IEEE Access
Multidisciplinary | Rapid Review | Open Access Journal

Received 28 December 2022, accepted 8 January 2023, date of publication 16 January 2023, date of current version 24 January 2023.

Digital Object Identifier 10.1109/ACCESS.2023.3237420

RESEARCH ARTICLE

Deep Reinforcement Learning for the Co-Optimization of Vehicular Flow Direction Design and Signal Control Policy for a Road Network

XIANGXUE ZHAO[1], DOMINIC FLOCCO[2], SHAPOUR AZARM[1], AND BALAKUMAR BALACHANDRAN[1], (Senior Member, IEEE)

[1]Department of Mechanical Engineering, University of Maryland, College Park, MD 20740, USA
[2]Department of Mathematics, University of Maryland, College Park, MD 20740, USA

Corresponding author: Dominic Flocco (dflocco@umd.edu)

This work was supported in part by a generous gift from Dr. Alex Mehr, who completed his Ph.D. at the University of Maryland, in 2003, through the Design Decision Support Laboratory Research and Education Fund.

ABSTRACT Reinforcement Learning (RL) is a popular approach for deciding on an optimum traffic signal control policy to alleviate congestion in a road network. However, the traffic signal control policy can also be optimized in conjunction with the design of vehicular flow directions to further improve traffic performance. The design of vehicular flow directions refers to the right of way or directional restriction imposed in a road network. Here, a new RL-based technique is presented for co-optimization of the design of vehicular flow directions and control policy for traffic signals. This technique consists of a two-step iterative process, wherein a set of vehicular flow directions for a road network is generated, then a RL-based approach is used to train the traffic signal control policy over the given set of vehicular flow directions. Following the proposed technique, the vehicular flow directions with poor traffic performance are iteratively eliminated, while new vehicular flow directions are generated to achieve better traffic performance and realize convergence to a maximum possible expected traffic performance. The proposed RL-based technique is evaluated by using two examples under rush hour and non-rush hour traffic conditions. It is found that, compared to a RL-based approach in which only traffic signal control policy is considered, the proposed approach can be used to obtain a better traffic performance in terms of vehicular queue length and throughput.

INDEX TERMS Co-optimization, reinforcement learning, vehicular flow direction design, traffic signal control, deep neural networks.

I. INTRODUCTION

Increasing population growth and corresponding vehicle ownership have resulted in heightened traffic congestion, presenting significant challenges for transportation authorities. Traffic congestion can lead to vehicular queueing, travel delays, fuel consumption, economic loss, and pollution, particularly in urban areas, placing a newfound emphasis on the importance of intelligent urban planning [2]. Two key factors that contribute to traffic congestion in road networks are (i) design of vehicular flow directions and (ii) the control policy used for traffic signals. Vehicular flow direction design refers to the right of way directions for roads in a network [3]. For example, urban road networks are composed of a combination of roads with one- or two-way vehicular flow directions with turning restrictions at their intersections [4], [5]. With vehicular flow design, one specifies whether the road is restricted to a one-way or two-way direction, while the traffic signal control policy dictates traffic signal patterns for non-conflicting directions at intersections, controlling

The associate editor coordinating the review of this manuscript and approving it for publication was Daniel Augusto Ribeiro Chaves.

the vehicular flow between connected roads. Traffic signal control policy can be adjusted according to real-time traffic information, including data collected from GPS-equipped vehicles, navigation systems, and sensors [6].

Traffic performance in congested areas is significantly impacted by vehicular flow design and control of traffic signals, whose effects are inherently coupled together. Different vehicular flow directions can result in distinct traffic signal control policies and vice versa. Due to their interdependence, the authors propose the simultaneous consideration of the design of vehicular flow directions and traffic signal control policies in order to optimize traffic performance in urban road networks.

In the literature, approaches to traffic signal control may be broken down into two broad categories: model-based optimization and model-free Reinforcement Learning (RL) techniques. Cong et al. [7] propose four model-based optimization approaches to jointly find an optimal road network topology and traffic signal controller policy. However, model-based techniques often rely on strong assumptions and specific traffic control models, making the generalization of this approach difficult. On the other hand, model-free RL techniques [8], [9] have two key benefits. First, they require less restrictive assumptions and optimal control policies can be learned from available data, such as GPS-equipped vehicles and loop detector sensors, lending the approach to more general problem instances. Within the RL framework, optimal performance can be achieved through constant back-and-forth interaction between an agent and the traffic environment. Additionally, function approximation techniques can be used to improve computational efficiency in high-dimensional state-action spaces [10]. The efficiency of model-free RL approaches is especially important in traffic optimization problems, as the number of variables grow exponentially with the size of the network.

RL-based techniques have been successfully used for solving traffic signal control problems [8], [9]. Such approaches dominate much of the literature and differ in their characterizations of the state-action space and reward function, and the RL algorithm variations employed. Conventionally, the state space is defined in terms of real-time traffic information such as vehicular flow rate [8], [11], queue length [12], [13], and average delay time [14], [15]. Recent efforts incorporate image-like features [15], [16] to provide a more comprehensive description of the traffic conditions, with positions of vehicles along the lanes represented in the form of a binary matrix. The action space is traditionally defined as a set of traffic phase patterns. A traffic signal phase is used to specify the timing of the permission (green light) or restriction (red light) of vehicular flow directions. Generally speaking, the action space may be (i) step-based [11], [17], in which the traffic controller decides whether to switch or stay in a traffic phase for a pre-determined time duration, or (ii) phase-based [8], [18], [19], wherein the controller is used to decide the time duration for each traffic phase. The reward function, which is a measure of traffic performance, is typically defined as a weighted sum of several metrics, such as travel time [11], [20], queue length [13], [14] and vehicular throughput. Additionally, various RL algorithms have been investigated for learning optimal traffic signal control policies, namely Deep Q-learning Network (DQN) [10], [14], [21], policy gradient method [22], and actor-critic method [23]. Problem instances for evaluation also vary, spanning single intersection control [14], multi-intersection control [24], [25], and even massive-scale scenarios, such as the road network of Manhattan with 3,971 traffic signals [9]. Even though significant progress has been made within this domain, popular techniques only focus on the optimization of the traffic signal controller assuming a pre-determined or fixed design of vehicular flow directions.

The task of designing vehicular flow directions is commonly viewed as an endeavor disconnected from traffic signal control policy optimization. Nonetheless, the urban transportation network design problem is a popular topic in the literature, and is concerned with building new streets, expanding existing road capacity, designing public transportation networks (i.e., bus networks) and restricting turning directions based on current traffic scenarios [26]. The scope of road network design problems may be classified into three broad categories: strategic decisions, tactical decisions, and operational decisions. These resolutions range from long-term decisions that relate to the design of new infrastructures [27] to short-term decisions that improve the flow of traffic in preexisting networks [3]. The proposed approach has applications to both classes of road design problems, as it may be applied to find optimal vehicular flow design patterns and a corresponding optimal control policy for new networks or extensions to existing networks; furthermore, the proposed co-optimization framework can be applied to make real-time traffic flow modifications such as imposing turning restrictions based on traffic demand [28], [29]. The problem of simultaneously optimizing traffic control and vehicular flow design in such application cases is one of the main motivations of this work.

Another key motivation behind this paper is that the traditional RL-based approaches, which are formulated as standard Markov Decision Processes (MDP) [30], do not consider vehicle flow direction design during the learning procedure. To resolve this issue, the authors couple the design and control optimization and employ a bi-level technique; one in which, first generates a set of candidate vehicular flow directions for the road network, and then optimizes the traffic signal controller for each design. In this way, the authors are able to co-optimize the design and control while capturing their interdependent relationship.

The contributions made in this paper are as follows:

1) For the first time in the literature, an integrated approach to the traffic optimization problem is considered in this paper. The authors integrate the design of vehicular flow directions into the RL-based traffic signal control problem. This new extension helps

simultaneously perform a heuristic search of the design space while optimizing the control policy, which has not been considered in the conventional MDP-based approaches. In this way, the proposed bi-level optimization approach is a new extension of a standard MDP [31], [32]. Furthermore, the proposed approach is data-driven and requires less assumptions than those made in conventional model-based techniques, for example, [7].

2) The incorporation of two new modeling schema. First, a directed graph is used to model the road network, which permits the exhaustive generation of feasible vehicular flow designs [33]. Second, conventional Deep Q-Learning is extended to approximate traffic signal control performance for different vehicular flow direction designs and combined with a decaying random search strategy to explore the design space. In this way, the proposed DQN approach can be used to explore a diversified set of feasible design alternatives, and eventually converge to the best combination of the vehicular flow direction design and traffic signal control policy.
3) The merits and versatility of the proposed approach is illustrated by applying the obtained solution to two road network topologies, namely 4- and 12-intersection grid networks. Furthermore, the application allows for the specification of designed roads and incorporates a signal synchronization scheme, extending the applicability of the model. Through this application, the authors illustrate that the co-optimization of vehicular flow design and traffic signal control can scale up to larger networks and, as evidenced in the applications, outperform conventional RL-based traffic signal control only approach.

The rest of the paper is organized as follows. In Section II, the authors present an overview of MDP and DQN to establish the necessary background. The proposed RL-based approach is outlined in Section III, and includes the problem definition, a detailed description of the algorithm and the accompanying deep neural network, an overview of parameter selection procedures, and an outline of the model assumptions. Following that, in Section IV, the performance of the proposed technique is demonstrated by using two examples. Finally, some concluding remarks are offered in the last section, Section V.

II. BACKGROUND

To establish the necessary background, the authors offer a brief review of Markov Decision Process (MDP) [30], [34] and Deep Q-learning Network (DQN) concepts [35], [36], [37], [38].

