Online Learning in Autonomic Multi-Hop Wireless Networks for Transmitting Mission-Critical Applications

Hsien-Po Shiang and Mihaela van der Schaar, *Fellow, IEEE*

***Abstract*—In this paper, we study how to optimize the transmission decisions of nodes aimed at supporting mission-critical applications, such as surveillance, security monitoring, and military operations, etc. We focus on a network scenario where multiple source nodes transmit simultaneously mission-critical data through relay nodes to one or multiple destinations in multi-hop wireless Mission-Critical Networks (MCN). In such a network, the wireless nodes can be modeled as agents that can acquire local information from their neighbors and, based on this available information, can make timely transmission decisions to minimize the end-to-end delays of the mission-critical applications. Importantly, the MCN needs to cope in practice with the time-varying network dynamics. Hence, the agents need to make transmission decisions by considering not only the current network status, but also how the network status evolves over time, and how this is influenced by the actions taken by the nodes. We formulate the agents' autonomic decision making problem as a Markov decision process (MDP) and construct a distributed MDP framework, which takes into consideration the informationally-decentralized nature of the multi-hop MCN. We further propose an online model-based reinforcement learning approach for agents to solve the distributed MDP at runtime, by modeling the network dynamics using priority queuing. We compare the proposed model-based reinforcement learning approach with other model-free reinforcement learning approaches in the MCN. The results show that the proposed model-based reinforcement learning approach for mission-critical applications not only outperforms myopic approaches without learning capability, but also outperforms conventional model-free reinforcement learning approaches.**

***Index Terms*—multi-user mission-critical transmission, autonomic multi-hop wireless networks, distributed Markov decision process, online reinforcement learning.**

I. Introduction

A PLETHORA of mission-critical applications such as battlefield videoconferencing, surveillance and security monitoring are emerging, e.g. in SOSANETs [1], where real-time response and actions to the acquired critical data becomes vital. This critical data needs to be reliably and timely relayed to one or multiple decision makers, possibly located at different destinations. To connect the various sources to the destinations, a rapidly deployable solution can be provided using multi-hop autonomic wireless networks. A key advantage of such flexible infrastructures is that the same network can be re-used and reconfigured to relay critical data to multiple destinations. The mission-critical applications require the network to support various transmission priorities, security, robustness requirements, and stringent transmission delay deadlines [6][8]. In this paper, we focus on minimizing the network delays of the mission-critical applications, and rely on related work (such as [8][9]) for the security and reliability requirements of the mission-critical applications.

Autonomic wireless networks are composed of autonomic wireless nodes (also interchangeably referred to as agents in this paper) endowed with the capability of individually sensing the network environment, learning the dynamic network changes based on their local information, and promptly adapting their transmission actions in an autonomous manner to optimize the utility of the applications which they are serving [10]. The dynamic network changes include variations in network topology, wireless channel conditions, application requirements, etc. When these network dynamics occur, the autonomic nodes can self-configure themselves and immediately react to these changes, without the need of propagating messages back and forth to a centralized coordinator. Autonomic wireless networks are especially suitable for mission-critical applications, since the autonomic behavior allows the wireless nodes to promptly discover local network changes and instantaneously react to these changes, such that the important data packets they are relaying will arrive at their destinations within their delay deadlines. Moreover, autonomic wireless nodes endowed with online learning capabilities can successfully model the network dynamics and foresightedly adapt their packet transmission to maximize the utility of the mission-critical applications.

In the MCN, the autonomic nodes need to coordinate their transmission decisions [7]. For example, in [25], it is shown that the performance degradation is unavoidable if the agents do not optimize their routing decisions in a cooperative manner. In [26][27], the Network Utility Maximization (NUM) framework is introduced and it is shown that by allowing agents to cooperatively exchange information, they can optimize their transmission actions in a distributed manner, such that a Pareto-efficient solution can be reached. However, such solutions assume a static network setting and they cannot address the dynamic nature of the MCN. Dynamic transmission policies based on local information feedback are proposed (for example, based on QoS state information

Manuscript received 4 April 2009; revised 22 December 2009. This work was funded by an ONR grant and by NSF CAREER CCF 0541867 grant.

Hsien-Po Shiang and Mihaela van der Schaar are with the Department of Electrical Engineering, UCLA (e-mail: hpshiang@ucla.edu, mihaela@ee.ucla.edu).

Digital Object Identifier 10.1109/JSAC.2010.100610.

[2] and queuing backpressure [4][5]), which ensure that the delays of the mission-critical applications are bounded as long as the rate allocations are inside the capacity region of the network. However, computing the capacity region requires a high computational complexity [32] and, moreover, does not guarantee that the required delay constraints of the mission-critical applications are met. In [3], a QoS-aware protocol with priority-based queuing model was proposed to support real-time traffic in wireless sensor networks. The protocol allocates energy-efficient paths to the applications that meet their end-to-end delay requirements. Also, other alternative QoS-aware solutions can be found in [11] for supporting various applications in wireless sensor networks. However, most of these solutions are mainly concerned with minimizing the energy consumption.

Importantly, in the distributed setting, an agent's decision impacts and is impacted by the decisions of the neighboring agents. We refer to this coupling effect as the *spatial dependency* among the agents. Although the abovementioned solutions consider the spatial dependency, they only react to the network changes in a "myopic" way. They merely optimize the transmission decisions based only on the information about the current network status and application requirements. In the dynamic MCN, however, the agents need to adopt "foresighted" adaptation by considering not only the immediate network status, but also how the network status evolves over time (referred to as the network dynamics in this paper), in order to make optimal transmission decisions. Hence, in addition to the spatial dependency, agents need also to consider the temporal dependency among their sequential decisions (performed over time).

Moreover, in practice, the network dynamics may not be known. Reinforcement learning solutions have been proposed for the nodes to learn the network dynamics and optimize the performance in routing [30] and admission control [31] solutions at runtime. However, these solutions do not minimize the delays of the mission-critical applications. Moreover, the majority of these solutions focus on model-free reinforcement learning approaches, which are not suitable for the mission-critical applications due to their slow convergence rates [15].

In summary, there is no integrated framework that considers the spatio-temporal dependencies among the agents in the MCN to minimize the end-to-end delays of the mission-critical applications, based on application priorities, packet-based delay deadlines, and the network dynamics. In this paper, we provide a systematic framework based on which agents (the nodes in the MCN) can optimize their cross-layer transmission actions and minimize the delays of the mission-critical applications, while considering the spatio-temporal dependencies among their actions. We assume that all the source and relay nodes are able to make their own cross-layer transmission decisions, which are the packet-based scheduling decisions in the application layer and the routing decisions in the network layer. In [28], it has been shown that Markovian models (e.g. finite-state markov model [29]) can be applied for both traffic state transition and channel state transition. Also in [30], it was shown that routing protocols in mobile ad hoc networks can be further improved by allowing the agents to make their decisions using Markov Decision Process (MDP) [18]. Based on the MDP, the agents are able to forecast the future network status and optimize their cross-layer transmission actions that consider the MCN dynamics. However, unlike in [30], which focuses on optimizing the overall throughput of the network, in this paper, the agents minimize the expected end-to-end delays of the mission-critical applications. The expected end-to-end delay is referred to in this paper as the *MDP delay value*.

Overall, the paper makes the following contributions:

1) Distributed MDP framework that considers the spatio-temporal dependencies in MCN. To account for the dynamic nature of the MCN, we construct an MDP framework which minimizes the MDP delay values of the mission-critical applications. To address the informationally-decentralized nature of the multi-hop MCN, the MDP needs to be formulated in a distributed manner, such that each agent in the MCN can deploy its own cross-layer transmission policy based on only local information exchanges with its neighboring agents. The proposed distributed MDP minimizes the delays of the mission-critical applications while capturing the spatio-temporal dependency in the MCN.

2) Model-based online learning approach to solve the distributed MDP in MCN. We propose an online model-based learning approach for the agents in MCN to solve the distributed MDP at runtime, when the network dynamics are unknown. Unlike the conventional model-free reinforcement learning approaches for solving MDPs (as in [16][17]), the proposed model-based learning algorithm adopts a preemptive-repeat priority M/G/1 queuing model [21], which enables a faster convergence rate and shorter delays for the mission-critical applications. The upper and lower bounds of the resulting MDP delay value are provided to verify the accuracy of the proposed model-based online learning approach at different network locations. Moreover, we compare the proposed model-based reinforcement learning approach with the model-free reinforcement learning approaches in terms of delay performance, computational complexity, and the required information exchange overheads.

This paper is organized as follows. In Section II, we discuss the network settings and the cross-layer transmission actions of the autonomic wireless nodes, and formulate the autonomic decision making problem in the MCN. In Section III, we discuss the distributed MDP framework that addresses both the dynamic and information-decentralized nature of the MCN. In Section IV, we propose a model-based online learning approach for the autonomic wireless nodes to solve the distributed MDP at runtime, which is suitable for the mission-critical applications. Section V provides simulation results and Section VI concludes the paper.

II. Autonomic Decision Making Problem Formulation in MCN

A. Mission-critical application characteristics

Unlike most cross-layer design papers that consider only a single application, we assume that there are multiple sources transmitting simultaneously delay-critical information over the MCN. Let $\mathbf{V} = \{V_i\}$ represent the set of the mission-critical applications. We assume that the packets of an application

V_i are prioritized into K_i priority classes. The total number of the priority classes in the network is $K = \sum_{i=1}^{|\mathbf{V}|} K_i$. Let $\{C_k, k = 1, ..., K\}$ represent all the priority classes in the network. In the subsequent part of the paper, we label the K classes (across all applications) in descending order of their priorities, i.e. C_1 is the highest priority class. A priority class C_k is characterized by the following parameters $\{D_k, R_k, L_k\}$[1]. D_k represents the delay deadline of the packets in class C_k. A packet of a mission-critical application is useful only if it is received at the destination before its delay deadline. R_k is the average source rate of the packets in class C_k. Based on the source rate, the source node generates a certain number of packets per unit time, which impacts the traffic load of the MCN. L_k is the average packet length of the packets in class C_k, which directly impacts the packet error rate and the transmission rate of sending a class C_k packet. Let $Delay_k$ represent the end-to-end delay that is required for the transmission of the traffic in class C_k. These required delays are mandated by the mission and the deployed applications, and the MCN agents need to prioritize the traffic and minimize their end-to-end delays according to the assigned priorities [8]. For example, in a battlefield mission-critical network, instructions from a command center are mission-critical and should have higher priority than any other traffic, e.g. response notification, surveillance results, etc.

B. Multi-hop MCN settings

The MCN is represented by a network graph $\mathcal{G}(\mathbf{V}, \mathbf{M}, \mathbf{E})$, where $\mathbf{M} = \{m_1, ..., m_{|\mathbf{M}|}\}$ represents the set of agents and $\mathbf{E} = \{e_1, ..., e_{|\mathbf{E}|}\}$ represents a set of edges (transmission links) that connect the various agents. There are two types of agents defined in this paper:

1) Autonomic Source Agents (ASs). Each AS generates a mission-critical application and would like to transmit the application to a predetermined destination node.

2) Autonomic Relay Agents (ARs). ARs relay the packets from the AS to the corresponding destination node. Unlike the ASs, the ARs do not generate their own traffic. They make their cross-layer transmission decisions and forward the packets for the ASs.

