Engineering Applications of Artificial Intelligence 52 (2016) 203–212

Contents lists available at ScienceDirect

Engineering Applications of Artificial Intelligence

journal homepage: www.elsevier.com/locate/engappai

Traffic flow optimization: A reinforcement learning approach

Erwin Walraven [a,*], Matthijs T.J. Spaan [a], Bram Bakker [b]

[a] Delft University of Technology, Mekelweg 4, 2628 CD Delft, The Netherlands
[b] Cygnify BV, Bargelaan 200, 2333 CW Leiden, The Netherlands

ARTICLE INFO

Article history:
Received 19 June 2015
Received in revised form
2 December 2015
Accepted 4 January 2016

Keywords:
Traffic flow optimization
Traffic congestion
Variable speed limits
Reinforcement learning
Neural networks

ABSTRACT

Traffic congestion causes important problems such as delays, increased fuel consumption and additional pollution. In this paper we propose a new method to optimize traffic flow, based on reinforcement learning. We show that a traffic flow optimization problem can be formulated as a Markov Decision Process. We use *Q*-learning to learn policies dictating the maximum driving speed that is allowed on a highway, such that traffic congestion is reduced. An important difference between our work and existing approaches is that we take traffic predictions into account. A series of simulation experiments shows that the resulting policies significantly reduce traffic congestion under high traffic demand, and that inclusion of traffic predictions improves the quality of the resulting policies. Additionally, the policies are sufficiently robust to deal with inaccurate speed and density measurements.

© 2016 Elsevier Ltd. All rights reserved.

1. Introduction

Traffic congestion is a problem many people are faced with almost every day, causing not only delays, but also pollution and increased fuel consumption. In the United States, travel delay increased from 1.1 billion hours in 1982 to 5.5 billion hours in 2011 (Schrank et al., 2012). At the same time, the amount of wasted fuel increased from 1.9 billion liters to 11 billion liters. The total congestion costs were 121 billion dollars in 2011. Expanding the road network to increase its capacity would be a straightforward solution, but this is not always feasible in practice because of space and budget limitations.

Rather than increasing the capacity of the road network to reduce congestion, variable message signs can be installed, which communicate speed limits to car drivers. These speed limits can be adjusted depending on the current traffic conditions, and such speed limits have a positive influence on traffic flow (Papageorgiou et al., 2008). Although several control algorithms for variable message signs have been developed, most of these approaches are reactive in the sense that speed limits are assigned when congestion is actually detected. In this paper we show that artificial intelligence techniques can play an important role in realizing proactive control for assigning speed limits.

The application of artificial intelligence in the area of traffic and transportation can be motivated by observing that several systems in these areas start to rely more on autonomous and intelligent decision making. For example, autonomously driving vehicles (Bai et al., 2015) need to be able to determine an appropriate driving speed, taking into account the distance to other objects and potential congestion in case the vehicle density on the road increases. Intelligent lane keeping methods need to be able to reason about vehicles in the vicinity, such that appropriate control actions are taken (Lu et al., 2007). Such applications need to decide autonomously how problems are solved, need to respond to a dynamic and changing environment and should be adaptive in the sense that systems should continuously reflect the preferences of their users. All these characteristics motivate the introduction of intelligent software systems in this domain, also known as intelligent agents (Jennings and Wooldridge, 1998). Proactively assigning speed limits to highways also requires an adaptive control system which dynamically responds to sensor data and traffic predictions. In this paper we take a step in this direction by presenting a reinforcement learning algorithm which automatically learns when speed limits should be assigned to highways to reduce congestion, based on the characteristics of the highway as well as demand volumes occupying the highway and predictions regarding future traffic conditions. The resulting algorithm is able to learn proactive control rules in a highly complex domain, and shows that reinforcement learning has the potential to address traffic congestion problems.

The main contributions of our paper can be summarized as follows. First, we formulate a traffic flow optimization problem as a Markov Decision Process (Puterman, 1994), and we show that *Q*-learning (Watkins, 1989) can be applied to find policies dictating how speed limits should be assigned to highway sections to

* Corresponding author. Tel.: +31 15 27 86206.
E-mail address: e.m.p.walraven@tudelft.nl (E. Walraven).

http://dx.doi.org/10.1016/j.engappai.2016.01.001
0952-1976/© 2016 Elsevier Ltd. All rights reserved.

reduce traffic congestion. Second, we show how traffic predictions can be included in our method. Third, we discuss how artificial neural networks (Haykin, 1999) can be used to approximate policies defining the speed limits. Using simulations we show that our methods are able to reduce travel time and congestion under high traffic demand in small road networks, and the policies are sufficiently robust to deal with inaccurate speed and density measurements.

The structure of the paper is as follows. Section 2 discusses related work in the area of artificial intelligence, focusing on problems in traffic and transportation domains. Section 3 introduces the traffic flow optimization problem under consideration. Section 4 provides background information regarding traffic flow modeling and reinforcement learning. The traffic flow optimization problem is formulated as a Markov Decision Process in Section 5 and Section 6 introduces the reinforcement learning algorithm which can be used to obtain policies. Section 7 describes a series of experiments and Section 8 provides a discussion of our work. Section 9 summarizes our results and conclusions.

2. Related work

In this section we give an overview of related work that applies artificial intelligence methods to solve problems in traffic and transportation domains.

2.1. Traffic flow control using artificial intelligence methods

Artificial intelligence has been applied to regulate the number of vehicles entering a highway. For example, ramp metering devices can be controlled using reinforcement learning algorithms (Fares and Gomaa, 2014, 2015). Similar to our work, optimization of traffic flow is formulated as a sequential decision making problem, but these methods explicitly aim to keep the density of the vehicles close to the critical density, such that flow is optimized. Another method to control ramp metering devices with Q-learning can be found in work by Rezaee et al. (2012), where the number of vehicles passing a loop detector near an on-ramp is optimized. Similar work by Davarynejad et al. (2011) takes queue lengths of on-ramps into account during learning, which is not considered in our work. In contrast to ramp metering approaches, we control the speed of vehicles rather than controlling the number of vehicles entering a highway.

Neural networks have been applied in existing work to create controllers for traffic lights (Spall and Chin, 1994; Wei and Zhang, 2002). The road networks used for learning were relatively simple and demand volumes were assumed to be fixed. Neural networks have also been used to predict the exit demand of highways (Kwon and Stephanedes, 1994). Although the application of neural networks is related to our work, the authors did not combine neural networks with reinforcement learning.

Optimization of traffic lights in the urban area is also important to reduce congestion. Kuyer et al. (2008) discuss coordination of traffic lights in the urban area using reinforcement learning, where traffic lights are intelligent agents that coordinate their behavior. A multiagent reinforcement learning controller for traffic lights with multiple objectives is discussed in work by Khamis and Gomaa (2012, 2014), where the method minimizes travel time, increases safety and controls speed in such a way that less fuel is consumed. When modeling control of traffic lights as multiagent system, the system may become complex when scaling up to a large number of intersections with cooperative traffic lights. Abdoos et al. (2013) discuss a hierarchical control method to address this problem. A survey by Zhao et al. (2012) discusses intelligent solutions for control of traffic lights, such as neural networks, and acknowledges that additional research is needed for traffic control using intelligent systems.