A. MARKOV DECISION PROCESS (MDP)

MDPs are formally defined by the tuple $\{S, A, T, R, \gamma\}$, with finite environment state space S, action space A, transition function $T : S \times A \times S \rightarrow [0, 1]$, reward function $R : S \times A \times S \rightarrow \mathbb{R}$ and discount factor γ. Given an MDP, an agent observes the state of the environment $s_t \in S$ at timestep t and takes an action $a_t \in A$ according to a control policy $\pi : S \times A \rightarrow [0, 1]$. The control policy $\pi(a_t|s_t)$ returns the probability of taking an action a_t given state s_t. The transition function $T(s_t, a_t, s_{t+1}) = P(s_{t+1}|s_t, a_t)$ dictates the transition from one state to the next at timestep t by considering immediate reward $r_t = R(s_t, a_t, s_{t+1})$. The objective of the agent is to learn a control policy π^* that maximizes the cumulative discounted reward at timestep t

$$R_t = \sum_{i=0}^{T-1} \gamma^i r_{t+i}, \tag{1}$$

where $\gamma \in [0, 1)$ is a discount factor that weights the effect of immediate and future rewards, and T is the terminal timestep. With MDP algorithms, one learns an optimal control policy based on a Q-value function, which quantifies the estimated expected reward for the agents action in a particular state. The Q-value function is given by

$$Q^\pi(s, a) = \mathbb{E}[R_t \mid s, a, \pi] \tag{2}$$

where $\mathbb{E}$ is the expected value operator. The function computes an expected reward starting from state s_t, taking an action a_t, and thereafter following policy π. The agent then chooses the action that yields the maximum Q-value at each timestep. Thus, the optimal Q-function, $Q^*(s, a) = \max_\pi Q^\pi(s, a) = \max_\pi \mathbb{E}[R_t \mid s, a, \pi]$ provides the optimum policy π^* by selecting the action a which maximizes the Q-value for the state s according to

$$\pi^*(s) = \arg\max_a Q^*(s, a), \quad \forall s \in S. \tag{3}$$

Through the lens of dynamic programming, the Q-value for state-action pairs is learned via the Bellman equations [34]

$$Q^\pi(s_t, a_t) = \mathbb{E}\left[r_t + \gamma Q^\pi(s_{t+1}, \pi(s_{t+1}))\right]. \tag{4}$$

In practice, the estimates for Q^π are updated with a learning rate α as

$$Q^\pi(s_t, a_t) \leftarrow Q^\pi(s_t, a_t) + \alpha(y_t - Q^\pi(s_t, a_t)), \tag{5}$$

where y_t is the Temporal Difference (TD) target, which is used to specify the target reward value based on previous estimates. As an off-policy model, actions of the agent in Q-learning are updated by maximizing Q-values over the action using a greedy approach.

B. DEEP Q-LEARNING

Deep Q-learning is an improved version of classical reinforcement learning wherein the Q-value function is approximated with a deep neural network, or Deep Q-Learning Network (DQN) [35]. Deep RL algorithms lend themselves to high-dimensional and complex systems, for which standard RL approaches are ineffective at learning necessary features for function approximation. Within the DQN framework, the

Q-value function is approximated as $Q(s, a; \theta) \approx Q^*(s, a)$, where θ are the neural network hyperparameters [37]. Conventional DQNs take in the state of the environment as input, estimate the Q-value by using a fully connected neural network architecture to train the network, and then output an action according to (3).

While many modifications to traditional Deep RL algorithms exist, two novel techniques introduced by Minh et al. [36] have been shown to significantly stabilize learning in DQNs: experience replay and target network. Experience replay is used to update the Q-network based on past experiences, so as to mitigate the potential of harmful correlations leading to diverging action values. At each timestep, the DQN agent interacts with the environment, obtains data (s_t, a_t, r_t, s_{t+1}) and stores the data in memory store $\mathcal{D}$. During training, mini-batches are sampled uniformly from $\mathcal{D}$, and the Q-network is updated according to a loss function L

$$L(\theta) = \mathbb{E}_{(s,a,r,s')\sim U(\mathcal{D})}\Big[\big(r + \gamma \max_{a'} Q(s', a'; \theta^-) - Q(s, a; \theta)\big)^2\Big], \tag{6}$$

where θ^- represents the parameters of the target neural network and $\mathbb{E}$ is the expected value over the uniform distribution $U(\mathcal{D})$ of replay memory $\mathcal{D}$. A target network is further employed to stabilize learning by designating two separate networks in the DQN: the main network that approximates the Q-function and the target network that computes the Temporal Difference (TD) target update for the main network. While the main network parameters θ are updated during each iteration in training, the target network parameters θ^- are updated only after a user-specified number of timesteps. The use of a target network allows for the utilization of a Double Dueling DQN (DDQN), which extends the standard Q-learning algorithm with a single estimator to one with two estimators [38].

III. PROPOSED REINFORCEMENT LEARNING APPROACH

In the following section, the authors outline the co-optimization approach to traffic design and signal control. First, a description of the RL environment is offered by formally defining the state-action space, and reward function. Next, the authors propose a new extension of a MDP, which formulates the co-optimization problem at hand in an RL setting. Then, a directed graph model is presented, which is used to determine the feasibility of vehicular flow directions. Finally, the co-optimization framework and corresponding assumptions are discussed.

A. ENVIRONMENT DEFINITION

The problem objective is to co-optimize the design of vehicular flow directions and control of traffic signals at one or multiple intersections such that the overall traffic performance in the road network is maximized. For traffic performance, one considers the number of vehicles that safely passing through intersections and the congestion of vehicles in the road network. To solve this traffic optimization problem, the authors use a RL framework that is defined by the state space, the action space, and a reward function. The state space S represents the traffic state in the road network and the action space A is a set of actions the agent may take at each time step. The reward function R, which drives the learning process of the agent, quantifies the current traffic state. The goal of the RL agent is to learn a policy that maximizes traffic performance by observing the current state of the system and choosing an action at each timestep.

1) STATE

The state space captures the information received during the agent's observation of the environment. Suppose there are m intersections in the road network, where the intersection i is composed of n_i roads. The state of the environment is defined by three variables: the head-car distance from intersection of vehicles, the number of vehicles driving towards each intersection, and the head-car turning intention. Formally, the observed state at time t is given by the matrix $s_t \in \mathbb{R}^{N\times 3}$, where $N = \sum_{i=1}^{m} n_i$ is the number of roads flowing into all intersections, referred to as in-roads. Row j of the matrix, where $1 \le j \le N$, represents the observation of in-road j and is stored as the tuple $s_t^j = (d_t^j, v_t^j, \tau_t^j)$, where $d_t^j \in \mathbb{R}$ is the distance between the leading vehicle on in-road j and the intersection, referred to as the head-car distance, $v_t^j \in \mathbb{N}$ is the number of cars on road j and $\tau_t^j \in \{-1, 0, 1\}$ is the head-car intention (left-turn, straight ahead or right-turn). At each timestep $0 \le t \le T$, the agent observes the current state s_t and receives a reward based on this current state.

2) REWARD

The traffic performance is calculated based on a weighted sum of two normalized criteria, where the weights are assumed to be predetermined by a traffic authority. The reward function $R : S \times A \times S \to \mathbb{R}$ is used to quantify the current traffic performance, where higher values indicate a better traffic condition. The first criterion is a cumulative vehicle throughput for all intersections in a road network. Let η_t^i be the number of vehicles that pass through intersection i during timestep t. The cumulative vehicle throughput is defined as the sum of the number of vehicles passing through all intersections per unit of time: $F_t = \sum_{i=1}^{m} \eta_i$. The second criterion is cumulative queue length of vehicles' per unit time for all intersections. The queue length q_t^j of in-road j, where $1 \le j \le N$, at time t is defined as the number of vehicles waiting to be served by a traffic signal. The cumulative queue length is then calculated by

$$Q_t = \sum_{i=1}^{m} \max_{n_i} q_t^j, \tag{7}$$

where n_i is the indices of in-roads to intersection i. Thus, the cumulative queue length measures the sum of the worst, or longest, queue length at each intersection. The reward r_t at

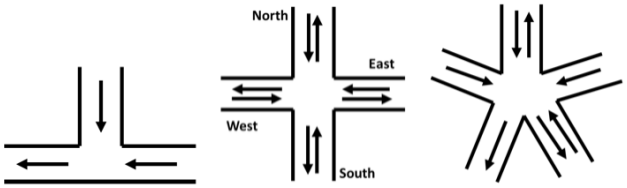


FIGURE 1. Examples of (a) T-junction, (b) four-legged intersection and (c) five-legged intersection types.

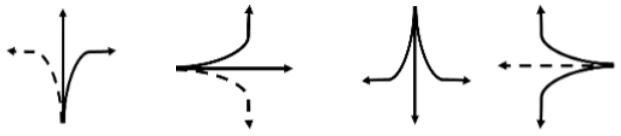


FIGURE 2. Example of four traffic signal for (a) south road, (b) west road, (c) north road and (d) east road, in a four-legged intersection (Figure 1b). A dashed line indicates a restricted direction, while a solid line indicates a non-restricted direction.

time t is calculated by using the weighted sum

$$r_t = R(s_t, a_t, s_{t+1}) = w_f F_t - w_q Q_t, \tag{8}$$

where w_f and w_q are predetermined parameters for weighting set by traffic authorities. The objective of the agent is to follow a policy that maximizes this reward and thereby improve traffic performance in the network.