To enable us to better discuss the various networking solutions, we label the agents using a directed acyclic graph [13] as shown in Figure 1, which consists of H hops from the ASs to the destination nodes[2]. We assume that M_h is the number of agents at the h-th hop ($0 \leq h \leq H-1$), and $M_0 = M_H = |\mathbf{V}|$. Each agent at the h-th hop will be tagged with a distinct number $m_h (1 \leq m_h \leq M_h)$. Let $\mathbf{M}_h \subseteq \mathbf{M}$ represent the set of agents at the h-th hop. The agent m_h processes a priority queue and it can only transmit the packets in the queue to a subset of ARs in $\mathbf{M_{h+1}}$. Through periodic information exchange (e.g. hello message exchange in [24]), we assume that each agent m_h knows the existence of its neighboring nodes (i.e. the other agents $m'_h \in \mathbf{M}_h$ in the same hop and the agents $m_{h+1} \in \mathbf{M}_{h+1}$ in the next hop), as well as the interference matrix [21] of the current hop that defines whether or not two different links of neighboring nodes can transmit simultaneously.

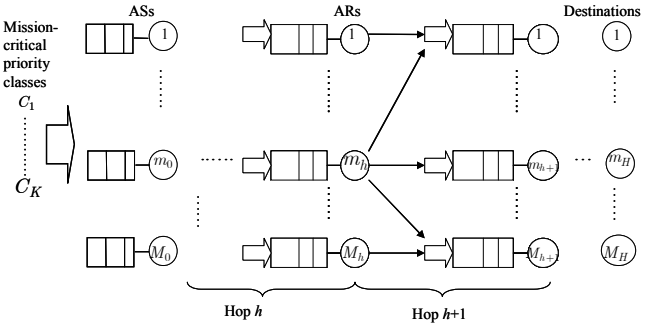


Fig. 1. Considered multi-hop wireless network [10]

C. Effective transmission rate over the multi-hop MCN

We denote the maximum transmission rate over the link (m_h, m_{h+1}) as $T_{k,m_h,m_{h+1}}$ for traffic class C_k. Assuming a memory-less packet erasure channel as in [20][21], and given the Signal-to-Interference-Noise-Ratio (SINR) $x_{m_h,m_{h+1}}$, we can compute the packet error rate $p_{k,m_h,m_{h+1}}(x_{m_h,m_{h+1}})$ over the link. If the agent m_h selects m_{h+1} as its next relay, the effective transmission rate (goodput) can be approximated using the sigmoid function [20]:

$$\begin{aligned} &T^{goodput}_{k,m_h,m_{h+1}}(x_{m_h,m_{h+1}}) \\ &= T_{k,m_h,m_{h+1}}(1 - p_{k,m_h,m_{h+1}}(x_{m_h,m_{h+1}})), \\ &p_{k,m_h,m_{h+1}}(x_{m_h,m_{h+1}}) = \frac{1}{1+e^{\zeta(x_{m_h,m_{h+1}} - \delta)}}, \end{aligned} \quad (1)$$

where ζ and δ are constants corresponding to the modulation and coding schemes for a given packet length L_k. This goodput is determined by the actions of the agent m_h, which influences the delay of the applications (see Section III.A for more details).

D. Actions of the autonomic wireless nodes

An agent's *cross-layer transmission action* varies when transmitting different priority class traffic. Denote $A_{m_h} = \{A_{k,m_h}, \forall C_k\}$ as the cross-layer transmission action of agent m_h, where $A_{k,m_h} = \{\pi_{k,m_h}, \beta_{k,m_h,m_{h+1}}, m_{h+1} \in \mathbf{M}_{h+1}\} \in \mathcal{A}_{m_h}$ represents the action of agent m_h when sending packets in class C_k. $\mathcal{A}_{m_h}$ represents the set of feasible actions for the agent m_h. In this paper, we assume that the cross-layer transmission action includes the application layer packet scheduling π_{k,m_h} of transmitting packets in class C_k, and the network layer relay selecting parameter $\beta_{k,m_h,m_{h+1}}$, which determines the probability of selecting a node $m_{h+1} \in \mathbf{M}_{h+1}$ in the next hop as the next relay. Denote $\mathbf{A} = \{A_{m_h}, \forall m_h \in \mathbf{M}\}$ as the actions of all the agents in the MCN. Note that the delay $Delay_k(\mathbf{A})$ of packets in class C_k is a function of all agents' actions.

[1]We refer the interested readers to our previous work [21] for more details on these parameters.

[2]Note that such a directed acyclic network can be deployed over any physical network topologies as an overlay network (see [13] for more details about how to deploy the directed acyclic graph over a multi-hop wireless network).

E. Problem formulation

In this subsection, we discuss several ways to determine the cross-layer transmission decisions for transmitting the mission-critical applications over the MCN.

- Centralized decision making

The majority of the cross-layer design papers assume a centralized optimization, in which a central controller collects global network information $\mathcal{G}$ and make transmission decisions for all the agents in the MCN. Since minimizing end-to-end delay is the key objective in the MCN, the centralized optimization needs to minimize the end-to-end delays for the various applications [12][13]. An advantage of such delay-driven approach is that the optimization only needs to be done for the higher priority classes, and the packets of the lower priority classes can be simply dropped if their delay constraints cannot be met[3]. Let $a_k = [A_{k,m_h}, \forall m_h \in \mathbf{M}]$ represent the actions of all the agents sending traffic class C_k. The actions for transmitting the priority class C_k can be computed after the actions for the higher priority classes $\{a_1, ..., a_{k-1}\}$ are determined and the action a_k will not affect any of the actions for $\{a_1, ..., a_{k-1}\}$. Specifically, the following delay constrained optimization is considered for the priority class C_k:

$$\begin{aligned} a_k^{opt} &= \arg\min_{a_k} Delay_k(a_k, \{a_1, ..., a_{k-1}\}, \mathcal{G}) \\ \text{s.t.} &Delay_k(a_k, \{a_1, ..., a_{k-1}\}, \mathcal{G}) \le D_k \end{aligned} \quad (2)$$

However, in mission-critical applications, which have stringent delay deadlines, it is impractical to assume that the global information $\mathcal{G}$ can be gathered in time at a central controller. Hence, it is important to decompose the optimization in equation (2) in such a way that each agent m_h can make timely decision based on local information $\mathcal{L}_{m_h}$.

- Distributed decision making for the agent m_h

Let $E[Delay_{k,m_h}(A_{k,m_h}, \mathcal{L}_{m_h})]$ represent the expected delay from m_h to the destination node of the traffic class C_k, which is a function of the transmission action A_{k,m_h} and its local information $\mathcal{L}_{m_h}$. Let $Delay_{k,m_h}^{PASS}$ represent the delay that has already passed when the class C_k packet arrives at the agent m_h. This can be computed based on the information that is encapsulated in the packet header. Since agent m_h cannot influence $Delay_{k,m_h}^{PASS}$, it can only minimize the delay for the highest priority class C_k in its queue using the following optimization [21]:

$$\begin{aligned} A_{k,m_h}^{opt}(\mathcal{L}_{m_h}) &= \arg\min_{A_{k,m_h}} E[Delay_{k,m_h}(A_{k,m_h}, \mathcal{L}_{m_h})] \\ \text{s.t.} &Delay_{k,m_h}(A_{k,m_h}, \mathcal{L}_{m_h}) \le D_k - Delay_{k,m_h}^{PASS} \end{aligned} \quad (3)$$

Figure 2(a) illustrates this conventional distributed decision making. First, the agent evaluates the utility (i.e. the expected delay $E[Delay_{k,m_h}(A_{k,m_h}, \mathcal{L}_{m_h})]$), which it can obtain from taking various actions based on the local information $\mathcal{L}_{m_h}$. Then, the agent determines its transmission action by solving the optimization in equation (3). The required local information $\mathcal{L}_{m_h}$ for computing $E[Delay_{k,m_h}(A_{k,m_h}, \mathcal{L}_{m_h})]$ will be discussed later in Section III.B.

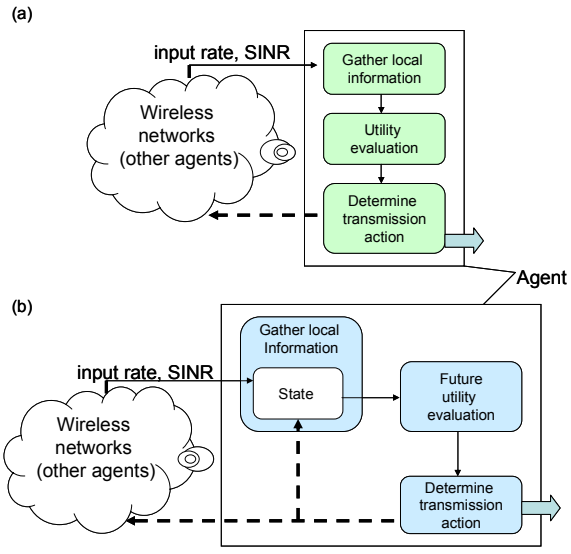


Fig. 2. (a) Conventional distributed decision making of an agent.(b) Proposed foresighted decision making of an agent.

However, due to the dynamic nature of the MCN, the gathered local information is changing over time. Hence, it is important for the agents to consider not only the current expected delay, but also the future expected delay as the network dynamics evolve. Figure 2(b) illustrates how an agent anticipates the evolution of the network dynamics by considering the impact of its current transmission action on the future network state (which will be defined in Section III.A), and based on it, makes foresighted transmission decisions to transmit mission-critical applications. Next, we formulate this foresighted decision making of an agent in the MCN.

- Proposed foresighted decision making for the agent m_h

Assume $E[Delay_{k,m_h}^{t_0}]$ as the expected delay of agent m_h at current service interval t_0. Given the current local information $\mathcal{L}_{m_h}^{t_0}$, agent m_h makes foresighted decisions by taking into account the impact of its actions not only on the current expected delay, but also on the discounted expected delays in the future service intervals, i.e.

$$\begin{aligned} &\mu_{k,m_h}(\mathcal{L}_{m_h}^{t_0}) = \\ &\arg\min_{A_{k,m_h}} \left\{ \sum_{t=t_0}^{\infty} \gamma^{t-t_0} E[Delay_{k,m_h}^{t}(A_{k,m_h}, \mathcal{L}_{m_h}^{t})] \right\} \end{aligned} \quad (4)$$

where $0 < \gamma < 1$[4] represents the discount factor to decrease the utility impact of the later transmitted packets. If the discount factor $\gamma = 0$, the optimization in equation (4) becomes a myopic decision making, similar to the one in [21]. We refer to the function $\mu_{k,m_h}(\mathcal{L}_{m_h})$ as the cross-layer transmission policy given the local information $\mathcal{L}_{m_h}$. In the

[3]The action $A_{k,m_h} = \{\beta_{k,m_h,m_{h+1}}, m_{h+1} \in \mathbf{M}_{h+1}\}$ hereafter does not include the application layer scheduling, since the highest priority packet is selected to be transmitted. To simplify the notation, we use the same notation for the cross-layer transmission actions and assume that the class C_k is the highest priority class existing in the queue of the agent m_h when taking the action A_{k,m_h}.

[4]$1-\gamma$ can be regarded as the probability that the priority class ends in a certain service interval. Note that different discount factors γ_k can be considered for different priority classes. However, to simplify the exposition, we consider here the same γ for all priority classes.

next section, we will discuss how to compute this cross-layer transmission policy.