2.2. Reinforcement learning for other transportation problems

Besides controlling vehicles from an infrastructural point of view (e.g., traffic lights and speed limits), reinforcement learning has been applied to make routing decisions to guide vehicles through a city (Zolfpour-Arokhlo et al., 2014). Similar to our work, significant travel time reductions can be obtained using reinforcement learning algorithms. Another traffic-related application of reinforcement learning can be found in the area of air traffic management (Tumer and Agogino, 2007), where major delay reductions can be realized. A more general application of reinforcement learning in this domain can be found in work by Cruciol et al. (2013), where different reward functions are investigated for decision-making in air traffic flow management with several stakeholders. In transportation and logistics, reinforcement learning has been applied to control ship unloading (Scardua et al., 2002). Similar to our work, control policies are learned using Markov Decision Processes, Q-learning and neural networks.

3. Traffic flow optimization problem

The problem of traffic congestion on highways occurs if the demand volume exceeds the highway capacity (Papageorgiou et al., 2003). Consequently, the density of the vehicles exceeds the critical density of the highway, the distance between vehicles decreases, and a lower speed is necessary to preserve safety. Furthermore, a congested highway leads to vehicular queueing, an increased travel time, increased cost, more fuel consumption and additional pollution in the environment. If an highway is not affected by congestion, then the flow on the highway is called free flow.

The delay incurred by vehicles as a consequence of congestion can be measured by computing the number of vehicle hours (Zhang et al., 2006). This is a metric that directly relates congestion to the travel time of vehicles. One vehicle hour represents one vehicle driving on a highway for 1 h, but can also be interpreted as 60 vehicles driving on a highway for 1 min. By aggregating over all vehicles, the total number of vehicle hours can be obtained. A similar metric is the total vehicle delay time, which is the total additional travel time compared to the travel time in case of free flow. For instance, if the travel time of a vehicle is 60 min in case of congestion, instead of 45 min in free flow, then its delay time is 15 min.

As mentioned in the introduction, expansion of the road network is not always feasible, and therefore other solutions are required. An example solution is assigning speed limits, which has shown to be able to reduce congestion on highways (Zhang et al., 2006). The problem to assign speed limits to highways is visualized in Fig. 1, where the gray area represents a congested area near the on-ramp and arrows indicate the direction of the vehicle flow. If the traffic demand volume of the on-ramp is high, speed limits can be assigned to upstream sections to reduce congestion. Assigning speed limits is not straightforward, however, because it is difficult to decide when speed limits should be issued and in which parts of the highway speed should be reduced. Additionally, speed limits should be increased and decreased smoothly to preserve safety and alternating sequences of speed limits should be prevented.

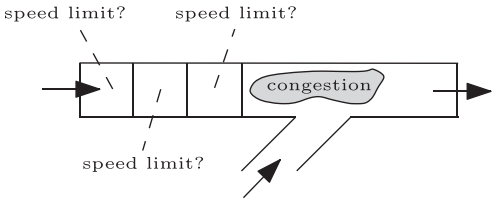

Fig. 1. Example highway stretch with congestion near the on-ramp.

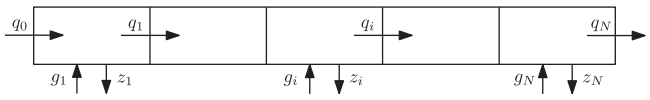

Fig. 2. Highway containing N sections.

4. Traffic flow models and reinforcement learning

In this section we provide an introduction to reinforcement learning and traffic flow modeling using METANET.

4.1. Traffic flow modeling

In the traffic flow domain, a considerable body of research has been performed related to modeling and optimization of traffic flow on highways. Two categories of traffic flow models can be distinguished: microscopic and macroscopic. Microscopic models define the behavior of traffic flow for individual vehicles in terms of their speed, position and the characteristics of the vehicle itself (e.g., maximum speed and acceleration). Such models make it possible to set up an accurate simulation of traffic flow. Unfortunately, they have high computational cost, which makes it infeasible to use them for several applications. Macroscopic models, on the other hand, model traffic flow using average speed, flow and density of highway sections. They are relatively easy to implement and have a low time complexity, because the number of calculations involved is fixed and does not depend on the number of vehicles on the highway. Moreover, the analytical aspects of such models make them suitable for the design of traffic control systems (Kotsialos et al., 2002) and macroscopic models provide a well-balanced tradeoff between the desired accuracy and computational complexity (Van den Berg et al., 2004).

We use an adapted version of the macroscopic METANET model (Papageorgiou, 1983; Messner and Papageorgiou, 1990). The METANET model computes speed, density and flow values of highway sections in closed form, which depend on the current traffic conditions and the traffic demand volumes of the on-ramps and off-ramps. The adapted model supports multiple driving lanes and contains additional boundary conditions (Zhang et al., 2006). We consider a unidirectional highway consisting of N sections, shown in Fig. 2. In the model, T represents the time step length; $k_i(n)$ represents the density in section i at time nT, where n denotes the time step index. The variable $v_i(n)$ is the mean speed of vehicles in section i at time nT, and the variable $q_i(n)$ represents the traffic volume leaving section i and entering section $i+1$ at time nT. The variables M_i and L_i denote the number of lanes and the length of section i, respectively. The variable $w_i(n)$ represents the queue length of the on-ramp associated with section i at time nT. An important parameter of the model is the jam traffic density k_{jam}, for which maximum occupancy is reached and speed approaches zero. The free flow speed v_f is the desired speed of vehicles if there are no constraints imposed by the driving behavior of other vehicles. The notation that we introduced will be used in our MDP formulation in Section 5.

4.2. Reinforcement learning

Reinforcement learning concerns agents interacting with their environment to maximize a cumulative reward signal they receive. A reinforcement learning problem can be modeled as a Markov Decision Process (MDP), which is a mathematical framework to model sequential decision making problems (Puterman, 1994). An MDP consists of a finite set of environment states S, and a finite set of actions A, which can be executed to change the state of the environment. $T_a(s,s')$ is the probability of making the transition from state s to s' by executing action a. State transitions of an MDP are said to be Markovian if any state transition only depends on the last state and is conditionally independent of all other previous states. The reward received from the environment when the state changes from s to s' after executing action a is given by $R_a(s,s')$.

The solution to an MDP consists of a policy $\pi : S \rightarrow A$ that defines the action that should be executed in each state to maximize the expected discounted reward that the agent receives in the future. For each state s and action a, the expected discounted reward when executing action a in state s is denoted by $Q(s,a)$, which is also known as the Q-value function. The optimal policy π^* can be expressed in terms of Q-values as $\pi^*(s) = \text{argmax}_{a \in A} Q^*(s,a)$, with $Q^*(s,a)$ defined as:

$$Q^*(s,a) = \sum_{s' \in S} T_a(s,s') \left(R_a(s,s') + \gamma \max_{a' \in A} Q^*(s',a') \right), \tag{1}$$

where $\gamma \in [0,1)$ is the discount rate.

Optimal Q-values and hence MDP solutions can be computed using dynamic programming (e.g., the value iteration algorithm). However, if the transition probabilities and reward values are initially unknown, then model-free solution techniques can be applied to learn a policy. For instance, the Q-learning algorithm (Watkins, 1989) learns a Q-value function by executing actions in an environment and observing rewards. If the agent is currently in state s, action a is executed to transition to another state s', and reward r is observed, then the following rule is used to update the estimate of $Q(s,a)$:

$$Q(s,a) \coloneqq Q(s,a) + \alpha_q \left(r + \gamma \max_{a' \in A} Q(s',a') - Q(s,a) \right), \tag{2}$$

where $\alpha_q \in [0,1]$ is the learning rate.