3) ACTION

Given the vehicular flow direction design for a road network, a centralized traffic control policy π is assumed to control the traffic signal for all intersections. Each traffic signal can take on a phase from a predetermined, finite set of phase patterns; therefore, the agent is used to take phase-based actions. For the road networks considered in this paper, it is assumed that a traffic signal controls the flow of traffic for each road entering the intersection. A phase is used to assign the permission (green light) or restriction (red light) to a combination of non-conflicting vehicular flow directions through an intersection. For example, in the four-legged intersection shown in Figure 1b, there can be four signal phases as shown in Figure 2. In general, an n_i-legged intersection is controlled by n_i traffic signals with n_i phase options. The goal of the centralized traffic controller is to choose a signal phase for each intersection at each timestep. For a road network with m multi-legged intersections, the traffic signal controller is used to decide on a combination of signal phases $a_t = (a_t^1, a_t^2, \ldots, a_t^m)$ for each intersection. Thus, at each time t, the total number of signal phase combinations is $n_1 \times n_2 \times \ldots \times n_m$.

4) DESIGN

The vehicular flow direction design is defined by the vector $x \in \{-1, 0, 1\}^N$, where N is the number of roads to be designed in the network, and -1, 0 and 1 represent one-way clockwise, two-way and one-way counterclockwise flow directions, respectively. The vehicular flow direction of a road can be designed to have a one- or two-way direction, for a total of three design options per road. Examples of vehicular flow directions are shown in Figure 1. Consider the design of vehicular flow directions for the roads in a road network composed of N roads and m multi-legged intersections, where intersection i has n_i roads. Since each road has three design options (two one-way vehicular flow directions and one two-way flow direction), there are 3^N possible design options for the network. However, not all design options are feasible, since conflicts in vehicle flow may arise, especially, as the size of the network grows. Further, note that the subset of feasible signal phases also depends on the flow direction on roads into intersections. Within the co-optimization framework, the vehicular flow design acts as a component of the state observation space.

B. MDP EXTENSION

The co-optimization of vehicular flow direction design and traffic signal control is formulated as an extension of a conventional MDP, defined by the tuple $\{X, S, A, T, R, \gamma\}$, where X is the design space. In the proposed approach, a heuristic search of the design space is integrated with at traditional MDP to create a bi-level optimization framework, as illustrated in Figure 4. The action $a_t \in A$ taken by the agent dictates the traffic signal control policy π and this action is taken based on both the state of the environment and the vehicular flow design variable. Thus, the policy is redefined as the function $\pi : S \times A \times X \to [0, 1]$, where $\pi(a_t|s_t, x)$ is used to compute the probability of taking action a_t given current state s_t and design x. The vehicular flow design in a road network governs the traffic controller's interaction with the environment through a transition probability function $T(s_t, a_t, s_{t+1}, x) = P(s_{t+1}|s_t, a_t, x)$. The transition function, formally defined as $T : S \times A \times S \times X \to [0, 1]$, dictates the transition from one state to the next at each timestep. In this way, the reward function $R : S \times A \times S \times X \to \mathbb{R}$ is also dependent on the vehicular flow direction design, which is given by $r_t = R(s_t, a_t, s_{t+1}, x)$.

The objective is to maximize the expected future reward R by co-optimizing the vehicular flow direction design x and traffic signal control policy π, with the optimal Q value

$$Q^* = \max_{x,\pi} \mathbb{E}\left[R(s_t, a_t, s_{t+1}) \mid s = s_0, a_t = \pi(s_t, x), x\right] \tag{9}$$

where s_0 is the initial state. As before, the agent's objective is to learn a control policy π and design x that achieves this maximum expected future reward, which is approximated by the Q-value function in (2) and (4).

C. DIGRAPH MODEL FOR FEASIBLE DESIGN CLASSIFICATION

The design of the vehicular flow directions for a road network can be modeled by a directed graph $G = (V, E)$, with

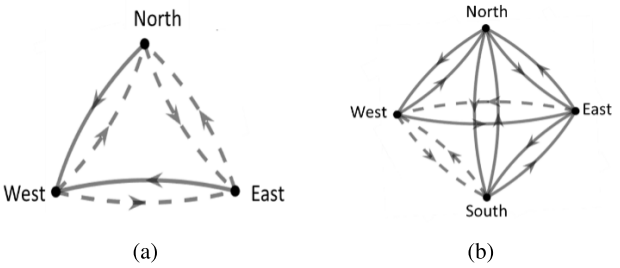


FIGURE 3. Directed graph representation of vehicular flow directions for (a) T-junction and (b) four-legged intersection, where a dashed lines and solid lines indicate restricted and permitted directions, respectively.

the vertex set V and the edge set E. In the model, vertices represent roads in the network and edges represent vehicular flow between roads. Specifically, edge $e_{ii'} \in E$ corresponds to a vehicular flow direction from road v_i to road $v_{i'}$. In this way, the vehicular flow direction design for a road network can be visualized as a directed graph, as shown in Figure 3. The directed graph model allows for the classification of feasible designs [39]. A road design is feasible if the injection roads into the network are strongly connected. That is, if there is a path between all pairs of vertices that represent an injection road. To determine whether injection roads are strongly connected, the authors employ Kosaraju's algorithm [33], in which one uses depth-first search to recursively traverse the directed graph to find strongly connected components. The algorithm is run with time complexity $\mathcal{O}(|V| + |E|)$, where $|V| = m$ and $|E| = m(m-1)$ for a network of m intersections, so the complexity of Kosaraju's algorithm is $\mathcal{O}(m^2)$. In this way, a feasible set of vehicular flow directions can be determined efficiently.

D. CO-OPTIMIZATION FRAMEWORK

The DQN approach to the aforementioned RL technique may be extended to account for the co-optimization of design and control as well. In addition to the traffic signal controller inputs, the Q-value function is extended to account for the design variables as additional inputs: $Q(s, a, x; \theta) \approx Q^*(s, a, x)$, with neural network parameters θ. As such, the objective function is formulated as

$$\max_{x,\theta} \mathbb{E}_\pi [Q(s, a, x; \theta) | a = \pi(s, x)]. \tag{10}$$

Following this formulation, the optimization of vehicular flow direction design and traffic signal control are coupled. The Q-value function is used to estimate the traffic control performance for different designs, which is then used to eliminate poor performing feasible designs until an optimal design is achieved. Once a vehicular flow direction design is fixed, the optimal value of the Q-value function $Q^* = \max_\pi Q_\pi$, yields an optimal control policy $\pi^*(a_t \mid s_t, x)$ for this optimal design. The combination of signal phases a_t for multiple intersections is decided by obtaining the maximum Q-value as $a_t \in \arg\max_{a_t} Q^*(s_t, a_t, x)$.

The co-optimization is done by using a co-learning approach, in which, one trains the Q-value function to simultaneously optimize the vehicular flow direction design and traffic signal controller. In the approach, first, the Q-value function is approximated by using a neural network $Q(s, a, x; \theta)$, where the parameters of the neural network θ are randomly initialized. Then, an iterative optimization process follows, for updating to the Q-value function. The computational framework is illustrated in the flowchart of Figure 4 and outlined in Algorithm 1.

Algorithm 1 Pseudocode for Co-Optimization of Vehicular Flow Direction Design and Traffic Signal Control

1: *Input* Sample size N, batch size d, exploration rate ϵ, number of timesteps T, update frequency M
2: *Initialize* Replay memory D
3: *Initialize* Main and target network weights θ and θ^-
4: *Initialize* Feasible design set X by Kosaraju's algorithm
5: **while** not converge **do**
6: Generate sample of N feasible designs from $U(X)$
7: **for** $i = 1, \ldots, N$ **do**
8: *Initialize* Design $x \sim U(X)$
9: **for** $t = 1, \ldots, T$ **do**
10: Take action $a_t = \arg\max_a Q^\pi(s_t, a, x; \theta)$ with probability $1 - \epsilon$
11: Store tuple $(s_t, a_t, r_t, s_{t+1}, x)$ in replay memory D
12: Sample D tuples randomly from $\mathcal{D}$
13: *Set* $y_t = r_t + \gamma \max_a Q^\pi(s_{t+1}, a_{t+1}, x; \theta)$
14: Perform SGD to update θ
15: *Set* $\theta^- = \theta$ every M steps
16: **end for**
17: **end for**
18: Eliminate bottom k designs from X based Q-value estimation
19: *Set* $N = N \cdot \delta$
20: Check whether converged to one design option
21: **end while**

Algorithm 1 is begun by generating all possible designs and determining a set of feasible designs according to the directed graph model outlined in Line 4. Following these initialization steps, the learning process is started by generating a sample of N designs from a uniform distribution of all feasible designs in Line 6. At this point, the RL training process to learn an optimal policy π is begun in Lines 5-21. By choosing a new design $x \sim U(X)$ at the beginning of each training iteration in Line 12, the agent attempts to learn a control policy that is optimal for all feasible design candidates. At each timestep, the centralized traffic signal controller decides on a combination of signal phases a_t according to the current control policy $\pi(a_t|s_t, x)$ such that $a_t \in \arg\max_{a_t} Q(s_t, a_t, x)$ (Line 10). The agent receives an immediate traffic performance reward r_t and observes the resulting traffic state s_{t+1} in Line 11. After completing one training iteration, a set of traffic data for vehicular flow direction design x is collected as

$\{s_0, a_0, r_0, s_1 \ldots s_{T-1}, a_{T-1}, r_{T-1}, s_T, x\}$. In the next step, the Q-value function is updated with the collected traffic data using experience replay in Line 13. A mini-batch of D traffic datapoints is uniformly drawn at random from the stored traffic data $\mathcal{D}$. Then, the stochastic gradient-decent (SGD) method is used in Line 14 to update the Q-value function parameters θ by minimizing the following loss function as

$$L(\theta) = \mathbb{E}_{(s,a,r,s')\sim U(\mathcal{D})}\Big[\big(r + \gamma \max_{a'} Q(s', a', x; \theta^-) - Q(s, a, x; \theta)\big)^2\Big], \tag{11}$$

where θ^- is the neural network parameters at the previous training iteration [37]. During the procedure, the Q-value function is updated based on data from a diversified set of feasible vehicular flow direction designs. As such, the agent learns a policy π that is optimal for all designs in the current feasible set.