III. Distributed Markov Decision Process Framework

In this section, we discuss how to systematically compute the cross-layer transmission policy $\mu_{k,m_h}(\mathcal{L}_{m_h})$ for the agents in the MCN. First, we define the state of the agents in Section III.A. Then, in Section III.B, we propose the distributed MDP which allows all the agents to make their own decisions.

A. States of the autonomic wireless nodes

We define the network state at agent m_h as $s_{m_h} = \{[\eta_{k,m_h}, \forall C_k], [x_{m_h,m_{h+1}}, \forall (m_h, m_{h+1})]\} \in \mathcal{X}_{m_h}$, where $x_{m_h,m_{h+1}}$ represents the channel condition (see Section II.C) and η_{k,m_h} represents the arrival rate of the class C_k packets at agent m_h. To evaluate the expected delay $E[Delay_{k,m_h}]$, agent m_h needs to first compute the expected queuing delay $E[W_{k,m_h}]$ for which the packets in class will be queued at m_h. The state includes sufficient statistics for computing the expected queuing delay $E[W_{k,m_h}]$, when an action A_{k,m_h} is taken. Note that the first two moments of the service rate can be obtained as: and

$$\begin{aligned} E[X_{k,m_h}] &= \frac{L_k}{T_{k,m_h,m_{h+1}}(1-p_{k,m_h,m_{h+1}}(x_{m_h,m_{h+1}}))} \\ E[X^2_{k,m_h}] &= \frac{L_k^2(1+p_{k,m_h,m_{h+1}}(x_{m_h,m_{h+1}}))}{T^2_{k,m_h,m_{h+1}}(1-p_{k,m_h,m_{h+1}}(x_{m_h,m_{h+1}}))^2} \end{aligned} \quad (5)$$

Together with the arrival rate η_{k,m_h}, the expected queuing delay $E[W_{k,m_h}]$ can be computed using an priority M/G/1 queuing model [21]. We assume that each agent will feed back its expected delays to all the agents in the previous hop (similar to DSDV protocols [24]). Hence, the agent m_h is able to select the next relay that minimizes the sum of current queuing delay and the expected delay from the next hop to the destination node of class C_k, i.e.

$$\begin{aligned} &E[Delay_{k,m_h}(A_{k,m_h}, s_{m_h})] \\ &= \sum_{h'=h}^{H-1} E[W_{k,m_{h'}}(A_{k,m_{h'}}, s_{m_{h'}})] \\ &= E[W_{k,m_h}(A_{k,m_h}, s_{m_h})] + E[Delay_{k,m_{h+1}}(A_{k,m_{h+1}})] \end{aligned} \quad (6)$$

Importantly, the agent m_h's transmission action will impact the information feedback $E[Delay_{k,m_{h+1}}]$, since it will select the next relay $m_{h+1} \in \mathbf{M}_{h+1}$ that feeds back different expected delay values. Moreover, the expected delay $E[Delay_{k,m_h}]$ will be fed back to the agents in the previous hop and hence impact their transmission actions. Hence, the agent m_h's action A_{k,m_h} will affect its own future state s_{m_h} and also will influence the future expected delay as the network dynamics evolve. As in [30], we denote the probability that the agent m_h has a state $s^{t+1}_{m_h}$ in service interval $t+1$ as $p(s^{t+1}_{m_h})$, which is modeled as a function of agent m_h's current state $s^t_{m_h}$ and current action A^t_{k,m_h}, i.e.

$$p(s^{t+1}_{m_h}) \cong \hat{\mathcal{F}}_{s^{t+1}_{m_h}}(s^t_{m_h}, A^t_{k,m_h}) \quad (7)$$

Note that the real $p(s^{t+1}_{m_h})$ can be very complicated in a real network, since it is impacted by the decisions of all the agents in the previous hop as well as the interference among the agents in the current hop. Note that in our solution, the agents do not need to know the exact form of $p(s^{t+1}_{m_h})$. Online learning approaches will be discussed in Section IV for the agents to learn the state transition function in equation (7). Next, we formulate the cross-layer optimization of the agent as an MDP for *each* class.

B. Distributed MDP for class C_k

For class C_k, the MDP at the agent m_h is defined by a tuple $\langle \mathcal{X}_{m_h}, \mathcal{A}_{m_h}, \mathcal{I}_{m_h}, \mathcal{T}_{m_h}, \mathcal{U}_{m_h}, \gamma \rangle$:

- States: Recall that the state is defined in Section III.A as $s_{m_h} = \{[\eta_{k,m_h}, \forall C_k], [x_{m_h,m_{h+1}}, \forall (m_h, m_{h+1})]\} \in \mathcal{X}_{m_h}$.

- Actions: Recall that the action is defined as $A_{k,m_h} = \{\beta_{k,m_h,m_{h+1}}, m_{h+1} \in \mathbf{M}_{h+1}\} \in \mathcal{A}_{m_h}$ in Section II.C. To simplify the notation, we will afterward use A_{m_h} instead of A_{k,m_h}.

- Information exchange: Let $\mathcal{I}_{m_h} = \{F^b_h, F^f_h\}$[5] represent the information exchange of the agents in the h-th hop to the previous hop and to the next hop. Denote $F^{b,t}_h(m_h) = E[Delay^t_{k,m_h}]$ as the feedback information from agent m_h to the agents in the previous hop (see equation (6)) and let $F^{b,t}_h = [F^{b,t}_h(m_h), m_h \in \mathbf{M}_h]$ represents the feedback information in the h-th hop in the service interval t. Denote $F^{f,t}_h(m_h) = \{Delay^{PASS}_{k,m_h}, \eta_{k,m_h}\}$ as the feedforward information from node to the selected relay in the next hop and let $F^{f,t}_h = [F^{f,t}_h(m_h), m_h \in \mathbf{M}_h]$ represent the feedforward information in the h-th hop. Given the feedforward information $F^{f,t}_{h-1}$, the agent m_h computes the average delay $Delay^{PASS}_{k,h-1}$ of passing through the previous hops as:

$$Delay^{PASS}_{k,h-1} = \sum\nolimits_{m_{h-1}=1}^{M_{h-1}} \frac{\eta_{k,m_{h-1}}}{R_k} Delay^{PASS}_{k,m_{h-1}} \quad (8)$$

If $Delay^{PASS}_{k,h-1}$ exceeds the delay deadline D_k, the packet in class C_k should be dropped and no MDP is needed for traffic class C_k at the agent m_h.

- State transition probabilities: Let $T_{s_{m_h} s'_{m_h}}(A_{m_h}) \in \mathcal{T}_{m_h} : \mathcal{X}_{m_h} \times \mathcal{X}_{m_h} \times \mathcal{A}_{m_h} \rightarrow [0,1]$ represent the stationary state transition probabilities from state s_{m_h} to state s'_{m_h} when action A_{m_h} is taken. Based on the state transition models in equation (7), we compute the state transition probabilities as $T_{s_{m_h} s'_{m_h}}(A_{m_h}) \cong \hat{\mathcal{F}}_{s'_{m_h}}(s_{m_h}, A_{m_h})$.

- Cost: The expected delay $E[Delay_{k,m_h}(s_{m_h}, A_{m_h})] \in \mathcal{U}_{m_h}$ represents the cost function. As mentioned in Section III.A, we rely on a priority-based queuing model to compute the cost function (see equation (6)). Note that the expected delay of a higher priority class will not be influenced by the other lower priority classes. However, if the class is one of the lower priority classes, the influence of the higher priority classes is taken into account based on the priority-based queuing model [21] (given the actions and states associated with the higher priority classes).

- Discount factor: Recall that γ is the same discount factor as in equation (4).

Based on the information feedback F^b_{h+1}, we modify the

[5]The superscript b and the superscript f represent backwards and forwards information, respectively.

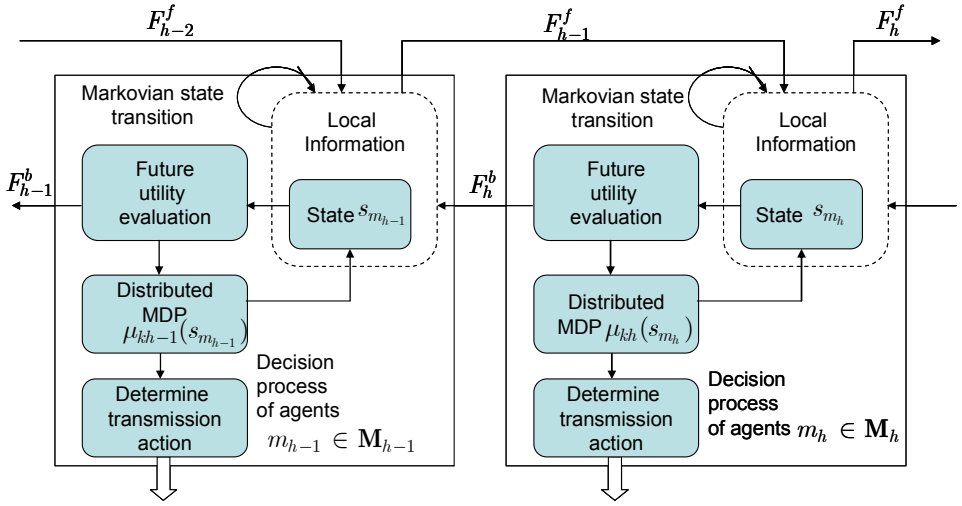


Fig. 3. Proposed decentralized MDP framework and the necessary information exchange among the agents

Bellman equation [19] of the MDP as:

$$\begin{aligned} & V^*_{k,m_h}(s_{m_h}, F^b_{h+1}) \\ &= \min_{A_{m_h}\in\mathcal{A}_{m_h}} \left\{ \sum_{t=1}^{\infty} \gamma^{t-1} E[Delay^t_{k,m_h}(s_{m_h}, A_{m_h})] \right\} \\ &= \min_{A_{m_h}\in\mathcal{A}_{m_h}} \left\{ \begin{array}{l} E[W_{k,m_h}(s_{m_h}, A_{m_h})] + F^b_{h+1}(A_{m_h}) + \\ \gamma \sum_{s_{m_h}'} T_{s_{m_h}s_{m_h}'}(A_{m_h}) V^*_{k,m_h}(s'_{m_h}, F^b_{h+1}) \end{array} \right\} \end{aligned} \quad (9)$$

where V^*_{k,m_h} is referred to as the *MDP delay value*, which is a discounted version of the long-term expected delay. To solve this feedback-modified Bellman equation, the agent m_h adopts value iteration [19] by updating the MDP delay value:

$$V^{t+1}_{k,m_h}(s_{m_h}, F^{b,t}_{h+1}) = \min_{A_{m_h}\in\mathcal{A}_h} Q^t_{k,m_h}(s_{m_h}, A_{m_h}, F^b_{h+1}), \quad (10)$$

where $Q^t_{k,m_h}(s_{m_h}, A_{m_h}, F^b_{h+1}) = E[W^t_{k,m_h}(s_{m_h}, A_{m_h})] + F^{b,t}_{h+1}(A_{m_h}) + \gamma \sum_{s_{m_h}'} T_{s_{m_h}s_{m_h}'}(A_{m_h}) V^t_{k,m_h}(s'_{m_h}, F^{b,t-1}_{h+1})$ is the Q-value at the agent m_h when a cross-layer transmission action A_{m_h} is taken in state s_{m_h}. The stationary policy can be written as: $\mu^t_{k,m_h}(\mathcal{L}^t_{m_h}) = \arg \min_{A_{m_h}\in\mathcal{A}_{m_h}} Q^t_{m_h}(s_{m_h}, A_{m_h}, F^{b,t}_{h+1})$.