5. Formulation as Markov Decision Process

In this section we show that optimizing traffic flow can be formulated as a sequential decision making problem using the Markov Decision Process framework. We present a reinforcement learning method to automatically learn speed limits for highways. We discuss a state representation for this problem, an action space and the reward function, and we also explain how traffic predictions can be included in states. The resulting model will be used in the algorithm that we present in Section 6.

5.1. State description

In this section we define highway states, which characterize the state of the highway at a specific point in time. It is assumed that speed limits are changed at so-called control time steps. We define a control time step size $T_c = c \cdot T$ with $c \in \mathbb{N}$, which is a multiple of the simulation time step size T. This means that speed limits can only be changed if the METANET simulation step is a multiple of c.

We let s_t denote the state at simulation time step ct. The state characterizes the traffic conditions of the highway and is defined below. In our definition we use the variables introduced in Section 4.1

and Fig. 2.

$$s_0 = \left(\frac{v_{\max}}{v_f}, \frac{v_{\max}}{v_f}, \frac{v_1(0)}{v_f}, \ldots, \frac{v_N(0)}{v_f}, \frac{k_1(0)}{k_{jam}}, \ldots, \frac{k_N(0)}{k_{jam}}\right) \quad (3)$$

$$s_t = \left(\frac{a_{t-1}}{v_f}, s_{t-1}(0), \frac{v_1(ct)}{v_f}, \ldots, \frac{v_N(ct)}{v_f}, \frac{k_1(ct)}{k_{jam}}, \ldots, \frac{k_N(ct)}{k_{jam}}\right) \quad (4)$$

The first and second state variables represent the current and previous speed limit assigned to the highway, respectively. In the initial state these speed limits are assumed to be equal to $v_{\max}$, which is the maximum speed limit that can be assigned to the highway. In the next section we will use the first two state variables to restrict the action space. The remaining state variables represent the speed and density of highway section 1 to N, corresponding to the highway sections in Fig. 2. State variables are normalized with respect to the free flow speed v_f and jam density k_{jam}, such that the state variables are in the interval [0, 1]. For example, if the speed of section 1 is equal to 60 km/hr and if the free flow speed equals 120 km/hr, then the third state variable will be 0.5. The speed and density information corresponding to the highway sections is relevant to include in the state description since it influences the ability of the algorithm to detect an incoming traffic jam in case the highway becomes congested. When larger parts of the road network need to be considered, the state description can be extended by adding information about additional highway sections. It should be noted that the state space of our MDP is not Markovian, which we further discuss in Section 6.1.

5.2. Action space

The action space A contains speed limit values that can be assigned to highway sections and should be defined in accordance with traffic regulations and rules. The action $a_t \in A$ is the action selected at simulation time step ct. An example action space is $A = \{60, 80, 100, 120\}$, which is a set of speed limits that can be assigned to highway sections. In order to increase and decrease speed limits smoothly, and to prevent oscillating speed limits, the action space can be restricted by making it state dependent. We let $A(s) \subseteq A$ denote the set of actions that can be executed in state $s \in S$, depending on the first two state variables in s_t. For example, if we assume that the current speed limit is 120 in state s_t, then the restricted action space can be defined as $A(s_t) = \{80, 100\}$ to ensure that speed limit 60 cannot be chosen. Speed limits can also be assigned to a specific part of the highway, rather than assigning a speed limit to the entire highway. For instance, speed limits are typically assigned to a congested area near an on-ramp or road interchange, and a few kilometers upstream.

5.3. Reward function

The reward function can be used to encode the objective function that needs to be optimized. In this paper we reduce traffic congestion and our reward function is directly related to the delay incurred by vehicles as a consequence of congestion. We compute the reward function based on the number of vehicle hours, which is the amount of time vehicles spend on the highway (Zhang et al., 2006). The reward r_t depends on the traffic conditions from time tc until time $(t+1)c$ and is defined as follows:

$$r_t = \begin{cases} 0 & \min\{v_i((t+1)c) \mid i = 1, \ldots, N\} > u \\ -h(tc, (t+1)c) & \text{otherwise} \end{cases}, \quad (5)$$

where u is a threshold. The function $h(b, e)$ computes the number of vehicle hours between METANET simulation step b and e and is defined as:

$$h(b, e) = T \sum_{p=b}^{e} \left(\sum_{i=1}^{N} [M_i L_i k_i(p)] + \sum_{i=0}^{N} w_i(p) \right). \quad (6)$$

The reward represents a punishment proportional to the number of vehicle hours since the last control time step (i.e., the previous speed limit assignment). Therefore, the number of vehicle hours from timestep tc to $(t+1)c$ is computed using the function h. The threshold u can be defined in such a way that there is no punishment if the minimum speed on the highway exceeds a threshold. In all other cases the punishment is proportional to the number of vehicle hours.

5.4. Including traffic predictions

The state description that we introduced in Section 5.1 contains the current speed and density of highway sections, but states do not contain explicit information regarding the future. Taking traffic predictions into account when making control decisions can be beneficial, however, since proactively reducing speed may harmonize flow on highways even before the highway has become congested. Value functions of an MDP already account for discounted future rewards, but states can also be enriched with predictive information.

Now we present an extension of the state description introduced in Section 5.1, which also includes the predicted speed and density of highway sections. The modified state description is shown below, where s'_t denotes the state at simulation step ct, which extends s_t with predictive information. The predicted speed is denoted by v', and the density predictions are denoted by k'

$$s'_t = \left(s_t, \frac{v'_1(ct)}{v_f}, \ldots, \frac{v'_N(ct)}{v_f}, \frac{k'_1(ct)}{k'_{jam}}, \ldots, \frac{k'_N(ct)}{k'_{jam}}\right) \quad (7)$$

For real highways traffic predictions can be obtained using traffic flow forecasting methods (Smith and Demetsky, 1994). In macroscopic simulations the future highway state can be predicted by running a separate METANET simulation based on the current highway state, and the expected demand volumes of the origin and on-ramps. This process is illustrated in Fig. 3, which shows a state transition from s_{t-1} to s_t. A METANET simulation starting from state s_t is performed to predict the state s_{t+1}, which is visualized as a dashed arrow. Subsequently, information from state s_{t+1} can be integrated in state s'_t such that it includes information regarding the future state of the highway.

6. Learning policies using Q-learning and neural networks

In this section we present an algorithm to learn speed limit policies, based on the Q-learning algorithm. We also discuss two existing methods to approximate Q-value functions during learning. The resulting algorithm can be used to compute speed limits, given the current and predicted traffic conditions of the highway.

6.1. Speed limit policy learning algorithm

Our algorithm is schematically shown in the flowchart in Fig. 4. The algorithm runs several Q-learning episodes, during which a METANET simulation is executed. When starting an episode, the initial state is defined, and the first speed limit is assigned (first block). Then a traffic simulation is executed using METANET, taking into account the speed limit assignment (second block). After the simulation a new MDP state can be defined, a Q-value is updated in memory using Eq. (2), and a new speed limit is assigned (third block). If the episode ends, the algorithm starts

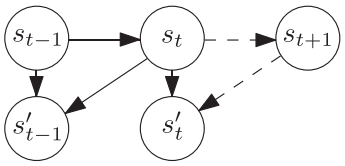

Fig. 3. Predicting the highway state s_{t+1} at simulation step tc to define predictive state s'_t.