Next, the updated Q-value function is used to approximate the traffic performance on the current set of feasible designs. The estimation of traffic performance for a single design x is calculated by the summation of the Q-values for a set of traffic states, given the current control policy

$$Q_x = \sum_{s_x} Q(s, a, x; \theta | a = \pi(s, x)), \tag{12}$$

where s_x is a randomly generated set of traffic states that corresponds to design x. Once a full training cycle is complete, the Q-value estimation is used to quantify the performance of the sample of vehicular flow direction designs. Based on this estimation, a portion of the design alternatives with the lowest estimated traffic performance is removed from further consideration in Line 18. This portion is dictated by an elimination rate $k \in (0, 1)$, which dictates the number of designs that are eliminated from the optimization. The elimination rate is determined via experimentation to balance the speed of the convergence and quality of the solution obtained.

The final step is to check whether there is only a single vehicular flow direction design remaining. If not, the approach is used to explore the remaining design candidates by sampling another round of design and continuing to train the signal control policy. This process is repeated until only one candidate design remains. Thereafter, the traffic signal controller is trained by using that design until the agent converges to an optimal control policy.

The computational complexity of the proposed approach is dependent on the number of timesteps T in one simulation, the sample size of designs N and the size of the action space $|A|$, which is a function of the number of intersections m. Namely, in a network with m L-legged intersections, one has $|A| = L^m$. To analyze the complexity of Algorithm 1, the authors consider consider the number of function calls performed in one training iteration. A function call is defined as an instance when the agent calls the reward function or Q-values function [40], [41]. As a result, if one neglects the one-time cost of Kosaraju's algorithm in Line 4 and neural network parameter updates via SGD, the worst case cost at time t is L^m. Overall, in a network with sample size N and T timesteps, the algorithm requires $N \cdot T \cdot L^m$ or $\mathcal{O}(L^m)$ function calls per iteration. The authors explore ways to mitigate this complexity in Section III-G.

E. NEURAL NETWORK ARCHITECTURE

In the proposed co-optimization framework, the DQN plays a similar role as in traditional RL-based traffic control algorithms. The DQN outputs an estimated Q-value for each action $a \in A$, denoted by $Q(s, a, x; \theta)$, which approximates the value of taking action a, given the current state of the environment. The agent uses this estimation to choose an action $a \in \arg\max_{a_t} Q(s, a, x)$, which maximizes the future expected reward. The input of the DQN is extended to include the design variable x in addition to the current state of the environment s. As such, the input layer is a vector of size $3N + K$, where K is the number of roads to design in the network. The output is then a vector of size $\prod_{i=1}^{m} n_i$, assuming that intersection i has n_i possible phase options. The hidden layers of the DQN are fully connected layers with Rectified Linear Unit (ReLU) activation functions, the size of which are determined via a grid search of hyperparameters, outlined in Section III-F2. A target network is employed to stabilize training, as outlined in Section II-B. This network has the same architecture as the main network, shown in Figure 5; however, it takes in previous state s' and design x, and outputs $Q(s', a', x; \theta^-)$ to evaluate successive action a'. In the proposed technique, the authors also make use of DDQN and use the target network estimation to evaluate the action selected by the main network, by updated the TD target according to (5).

F. PARAMETER SELECTION EXPERIMENTS

The proposed co-optimization approach is dependent on a number of parameters that impacts its performance and the stability of the learning process. In particular, there are two sets of parameters that impact algorithm performance: design sampling parameters and DQN hyperparameters. The former influences how designs are sampled from the feasible design set, and the latter effects the approximation of Q-values in the deep neural network. Both sets of parameters are determined based on the results of a grid search of the parameter space, as outlined below. For both experiments, the impact on stability, measured by the standard deviation of the agents reward at each training episode, traffic performance, which is quantified by the average episode reward received by the agent, is recorded.

1) DESIGN SAMPLING PARAMETERS

Design sampling parameters govern the manner with which designs are chosen from the feasible design set at the beginning of each training iteration. The following four design sampling parameters are tested for their impact on learning stability and performance: sample size N, elimination rate k, decreasing factor δ, and distribution of designs (weighted or

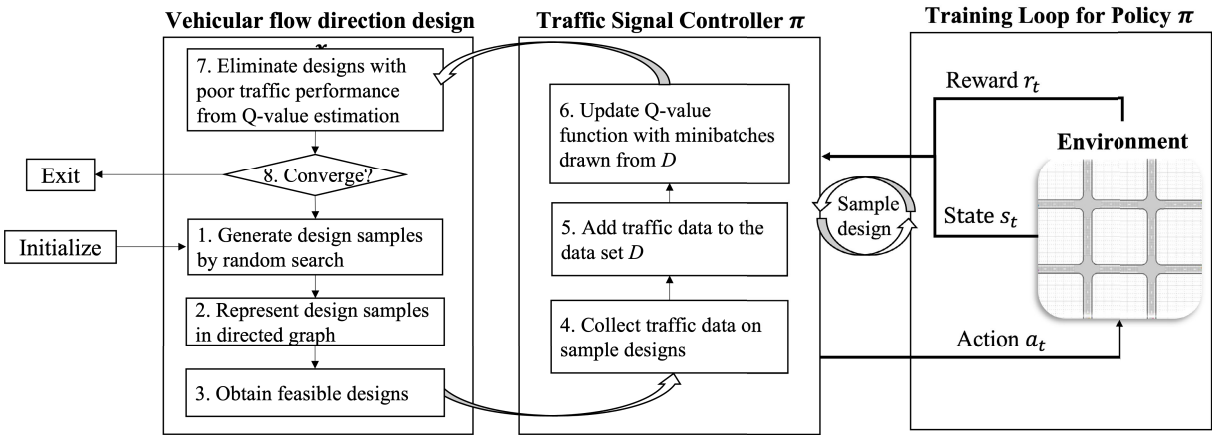


FIGURE 4. RL-based flowchart for co-optimization of vehicular flow direction design and traffic signal control. Each iteration is started by sampling a set of feasible designs (Steps 1-3), then the control policy π is trained by using a standard Deep RL approach with each design in the sample (Steps 4-6). Finally, poor performing designs, as measured by Q-value approximations, are eliminated. The process is repeated until the algorithm converges to a single feasible design.

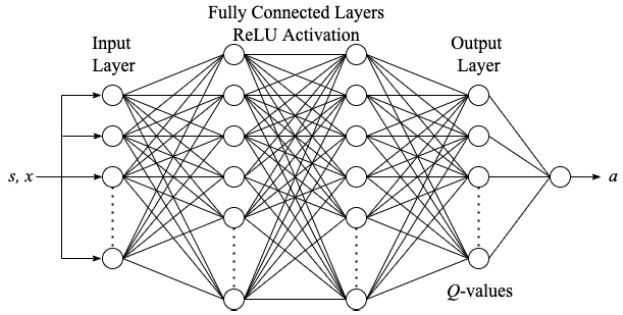


FIGURE 5. Deep Q-Network architecture with two fully connected hidden layers used in the implementation of the Deep RL approach. The network takes in state s and design x and outputs the Q-values associated with taking each action $a \in A$.

uniform). The sample size $N \in \mathbb{Z}$ dictates how many designs are sampled from the feasible design set X at Step 6 in Algorithm 1. In all cases, the sample size N exceeds the number of feasible designs, thus, the traffic controller agent trains on the same design for multiple iterations. At the beginning of each training episode, a feasible design is sampled from a distribution of candidate designs, which dictates the design of vehicular flow directions in the agent's environment for all timesteps in the ensuing episode (Step 8 in Algorithm 1). The elimination rate $k \in (0, 1)$ determines the ratio of designs that are eliminated from consideration after each training cycle. Poor performing designs, as determined by the cumulative estimated Q-values of the previous iteration according to (12), are eliminated from the candidate design set at the end of each training cycle until one design remains. As such, the elimination rate directly impacts the speed of design convergence in the proposed algorithm. The decreasing factor $\delta \in (0, 1]$, governs the update to the sample size made after each training cycle in Step 19 of Algorithm 1. The methodology behind the decreasing factor δ is that the sample size of feasible designs should decrease with the number of feasible candidate designs. Finally, the distribution used in the sampling dictates how designs are sampled from the feasible candidate set X. A uniform sampling distribution selects design x_i with equal probability $p_i = 1/|X|$, while a weighting distribution selects design x with probability

$$p_i = \frac{\overline{R}_i}{\sum_{x_j \in X} \overline{R}_j}, \tag{13}$$

where $\overline{R}_i$ is the average reward obtained with design x_i during the previous training. Defined in this way, a weighted sampling will sample better performing designs with higher probability.

2) DQN HYPERPARAMETERS

Deep Q-learning approaches are sensitive to the tuning of various hyperparameters that dictate the structure of the neural network and its ability to learn features during training. Such parameters include the quantity and size of layers in the network, the learning rate α, the ϵ decay rate, experience replay buffer length, and batch size M. A combination of these five hyperparameters were chosen through experimentation, for use with each application network through a grid search of the parameter space. The standard deviation of episode rewards during training and the average episode reward were used as metrics of comparison to quantify the impact of each hyperparameter on learning stability and performance. Results and analysis from hyperparameter experimentation are presented in Section IV-A.