The feedback-modified Bellman equation in equation (9) can be solved using value iteration, if the agent m_h has complete knowledge about $E[W_{k,m_h}(s_{m_h}, A_{m_h})]$ and $T_{s_{m_h}s_{m_h}'}(A_{m_h})$. Table I presents the detailed implementation of the distributed MDP and Figure 3 shows the considered system diagram of the distributed MDP that allows the agents to exchange information with the nodes in the neighboring hops.

IV. Online Model-Based Learning for Solving the Distributed MDP

In order to solve the Bellman equations, the agents need to know the state transition probabilities $T_{s_{m_h}s_{m_h}'}(A_{m_h})$ in the updating equation (10). However, the state transition probabilities may not be known to the agents a priori. In this section, we discuss online learning approaches for solving the distributed MDP introduced in the previous section at runtime. We propose a novel model-based reinforcement learning approach that is suitable for the agents to transmit mission-critical applications over the MCN. The proposed model-based reinforcement learning approach adopts the priority queuing model $E[W_{k,m_h}(s_{m_h}, A_{m_h})]$ for the cost and directly estimates the state transition probabilities $T_{s_{m_h}s_{m_h}'}(A_{m_h})$ to solve the distributed MDP. In Section IV.B, we show that the proposed model-based learning methods converge faster than the model-free learning approaches, since it takes less time for the autonomic node to explore different states and correctly evaluate the Q values.

A. Conventional model-free reinforcement learning

The model-free learning methods, e.g. Q-learning [16][17], can be applied at an agent m_h to learn the next Q values $[Q^{t+1}_{k,m_h}(s_{m_h}, A_{m_h}), \forall s_{m_h} \in \mathcal{X}_{m_h}]$ without characterizing the state transition probabilities $T_{s_{m_h}s_{m_h}'}(A_{m_h})$. Taking Q-learning as an example, given the feedback value $F^{b,t}_{h+1}$, the autonomic node m_h updates the Q-value using the following updating equation:

$$\begin{aligned} & Q^{t+1}_{k,m_h}(s_{m_h}, A_{m_h}) = (1-\rho_t) Q^t_{k,m_h}(s_{m_h}, A^t_{m_h}) + \\ & \rho_t \left\{ Cost^t_{k,m_h} + F^{b,t}_{h+1}(A^t_{m_h}) + \gamma \min_{A_{m_h}} Q^t_{k,m_h}(s^{t+1}_{m_h}, A_{m_h}) \right\} \end{aligned} \quad (11)$$

where $0 < \rho_t < 1$ represents the learning rate, and $\sum_t \rho_t = \infty$ and $\sum_t (\rho)^2 < \infty$ are ensured for the convergence of the Q-value [16]. The $Cost^t_{k,m_h}$ represents the delay measurement (e.g. by measuring the queue size) of sending packets in class C_k and $s^{t+1}_{m_h}$ represents the next state after the agent m_h takes the cross-layer transmission action $A^t_{m_h}$. For exploration purposes, instead of following the optimal stationary policy $\mu^t_{k,m_h}(s_{m_h}) = \arg \min_{A_{m_h}\in\mathcal{A}_{m_h}} Q^t_{k,m_h}(s_{m_h}, A_{m_h})$, the next action is selected according to a soft-min policy. Assume $\pi^t_{k,m_h}(s_{m_h}, A_{m_h})$ denotes the probability for agent m_h to take the action A_{m_h} given the state s_{m_h}. The soft-min policy $\mu^t_{k,m_h}(s_{m_h}) = [\pi^t_{k,m_h}(s_{m_h}, A_{m_h}), \forall A_{m_h} \in \mathcal{A}_{m_h}]$ is defined

TABLE I
IMPLEMENTATION OF THE DISTRIBUTED MDP

Step 1. Gather local information. From the information feedforward $F_{h-1}^{f,t}$ from the previous hop, the agent m_h computes $Delay_{k,h-1}^{PASS}$ and determine whether the distributed MDP should be performed for traffic class C_k. Then, gathers the local information $\mathcal{L}_{m_h}^t = \left\{ s_{m_h}^t, F_{h+1}^{b,t} \right\}$.
Step 2. Evaluate queuing delay and state transition probabilities. Based on state s_{m_h} and action A_{m_h} the agent m_h evaluates $E[W_{k,m_h}^t]$. The state transition probabilities are modeled using $T_{s_{m_h}s_{m_h}'}(A_{m_h}) = \hat{\mathcal{F}}_{s_{m_h}^{t+1}}(s_{m_h}^t, A_{m_h}^t)$ in equation (7).
Step 3. Update the transmission policy. The agent m_h updates the MDP delay value $V_{m_h}^{t+1}(s_{m_h}, F_{h+1}^{b,t})$ using equation (10). The stationary policy of the agent m_h is $\mu_{k,m_h}^t(\mathcal{L}_{m_h}^t) = \arg\min_{A_{m_h} \in \mathcal{A}_{m_h}} Q_{m_h}^t(s_{m_h}, A_{m_h}, F_{h+1}^{b,t})$.
Step 4. Update the information exchange. After the policy $\mu_{k,m_h}^t(\mathcal{L}_{m_h}^t)$ is determined, the next relay m_{h+1} is selected and m_h can then update the feedback information $F_h^{b,t+1}(m_h) = \sum_{m_{h+1} \in \mathbf{M}_{h+1}} \beta_{k,m_h,m_{h+1}} F_{h+1}^{b,t}(m_{h+1}) + E[W_{k,m_h}^t(s_{m_h}^t, \mu_{k,m_h}^t(\mathcal{L}_{m_h}^t))]$. The wireless node m_h also needs to update its feedforward information $F_h^{f,t+1}(m_h) = Delay_{k,h-1}^{PASS} + E[W_{k,m_h}^t(s_{m_h}^t, \mu_{k,m_h}^t(\mathcal{L}_{m_h}^t))]$.

using the Boltzmann distribution [14][15][16]:

$$\pi_{k,m_h}^t(s_{m_h}, A_{m_h}) = \frac{\exp(\frac{-Q_{k,m_h}^t(s_{m_h}, A_{m_h})}{\tau})}{\sum_{\forall A_{m_h} \in \mathcal{A}_{m_h}} \exp(\frac{-Q_{k,m_h}^t(s_{m_h}, A_{m_h})}{\tau})} \tag{12}$$

where τ is the temperature parameter. A small τ provides a greater probability difference in selecting different actions. If $\tau \to 0$, the approach reduces back to $\mu_{k,m_h}^t(s_{m_h}) = \arg\min_{A_{m_h} \in \mathcal{A}_{m_h}} Q_{k,m_h}^t(s_{m_h}, A_{m_h})$. On the other hand, a larger τ allows the agents to explore various actions with higher probabilities[6]. We provide detailed steps of the model-free reinforcement learning in Algorithm 1 in Table VI. Table II summarizes the required local information, memory complexity, and computational complexity of the model-free reinforcement learning approaches. In each service interval, the model-free reinforcement learning approaches need to update the Q-values of $\forall s_{m_h} \in \mathcal{X}_{m_h}, \forall C_k$, and for each state, $Q_{k,m_h}^t(s_{m_h}^{t+1}, A_{m_h})$ over $\forall A_{m_h} \in \mathcal{A}_{m_h}$ is calculated. Hence, the computational complexity is $O(|\mathcal{X}_{m_h}||\mathcal{A}_{m_h}|K)$. Note that the dynamics in the MCN may change before the updated policy converges when using a model-free learning approach. Hence, we consider alternative model-based reinforcement learning in the next subsection, which is more suitable for the agents in the MCN due to a faster convergence rate.

B. Proposed model-based reinforcement learning

In this section, we propose our model-based learning approach that enables the agent m_h to directly model the expected queuing delay $E[W_{k,m_h}(s_{m_h}, A_{m_h})]$ and estimate the state transition probabilities $\hat{T}_{s_{m_h}s_{m_h}'}(A_{m_h})$ to solve the Bellman equation through value iteration [19]. Figure 4 provides a system block diagram of the proposed online learning approach at the agent m_h. Our approach is similar to the Adaptive-RTDP in [14], where the state transition probabilities are determined using maximum-likelihood estimation. Specifically, let $\hat{T}_{s_{m_h}s_{m_h}'}(A_{m_h})$ denote the estimated state transition probability at , which is updated at each service interval. The Q-value is also updated as:

$$Q_{k,m_h}^{t+1}(s_{m_h}, A_{m_h}) = (1-\rho_t) Q_{k,m_h}^t(s_{m_h}, A_{m_h}) + \rho_t \left\{ \begin{array}{l} E[W_{k,m_h}^t(s_{m_h}, A_{m_h})] + F_{h+1}^{b,t}(A_{m_h}) + \\ \gamma \min_{A_{m_h} \in \mathcal{A}_h} \sum_{s_{m_h'}} \hat{T}_{s_{m_h}s_{m_h'}}^t(A_{m_h}) Q_{k,m_h}^t(s_{m_h'}, A_{m_h}) \end{array} \right\} \tag{13}$$

$s_{m_h'}$ represents the next state to which agent m_h transits, after it takes the cross-layer transmission action A_{m_h}. We provide the detailed steps of the proposed model-based reinforcement learning in Algorithm 2 in Table VII. The main differences between the model-based online learning approach and model-free learning approaches are the following:

1) We model the expected queuing delay $E[W_{k,m_h}(s_{m_h}, A_{m_h})]$ with an action realized from the policy μ_{k,m_h}^t using the preemptive-repeat priority M/G/1 queuing model as in [21]:

$$E[W_{k,m_h}(s_{m_h}, A_{m_h})] = \left\{ \begin{array}{l} \frac{\sum_{i=1}^{k} \eta_{i,m_h} E[X_{i,m_h}^2]}{2\left(1-\sum_{i=1}^{k-1} \eta_{i,m_h} E[X_{i,m_h}]\right)\left(1-\sum_{i=1}^{k} \eta_{i,m_h} E[X_{i,m_h}]\right)} \\ \text{, if} E[W_{k,m_h}] \le D_{k,m_h}^{rem} \\ \infty, \text{otherwise} \end{array} \right. \tag{14}$$

From equation (14), we know that if the queuing time exceeds the remaining delay deadline $D_{k,m_h}^{rem} = D_k - Delay_{k,h-1}^{PASS}$, the expected queuing time $E[W_{k,m_h}]$ becomes infinite, since the packets will be useless (no utility gain) and they will be dropped at the agent m_h. Unlike Q-learning that can only update one Q-value of a state-action pair at each service interval, with the priority queuing model, our model-based learning approach provides accurate estimation for any state-action pairs. Hence, the priority queuing model enables a faster learning capability, which is very important in order to satisfy

[6] τ provides an exploration and exploitation tradeoff between exploring different actions and exploiting the Q-values of taking an action. Such tradeoff is important in the MCN, since it significantly impacts the convergence rate and the performance of the learning approach.

