
| $v_m^V$ | Vector of instantaneous speed of the $m$th sample of vehicle unit $V_\sigma$, $m \in [1, k]$ |
| $L_{V_\sigma}$ | Length of road section of vehicle unit $V_\sigma$ in time interval $t$ |
| $T_{V_\sigma}$ | Experienced travel time of vehicle unit $V_\sigma$ in time interval $t$ |
| VRA | Abbreviation of the current road name of the vehicle units |
| VST | Abbreviation of the time of vehicle unit switching state |
| VRN | Abbreviation of the number of remaining roads of the vehicle unit |
| VQL | Abbreviation of the queue length of vehicle units |
| IWJ | Abbreviation of the judgement of whether to wait at the intersection |
| GPD | Abbreviation of the duration of the green phase |
| VP | Abbreviation of the priority of vehicle units |
| WT | Abbreviation of the waiting time of the vehicle units |
Intersection delay caused by traffic signal control

\(N_V\) Set of all vehicle units, \(\forall V_\sigma \in N_V\)

\(D_{score}\) Reward score

\(T_t\) The maximum time to ensure the drivers’ driving experience

\(\Theta_{N_\sigma}\) Set of road sections of vehicle unit \(V_\sigma\)

\(\Xi_{N_\sigma}\) Set of waiting state of vehicle unit \(V_\sigma\)

\(\chi_{(i)}\) Decision factor for the reward score

\(n_{V_\sigma}\) Number of times vehicle unit \(V_\sigma\) arrives at intersection \(i\)

\(\Phi\) Set of all intersections

\(\phi_{ij}\) Vector of the road section starting from intersection \(i\) and ending at intersection \(j\)

\(\Psi\) Set of all road sections

\(\omega_{\phi_{ij}}\) The queuing sequence of vehicle unit \(V_\sigma\) waiting for the green light in road section \(\phi_{ij}\)

\(\Omega_{\phi_{ij}}\) The identifier of the state of the green traffic light in road section \(\phi_{ij}\)

\(\Phi_{\sigma\rightarrow j}\) Set of all intersections leading to intersection \(j\)

\(\mathcal{WT}_\kappa\) The average waiting time of vehicle units within the coverage area of edge server \(\kappa\) in time interval \(t\)

\(N_{\kappa}\) Set of all the vehicle units within the coverage area of edge server \(\kappa\)

\(N_{\kappa\mathcal{WT}}\) Set of WT of the vehicle units within the coverage area of edge server \(\kappa\)

\(N_{\kappa\mathcal{ES}}\) Set of all the edge servers, \(\forall \kappa \in N_{\kappa\mathcal{ES}}\)

\(\kappa\) Edge server

\(r_{\kappa,\text{local}}\) Local reward feedback for edge server \(\kappa\)

\(r_{\kappa,\text{joint}}\) Joint reward feedback for edge server \(\kappa\)

\(d_{\kappa\rightarrow\kappa}\) Spatial attenuation factor

\(K_{\kappa}\) Set of the adjacent edge servers of edge server \(\kappa\)

\(\rho_{\kappa}\) The traffic density within the coverage area of edge server \(\kappa\) in time interval \(t\)

\(\Phi_{\kappa}\) Set of all the road sections within the coverage area of edge server \(\kappa\)

\(N_{\phi,t}\) Number of vehicle units on the road section \(\phi\) within the coverage area of edge server \(\kappa\) in time interval \(t\)

\(L_{\kappa}\) Length of the road section \(\phi\) within the coverage area of edge server \(\kappa\)

\(A\) Matrix of Co-factor set

\(d_{e}\) Benchmark distance

\(|d_{ij}|\) Distance between edge server \(i\) and edge server \(j\)

\(\alpha, \gamma\) Attenuation factors

\(\beta\) Experiential factor

\(v_{ij}^{m+1}\) Vector of the velocity of the particle

\(E_{ij}^{m}\) Individual extremum

\(Q_{ij}^{m}\) Global extremum

\(c_1, c_2\) Learning factors

\(\omega\) Inertia weight

\(\omega_{\text{max}}\) Maximum of inertia weight

\(\omega_{\text{min}}\) Minimum of inertia weight

\(P_{\text{max}}\) Maximum iteration

References

1. Hewton, J.T. The metropoliton toronto traffic control signal system. Proc. IEEE 1968, 56, 577–599. [CrossRef]

2. Gazis, D.C. Traffic control: From hand signals to computers. Proc. IEEE 1971, 59, 1090–1099. [CrossRef]

3. Inose, H. Road-Traffic Control with the Particular Reference to Tokyo Traffic Control and Surveillance System. Proc. IEEE 1976, 64, 1028–1039. [CrossRef]

4. Diakaki, P.; Kotsialos, D.; Wang, Y. Review of road traffic control strategies. Proc. IEEE 2004, 91, 2041–2042. [CrossRef]

5. Ganin, A.A.; Mersky, A.C.; Jin, A.S.; Kitsak, M.; Keisler, J.M.; Linkov, I. Resilience in Intelligent Transportation Systems (ITS). Transp. Res. Part C Emerg. Technol. 2019, 100, 318–329. [CrossRef]

6. Wang, T.; Cao, J.; Hussain, A. Adaptive Traffic Signal Control for large-scale scenario with Cooperative Group-based Multi-agent reinforcement learning. Transp. Res. Part C Emerg. Technol. 2021, 125, 103046. [CrossRef]