G. MODEL ASSUMPTIONS

While the proposed approach is applicable to general road networks, since the design and action space scale exponentially to the number of roads and intersections, respectively, reductions on these spaces must be imposed to allow for application to larger road networks. Furthermore, the following assumptions were made during the implementation of the

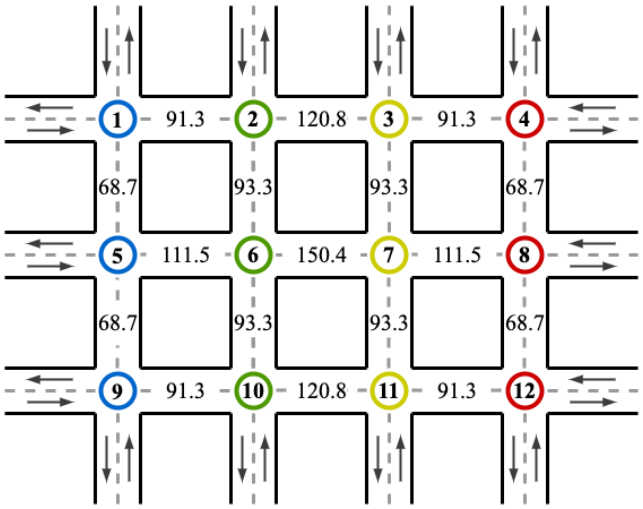


FIGURE 6. Signal synchronization scheme and road importance measures on 12-node grid network. Signals are placed in clusters $S = \{(1, 5, 9), (2, 6, 10), (3, 7, 11), (4, 8, 12)\}$, where signals in the same cluster share phase patterns at each timestep. Road betweenness centrality are calculated according to (14) based on directed graph model.

model: (i) A subset of roads are designated as injection roads, which are used to inject vehicles into the road network during simulation. Injection roads are fixed to be two-way roads by which cars enter and exit the road network. The injection rate for each injection roads can be modified to simulate rush hour or non-rush hour scenarios; (ii) A vehicular flow direction design is feasible if a vehicle that enters from any injection road can exit from all other roads, including the road from which the vehicle injects; (iii) One-way roads are assumed to have a single lane, while bi-directional roads have one lane for each direction; (iv) A centralized traffic controller is used to control signal phases for all intersections; (v) Sensors (e.g., inductive-loop traffic detectors) located at each intersection are used to collect traffic data such as the number of vehicles passing through the intersection and distance of the lead vehicle from the intersection; and (vi) drivers' turning intentions are known before arriving at an intersection.

To increase the applicability of the approach and address the scalability of the model, two simplifications are made. The use of traffic signal synchronization is used to control traffic signals in clusters wherein all signals in one cluster follow the same phase pattern. This is a popular approach in the literature, as traffic signal synchronization may be used to improve the flow of traffic along heavily traveled routes [42]. For networks composed of more than 4 intersections, a signal synchronization scheme of k clusters is fixed, as shown in Figure 6 for a 12-node network. In this example, 4 clusters of 3 traffic signals is chosen to reduce the sufficiently reduce the action space to $4^4 = 256$ to ensure the problem is computationally feasible. In road networks with more diverse intersection types, the dimension of the action space becomes $|A| = \Pi_{i=1}^{k} n_i$, where n_i is the number of incoming roads to intersections in cluster i. Continuing the discussion on computational complexity from Section III-D, by using $M \ll m$ signal clusters in a network with L-legged intersections, one reduces the number of function calls per timestep to L^M. Additionally, the model may be generalized to consider the design of a subset internal roads in the network. This could allow practitioners to fix the design of certain roads in the network, while including a smaller set of roads in the design space, possibly permitting the redesign of particular roads in a network as in reference [43]. For implementation, roads in the design space are decided by the importance of each internal road. By using the directed graph model of road networks presented in Section III-C, the importance of a road can be measured by calculating the betweenness centrality of each node

$$c(u) = \sum_{s,t \neq u} \frac{n_{st}(u)}{N_{st}}, \tag{14}$$

where $n_{st}(u)$ is the number of shortest paths from node s to node t that pass through node u, and N_{st} is the total number of shortest paths from s to t [44]. The betweenness centrality of a road is a measure of the importance of a road as the likelihood of a vehicle driving along a particular road based on network topology. By using this metric, only a subset of roads is chosen as the most important and used as the design space in the co-optimization framework.

IV. APPLICATION & RESULTS

The proposed RL-based approach is implemented by using a traffic simulation tool OpenTrafficLab [45], which is built in MATLAB^{R} [46] by using the Automated Driving Toolbox^{TM} [47]. To evaluate performance, the authors implement the model on two road network sizes with two different traffic scenarios. In the first application case, the model is implemented on a grid-shaped road network with 4 intersections, as shown in Figure 8a, as a proof of concept and to demonstrate the inner-workings of the algorithm. In the next example, the authors consider a road network composed of twelve-intersections, as shown in 8b, to demonstrate the scalability of the problem and application to complex road networks. Additionally, the authors apply the approach in two traffic scenarios: (1) symmetric high traffic flow from all injection roads with a 4-node network, shown in Figure 8a, meant to simulate rush-hour traffic across the road network, and (2) asymmetric high traffic flow from a subset of injection roads with a 12-node network, meant to simulate concentrated rush-hour traffic. In each case, the co-optimization approach is compared to three control agents with random designs, which are trained using the conventional RL-based for traffic signal optimization. Through these two examples, one can quantify the usefulness of the proposed approach and exhibit the interdependence of signal control and vehicular flow direction design in road networks.

The co-optimization agent undergoes offline training to learn an optimal design and control policy by interacting with the traffic environment. The traffic environment is defined by the road network topology and a vehicle traffic simulation,

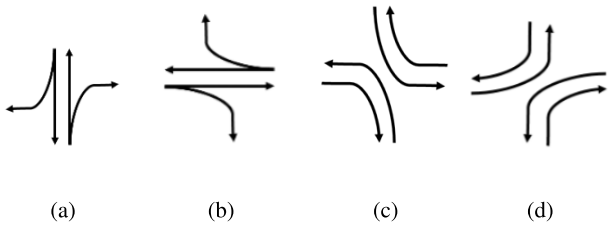


FIGURE 7. Four non-conflicting traffic signal phases for a four-legged intersection: (a) straight and right turn from North and South; (b) straight and right turn from West and East; (c) left turn from North and South, and right turn from West and East; and (d) left turn from West and East, and right turn from North and South.

which imitates real-world traffic scenarios. During the simulation, vehicles enter the road network from injection roads according to a Poisson distribution. As a vehicle approaches an intersection, it is allowed to randomly choose one of the following actions: (1) turn right, (2) turn left or (3) continue straight, each with equal probability. A vehicle that enters the network from an injection road acts in this manner until it exits the road network and leaves the simulation. The reward function is computed as the weighted sum of traffic performance criteria according to (8) at each timestep. After the algorithm converges to an optimal design, the signal control policy is further optimized with the design fixed. At each training iteration, the traffic performance reward is recorded to quantify learning progress.

A. PARAMETER SELECTION

In order to effectively train the deep reinforcement learning agent within the co-optimization framework, the authors investigate the impact on design sampling and DQN parameters on offline training, as outlined in Section III-F. To begin, a set of DQN hyperparameters is fixed and the offline training scheme in Algorithm 1 is run with different design sampling parameters until the algorithm converges to a single design. Next, by using this set of design sampling parameters, the DQN hyperparameter space on the same algorithm is searched. The following investigation was performed separately for the 4- and 12-node road network typologies, and the parameters are outlined in Table 1. The authors search all possible permutations of the parameters. Thus, the experiment consisted of 24 design sampling parameter sets and 16 DQN hyperparameter sets for both 4- and 12-node road networks. Each set of parameters is evaluated based on its offline traffic performance, as measured by the reward function, and the stability of learning, as measured by the standard deviation of the agent's offline learning curve.

Design sampling parameters directly impact both the speed of convergence and the quality of the design determined in the co-optimization framework. For example, a high elimination rate will lead to faster convergence but at the cost of lower exploration, which impacts the quality of the converged design. From the grid search outlined above, the authors found the optimal set of design sampling parameters for the

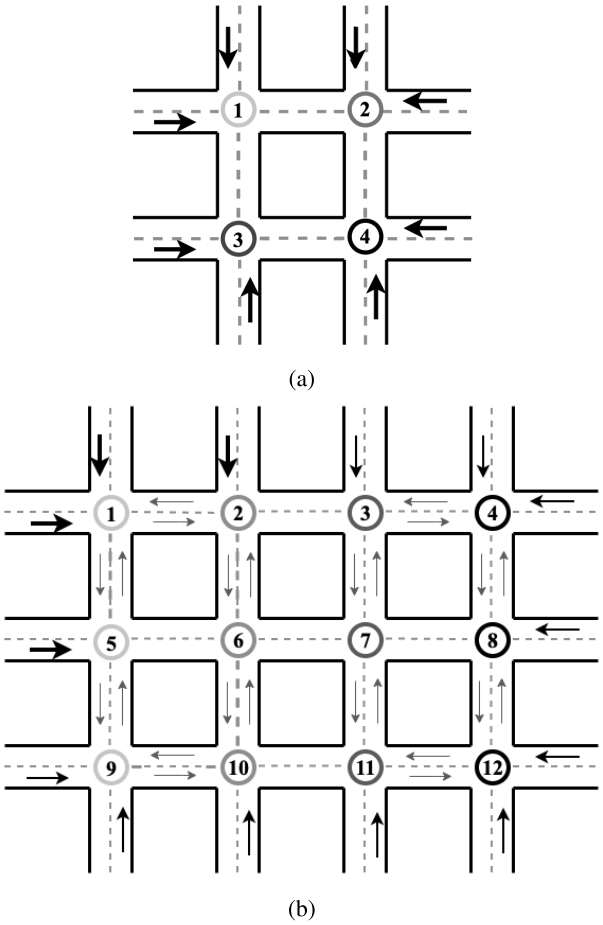


FIGURE 8. Road Network Scenarios used during implementation, where bold and thin arrows indicate rush-hour and non-rush-hour traffic flow, respectively, and each color shade of intersections corresponds to clusters of traffic signals whose phase is synchronized: (a) Four-Node Road Network with symmetric injection rates across injection roads and four independent traffic signal phases; (b) Twelve-Node Road Network with asymmetric injection rates and four signal phase clusters.