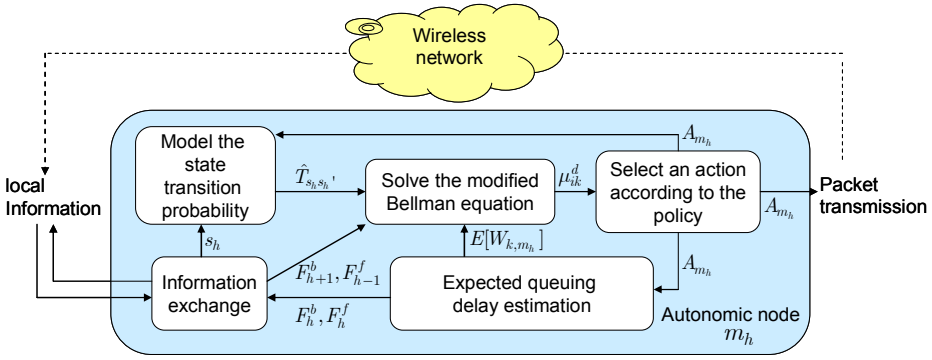


Fig. 4. System diagram of the proposed model-based online learning approach at the agent m_h

TABLE II
COMPLEXITY SUMMARY OF THE MODEL-FREE REINFORCEMENT LEARNING

Required local information	$\mathcal{L}^t_{m_h} = \{s^t_{m_h}, \{Cost^t_{k,m_h}, \forall C_k\}, F^{f,t}_{h-1}, F^{b,t}_{h+1}\}$		
Memory complexity	Transmission policy	State transition	Q-value
	$\lvert\mathcal{X}_{m_h}\rvert\lvert\mathcal{A}_{m_h}\rvert K$	Not required	$\lvert\mathcal{X}_{m_h}\rvert\lvert\mathcal{A}_{m_h}\rvert K$
Computational complexity	$O(\lvert\mathcal{X}_{m_h}\rvert\lvert\mathcal{A}_{m_h}\rvert K)$		

$$\mathcal{B}_{m_h} = \{s_{m_h} : x_{m_h,m_{h+1}} \geq \delta - \ln\left(\frac{T_{k,m_h,m_{h+1}} D^{rem}_{k,m_h}}{L_k} - 1\right)^{\frac{1}{\xi}}\} \tag{15}$$

the stringent delay constraints of mission-critical applications. 2) We apply the maximum-likelihood state-transition probabilities [14] in Algorithm 2 to update the state transition probabilities $\hat{T}^t_{s_{m_h}s_{m'_h}}(A_{m_h})$, instead of using the Q-value of the next state $s^{t+1}_{m_h}$ at each service interval. In Algorithm 2, $n^t_{s_{m_h}s_{m'_h}}(A_{m_h})$ represents the observed number of times before service interval t that the action A_{m_h} is taken when the state was in s_{m_h} and made a transition to $s_{m'_h}$ and $n^t_{s_{m_h}}(A_{m_h}) = \sum_{s_{m'_h}\in\mathcal{X}_{m_h}} n^t_{s_{m_h}s_{m'_h}}(A_{m_h})$ represents the observed number of times before service interval t that the action A_{m_h} is taken when the state was s_{m_h}.
3) Unlike regular value iteration and Q-learning, instead of updating the value $Q^{t+1}_{k,m_h}(s_{m_h}, A_{m_h})$ for $\forall s_{m_h} \in \mathcal{X}_{m_h}$, we only update the value for states in a particular set $\mathcal{B}_{m_h}$. The rest of the states $s_{m_h} \notin \mathcal{B}_{m_h}$ have insufficient SINR values to keep the transmission time within the remaining delay deadline D^{rem}_{k,m_h}. In other words, the condition $L_k \Big/ T^{goodput}_{k,m_h,m_{h+1}} \leq D^{rem}_{k,m_h}$ must hold to support the transmission of traffic clas C_k at agent m_h. Hence, the set is defined as in (15), which depends on the physical layer parameters δ and ξ of the agent m_h (see equation (1)). We only update the Q-values of the states $s_{m_h} \in \mathcal{B}_{m_h}$ in Algorithm 2. Table III summarizes the required local information, memory complexity, and computational complexity of the proposed model-based reinforcement learning approach.

The proposed model-based reinforcement learning approach has higher computational complexity than model-free reinforcement learning approaches. However, the computational complexity is a minor concern in the MCN compared with satisfying the delay constraints of the mission-critical applications. For the proposed model-based reinforcement learning approach, the Q-values of $\forall s_{m_h} \in \mathcal{B}_{m_h}, \forall C_k$ need to be updated in each service interval, and for each state over $\forall A_{m_h} \in \mathcal{A}_{m_h}$, the last term $\min_{A_{m_h}\in\mathcal{A}_{m_h}} \sum_{s_{m'_h}} \hat{T}^t_{s_{m_h}s_{m'_h}}(A_{m_h}) Q^t_{k,m_h}(s_{m'_h}, A_{m_h})$ in equation (13) is calculated. Although the computational complexity is larger, the convergence rate of the proposed model-based reinforcement learning approach is much faster than the model-free reinforcement learning approaches. In Section V.B, we compare the convergence speeds of different learning methods through extensive simulation results. Hence, the MCN nodes can choose to implement this higher complexity learning to improve their performance. In Section V.C, we investigate the case where nodes deploy heterogeneous learning methods and determine the resulting performance.

C. Upper and lower bounds of the model-based learning approach

Since the maximum-likelihood state-transition probabilities $\hat{T}^t_{s_{m_h}s_{m'_h}}(A_{m_h})$ are used in the proposed model-based learning approach, there is no guarantee that the resulting MDP delay value can converge to the optimal value $V^*_{k,m_h}(s_{m_h}, F^b_{h+1})$ in equation (9). In this subsection, we investigate the accuracy of the proposed model-based learning in terms of the resulting MDP delay value. Let $\overline{V}^t_{k,m_h}(s_{m_h}, F^{b,t-1}_{h+1})$ and $\underline{V}^t_{k,m_h}(s_{m_h}, F^{b,t-1}_{h+1})$ denote the

TABLE III
COMPLEXITY SUMMARY OF THE MODEL-BASED REINFORCEMENT LEARNING

Required local information	$\mathcal{L}^t_{m_h} = \{s^t_{m_h}, F^{f,t}_{h-1}, F^{b,t}_{h+1}\}$		
Memory complexity	Transmission policy	State transition	Q-value
	$\lvert\mathcal{B}_{m_h}\rvert\lvert\mathcal{A}_{m_h}\rvert K$	$\lvert\mathcal{B}_{m_h}\rvert^2\lvert\mathcal{A}_{m_h}\rvert K$	$\lvert\mathcal{B}_{m_h}\rvert\lvert\mathcal{A}_{m_h}\rvert K$
Computational complexity	$O\left(K\lvert\mathcal{B}_{m_h}\rvert^2\lvert\mathcal{A}_{m_h}\rvert\right)$		

upper and the lower bounds of the value, respectively, using $\hat{T}^t_{s_{m_h}s_{m'_h}}(A_{m_h})$ in the proposed model-based learning approach in service interval t. We define ε as the $(1-\delta)$-confidence interval of the real MDP delay value (using the unknown $\hat{T}^t_{s_{m_h}s_{m'_h}}(A_{m_h})$ in Section III) in service interval t, i.e. $\text{Prob}(\overline{V}^t_{k,m_h}(s_{m_h}, F^{b,t-1}_{h+1}) - V^t_{k,m_h}(s_{m_h}, F^{b,t-1}_{h+1}) \geq \varepsilon) \leq 1-\delta (0<\delta<1)$.

Proposition: There exists a $(1-\delta)$-confidence interval ε, such that an agent m_h can update the upper bound of value $\overline{V}^t_{k,m_h}(s_{m_h}, F^{b,t}_{h+1})$ using

$$\overline{V}^{t+1}_{k,m_h}(s_{m_h}, F^{b,t}_{h+1}) = \min_{A_{m_h}} \left\{ \begin{array}{l} E[W^t_{k,m_h}(s_{m_h}, A_{m_h})] + F^{b,t}_{h+1}(A_{m_h}) + \\ \gamma \sum\limits_{s_{m'_h}} \hat{T}^t_{s_{m_h}s_{m'_h}}(A_{m_h}) \overline{V}^t_{k,m_h}(s'_{m_h}, F^{b,t-1}_{h+1}) + \varepsilon \end{array} \right\} \tag{16}$$

and update the lower bound $\underline{V}^t_{k,m_h}(s_{m_h}, F^{b,t}_{h+1})$ using

$$\underline{V}^{t+1}_{k,m_h}(s_{m_h}, F^{b,t}_{h+1}) = \min_{A_{m_h}} \left\{ \begin{array}{l} E[W^t_{k,m_h}(s_{m_h}, A_{m_h})] + F^{b,t}_{h+1}(A_{m_h}) + \\ \gamma \sum\limits_{s_{m'_h}} \hat{T}^t_{s_{m_h}s_{m'_h}}(A_{m_h}) \underline{V}^t_{k,m_h}(s'_{m_h}, F^{b,t-1}_{h+1}) - \varepsilon \end{array} \right\} \tag{17}$$

and the following two conditions are satisfied:
1) $n^t_{s_{m_h}}(A_{m_h}) = \left\lceil \frac{1}{2} \ln \left(\frac{|\mathcal{A}_{m_h}||\mathcal{B}_{m_h}|}{\delta} \right) \left(\frac{V_{\max}}{\varepsilon} \right)^2 \right\rceil, \forall A_{m_h} \in \mathcal{A}_{m_h}$, where $V_{\max} = \frac{\max\limits_k D^{rem}_{k,m_h}}{1-\gamma}$ represents the largest MDP delay value.
2) $\underline{V}^*_{k,m_h}(s_{m_h}, F^b_{h+1}) \leq V^*_{k,m_h}(s_{m_h}, F^b_{h+1}) \leq \overline{V}^*_{k,m_h}(s_{m_h}, F^b_{h+1})$ with probability at least $1-2\delta$.
Proof: See Appendix.

This proposition shows that the estimated values $V^{t+1}_{k,m_h}(s_{m_h}, F^{b,t}_{h+1})$ become more accurate as $n^t_{s_{m_h}}(A_{m_h})$ becomes larger than $\left\lceil \frac{1}{2} \ln \left(\frac{|\mathcal{A}_{m_h}||\mathcal{B}_{m_h}|}{\delta} \right) \left(\frac{V_{\max}}{\varepsilon} \right)^2 \right\rceil$. Moreover, the closer the agent m_h is to the destination node, the remaining path becomes shorter and provides a smaller and leads to a smaller $V_{\max}$ requirement on $n^t_{s_{m_h}}(A_{m_h})$. Hence, using the same proposed model-based learning approach to accumulate $n^t_{s_{m_h}}(A_{m_h})$, the learning approach provides a more accurate MDP delay value for an agent that is closer to its destination node, which is also verified in the simulation results in Section V.D.

V. SIMULATION RESULTS

In this section, we simulate the performance of the proposed model-based reinforcement learning for solving the distributed MDP for the mission-critical applications.

A. Simulation results for different network topologies

We simulate first a 6-hop MCN with a topology shown in Figure 5(a) with two ASs and 18 ARs. Such MCN is commonly adopted in various areas, such as battlefield sensing, security monitoring, and healthcare applications, where prioritized data packets need to be relayed to the remote destinations in a timely manner. Two groups of mission-critical applications are sent in different priority classes ($K=8$). The characteristic parameters of these mission-critical applications are given in Table IV. Various mission-critical applications can be supported, e.g. video streams from surveillance cameras [21], delay-sensitive monitoring report such as forest fire detection, or patient monitoring [1]. Group 1 mission-critical applications are sent through the AS m_1 to the destination node D1 and group 2 mission-critical applications are sent from the other AS m_2 to its destination node D2. The agents are assumed to be able to select a set of modulation and coding schemes that support a transmission rate $T=1$ Mbps for all the transmission links in the network [20]. Each receiver of the transmission links receives a random SINR x that results in a packet error rate ranging from 5% to 30%.