7. Wu, Q.; Wu, J.; Shen, J.; Yang, B.; Zhou, Q. An Edge Based Multi-Agent Auto Communication Method for Traffic Light Control. Sensors 2020, 20, 4291. [CrossRef]
8. Shao, M.L.; Cao, E.; Hu, M.; Zhang, Y.; Chen, W.J.; Chen, M.S. Traffic light optimization control method for priority vehicle awareness. *Ruian Ian Xue Bao*, Softw. 2021, 32, 2425–2438. Available online: http://www.jos.org.cn/1000-9825/6191.htm (accessed on 7 February 2021). (In Chinese)

9. Yang, F.; Wang, S.; Li, J.; Liu, Z.; Sun, Q. An overview of Internet of Vehicles. *China Commun.* 2014, 11, 1–15. [CrossRef]

10. Shi, W.; Sun, H.; Cao, J.; Zhang, Q.; Liu, W. Edge computing- An emerging computing model for the Internet of everything era. *J. Comput. Res. Dev.* 2017, 54, 907–924. (In Chinese)

11. Liu, F.; Tang, G.; Li, Y.; Cai, Z.; Zhang, X.; Zhou, T. A Survey on Edge Computing Systems and Tools. *Proc. IEEE 2019*, 107, 1537–1562. [CrossRef]

12. Tang, C.; Wei, X.; Zhu, C.; Wang, Y.; Jia, W. Mobile Vehicles As Fog Nodes For Latency Optimization In Smart Cities. *IEEE Trans. Veh. Technol.* 2020, 69, 9364–9375. [CrossRef]

13. Letaief, K.B.; Chen, W.; Shi, Y.; Zhang, J.; Zhang, Y.J.A. The Roadmap to 6G—AI Empowered Wireless Networks. *IEEE Commun. Mag.* 2019, 57, 84–90. [CrossRef]

14. Tang, F.; Kawamoto, Y.; Kato, N.; Liu, J. Future Intelligent and Secure Vehicular Network Toward 6G: Machine-Learning Approaches. *Proc. IEEE 2020*, 108, 292–307. [CrossRef]

15. Mu, S.; Xiong, Z.; Tian, Y. Intelligent Traffic Control System Based on Cloud Computing and Big Data Mining. *IEEE Trans. Ind. Inform.* 2019, 15, 6583–6592.

16. Yang, S.R.; Su, Y.J.; Chang, Y.Y.; Hung, H.N. Short-Term Traffic Prediction for Edge Computing-Enhanced Autonomous and Connected Cars. *IEEE Trans. Veh. Technol.* 2019, 68, 3140–3153. [CrossRef]

17. Zhang, J.; Zheng, Y.; Sun, J.; Qi, D. Flow Prediction in Spatio-Temporal Networks Based on Multitask Deep Learning. *IEEE Trans. Knowl. Data Eng.* 2019, 32, 468–478. [CrossRef]

18. Comert, G.; Khan, Z.; Rahman, M.; Chowdhury, M. Grey Models for Short-Term Queue Length Predictions for Adaptive Traffic Signal Control. *Expert Syst. Appl.* 2019, 185, 115618. [CrossRef]

19. Song, X.; Guo, Y.; Li, N.; Zhang, L. Online Traffic Flow Prediction for Edge Computing-Enhanced Autonomous and Connected Vehicles. *IEEE Trans. Veh. Technol.* 2021, 70, 2101–2111. [CrossRef]

20. Wu, L.B.; Nie, L.; Liu, B.Y.; Wu, N.; Zou, Y.F.; Ye, L.Y. An Intelligent Traffic Signal Control Method in VANET. *Jisuanji Xuebao/Chin. J. Comput.* 2016, 6, 1105–1119. Available online: http://cjc.ict.ac.cn/qwjs/view.asp?id=4662 (accessed on 15 January 2016). (In Chinese)

21. Yan, G.; Qin, Q. The Application of Edge Computing Technology in the Collaborative Optimization of Intelligent Transportation System Based on Information Physical Fusion. *IEEE Access 2020*, 8, 153264–153272. [CrossRef]

22. Moubayed, A.; Shami, A.; Heidari, P.; Larabi, A.; Brunner, R. Edge-enabled V2X Service Placement for Intelligent Transportation Systems. *IEEE Trans. Mob. Comput.* 2020, 20, 1380–1392. [CrossRef]

23. Chen, H.; Qiu, T.Z. Distributed Dynamic Route Guidance and Signal Control for Mobile Edge Computing-Enhanced Connected Vehicle Environment. *IEEE Trans. Intell. Transp. Syst.* 2021, 1–12. [CrossRef]

24. Zadeh, L.A. Fuzzy sets. *Inf. Control.* 1965, 8, 338–353. [CrossRef]

25. Chou, C.H.; Teng, C. A fuzzy logic controller for traffic junction signals. *Inf. Sci.* 2002, 143, 73–97. [CrossRef]

26. Eberhart, R.; Kennedy, J. Particle swarm optimization. In *Proceedings of the IEEE International Conference on Neural Networks*, Perth, WA, Australia, 27 November–1 December 1995.

27. Lopez, P.A.; Behrisch, M.; Bieker-Walz, L.; Erdmann, J.; Flöteröd, Y.; Hilbrich, R. Microscopic traffic simulation using sumo. In Proceedings of the 2018 21st international conference on intelligent transportation systems (ITSC), Maui, HI, USA, 4–7 November 2018; pp. 2575–2582.

28. Haklay, M.; Weber, P. OpenStreetMap: User-generated street maps. *IEEE Pervasive Comput.* 2008, 7, 12–18. [CrossRef]