TABLE 1. Design sampling and Deep Q-Network parameter space for grid search on $m = 4$, 12-node road networks. Design sampling parameters are sample size N, elimination rate k, sampling distribution, and descending N rate δ. DQN hyperparameters are neural network layers architecture, learning rate α, decay rate ϵ and mini-batch size D.

	Design Sampling			
m	N	k	Dist.	δ
4	$\{200, 500, 1000\}$	$\{0.5, 0.75\}$	$\{U(X), W(X)\}$	$\{1.0, 0.75\}$
12	$\{2000, 3000, 5000\}$	$\{0.5, 0.75\}$	$\{U(X), W(X)\}$	$\{0.6, 0.8\}$

	DQN Hyperparameters			
	Layers	α	ϵ	D
4	$\{[512, 400], [400, 300]\}$	$\{5 \times 10^{-3}, 5 \times 10^{-4}\}$	$\{10^{-3}, 10^{-4}\}$	$\{512, 200\}$
12	$\{[512, 400], [400, 300]\}$	$\{5 \times 10^{-4}, 5 \times 10^{-5}\}$	$\{10^{-4}, 10^{-5}\}$	$\{512, 200\}$

4-node road network to be $N = 1000$, $k = 0.75$, dist $= W(X)$ and $\delta = 0.75$, and $N = 5000$, $k = 0.75$, dist $= W(X)$ and $\delta = 0.8$ for the 12-node road network. The high sampling size N allows the agent to explore a large set of designs, while a high elimination rate and low descending N rate eliminates poor designs more quickly, allowing for faster and more stable convergence to high-quality designs.

By using the aforementioned optimal design sampling parameters, the authors evaluate the impact of DQN hyperparameter sets. For the 4-node intersection, the authors found the optimal set of DQN parameters to be a 2-layer neural network with 512 nodes in layer 1 and 400 nodes in layer 2, learning rate $\alpha = 5 \times 10^{-5}$, decay rate $\epsilon = 10^{-3}$ and mini-batch size $D = 256$. Similarly, for the 12-node road network these parameters were 512 and 400 node layers, $\alpha = 5 \times 10^{-5}$, $\epsilon = 10^{-3}$ and $D = 512$. Due to the randomness of the vehicle scenario, it was found that a lower learning rate produced more stable learning from the agent, while a higher ϵ-decay rate lead to more exploration of the action space, producing more effective control policies during the co-optimization procedure. In addition to the DQN hyperparameters and design sampling parameters determined via experimentation, the deep RL approach is implemented with a discount factor $\gamma = 0.99$ and exploration rate $\epsilon = 0.9$.

B. FOUR-NODE ROAD NETWORK

The design-control co-optimization agent is trained in the four-node road network environment (Figure 8) following the proposed approach outlined in Section III by using the set of DQN parameters and design sampling parameters determined in Section IV-A. At each time step, the centralized traffic signal controller is used to choose a signal phase (or action) for each of the four intersections from the set of non-conflicting signal phases shown in Figure 7. These signal phases comprise the action space for the signal control policy, which has size 4^m for m independently operating traffic signals. In the 4 node road network, the action space has size $4^4 = 256$ at each time step t. At the beginning of the algorithm, all possible vehicular flow designs for the internal roads are generated, of which there are 3^M for a road network with M internal roads. Thereafter, the set of feasible designs is obtained through the directed-graph model outlined in Section III-C. In the four-node road network, the initial design space is comprised of $3^4 = 81$ designs, 31 of which are feasible. These 31 feasible designs comprise the initial design sampling space in the co-optimization algorithm.

1) OFFLINE TRAINING

The learning curve for the offline training of the co-optimization agent is presented in Figure 10a. The agent converged to a single feasible design after the first 5000 iterations, upon which the optimal design was fixed and the control optimization continued for another 4000 iterations. The traffic performance remained stagnant during the design optimization, likely due to the complex dynamics of the traffic environment and the symmetric injection rates. The training curve indicates that the agent struggled to improve traffic performance while optimizing control and design simultaneously in the first 5000 iterations; however, once the design converged the agent effectively improved traffic conditions in the continuation of the control optimization. It appears the agent was able to effectively use its past experiences to quickly improve the control policy after the design optimization terminates, eliminating the overhead of learning a control policy from scratch. To further explore the effectiveness of this approach, the authors compare these results to conventional RL-based signal control optimization agents.

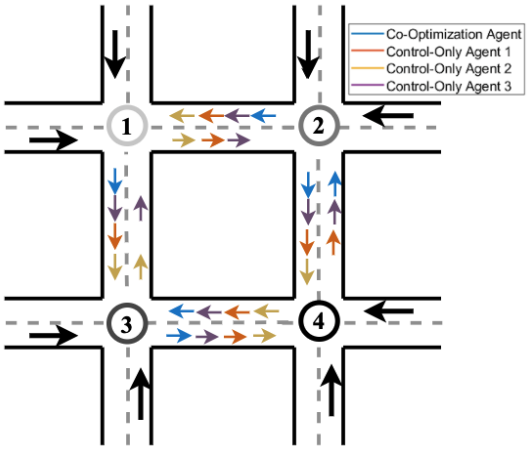
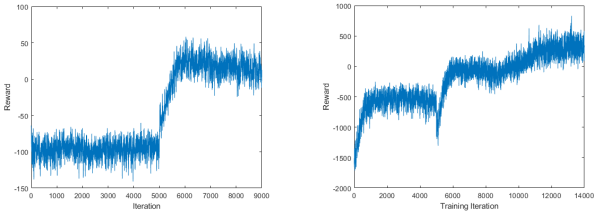


FIGURE 9. Design of 4-node network for co-optimization agents in online simulation comparison.

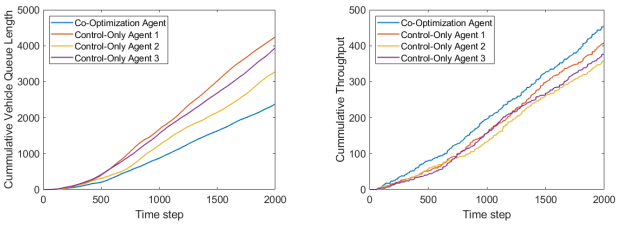


(a) (b)

FIGURE 10. Reward function at each iteration during offline training of co-optimization agents: (a) four-node road network with symmetric injection rates and (b) twelve-node road network with asymmetric injection rates.

2) ONLINE SIMULATION

To display the effectiveness of the proposed co-optimization approach, the authors train three traffic control agents by using a conventional RL-based approach with a randomly generated, fixed design, and compare the performance of the agents in an online traffic environment simulation. The control-only agents begin with a random feasible design, and train for 4000 iterations in an offline environment. The vehicular flow direction designs are presented in Figure 9. Next, a random vehicular flow pattern is initialized and each agents acts in the environment for 2000 time steps, with the traffic performance measured at each time step. To measure traffic performance, the authors record the cumulative vehicular throughput and cumulative queue length; the two metrics which define the reward function. To statistically analyze the results, the experiment is run by using 20 different randomly generated traffic scenarios. A plot of the results from one



(a) (b)

FIGURE 11. (a) Cumulative queue length and (b) cumulative vehicular throughput at each time step during online simulation of co-optimization agent and three control agents in four-node road network scenario.

TABLE 2. Mean and standard deviation of cumulative vehicular throughput F_{put} and vehicle queue length Q_{len} through $n = 20$ online traffic simulations of co-optimization and control-only agents.

	4-node	Co-Opt	Control 1	Control 2	Control 3
Mean	F_{put}	446.30	361.60	354.11	367.26
	Q_{len}	2367.7	4054.2	3232.4	3705.3
STD	F_{put}	25.40	23.86	25.69	32.9
	Q_{len}	250.72	490.44	290.85	500.31

	12-node	Co-Opt	Control 1	Control 2	Control 3
Mean	F_{put}	747.69	580.21	708.20	574.48
	Q_{len}	3604.6	4564.2	5169.7	7521.5
STD	F_{put}	41.936	37.040	81.746	60.995
	Q_{len}	408.4	556.8	532.4	944.7

online simulation is presented in Figure 11, and the averages across simulations are presented in Table 2.

It is observed that the co-optimization agent outperforms all three control-only agents on average, producing a higher cumulative vehicular throughput and lower queue length. This corresponds to more vehicles passing through intersections and less waiting time for vehicles in the network. Based on the averages and standard deviations summarized in Table 2, one can accept this result with 95% confidence, indicating that co-optimizing design and control produces a signal control policy that is more effective at improving traffic conditions compared to the conventional control-only approach.