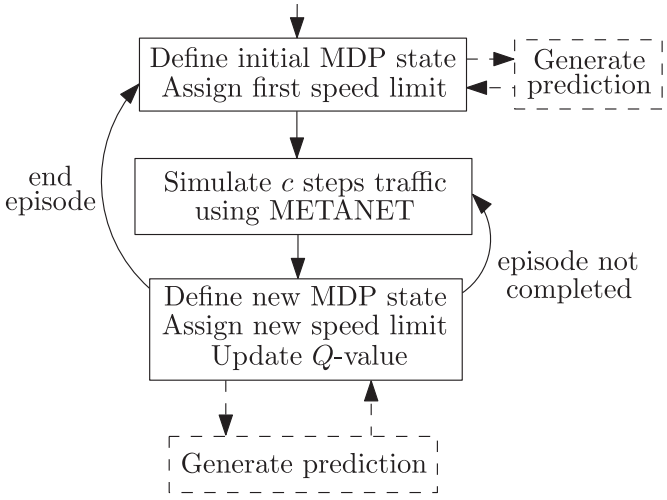

Fig. 4. Flowchart representing our speed limit policy learning algorithm based on Q-learning and the METANET simulation model. The second and third blocks will be executed several times, until an episode has ended, which is represented by the labeled arrows. Generation and inclusion of traffic predictions is optional, represented by the dashed arrows and blocks.

from the initial traffic state and starts a new episode. In all other cases it continues the episode and runs a METANET simulation. These conditional steps are represented by the labeled arrows. Predictions of the future highway state can be included in the state as well, which is visualized as an optional step (dashed blocks).

Learning speed limit policies using our MDP model can be considered as a hidden state task (Whitehead and Lin, 1995), since the state description introduced in Section 5.1 is not Markovian and some information about the state of the highway is hidden (i.e., not included in the MDP state). Defining a compact Markovian state representation for highways is difficult, however, because many different external factors influence traffic flow on highways, which are hard to model accurately. To be able to learn policies with non-Markov state representations, a state history can be maintained in memory to distinguish hidden states, even if the current state of the highway appears to be identical. The window-Q architecture introduced by Whitehead and Lin (1995) is a method to maintain a state history in memory with a sliding window, which can also be combined with the policy learning algorithm that we introduced in this section.

6.2. Approximating Q-value functions

In the process of learning speed limit policies, a Q-value function is stored and updated in memory. Function approximation can be applied to store an approximate value function in memory and has three purposes. First, a function approximator allows us to apply Q-learning when the state space is continuous. Second, value functions can be approximated to avoid memory requirements that scale exponentially in the number of state variables. Third, a function approximator improves learning efficiency because it generalizes learning experiences.

We use and compare two different value function approximation methods. We use a linear approximation method called tile coding (Sutton and Barto, 1998), which is a method to systematically discretize the state space. The second method that we use is a value function representation based on artificial neural networks, where neural networks are trained to obtain the desired input–output behavior of the value function. An example is given in Fig. 5, which shows four neural networks corresponding to the actions in action space $A = \{60, 80, 100, 120\}$. For a given state s, the value $Q(s, a)$ can be obtained by feeding the state variables as input to the neural network that corresponds to action a. Instead of applying Eq. (2) during learning, a Q-value error (i.e., the difference between the current and desired output) is propagated backwards through the neural network using the back-propagation algorithm (Bryson and Ho, 1975).

The MDP state variables given as input to a neural network are values between 0 and 1, but often neural networks learn more efficiently with binary input variables. Therefore, the input can be augmented with additional input variables that can be derived directly from the MDP state. For example, the binary indicator variable I_n defined below equals 1 if the minimum speed is below 100 at simulation step cn, and 0 otherwise.

$$I_n = \begin{cases} 1 & \min\{v_i(cn) \mid i = 1, \ldots, N\} < 100 \\ 0 & \text{otherwise} \end{cases} \tag{8}$$

Indicator variables representing the current speed limit and representing the density can be defined similarly. In the experimental evaluation we clearly define which indicator variables are used as additional neural network inputs.

7. Experiments

To evaluate the proposed learning algorithm, we run several experiments in which we show that our algorithm generates policies that reduce congestion under high traffic demand, and we show that flow can be controlled more efficiently when including predictions in states. We also study the robustness of policies in case of inaccurate speed and density measurements. First we explain the setup of our experiments, and then we discuss the individual experiments and corresponding conclusions.

7.1. Experimental setup

We consider the highway depicted in Fig. 6 for $N=8$ with two lanes and each section has a length of 3 km. The relevant parameters are given in Table 1. In most of the experiments we use METANET as our evaluation model. The METANET model parameters correspond to a real highway, as discussed in work by Karaaslan et al. (1990). In each episode we run a simulation for 1 h with step size 15 s, which means that we use METANET to calculate 241 steps. This includes the initial traffic state, in which the speed and density of each section is 120 km/h and 17 veh/km/lane, respectively. Speed limits are changed every 5 min, and therefore the variable c equals 20. In our evaluation we consider three traffic scenarios, which are shown in Fig. 7. We assume that $v_{\max}$ equals 120 and the action space is $A = \{60, 80, 100, 120\}$. The speed limits are only assigned to sections 2–6. These speed-controlled sections have been selected in such a way that speed control applies to the area near the first on-ramp and a few kilometers upstream. Hence, it allows us to control the number of vehicles approaching the area near the on-ramps, where congestion may arise if the incoming flow is high. In practice the length of the speed-controlled area may be selected depending on the distance between multiple consecutive on-ramps. The action space is state dependent, such that multiple consecutive speed limits do not alternate and

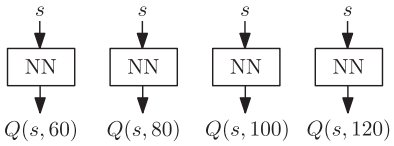

Fig. 5. Q-value function approximation with neural networks for action space $A = \{60, 80, 100, 120\}$.

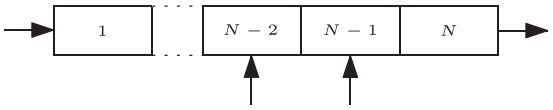

Fig. 6. Highway with N sections, an origin and two on-ramps.

increase and decrease smoothly. For example, if the current speed limit is 80, then the action corresponding to speed limit 120 cannot be chosen. This restriction can be made based on the first two state variables in Eqs. (3) and (4).

We apply Q-learning with an ε-greedy exploration strategy to learn policies, where the probability to select random actions decreases linearly from 1 to 0. To evaluate the quality of policies, we use them to control 1 h of traffic flow and we use the total number of vehicle hours in the simulation as a performance metric. The metric is equivalent to computing $h(1,241)$, because this is the total number of vehicle hours in a simulation, which should be minimized. We learn and evaluate policies using the same traffic flow model and simulation data, but in Section 7.7 of this paper we will run additional simulations in which the learning and evaluation model are distinct.