We assume that the nodes are exchanging hello messages (as in DSDV [24]) with the required information exchange every 10 ms (each service interval is 10 ms). Figure 5(b) shows the MDP delay values from the ASs to the destination nodes for the first 120 service intervals. Only the results of the first five priority classes are shown. The higher priority traffic has a smaller MDP delay value $V^t_{k,m}$. The results of centralized optimization are analytically computed by assuming that the global network information is known by a central controller, which is unrealistic in practice. On the other hand, the proposed model-based reinforcement learning determines the cross-layer transmission policy at each agent based on local information. We set $\gamma = 0.75$, which is appropriate for highly time-varying MCN (after 10 service intervals, the future is only about 5% of the cost). Note that our model-based learning provides the MDP delay values close to the centralized optimization results, especially for the priority classes C_1, C_2, C_3 that satisfy the condition $E[W_{k,m_h}] \leq D^{rem}_{k,m_h}$. These three priority classes converge to a steady state after $t=40$, since their end-to-end delays are within the delay deadline of the applications (the required performance level

TABLE IV
THE CHARACTERISTIC PARAMETERS OF THE MISSION-CRITICAL APPLICATIONS

	Group 1 mission-critical applications V_1				Group 2 mission-critical applications V_2			
C_k	C_1	C_4	C_6	C_8	C_2	C_3	C_5	C_7
R_k (Kbps)	556	333	334	445	500	300	300	400
D_k	1 sec							
L_k	1000 bytes							

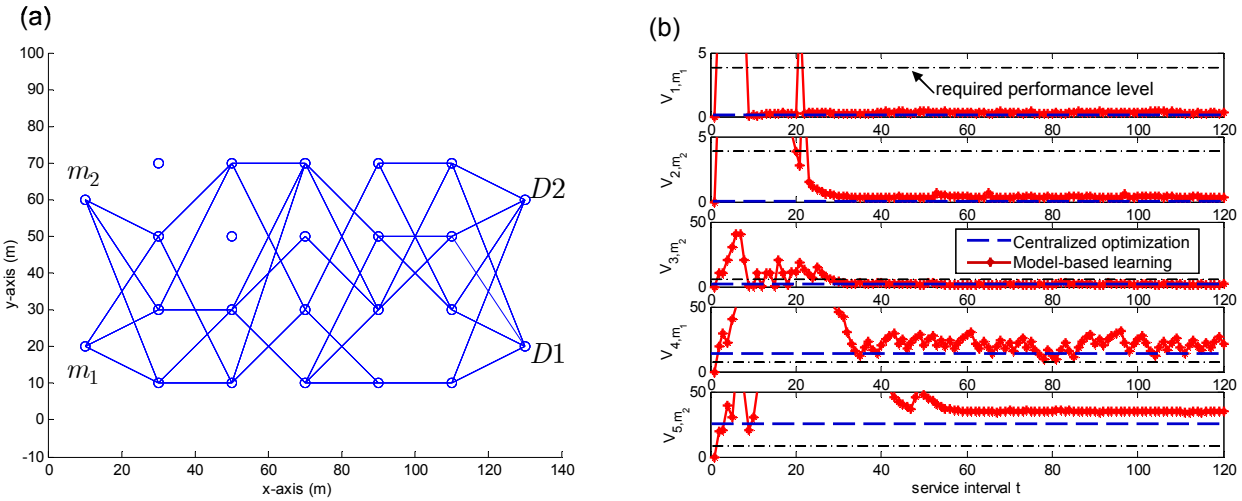


Fig. 5. (a) 6-hop network topology (b) MDP delay values of the first five priority classes

is set as $\sum_{t=1}^{\infty} \gamma^{t-1} D_k = \frac{D_k}{1-\gamma} = 4$ when the delay deadline of each future service interval is considered) and no packets are dropped. The results also show that the higher priority traffic converges faster than the lower priority traffic. This is because the queuing delay of the lower priority class traffic is impacted by the higher priority class traffic. Next, we simulate a skewed network topology that has two clusters of nodes shown in Figure 6(a). Such network topology with clusters of nodes can be common in the MCN due to landscape requirements. The network connections between the two clusters usually form a bottleneck to transmit the mission-critical applications. Figure 6(b) shows that the MDP delay values $V_{k,m}^t$ of all the priority classes increase. We observe that only the convergence rates of the higher priority classes decrease in the skewed network due to the impact of the bottleneck.

B. Comparison among the reinforcement learning approaches

In this subsection, we compare the proposed model-based reinforcement learning approach with Q-learning in [16] (a model-free reinforcement learning approach) and the myopic self-learning approach in [21] ($\gamma = 0$). We adopt the same network conditions as the previous simulations and the network topology shown in Figure 5(a). In Figure 7, the simulation results show that the proposed model-based reinforcement learning approach outperforms the other two learning approaches in terms of the MDP delay values for all the priority classes. Although Q-learning has the lowest computational complexity, it has the worst performance in terms of both the MDP delay value $V_{k,m}^t$ and the convergence rate. The delay of the C_1 traffic converges after $t = 20$ for the proposed model-based learning approach and converges only after $t = 40$ for Q-learning approach. The convergence is not guaranteed for the lower priority class traffic, especially for the myopic self-learning solution. Moreover, although the myopic approach has the fastest convergence rate, it results in a worse performance than the proposed model-based reinforcement learning approach.

In addition to the MDP delay values $V_{k,m}^t$, we directly compare expected end-to-end delays $E[Delay_k^t]$ of the mission-critical applications from the ASs to the destination nodes. The acceptance level for $E[Delay_k^t]$ is $D_k = 1$. In Figure 8, the simulation results show that by using the proposed model-based learning approach, the MCN is able to support up to three mission-critical classes, since the end-to-end delay must be within the delay deadline of the applications ($E[Delay_k^t] \leq D_k$), while by using the other two learning approaches, the network can only support two mission-critical classes.

Next, we simulate the expected delay of different classes in a source variation scenario, where the AS m_1 disappears right after service interval $t = 60$. Figure 9 shows the changes of expected delays over time for different classes using various learning approaches. Since the AS m_1 is the source node of packets in classes $\{C_1, C_4, C_6, C_8\}$, the expected delays $E[Delay_1]$ and $E[Delay_4]$ in Figure 8 vanish after $t = 60$. We

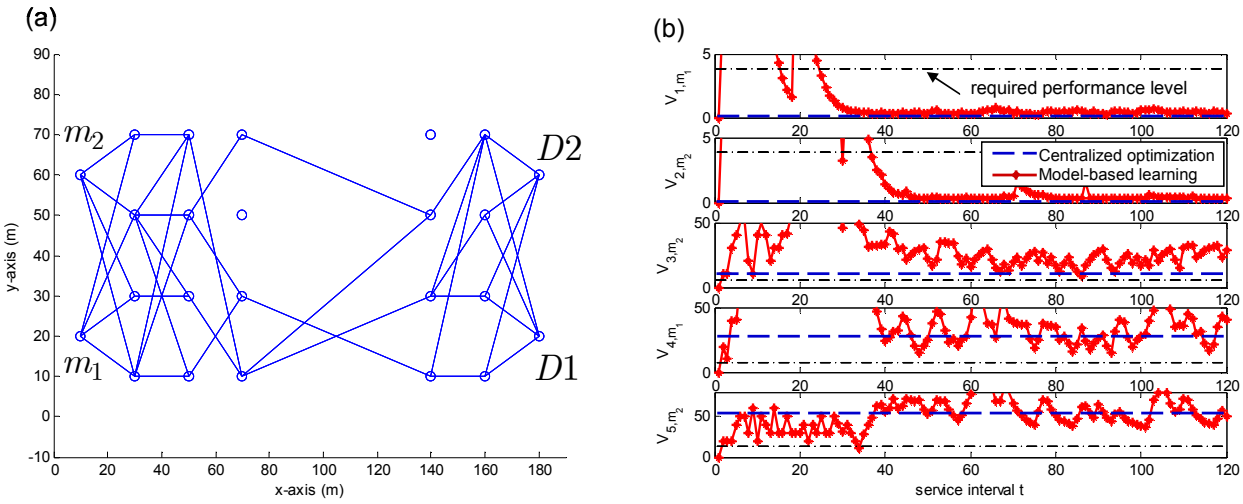


Fig. 6. (a) 2-cluster skewed network topology (b) MDP delay values of the first five priority classes

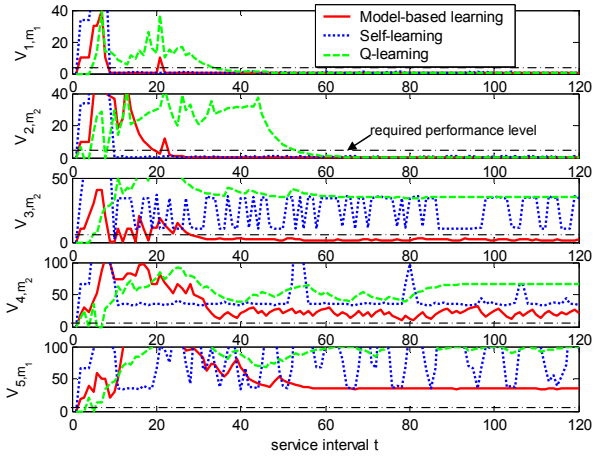


Fig. 7. Comparisons of the discounted end-to-end delay using different learning approaches that solves the distributed MDP

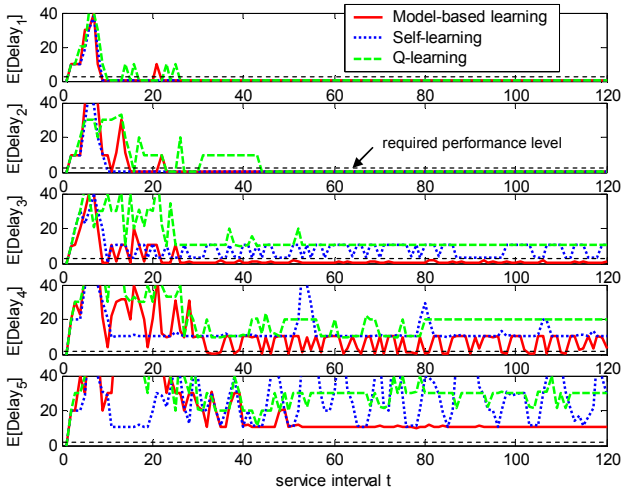


Fig. 8. Comparisons of the expected end-to-end delay using different learning approaches that solves the distributed MDP.

can observe that if Q-learning is applied, before $t = 60$, only class C_1 from m_1 can be delivered in time ($E[Delay_1] \leq D_1$). However, after $t = 60$, the class C_2 from m_2 can be supported by the MCN due to the alleviation of the traffic loading. By applying the proposed model-based learning approach, before $t = 60$, both classes C_1, C_2 can be delivered in time, and after $t = 60$, not only the class C_2 but also the class C_3 from m_2 can be supported by the MCN. This shows that the proposed model-based learning approach enables the MCN to support more mission-critical applications.

C. Heterogeneous learning

In the previous simulations, we assume that all the network nodes adopt the same learning approach to solve the distributed MDP. However in reality, the agents can adopt different learning approaches. We simulated different scenarios in which the agents have heterogeneous learning capabilities using the same network conditions as the previous simulation and the same network topology shown in Figure 5(a).