C. TWELVE-NODE ROAD NETWORK

To exhibit the scalability of the approach, the authors extend the approach to a more complex road network of twelve intersections, displayed in Figure 8b. In addition to the increased size of the road network topology, the injection rates are asymmetric, with incoming traffic flow concentrated in the top left of the grid system. In order to scale the approach to this larger road network, two techniques are taken to reduce the action and design space size. The first is a signal clustering scheme, summarized in Figure 6, which reduces the action space to $4^4 = 256$ at each time step for the centralized traffic controller. In the second technique, one reduces the design space by selecting a subset of internal roads to design, which is summarized in Section III-G. With

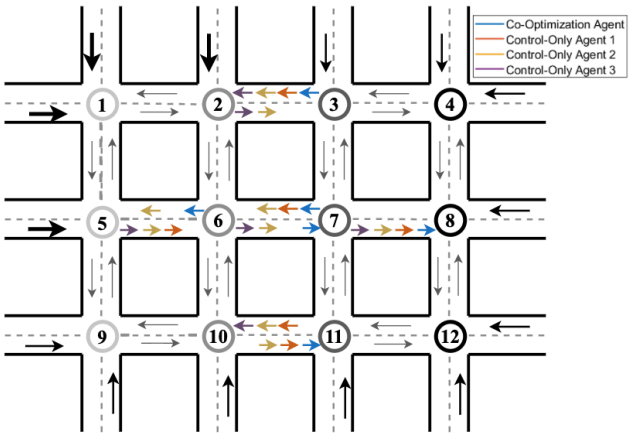


FIGURE 12. Design of 12-node network for co-optimization agents in online simulation comparison.

the reduced action space, the agent decides on an optimal design for 5 internal roads, while the other 12 internal roads are fixed to be bi-directional. Therefore, the design space is composed of $3^5 = 243$ possible designs, 181 of which are feasible. At the beginning of the algorithm, these 181 feasible designs comprise the design space which the agent searches throughout the co-optimization framework.

1) OFFLINE TRAINING

Similar to the implementation of the four-node network, the co-optimization agent is trained in an offline traffic environment by using the empirically determined parameters in Section IV-A. The offline learning curve for the agent in the twelve node environment is shown in Figure 10b. With a larger design space to search, the agent converged to a single design after 8000 iterations, and thereafter continued to optimize the control policy for another 6000 iterations. During the design and control optimization, the agent was able to effectively improve its reward through interactions with the environment. It appears that the asymmetry in the injection roads improved the agent's ability to learn while searching the feasible design space. After the optimal design was fixed, the agent continued to improve traffic performance while converging to an optimal control policy.

2) ONLINE SIMULATION

The performance of the co-optimization agent is compared to the conventional RL-based control only approach in the twelve-intersection road network scenario. As before, three feasible designs are selected from a uniform distribution and three control agents are trained for 6000 iterations using the same DQN hyperparamters with a fixed design, shown in Figure 12. A random vehicular flow pattern is initialized and each agent interacts with the environment for 2000 time steps. The traffic information is measured at each time step and is summarized in Figure 13. This process is repeated 20 times

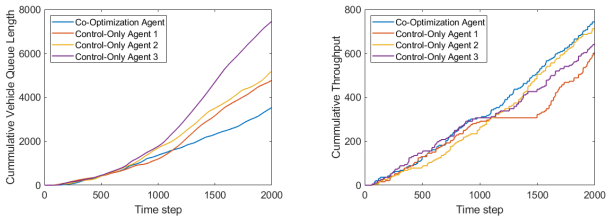


FIGURE 13. (a) Cumulative queue length and (b) cumulative vehicular throughput at each time step during online simulation of co-optimization agent and three control agents in twelve-node road network scenario.

and the mean and standard deviation of traffic performance over 20 simulations is summarized in Table 2.

Again, the co-optimization technique outperforms the three control-only agents, allowing more vehicles to pass through intersections and reducing vehicle waiting times at intersections. From the statistics provided in Table 2, the authors accept the true mean with 95% confidence, indicating that the co-optimization framework is effective in improving traffic conditions compared to conventional control-only approaches. Furthermore, the proposed technique is capable of capturing the dynamics of larger and more complex road networks. It should be noted that due to the increased size of the design space, the training process was slow when run on a high performance computing cluster, indicating that reducing computational cost is key to scaling up the approach to larger networks.

V. CONCLUSION

The goal of any intelligent transportation system is to improve traffic conditions in a road network and effectively capture the complex dynamics of urban transportation systems. To that end, the goal of the present work is to propose a novel RL-based approach to road network management successful at this task. The authors present a technique, in which the design of vehicular flow directions is integrated into the conventional RL-based traffic signal control problem. In the approach, other methods such as a directed graph model are leveraged to determine design feasibility, a centrality measure to quantify road importance, and a set of reasonable measures to reduce design and action space size. At a high level, this approach is an extension of the deep reinforcement learning framework to explore design options via random search, optimizing signal control while eliminating feasible designs based on performance. After a sufficient number of training iterations, the algorithm is found to converge to a best combination of vehicular flow direction design and a corresponding signal control policy.

The proposed approach is demonstrated in two problem instances: a four-node grid network with symmetric vehicle injection, and a twelve-node grid network with asymmetric vehicle injection. These applications are used to illustrate the algorithm's ability to optimize vehicular flow design for multiple roads and to learn an effective control policy implemented by a centralized traffic controller. In each instance, the design-control co-optimization approach is compared to several RL-based control techniques in an online traffic simulation. When acting in this environment, the co-optimization method is found to achieve better traffic performance than the conventional control approaches. This comparison not only illustrates the ability of the present technique to capture the complex dynamics of road networks; but further displays the interdependence on vehicular flow design and signal control; the results indicate that the simultaneous optimization of design and control produce a policy better equipped to reduce traffic congestion.

The present data-driven approach has many advantages. Compared to model-based approaches, the problem is easy to generalize to different road network topologies and traffic scenarios. For example, the approach can be modified to design a specified subset of roads, or to incorporate the synchronization of traffic signals. Although the authors were able to make realistic assumptions to scale the approach to mid-scale networks, the scalability of the approach is one potential limitation. As the role of a centralized traffic controller scales the problem exponentially with the number of independent traffic signals, further modifications to the approach, such as the implementation of a decentralized traffic controller, may be needed to scale the network to mega-scale networks present in the real-world. Nonetheless, the present co-optimization framework sheds light on the interdependence of the design and control of road systems, and effectively reduces traffic congestion in urban road networks.

ACKNOWLEDGMENT

This study was based on the dissertation work [1] of the first author, Dr. Xiangxue Zhao, while she was a Ph.D. student at the University of Maryland. It is acknowledged that the traffic simulator used in this article was created by Dr. Manuel Rodriguez who completed his Ph.D. degree at the University of Maryland, in 2021. The simulator is an extension of an open-source tool in GitHub: MathWorks/OpenTrafficLab: 1.0.0., which is available at this link: https://doi.org/10.5281/zenodo.4354100. The authors acknowledge the University of Maryland High-Performance Computing resources (http://hpcc.umd.edu) made available for conducting the research reported in this article.

REFERENCES

[1] X. Zhao, "Data-driven prediction, design, and control of system behavior using statistical learning," Ph.D. dissertation, Dept. Mech. Eng., Univ. Maryland College Park, College Park, MD, USA, 2021.

[2] I. Mayeres, S. Ochelen, and S. Proost, "The marginal external costs of urban transport," *Transp. Res. D, Transp. Environ.*, vol. 1, no. 2, pp. 111–130, Dec. 1996.

[3] J. Long, W. Y. Szeto, and H.-J. Huang, "A bi-objective turning restriction design problem in urban road networks," *Eur. J. Oper. Res.*, vol. 237, no. 2, pp. 426–439, Sep. 2014.