We use and compare two different value function approximation techniques: tile coding (Sutton and Barto, 1998) and artificial neural networks (Haykin, 1999). For approximations with tile coding we define 60 overlapping tilings, with four tiles in each dimension of the state space and each tile has width 1.3. The implementation with neural networks is using one bias node and learning rate 0.01. The hidden layer contains one more neuron than the number of input neurons, and back-propagation is used for training (Bryson and Ho, 1975). One previous state is maintained in a state history to deal with the non-Markovian state space (Whitehead and Lin, 1995), as we discussed in Section 6.1.

7.2. Policies for a single scenario: tile coding

In this section we investigate whether the learning algorithm based on Q-learning is able to learn appropriate policies for individual traffic scenarios. This means that a policy is learned and evaluated in a simulation run with the same traffic scenario. In the experiment we aim to show that policy learning algorithm delivers policies of good quality, reducing traffic congestion significantly. In the next section we repeat the same experiment with neural networks as value function approximator and a more extensive state description.

We use tile coding to approximate value functions and we use the state description introduced in Section 5. However, we only include speed values of highway sections 4–7. We generate 20 policies for each traffic scenario and we run 5000 learning episodes. For each policy, the number of vehicle hours is computed and the results are shown in Fig. 8. In the figure, the TC column shows the policy quality without predictions included in states, and the TC+predictions column shows the quality of policies learned with 5 min predictions included in states. Additional statistics regarding the policy quality can be found in Table 2.

To analyze the quality of the resulting policies, we compute the best fixed speed limit assignments for each scenario. We enumerate all action sequences $a_1, a_2, \ldots, a_{13}$, such that $|a_i - a_{i+1}| \leq 20$ for all $i = 1, 2, \ldots, 12$, and we compute the number of vehicle hours realized by each sequence to determine the minimum. Computing such an assignment is not always feasible in general, since the number of combinations grows rapidly if either the control step size is smaller or the scenario has a longer duration. The dashed horizontal lines in Fig. 8 represent the number of vehicle hours of the best possible fixed speed limit assignment, and the baseline, which is the number of vehicle hours without any speed control. The best fixed speed limit assignment and the baseline values are also shown in Table 2.

Table 1
METANET, Q-learning and MDP parameters used in the experiments.

v_f	k_{jam}	l	m	α	κ'	κ	τ
130	110	1.86	4.05	0.95	4	40	20.4
u	μ_1	μ_2	σ	ρ	T	α_q	γ
101	12	6	35	120	1/240	0.1	0.8

The results of the experiment show that the quality of the learned policies is close to the best fixed assignment, and policy quality improves when including predictive information in the state. For each scenario, the best policy found with predictive information included in states is better than the best policy found without predictive information in states. In comparison to the baseline values reported in Table 2, the policies give a significant improvement. In the next section we investigate whether the quality of policies can be further improved.

7.3. Policies for a single scenario: neural networks

In addition to the experiment from the previous section, we investigate the performance of the policy learning algorithm with neural networks as value function approximator and an extended state description.

The state description introduced in Section 5 is not Markovian, which means that inclusion of additional information in states may improve the performance of the algorithm. To include more information in states, we rely on neural networks as a value function approximation technique. The reason is that tile coding does not scale well in terms of running time.

The setup of this experiment is similar to the experiment described in the previous section, except that we use neural networks to approximate the value function and more information is included in state description. Besides the speed values of highway sections 4–7, we also include the density values of these sections. We also include binary indicator variables as additional inputs to the neural network, representing which speed limit is currently assigned and whether there is currently free flow or congestion. These binary indicator variables are useful to make learning with neural networks more efficient, as discussed in Section 6. The number of learning episodes is increased to 20 000.

The results of this experiment are also shown in Fig. 8 and Table 2, represented by the columns NN and NN+predictions. We conclude that the performance has improved in comparison to the results obtained with tile coding and the initial state description. This can also be seen in the table, where the number of vehicle hours is consistently lower in the second experiment with neural networks. This experiment shows that the resulting policies are able to control traffic more efficiently. Moreover, we conclude that inclusion of predictions in states can be beneficial because it leads to better performance in most cases.

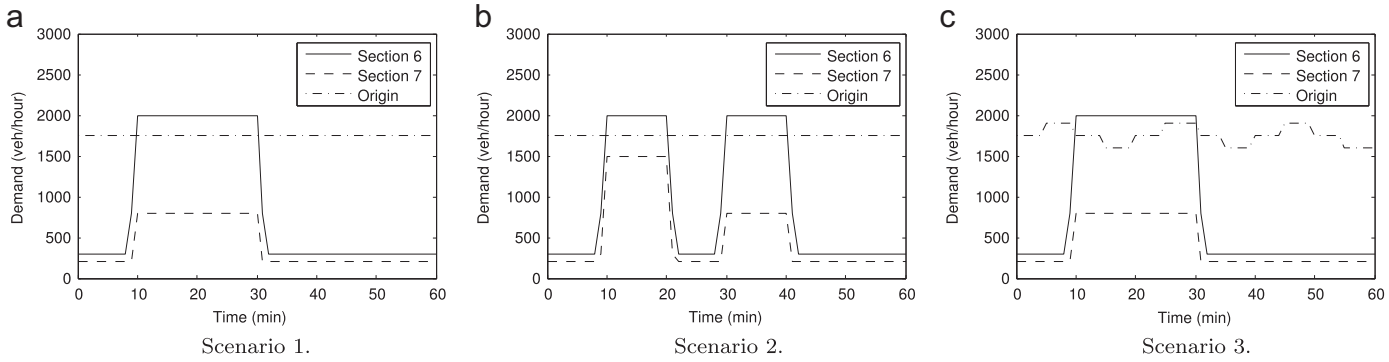

Fig. 7. Demand profiles defining traffic volumes for the origin and two on-ramps during 60 min.

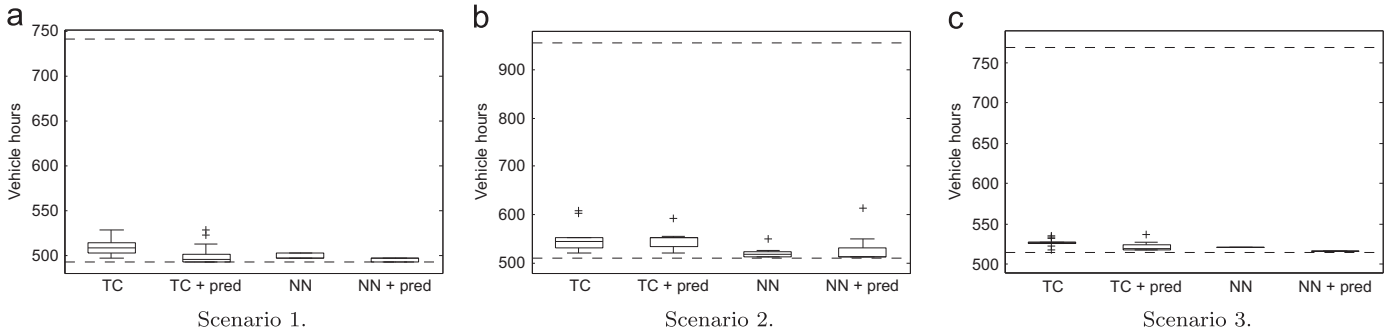

Fig. 8. Policy quality of the experiment with tile coding (TC) and neural networks (NN) as function approximator, with and without predictions included in states (labeled as 'pred').