In Table V, we assume that the agents in the same hop are using the same learning method. The model-based learning refers to the proposed model-based reinforcement learning approach and the model-free learning refers to the Q-learning in [16]. The simulation results show that adopting a model-based learning approach near the ASs is very important. The delays are smaller independent of the type of learning approaches the rest of the nodes. This is because the model-based learning approach provides a more accurate estimate of the expected delay feedback than the model-free learning

TABLE V
THE RESULTS OF HETEROGENEOUS LEARNING SCENARIOS

	Learning method (within 2 hops from ASs)	Learning method (outside 2 hops from ASs)	Expected delay of the first class traffic (sec)	Expected delay of the second class traffic (sec)
1	Model-based	Model-based	0.34	0.4535
2	Model-based	Both (random)	0.3411	1.5841
3	Model-based	Model-free	0.3461	1.9785
4	Model-free	Model-based	1.5507	2.9401
5	Model-free	Both (random)	1.6886	7.4319
6	Model-free	Model-free	1.8401	7.7301

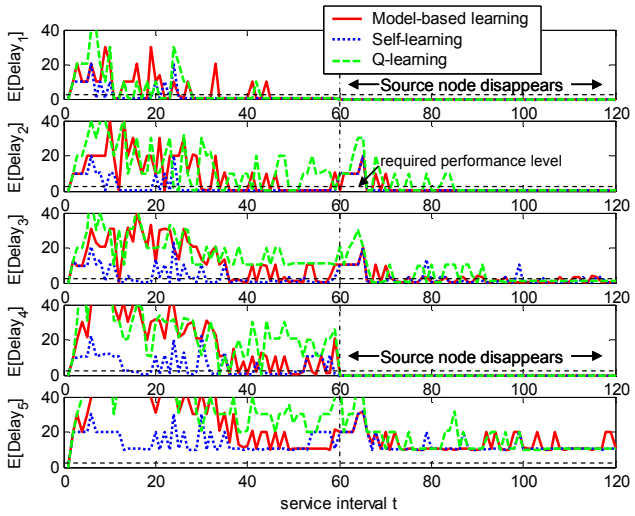


Fig. 9. Source node of packets in class C_1, C_4 disappears after $t = 60$.

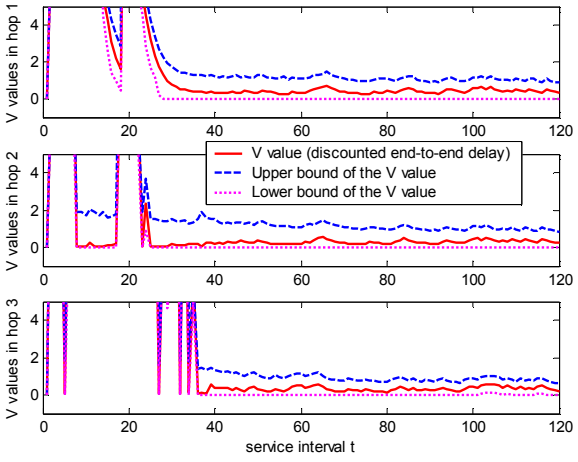


Fig. 10. The upper and the lower bounds of the discounted end-to-end delays for the first priority class traffic at different hops.

approach. Also, the model-based learning approach converges faster than the model-free learning approach. Hence, the more remaining nodes adopt the model-based learning approach, the higher the improvement in the delay performance. Moreover, the delays of the second priority class traffic vary more than the first priority class. This shows that the learning methods adopted by the agents can significantly impact the performance of mission-critical applications, especially the ones with lower priorities. In other words, the deployed learning approaches impact the number of mission-critical applications supported by the MCN.

D. Determining the upper and the lower bound

In this subsection, we provide simulation results to show the upper bound and the lower bound of the model-based reinforcement learning. We adopt the same network conditions and the 2-cluster network topology shown in Figure 6(a). Figure 10 shows the MDP delay values of the first priority class traffic at different hops. Since the real delay is proven to be bounded between the upper and the lower bounds, the result shows that the model-based reinforcement learning provides end-to-end delays that are more and more accurate over time as well as when the agents are getting closer to the destination nodes.

VI. Conclusion

In this paper, we investigated how the agents in the MCN should optimally select their cross-layer transmission actions in the MCN in order to minimize the end-to-end delays of mission-critical applications. To consider both the spatial and temporal dependency in the MCN, we formulate the network delay minimization problem using distributed MDP. To solve the distributed MDP in practice, we propose an online model-based reinforcement learning approach. Unlike the conventional model-free reinforcement learning approaches, the proposed model-based reinforcement learning approach has a faster convergence rate, since it takes advantage of the priority queuing model and requires less time for the autonomic node to explore different states to evaluate the Q-values. Our simulation results verify that the suitability of the proposed model-based learning approach supporting mission-critical applications by the agents in the MCN.

Appendix
Proof of the Proposition

We apply Heoffding inequality [22] to obtain the confidence interval ε, which basically states that given random variables $\{X_1, ..., X_m\}$ in range $[0, X_{\max}]$, the inequality holds:

$$\text{Prob}\left(\frac{1}{m}\sum_{i=1}^{m} X_i - \frac{1}{m}\sum_{i=1}^{m} E[X_i] \geq \varepsilon\right) \leq e^{-2m\left(\frac{\varepsilon}{X_{\max}}\right)^2} \quad (18)$$

From the first condition, we have $\varepsilon = V_{\max}\sqrt{\frac{-\ln\left(\frac{\delta}{|\mathcal{A}_{m_h}||\mathcal{B}_{m_h}|}\right)}{2n^t_{s_{m_h}}(A_{m_h})}}$. Denote $E[\overline{V}(s_{m_h}, A_{m_h})] = \sum_{s'_{m_h}} \hat{T}^t_{s_{m_h}s'_{m_h}}(A_{m_h})\overline{V}^t_{k,m_h}(s'_{m_h}, F^{b,t-1}_{h+1})$ as the average MDP delay upper bound based on the estimated $\hat{T}^t_{s_{m_h}s'_{m_h}}(A_{m_h})$ whenever state s_{m_h} is visited and action A_{m_h} is taken, and denote $E[V(s_{m_h}, A_{m_h})] = \sum_{s'_{m_h}} T_{s_{m_h}s'_{m_h}}(A_{m_h})V^t_{k,m_h}(s'_{m_h}, F^{b,t-1}_{h+1})$ as the average expected MDP delay value based on real $T^t_{s_{m_h}s'_{m_h}}(A_{m_h})$. Similar to the proof of lemma 3.2 in [23], equation (18) can be rewritten as:

$$\text{Prob}(E[\overline{V}(s_{m_h}, A_{m_h})] - E[V(s_{m_h}, A_{m_h})] \geq \varepsilon) \leq \exp\left\{-2n^t_{s_{m_h}}A_{m_h}\left(\frac{1}{V_{\max}}\right)^2\left(V_{\max}\sqrt{\frac{-\ln\left(\frac{\delta}{|\mathcal{A}_{m_h}||\mathcal{B}_{m_h}|}\right)}{2n^t_{s_{m_h}}A_{m_h}}}\right)^2\right\} = \left(\frac{\delta}{|\mathcal{A}_{m_h}||\mathcal{B}_{m_h}|}\right) \quad (19)$$

Hence,$\text{Prob}(\bar{V}^{t+1}_{k,m_h}(s_{m_h}, F^{b,t}_{h+1}) - V^{t+1}_{k,m_h}(s_{m_h}, F^{b,t}_{h+1}) \geq \varepsilon) \leq \delta$ for each state-action pair (the total number of the state-action pairs is $|\mathcal{A}_{m_h}||\mathcal{B}_{m_h}|$). Similar proof can be applied to the lower bound. Since $n^t_{s_{m_h}}(A_{m_h})$ in the last term of equations

TABLE VI
ALGORITHM 1: MODEL-FREE REINFORCEMENT LEARNING AT NODE m_h

Input: $\{F_{h+1}^{b,t}, \forall t\}$, $\{F_{h-1}^{f,t}, \forall t\}$, γ, τ; **Output:** μ_{k,m_h}^t, $F_h^{b,t}$, $F_h^{f,t}$;

Initialization: μ_{k,m_h}^0, $F_{h+1}^{b,0}$, $F_{h-1}^{f,0}$, $s_{m_h}^0$;

set $t \leftarrow 0$, $Q_{k,m_h}^0(s_{m_h}, A_{m_h}) = 0, \forall s_{m_h} \in \mathcal{X}_{m_h}, \forall A_{m_h} \in \mathcal{A}_{m_h}$;

Step 1: Check the head of line packet class and the delay deadline

Get the class C_k packet that has the highest priority in the queue, and check the packet header.

If $Delay_{k,h-1}^{PASS} > D_k$, drop the packet and repeat step 1, $t \leftarrow t+1$; otherwise, go to step 2.

Step 2: Select an action $A_{m_h}^t$ based on policy μ_{k,m_h}^t

Randomly select the action $A_{m_h}^t$ according to the probability distributions $[\pi_{k,m_h}^t(s_{m_h}^t, A_{m_h}), \forall A_{m_h} \in \mathcal{A}_{m_h}]$.

Step 3: Transmit the packet and observe the current cost $Cost_{k,m_h}^t$ and the new state $s_{m_h}^{t+1}$

Step 4: Update the Q-value

For $\forall s_{m_h} \in \mathcal{X}_{m_h}$, update the Q-values $Q_{k,m_h}^{t+1}(s_{m_h}, A_{m_h}^t)$ using equation (11).

Step 5: Update the policy

For $\forall s_{m_h} \in \mathcal{X}_{m_h}$, update the policy $\mu_{k,m_h}^{t+1}(s_{m_h})$ using equation (12).

Step 6: Update the feedback values and exchange information with the neighboring nodes

Update $F_h^{b,t+1}$ and $F_h^{f,t+1}$ as in Table II.

$t \leftarrow t+1$; go back to step 1.

TABLE VII
ALGORITHM 2: MODEL-BASED REINFORCEMENT LEARNING AT NODE m_h

Input: $\{F_{h+1}^{b,t}, \forall t\}$, $\{F_{h-1}^{f,t}, \forall t\}$, γ, τ; **Output:** μ_{k,m_h}^t, $F_h^{b,t}$, $F_h^{f,t}$;

Initialization: μ_{k,m_h}^0, $F_{h+1}^{b,0}$, $F_{h-1}^{f,0}$, $s_{m_h}^0$;

set $t \leftarrow 0$, $Q_{k,m_h}^0(s_{m_h}, A_{m_h}) = 0, \forall s_{m_h} \in \mathcal{X}_{m_h}, \forall A_{m_h} \in \mathcal{A}_{m_h}$;

Step 1: Check the head of line packet class and the delay deadline

Get the class C_k packet that has the highest priority in the queue, and check the packet header.

If $Delay_{k,h-1}^{PASS} > D_k$, drop the packet and repeat step 1, $t \leftarrow t+1$; otherwise, go to step 2.

Step 2: Select an action $A_{m_h}^t$ based on policy μ_{k,m_h}^t

Randomly select the action $A_{m_h}^t$ according to the probability distributions $[\pi_{k,m_h}^t(s_{m_h}^t, A_{m_h}), \forall A_{m_h} \in \mathcal{A}_{m_h}]$.