[4] E. Miandoabchi, F. Daneshzand, W. Y. Szeto, and R. Z. Farahani, "Multi-objective discrete urban road network design," *Comput. Operations Res.*, vol. 40, no. 10, pp. 2429–2449, Oct. 2013.
[5] H. Poorzahedy and F. Abulghasemi, "Application of ant system to network design problem," *Transportation*, vol. 32, no. 3, pp. 251–273, May 2005.
[6] A. Joyo, K. Yaqub, and N. Madamopoulos, "Managing traffic-light-duration by exploiting smart antenna technology (MATSAT) for coordinated multiple-intersections (CMI)," in *Proc. Int. Conf. Emerg. Technol. (ICET)*, Dec. 2015, pp. 1–7.
[7] Z. Cong, B. De Schutter, and R. Babuśka, "Co-design of traffic network topology and control measures," *Transp. Res. C, Emerg. Technol.*, vol. 54, pp. 56–73, May 2015.
[8] P. G. Balaji, X. German, and D. Srinivasan, "Urban traffic signal control using reinforcement learning agents," *IET Intell. Transp. Syst.*, vol. 4, no. 3, pp. 177–188, Sep. 2010.
[9] F.-X. Devailly, D. Larocque, and L. Charlin, "IG-RL: Inductive graph reinforcement learning for massive-scale traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 7, pp. 7496–7507, Jul. 2022.
[10] L. A. Prashanth and S. Bhatnagar, "Reinforcement learning with function approximation for traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 12, no. 2, pp. 412–421, Jun. 2011.
[11] I. Arel, C. Liu, T. Urbanik, and A. G. Kohls, "Reinforcement learning-based multi-agent system for network traffic signal control," *IET Intell. Transp. Syst.*, vol. 4, no. 2, pp. 128–135, 2010.
[12] M. Guo, P. Wang, C.-Y. Chan, and S. Askary, "A reinforcement learning approach for intelligent traffic signal control at urban intersections," in *Proc. IEEE Intell. Transp. Syst. Conf. (ITSC)*, Oct. 2019, pp. 4242–4247.
[13] L. Li, Y. Lv, and F.-Y. Wang, "Traffic signal timing via deep reinforcement learning," *IEEE/CAA J. Automat. Sinica*, vol. 3, no. 3, pp. 247–254, Apr. 2016.
[14] H. Wei, G. Zheng, H. Yao, and Z. Li, "IntelliLight: A reinforcement learning approach for intelligent traffic light control," in *Proc. 24th ACM SIGKDD Int. Conf. Knowl. Discovery Data Mining*. New York, NY, USA: Association for Computing Machinery, Jul. 2018, pp. 2496–2505.
[15] W. Genders and S. Razavi, "Asynchronous N-step Q-learning adaptive traffic signal control," *J. Intell. Transp. Syst.*, vol. 23, no. 4, pp. 319–331, Jul. 2019.
[16] J. Gao, Y. Shen, M. Ito, and N. Shiratori, "Bias based general framework for delay reduction in backpressure routing algorithm," in *Proc. Int. Conf. Comput., Netw. Commun. (ICNC)*, Mar. 2018, pp. 215–219.
[17] S. El-Tantawy, B. Abdulhai, and H. Abdelgawad, "Multiagent reinforcement learning for integrated network of adaptive traffic signal controllers (MARLIN-ATSC): Methodology and large-scale application on downtown Toronto," *IEEE Trans. Intell. Transp. Syst.*, vol. 14, no. 3, pp. 1140–1150, Sep. 2013.
[18] Y. Chin, N. Bolong, S. S. Yang, and K. T. K. Teo, "Q-learning based traffic optimization in management of signal timing plan," *Int. J. Simul., Syst., Sci. Technol.*, pp. 29–35, Jun. 2020.
[19] M. Abdoos, N. Mozayani, and A. L. C. Bazzan, "Holonic multi-agent system for traffic signals control," *Eng. Appl. Artif. Intell.*, vol. 26, nos. 5–6, pp. 1575–1587, 2013.
[20] E. Van der Pol and F. A. Oliehoek, "Coordinated deep reinforcement learners for traffic light control," in *Proc. 30th Conf. Neural Inf. Process. Syst. (NIPS)*, 2016, pp. 1–9.
[21] X. Liang, X. Du, G. Wang, and Z. Han, "A deep reinforcement learning network for traffic light cycle control," *IEEE Trans. Veh. Technol.*, vol. 68, no. 2, pp. 1243–1253, Feb. 2019.
[22] S. S. Mousavi, M. Schukat, and E. Howley, "Traffic light control using deep policy-gradient and value-function-based reinforcement learning," *IET Intell. Transp. Syst.*, vol. 11, no. 7, pp. 417–423, Sep. 2017.
[23] H. Pang and W. Gao, "Deep deterministic policy gradient for traffic signal control of single intersection," in *Proc. Chin. Control Decis. Conf. (CCDC)*, Jun. 2019, pp. 5861–5866.
[24] T. Tan, F. Bao, Y. Deng, A. Jin, Q. Dai, and J. Wang, "Cooperative deep reinforcement learning for large-scale traffic grid signal control," *IEEE Trans. Cybern.*, vol. 50, no. 6, pp. 2687–2700, Jun. 2019.
[25] M. A. Khamis and W. Gomaa, "Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework," *Eng. Appl. Artif. Intell.*, vol. 29, pp. 134–151, Mar. 2014.
[26] R. Z. Farahani, E. Miandoabchi, W. Y. Szeto, and H. Rashidi, "A review of urban transportation network design problems," *Eur. J. Oper. Res.*, vol. 229, no. 2, pp. 281–302, 2013.
[27] E. Miandoabchi, R. Zanjirani Farahani, W. Dullaert, and W. Szeto, "Hybrid evolutionary metaheuristics for concurrent multi-objective design of urban road and public transit networks," *Netw. Spatial Econ.*, vol. 12, pp. 1–40, Jul. 2011.
[28] G. E. Cantarella and A. Vitetta, "The multi-criteria road network design problem in an urban area," *Transportation*, vol. 33, no. 6, pp. 567–588, Nov. 2006.
[29] J. Long, Z. Gao, H. Zhang, and W. Y. Szeto, "A turning restriction design problem in urban road networks," *Eur. J. Oper. Res.*, vol. 206, no. 3, pp. 569–578, 2010.
[30] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*, 1st ed. Hoboken, NJ, USA: Wiley, 1994.
[31] K. Luck, H. Ben Amor, and R. Calandra, "Data-efficient co-adaptation of morphology and behaviour with deep reinforcement learning," in *Proc. Conf. Robot Learn.*, 2020, pp. 854–869.
[32] R. J. Williams, "Simple statistical gradient-following algorithms for connectionist reinforcement learning," *Mach. Learn.*, vol. 8, nos. 3–4, pp. 229–256, 1992.
[33] M. Sharir, "A strong-connectivity algorithm and its applications in data flow analysis," *Comput. Math. Appl.*, vol. 7, no. 1, pp. 67–72, 1981.
[34] R. Bellman and R. Kalaba, "Dynamic programming and statistical communication theory," *Proc. Nat. Acad. Sci. USA*, vol. 43, no. 8, p. 749, Aug. 1957.
[35] A. Haydari and Y. Yilmaz, "Deep reinforcement learning for intelligent transportation systems: A survey," *IEEE Trans. Intell. Transp. Syst.*, vol. 23, no. 1, pp. 11–32, Jan. 2020.
[36] V. Mnih, K. Kavukcuoglu, D. Silver, A. A. Rusu, J. Veness, M. G. Bellemare, A. Graves, M. A. Riedmiller, A. Fidjeland, G. Ostrovski, S. Petersen, C. Beattie, A. Sadik, I. Antonoglou, H. King, D. Kumaran, D. Wierstra, S. Legg, and D. Hassabis, "Human-level control through deep reinforcement learning," *Nature*, vol. 518, pp. 529–533, Feb. 2015.
[37] M. Otterlo and M. Wiering, "Reinforcement learning and Markov decision processes," in *Reinforcement Learning: State of the Art*. Amsterdam, The Netherlands: IOS Press, 2012, pp. 3–42.
[38] H. Hasselt, "Double Q-learning," in *Advances in Neural Information Processing Systems*, vol. 23, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds. Red Hook, NY, USA: Curran Associates, 2010.
[39] T. H. Cormen, C. E. Leiserson, R. L. Rivest, and C. Stein, *Introduction to Algorithms*, 3rd ed. Cambridge, MA, USA: MIT Press, 2009. [Online]. Available: http://mitpress.mit.edu/books/introduction-algorithms
[40] S. Arora and B. Barak, *Computational Complexity: A Modern Approach*, 1st ed. Cambridge, MA, USA: Cambridge Univ. Press, 2009.
[41] S. Koenig and R. G. Simmons, "Complexity analysis of real-time reinforcement learning," in *Proc. 11th Nat. Conf. Artif. Intell.*, 1993, pp. 99–105.
[42] L. Adacher and M. Tiriolo, "Performance analysis of decentralized VS centralized control for the traffic signal synchronization problem," *J. Adv. Transp.*, vol. 2020, pp. 1–19, Nov. 2020.
[43] T. Oguchi, "Redesign of transport systems on highways, streets and avenues," *IATSS Res.*, vol. 32, no. 1, pp. 6–13, 2008.
[44] M. Girvan and M. E. J. Newman, "Community structure in social and biological networks," *Proc. Nat. Acad. Sci. USA*, vol. 99, no. 12, pp. 7821–7826, 2002.
[45] A. Mavrommati, "Mathworks/OpenTrafficLab: 1.0.0," MathWorks, Natick, MA, USA, Tech. Rep., 2020, doi: 10.5281/zenodo.4354100.
[46] *Version 9.9.0 (R2021B)*, MATLAB, MathWorks, Natick, MA, USA, 2021.
[47] *Version 8.9.0 (R2020B)*, A. D. Toolbox, MathWorks, Natick, MA, USA, 2020.
[48] K. Shingate, K. Jagdale, and Y. Dias, "Adaptive traffic control system using reinforcement learning," *Int. J. Eng. Res.*, vol. V9, no. 2, pp. 443–447, Feb. 2020.
[49] T. Chu, J. Wang, L. Codecà, and Z. Li, "Multi-agent deep reinforcement learning for large-scale traffic signal control," *IEEE Trans. Intell. Transp. Syst.*, vol. 21, no. 3, pp. 1086–1095, Mar. 2020.
[50] C. Schaff, D. Yunis, A. Chakrabarti, and M. Walter, "Jointly learning to construct and control agents using deep reinforcement learning," in *Proc. Int. Conf. Robotics Automat. (ICRA)*, 2019, pp. 9798–9805.
[51] D. Ha, "Reinforcement learning for improving agent design," *Artif. Life*, vol. 25, no. 4, pp. 352–365, Nov. 2019.

XIANGXUE ZHAO received the bachelor's degree in electrical science and technology from the University of Electronic Science and Technology of China, the master's degree in industrial and system engineering from the University of Michigan–Dearborn, and the Ph.D. degree in reliability engineering from the University of Maryland (UMD), College Park, MD, USA. Her research interests include predicting, designing, and controlling of system behavior with simulation, experiment, and sensor data using statistical machine learning.

DOMINIC FLOCCO received the bachelor's degree in mathematics and computer science from Davidson College. He is currently pursuing the Ph.D. degree in applied mathematics and statistics, and scientific computation with the University of Maryland (UMD), College Park, MD, USA. His research interests include scientific computing, statistical machine learning, linear programming, and network optimization.

SHAPOUR AZARM is currently a Professor of mechanical engineering and a Professor with the Applied Mathematics and Statistics, and Scientific Computation Program, University of Maryland (UMD), College Park, MD, USA. He is also a Senior Advisor of *Structural and Multidisciplinary Optimization* journal. His research interests include predictive modeling, engineering optimization, and decision analysis. He was the Editor-in-Chief of the *Journal of Mechanical Design* (ASME).

BALAKUMAR BALACHANDRAN (Senior Member, IEEE) is currently a Professor and the Chair of the Department of Mechanical Engineering, University of Maryland (UMD), College Park, MD, USA. He is also a Professor with the Applied Mathematics and Statistics, and Scientific Computation Program, UMD. His research interests include applied mathematics, nonlinear phenomena, dynamics and vibrations, and control.