Table 2
Vehicle hours of policies controlling traffic in one scenario.

Control method	Scenario 1	Scenario 2	Scenario 3
Baseline (no control)	740.8	954.5	768.6
Best fixed assignment	491.6	511.2	514.5
Tile coding	509.0 ± 7.5	549.1 ± 25.9	526.7 ± 3.6
Tile coding + predictions	498.8 ± 10.4	546.4 ± 16.3	521.4 ± 4.8
Neural networks	498.5 ± 2.2	520.6 ± 8.6	521.0 ± 0.0
Neural networks + predictions	494.9 ± 2.5	525.2 ± 23.2	516.4 ± 0.0

Table 3
Vehicle hours of policies controlling multiple scenarios.

Control method	Scenario 1	Scenario 2	Scenario 3
Baseline (no control)	740.8	954.5	768.6
Best fixed assignment	491.6	511.2	514.5
Policy	492.1 ± 1.2	524.3 ± 1.7	516.5 ± 0.4

7.4. Policies for multiple scenarios

The policies in the previous experiments have been learned based on one traffic scenario. To mitigate the risk of overfitting, policies can also be learned by running episodes for different scenarios sequentially. In this experiment we use the policy learning algorithm, implemented as described in Section 7.3, to learn 20 policies. Each policy is trained sequentially on the three scenarios during 10 000 learning episodes, and therefore the number of episodes for each policy is 30 000 in total. For each scenario, the policies have been used to run a traffic simulation with speed control, and again the number of vehicle hours is used as a performance metric. The results of this evaluation are shown in Table 3, which shows the results obtained after applying 20 policies to the scenarios. From the table we conclude that the quality of the policies is still close to the best fixed speed limit assignment, and the improvement compared to the baseline values is significant. Moreover, the experiment shows that the algorithm is capable of learning more general policies that can be used to control traffic in several different scenarios.

7.5. Effects of traffic control on highway traffic

In our previous experiments the performance of the policy learning algorithm has been analyzed, where the performance is measured using the vehicle hours metric. Besides this theoretical analysis of the quality of a policy, we can also visualize how policies control traffic flow on highways in graphical form. In this experiment we use a policy that was learned using our algorithm combined with neural networks. Fig. 9 depicts how the speed on the highway changes over time in scenario 1, where speed is represented by a grayscale color ranging from white to black, where white represents free flow and black represents a fully congested section. The picture on the left shows how the speed evolves over time for each section if there is no traffic control with speed limits. The picture on the right shows that the speed of sections 2–6 can be temporarily decreased such that congestion has resolved after 60 min. From this figure we conclude that policies are able to recognize when the highway starts to become congested, such that appropriate speed limits can be assigned to reduce congestion on the highway.

7.6. Robustness of policies

Data received from loop detectors in a real road network may be affected by noise. Therefore, we describe an experiment to investigate whether policies behave well in case of such inaccurate measurements. For each scenario we run the algorithm to learn a policy during 5000 episodes, resulting in one policy for each scenario. These policies can be applied to control traffic in their corresponding scenarios, and during these evaluation runs we add noise to the speed and density values in the states, representing the inaccuracies in speed and density measurements that may occur in practice. The noise is drawn from a Gaussian distribution and added to the state variables representing speed and density. To determine how much noise is added to the speed and density measurements, we use a Gaussian distribution $N(\mu, \sigma)$, where μ is either speed or density, and σ ranges from 0μ to 0.3μ, which represents the noise percentage. We did not include noise during the learning episodes.

For each scenario and noise percentage, we run 50 simulations and for each run the number of vehicle hours is computed to assess whether a policy is still able to control traffic flow appropriately. Fig. 10 shows the results for each scenario that we defined, and the horizontal dashed lines represent the number of vehicle hours of the best fixed speed limit assignment, and the baseline value. From the result we can conclude that policies behave well for noise up to 10 percent in macroscopic simulations. If there is more noise, then speed and density measurements become inaccurate and policies can no longer be used to select appropriate speed limits. This can be concluded from Fig. 10, because in those cases the number of vehicle hours may exceed the baseline, which shows that the situation may be even worse in comparison to not controlling traffic at all.

7.7. Evaluation using distinct simulation model

In our previous experiments we learn and evaluate our policies using the same traffic flow model and simulation data. In order to evaluate the performance on other models and to address the risk of overfitting, we did another experiment using a microscopic simulation environment. The main purpose of this experiment is testing whether the policies also reduce congestion if we apply them to distinct simulation models, other than METANET.

We implemented a microscopic traffic simulation in Simulation of Urban MObility (SUMO), which simulates traffic flow at the level of individual vehicles (Behrisch et al., 2011). The experiment involves a simulation of vehicles driving near a road interchange in Eindhoven in the Netherlands. We use the map of the road network from OpenStreetMap and imported this data into SUMO. We derived a traffic demand pattern from NDW, the Dutch national database containing historical traffic data. This pattern was collected from sensors and loop detectors integrated in the pavement. Using this pattern we were able to define vehicle flows in the simulation similar to real traffic that was measured on October 21, 2013 from 7 to 8 am. In our simulation we create an artificial accident which causes traffic to slow down for 15 min. We use precomputed policies to determine appropriate speed limits during simulation, and assign them to approaching vehicles driving at most 8 km upstream in the direction of the accident. The policy learning procedure we used is similar to the procedure in Section 6 and is based on METANET.

The experiment has been repeated 20 times for several compliance rates. The compliance rate defines how many vehicle drivers comply with the speed limits. The results are shown in Fig. 11, in which we use the number of vehicle hours as a performance metric, similar to the evaluation of the policy learning algorithm. From the figure we can derive two conclusions. First, and most importantly, policies learned using the METANET simulation model are sufficiently general to reduce congestion in other road networks, not identical to the model used for learning. Second, if there is only limited compliance (e.g., 20 percent), then our policies already reduce congestion compared to the simulation without any speed control.

8. Discussion

In this section we provide a discussion of our contributions and its evaluation. Furthermore, we discuss aspects which we did not

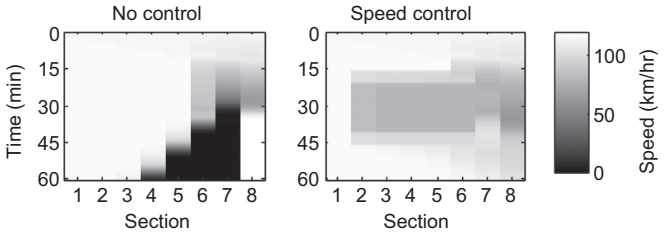

Fig. 9. Comparison between no control and speed control using a policy.

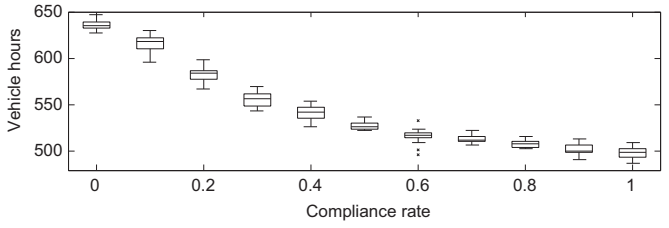

Fig. 11. Vehicle hours for several compliance rates in the microscopic simulation experiment.