Step 3: Transmit the packet, observe the new state $s_{m_h}^{t+1}$, and update the number of state transition

$$n_{s_{m_h}^t s_{m_h}^{t+1}}^{t+1}(A_{m_h}^t) = n_{s_{m_h}^t s_{m_h}^{t+1}}^{t}(A_{m_h}^t) + 1;\ n_{s_{m_h}^t}^{t}(A_{m_h}^t) = \sum\nolimits_{s_{m_h}' \in \mathcal{X}_{m_h}} n_{s_{m_h}^t s_{m_h}^{t+1}}^{t}(A_{m_h}^t).\ \hat{T}_{s_{m_h}^t s_{m_h}^{t+1}}^{t}(A_{m_h}^t) = \frac{n_{s_{m_h}^t s_{m_h}^{t+1}}^{t}(A_{m_h}^t)}{n_{s_{m_h}^t}^{t}(A_{m_h}^t)}.$$

Step 4: Evaluate the local queuing delay

Calculate $E[W_{k,m_h}^t(s_{m_h}, A_{m_h}^t)]$ using equation (14).

Step 5: Update the Q-value

For $\forall s_{m_h} \in \mathcal{B}_{m_h}$, update the Q-values $Q_{k,m_h}^{t+1}(s_{m_h}, A_{m_h}^t)$ using equation (13).

Step 6: Update the policy

For $\forall s_{m_h} \in \mathcal{B}_{m_h}$, update the policy $\mu_{k,m_h}^{t+1}(s_{m_h})$ using equation (12).

Step 7: Update the feedback values and exchange information with the neighboring nodes

Update $F_h^{b,t+1}$ and $F_h^{f,t+1}$ as in Table II.

$t \leftarrow t+1$; go back to step 1.

(16) and (17) goes to infinity as $t \to \infty$, we can show that both the upper bound and the lower bound converge under the same conditions, i.e. $\overline{V}_{k,m_h}^*(s_{m_h}, F_{h+1}^b) = \lim_{t\to\infty} \overline{V}_{k,m_h}^t(s_{m_h}, F_{h+1}^{b,t})$, and $\underline{V}_{k,m_h}^*(s_{m_h}, F_{h+1}^b) = \lim_{t\to\infty} \underline{V}_{k,m_h}^t(s_{m_h}, F_{h+1}^{b,t})$. Due to the symmetric structure of $\overline{V}_{k,m_h}^*(s_{m_h}, F_{h+1}^b)$ and $\underline{V}_{k,m_h}^*(s_{m_h}, F_{h+1}^b)$, we apply the union bound as in [23] to show that the probability $\text{Prob}(\left|\overline{V}_{k,m_h}^*(s_{m_h}, F_{h+1}^b) - V_{k,m_h}^*(s_{m_h}, F_{h+1}^b)\right| \geq \varepsilon) \leq 2\delta$ and complete the proof.

REFERENCES

[1] A. Rezgui and M. Eltoweissy, "Service-Oriented Sensor-Actuator Networks," *IEEE Commun. Mag.*, vol. 45, no. 12, pp 92-100, Dec 2007.

[2] S. Nelakuditi, Z. Zhang, R. P. Tsang, D. H. C. Du, "Adaptive Proportional Routing: A Localized QoS Routing Approach," *IEEE/ACM Trans. Netw.*, vol. 10, no. 6, pp. 790-804, Dec 2002.

[3] K. Akayya and M. Younis, "An Energy-Aware QoS Routing Protocol for Wireless Sensor Networks, " in the *Proc. IEEE Workshop on Mobile and Wireless Networks(MWN2003)*, Providence, RI, May 2003.

[4] P. Gupta and T. Javidi, "Towards Throughput and Delay-Optimal Routing for Wireless Ad-Hoc Networks," *Asilomar Conference on Signals, Systems and Computers*, Nov. 2007.

[5] M. J. Neely, E. Modiano, and C. E. Rohrs, "Dynamic Power Allocation and Routing for Time-Varying Wireless Networks", *IEEE J. Sel. Areas Commun.*, vol. 23. no. 1, Jan 2005.

[6] A. Tizghadam, A. Leon-Garcia, "On Congestion in Mission-Critical Networks," *IEEE INFOCOM 2008*, April 2008.

[7] M. Liotine, *Mission Critical Network Planning*, Artech House, Norwood, MA 2003.

[8] Y. Guan, X. Fu, D. Xuan, P. U. Shenoy, R. Bettati, and W. Zhao, "NetCamo: Camouflaging Network Traffic for QoS-Guaranteed Mission Critical Applications," *IEEE Trans. Syst., Man, Cybernet. A.*, vol. 31, no. 4, pp. 253-265, July 2001.

[9] Y. Huang, W. He, K. Nahrstedt, W. C. Lee, "Dos Resistant Broadcast Authentication with Low End-to-end Delay," *IEEE INFOCOM 2008*, April 2008.

[10] D. Marsh, R. Tynan, D. O'Kane, G. M. P. O'Hare, "Autonomic Wireless Sensor Networks," *Artificial Intelligence*, vol. 17, pp. 741-748, 2004.

[11] D. Chen and P. K. Varshney, "QoS support in wireless sensor networks: A survey" In *Proc. International Conference on Wireless Networks (ICWN)*, pp. 21-24, Las Vegas, NV, June 2004.

[12] J. Chakareski and P. Frossard, "Rate-Distortion Optimized Distributed Packet Scheduling of Multiple Video Streams Over Shared Communication Resource," *IEEE Trans. Multimedia*, vol. 8, no. 2, Apr, 2006.

[13] H.-P. Shiang and M. van der Schaar, "Informationally Decentralized Video Streaming over Multi-hop Wireless Networks," *IEEE Trans. Multimedia*, vol. 9, no. 6, pp. 1299-1313, Sep 2007.

[14] A. G. Barto, S. J. Bradtke and S. P. Singh, "Learning to act using real-time dynamic programming", *Artificial Intelligence*, vol. 72, no. 1-2, Jan 1995, pp. 81-138.

[15] P. Tadepalli and D. Ok, "Model-based average reward reinforcement learning", *Artificial Intelligence*, vol. 100, no. 1-2, Jan 1998, pp. 177-224.

[16] C. J. C. H. Watkins, P. Dayan, "Q-learning", *Machine Learning*, vol. 8, no. 3-4, pp. 279-292, May 1992.

[17] R. S. Sutton, "Learning to predict by the method of temporal differences," *Machine Learning*, vol. 3, no. 1, pp. 9-44, Aug. 1988.

[18] M. L. Puterman, *Markov Decision Process: Discrete Stochastic Dynamic Programming*, John Wiley & Sons, Inc. New York, 1994.

[19] D. P. Bertsekas, *Dynamic Programming and Optimal Control*, Athena Scientific, 1995.

[20] D. Krishnaswamy, "Network-assisted Link Adaptation with Power Control and Channel Reassignment in Wireless Networks," *3G Wireless Conference*, pp. 165-170, 2002.

[21] H. -P. Shiang and M. van der Schaar, "Multi-user video streaming over multi-hop wireless networks: A distributed, cross-layer approach based on priority queuing," *IEEE J. Sel. Areas Commun.*, vol. 25, no. 4, pp. 770-785, May 2007.

[22] W. Hoeffding, "Probability inequalities for sums of bounded random variables," *J. American Statistical Association*, vol. 58, no. 301, pp. 31-30, Mar. 1963.

[23] E. Even-Dar, S. Mannor, Y. Manour, "Action elimination and stopping conditions for reinforcement learning," *Proc. International Conference on Machine Learning (ICML 2003)*, 2003.

[24] C. E. Perkins, P. Bhagwat, "Highly Dynamic Destination-Sequenced Distance-Vector Routing (DSDV) for Mobile Computers," *ACM SIGCOMM Computer Communication Review*, vol. 24, no. 4, pp. 234-244, Oct. 1994.

[25] T. Roughgarden, E. Tardos, "How Bad is Selfish Routing?" *J. ACM*, vol. 49, no. 2, pp. 236-259, March 2002.

[26] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," *J. Operational Research Society*, vol. 49, no. 3, pp. 237-252, Mar. 1998.

[27] D. Xu, M. Chiang, and J. Rexford, "Link-state routing with hop-by-hop forwarding achieves optimal traffic engineering", *Proc. IEEE INFOCOM, 2008.*

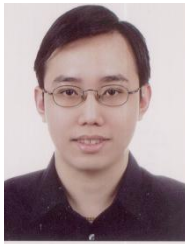
[28] F. Fu, M. van der Schaar, "A systematic framework for dynamically optimizing multi-user video transmission", technical report, http://arxiv.org/abs/0903.0207.

[29] Q. Zhang, S. A. Kassam, "Finite-state Markov Model for Reyleigh fading channels," *IEEE Trans. Commun.*, vol. 47, no. 11, Nov. 1999.

[30] J. Dowling, E. Curran, R. Cunningham, and V. Cahill, "Using Feedback in Collaborative Reinforcement Learning to Adaptively Optimize MANET Routing," *IEEE Trans. Syst., Man, Cybern. A.*, vol. 35, no. 3, pp. 360-372, May 2005.

[31] H. Tong, T. X. Brown, "Adaptive Call Admission Control under Quality of Service Constraints: A Reinforcement Learning Solution," *IEEE J. Sel. Areas Commun.*, vol. 18, no. 2, pp. 209-221, Feb 2000.

[32] S. Toumpis, A. J. Goldsmith, "Capacity Regions for wireless Ad Hoc Network", *IEEE Trans. Wireless Commun.*, vol. 2, no. 4, pp. 736-748, July 2003.



Hsien-Po Shiang is currently a Postdoctoral Scholar at the Department of Electrical Engineering, University of California, Los Angeles. He graduated from National Taiwan University with his B.S. and M.S. in Electrical Engineering in 2000 and 2002, respectively. In 2009, he received his Ph.D. degree from Electrical Engineering at University of California, Los Angeles. During his Ph.D. study, he worked at Intel Corp., Folsom CA in 2006, researching overlay network infrastructure over wireless mesh networks. He published several journal papers and conference papers on these topics and has been selected as one of the eight Ph.D. students chosen for the 2007 Watson Emerging Leaders in Multimedia awarded by IBM Research, NY. His research interests include cross-layer optimizations/adaptations, multimedia communications, and dynamic resource management for delay-sensitive applications.

Mihaela van der Schaar received the Ph.D. degree from Eindhoven University of Technology, Eindhoven, The Netherlands, in 2001. She is currently an Associate Professor at the Department of Electrical Engineering, University of California, Los Angeles. She holds 30 granted US patents. She is also the editor (with Phil Chou) of the book Multimedia over IP and Wireless Networks: Compression, Networking, and Systems (San Diego, CA: Academic Press, 2007). Dr. Van der Schaar has been an active participant in the International Organization for Standardization (ISO) MPEG standard since 1999, to which she made more than 50 contributions and for which she received 3 ISO recognition awards. She received the National Science Foundation CAREER Award in 2004, IBM Faculty Award in 2005 and 2007, the Okawa Foundation Award in 2006, the IEEE Transactions on Circuits and Systems for Video Technology Best Paper Award in 2005, and the Most Cited Paper Award from the European Association for Signal Processing Journal Signal Processing: Image Communication for 2004-2006. She was elected as an IEEE Fellow in 2010. Her research interests include wireless multimedia processing, communication and networking, game-theoretic approaches in multi-agent communication systems, and multimedia systems.