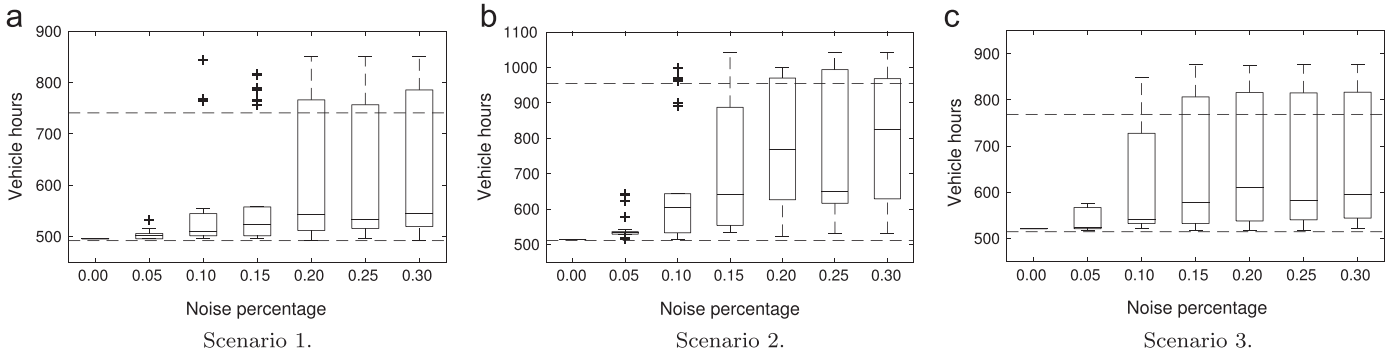

Fig. 10. Vehicle hours in case of noisy measurements.

study, and key problems that can be investigated further in future research.

Our method to reduce traffic congestion is primarily focused on speed control, which aims to reduce the speed along the highway such that traffic congestion is reduced. Even though speed control has shown to be able to reduce congestion in the literature (Zhang et al., 2006), it should be noted that speed control is not always sufficient to achieve effective congestion reduction. Traffic flow on highways can be characterized using speed, density and flow. This means that for a fixed and controlled maximum speed, there can be a large variety of flow values, which consequently influence the density and occurrence of traffic congestion. Therefore, in addition to speed control there is a need to explicitly control the flow on highways, such that both speed and flow can be controlled. Since our method purely focuses on the learning techniques that are needed to obtain appropriate speed limits, it currently does not account for explicit flow control. Complementary reinforcement learning techniques in the literature have shown to be able to control the on-ramp flow using ramp metering devices (Fares and Gomaa, 2014, 2015), and in future work it can be studied how such techniques can be combined with reinforcement learning algorithms based on speed control.

The evaluation of our method is currently based on small-scale experiments focusing on individual highways. Traffic congestion problems in practice, however, may affect multiple highways or an entire road network, because there can be many dependencies between traffic on several interconnected highways. For instance, reducing congestion in one part of the road network may cause congestion in other parts. In order to realize effective congestion reduction in larger road networks based on our approach, it can be investigated how our work can be extended to multiple highways. Our approach has been designed in such a way that extensions to larger road networks can be achieved without changing the operation of our reinforcement learning techniques. The macroscopic METANET model can be replaced by a more general model which models a road network as a directed graph, in which each edge represents a separate METANET highway (Van den Berg et al., 2007). The action space of our MDP formulation can be extended such that actions assign distinct speed limits in multiple parts of the road network. The Multiagent MDP (MMDP) framework (Boutilier, 1996) can be used to encode this setting. Moreover, it can be expected that it is not necessary to consider the full MMDP action space since several combinations of actions are infeasible in a real road network. Our reinforcement learning approach and its evaluation provides a starting point for congestion reduction using artificial intelligence methods, and its extensibility to larger road networks is one of the key aspects which can be considered in future research.

The application of reinforcement learning algorithms, combined with models from the traffic flow domain, shows that artificial intelligence has the potential to address traffic congestion problems. The macroscopic METANET model has been validated in several existing studies (Papageorgiou et al., 1990; Ngoduy et al., 2003), and the fact that it can be successfully combined with artificial intelligence techniques motivates further research and applications at the intersection of both domains.

9. Conclusions

In this paper we presented a new method to assign speed limits to highways, based on reinforcement learning. We defined a traffic flow optimization problem as a Markov Decision Process and we applied *Q*-learning to find policies to assign speed limits. In contrast to existing work, we have also shown that predictive information can be taken into account. To improve the scalability and performance aspects of our algorithm, we have shown that artificial neural networks are able to efficiently learn policies. Simulation experiments have shown that the resulting policies are able to reduce traffic congestion under high traffic demand in simple scenarios based on small road networks, and the policies are sufficiently robust in case speed and density measurements on highways are inaccurate. Our experimental evaluations have shown that the proposed reinforcement learning approach performs well in small road networks and it provides a starting point for large-scale evaluations, which are necessary when using reinforcement learning methods in real traffic engineering applications and larger road networks.

Our methods are relevant to accelerate the development of intelligent AI-based algorithms for intelligent transportation systems, which are systems that apply advanced communication, information and electronics technology to solve transportation problems (Figueiredo et al., 2001). First of all, in-car information displays can be used to provide personalized advice regarding driving speed, taking into account the predicted traffic conditions. This allows for proactive traffic flow control, rather than using traditional approaches which activate variable message signs after having detected congestion. Second, autonomously driving vehicles need to determine their optimal driving speed, and our method provides a starting point to intelligently make such decisions using AI-based algorithms. We conclude that our work contributes to the developments in the field of intelligent transportation systems and motivates the application of AI-driven methods in this real-world domain.

Acknowledgments

The work presented in this paper is part of the Brabant In-Car III program, financially supported by the Cityregion Eindhoven (SRE), the province of North Brabant and the Ministry of Infrastructure and the Environment in the Netherlands. We would like to thank Traxpert, Locatienet, Tessa Bouw Communicatie and Pieter Loof for their contributions to this research project.

References

Abdoos, M., Mozayani, N., Bazzan, A.L.C., 2013. Holonic multi-agent system for traffic signals control. Eng. Appl. Artif. Intell. 26 (5–6), 1575–1587.

Bai, H., Cai, S., Ye, N., Hsu, D., Lee, W.S., 2015. Intention-aware online POMDP planning for autonomous driving in a crowd. In: IEEE International Conference on Robotics and Automation, pp. 454–460.

Behrisch, M., Bieker, L., Erdmann, J., Krajzewicz, D., 2011. SUMO—Simulation of Urban MObility. In: The 3rd International Conference on Advances in System Simulation, pp. 55–60.

Boutilier, C., 1996. Planning, learning and coordination in multiagent decision processes. In: Proceedings of the 6th Conference on Theoretical Aspects of Rationality and Knowledge, pp. 195–210.

Bryson, A.E., Ho, Y.-C., 1975. Applied Optimal Control: Optimization, Estimation, and Control. Taylor & Francis, Abingdon, United Kingdom.

Cruciol, L.L., de Arruda, A.C., Weigang, L., Li, L., Crespo, A.M., 2013. Reward functions for learning to control in air traffic flow management. Transp. Res. Part C: Emerg. Technol. 35, 141–155.

Davarynejad, M., Hegyi, A., Vrancken, J., Van den Berg, J., 2011. Motorway ramp-metering control with queuing consideration using *Q*-learning. In: IEEE Conference Proceedings on Intelligent Transportation Systems, pp. 1652–1658.

Fares, A., Gomaa, W., 2014. Freeway ramp-metering control based on reinforcement learning. In: 11th IEEE International Conference on Control & Automation, pp. 1226–1231.

Fares, A., Gomaa, W., 2015. Multi-agent reinforcement learning control for ramp metering. In: Selvaraj, H., Zydek, D., Chmaj, G. (Eds.), Proceedings of the 23rd International Conference on Systems Engineering, pp. 167–173.

Figueiredo, L., Jesus, I., Tenreiro Machado, J.A., Rui Ferreira, J., Martins de Carvalho, J. L., 2001. Towards the development of intelligent transportation systems. Intell. Transp. Syst. 88, 1206–1211.

Haykin, S., 1999. Neural Networks: A Comprehensive Foundation. Prentice Hall, Upper Saddle River, NJ.

Jennings, N.R., Wooldridge, M., 1998. Applications of Intelligent Agents. Agent Technology. Springer, Springer-Verlag, Berlin, Heidelberg, pp. 3–28.
Karaaslan, U., Varaiya, P., Walrand, J., 1990. Two Proposals to Improve Freeway Traffic Flow. Technical Report. Institute of Transportation Studies, UC Berkeley.
Khamis, M.A., Gomaa, W., 2012. Enhanced multiagent multi-objective reinforcement learning for urban traffic light control. In: 11th International Conference on Machine Learning and Applications, pp. 586–591.
Khamis, M.A., Gomaa, W., 2014. Adaptive multi-objective reinforcement learning with hybrid exploration for traffic signal control based on cooperative multi-agent framework. Eng. Appl. Artif. Intell. 29, 134–151.
Kotsialos, A., Papageorgiou, M., Diakaki, C., Pavlis, Y., Middelham, F., 2002. Traffic flow modeling of large-scale motorway networks using the macroscopic modeling tool metanet. IEEE Trans. Intell. Transp. Syst. 3 (4), 282–292.
Kuyer, L., Whiteson, S., Bakker, B., Vlassis, N., 2008. Multiagent reinforcement learning for urban traffic control using coordination graphs. In: Daelemans, W., Morik, K. (Eds.), Machine Learning and Knowledge Discovery in Databases. Springer, Springer-Verlag, Berlin Heidelberg, pp. 656–671.
Kwon, E., Stephanedes, Y.J., 1994. Comparative evaluation of adaptive and neural-network exit demand prediction for freeway control. Transp. Res. Rec. (1446)
Lu, W., Wang, H., Wang, Q., 2007. A synchronous detection of the road boundary and lane marking for intelligent vehicles. In: 8th ACIS International Conference on Software Engineering, Artificial Intelligence, Networking, and Parallel/Distributed Computing, pp. 741–745.
Messner, A., Papageorgiou, M., 1990. METANET: a macroscopic simulation program for motorway networks. Traffic Eng. Control 31 (8–9), 466–470.
Ngoduy, D., Hoogendoorn, S., van Zuylen, J., 2003. An automated calibration procedure for macroscopic traffic flow models. In: Proceedings of the 10th IFAC Symposium on Control in Transportation Systems, pp. 295–300.
Papageorgiou, M., 1983. Applications of Automatic Control Concepts to Traffic Flow Modeling and Control. Springer-Verlag Inc., New York.
Papageorgiou, M., Blosseville, J.-M., Hadj-Salem, H., 1990. Modelling and real-time control of traffic flow on the southern part of boulevard peripherique in Paris. Part I: modelling. Transp. Res. Part A: Gen. 24 (5), 345–359.
Papageorgiou, M., Diakaki, C., Dinopoulou, V., Kotsialos, A., Wang, Y., 2003. Review of Road Traffic Control Strategies. Proceedings of the IEEE 91 (12), 2043–2067.
Papageorgiou, M., Kosmatopoulos, E., Papamichail, I., 2008. Effects of variable speed limits on motorway traffic flow. Transp. Res. Rec. 2047 (1), 37–48.
Puterman, M.L., 1994. Markov Decision Processes: Discrete Stochastic Dynamic Programming. John Wiley & Sons Inc., New York, NY.
Rezaee, K., Abdulhai, B., Abdelgawad, H., 2012. Application of reinforcement learning with continuous state space to ramp metering in real-world conditions. In: 15th International IEEE Conference on Intelligent Transportation Systems, pp. 1590–1595.
Scardua, L.A., Da Cruz, J.J., Costa, A.H.R., 2002. Optimal control of ship unloaders using reinforcement learning. Adv. Eng. Inf. 16 (3), 217–227.
Schrank, D., Eisele, B., Lomax, T., 2012. TTI's 2012 Urban Mobility Report. Technical Report. Texas A&M Transportation Institute.
Smith, B.L., Demetsky, M.J., 1994. Short-term traffic flow prediction: neural network approach. Transp. Res. Rec. 1453, 98–104.
Spall, J., Chin, D., 1994. A model-free approach to optimal signal light timing for system-wide traffic control. In: Proceedings of the 33rd Conference on Decision and Control, pp. 1868–1875.
Sutton, R.S., Barto, A.G., 1998. Reinforcement Learning: An Introduction. MIT Press, Cambridge, Massachusetts.
Tumer, K., Agogino, A., 2007. Distributed agent-based air traffic management. In: Proceedings of the 6th International Joint Conference on Autonomous Agents and Multiagent Systems, pp. 342–349.
Van den Berg, M., De Schutter, B., Hegyi, A., Hellendoorn, J., 2004. Model predictive control for mixed urban and freeway networks. In: Proceedings of the 83rd Annual Meeting of the Transportation Research Board, p. 19.
Van den Berg, M., Hegyi, A., De Schutter, B., Hellendoorn, H., 2007. Integrated traffic control for mixed urban and freeway networks: a model predictive control approach. Eur. J. Transp. Infrastruct. Res. 7 (3), 223–250.
Watkins, C.J.C.H., 1989. Learning from Delayed Rewards [Ph.D. thesis]. King's College, Cambridge, UK.
Wei, W., Zhang, Y., 2002. FL-FN based traffic signal control. In: Proceedings of the 2002 IEEE International Conference on Fuzzy Systems, pp. 296–300.
Whitehead, S.D., Lin, L.-J., 1995. Reinforcement learning of non-Markov decision processes. Artif. Intell. 73 (1–2), 271–306.
Zhang, J.Z.J., Chang, H.C.H., Ioannou, P., 2006. A simple roadway control system for freeway traffic. In: Proceedings of the American Control Conference, pp. 4900–4905.
Zhao, D., Member, S., Dai, Y., Zhang, Z., 2012. Computational intelligence in urban traffic signal control: a survey. IEEE Trans. Syst. Man Cybern. 42 (4), 485–494.
Zolfpour-Arokhlo, M., Selamat, A., Mohd Hashim, S.Z., Afkhami, H., 2014. Modeling of route planning system based on Q value-based dynamic programming with multi-agent reinforcement learning algorithms. Eng. Appl. Artif. Intell. 29, 163–177.