DRIVING: Distributed Scheduling for Video Streaming in Vehicular Wi-Fi Systems

Xi Chen
McGill University
xi.chen11@mail.mcgill.ca

Lei Rao
General Motors Company
lei.rao@gm.com

Qiao Xiang
Yale University
qiao.xiang@cs.yale.edu

Xue Liu
McGill University
xueliu@cs.mcgill.ca

Fan Bai
General Motors Company
fan.bai@gm.com

ABSTRACT

Video streaming has been dominating the mobile bandwidth, and is still expanding drastically. Its tremendous economic benefits have driven the automobile industry to equip vehicles with video streaming capacity. As a result, the new in-cabin Wi-Fi systems have been deployed, enabling each vehicle as a streaming hotspot on the wheels. A built-in Access Point (AP) bridges the communications between Wi-Fi devices inside and cellular networks outside. Distinct advantages offered by this system include a more powerful antenna array to improve multimedia quality, a constant energy source to power the streaming, etc. However, there exist two challenging features that may jeopardize the system performance. (1) The in-cabin Wi-Fi hotspots are mostly deployed on private vehicles, and thus are completely decentralized. (2) Video packets need to be delivered before their deadlines with small delays. Due to these features, existing algorithms may fail to efficiently schedule the in-cabin Wi-Fi video streaming. To fill the gap, we propose the Delay-awaRe dIstributed Video schedulING (DRIVING) framework. Being fully distributed and delay-aware, DRIVING not only increases the streaming goodput, but also reduces the delivery latency and deadline missing ratio. In a typical scenario, DRIVING increases the goodput by up to 27.0%, while reducing the queueing delay and the deadline missing ratio by up to 40.0% and 38.4%, respectively.

1. INTRODUCTION

Mobile multimedia services have been expanding significantly. Among them, video streaming is and will be the dominating one, accounting for over 75% of mobile data traffic by 2020 [1]. This leads to a huge and fast-growing market, which drives car manufacturers to actively deploy video streaming capacity to their vehicles. The new in-cabin Wi-Fi systems are thus developed to power all the Wi-Fi devices inside the cabin, bridging the communication gap between these devices and 3G/4G cellular networks [2]. A built-in Wi-Fi hotspot (we call it an in-cabin Wi-Fi AP in the rest of this paper) provides drivers and passengers a variety of multimedia services including the most popular video streaming. Comparing with LTE connected smart phones and tablets, the built-in Wi-Fi connection offers distinct advantages including a more powerful antenna array to improve signal quality, a constant energy source to power this connection, and an integrated design that is optimized for in-vehicle use.

Considering the high popularity and large bandwidth consumption of video streaming, in-cabin Wi-Fi APs on different vehicles must be efficiently scheduled. The design of a scheduling algorithm needs to consider two challenging features. (1) The in-cabin Wi-Fi APs are mainly deployed on private vehicles, serving different individuals in a non-cooperative manner. (2) The video streaming requires small delays and jitters, and low deadline missing ratio.

Existing scheduling algorithms may fail to handle the co-existence of these two features. (1) Centralized algorithms (e.g., those proposed in [3, 4, 5, 6]) could be inefficient for the non-cooperative and distributed in-cabin Wi-Fi APs. Meanwhile, distributed scheduling algorithms designed for stationary networks (e.g., those proposed in [7, 8, 9, 10]) may degrade largely in a vehicular scenario, where the topology of APs is fast-varying. Hence, a candidate scheduling scheme should be fully distributed and robust to the rapid changes in AP topology. (2) Existing distributed scheduling criteria, such as queue length [11], historical throughput [12] or energy efficiency [9, 13], could fail in fulfilling delay requirements of video streaming.Thus, the delay sensitive feature should be integrated into the distributed scheduling. To sum up, we are still in need of a scheduling scheme that is both **fully distributed** and **delay-sensitive**.

In this paper, we develop the **D**elay-awa**R**e D**I**stributed **V**ideo schedul**ING** (DRIVING) framework for in-cabin Wi-Fi systems. DRIVING carefully retrofits existing distributed Carrier Sense Multiple Access (CSMA) schemes to prioritize packets with large queueing delays. In this way, it addresses the unawareness of delays in existing distributed schemes (e.g., those proposed in [11, 12]), and reduces delays and jitters in video streaming. At the same time, DRIVING is lightweight, requiring only a software upgrade for deployment on commodity Wi-Fi APs.

The **main contribution** of this paper is twofold.

• We study the important problem of scheduling video streaming in in-cabin Wi-Fi systems. We develop a fully dis-

MM '16, October 15-19, 2016, Amsterdam, Netherlands
© 2016 ACM. ISBN 978-1-4503-3603-1/16/10. . . $15.00
DOI: http://dx.doi.org/10.1145/2964284.2964290

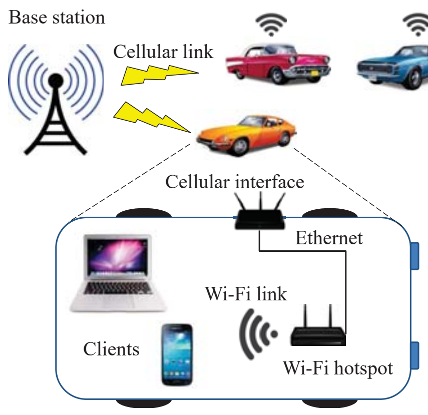


Figure 1: An example of In-cabin Wi-Fi systems.

tributed and delay-aware framework DRIVING to embrace the distinct features of in-cabin Wi-Fi video streaming.

• We characterize DRIVING with advanced cross-layer analytical models, which serve as the guidelines to tune this new framework for better performance.

In a typical scenario, DRIVING increases the average goodput by up to 27.0% while reducing the average queueing delay and the average deadline missing ratio by up to 40.0% and 38.4%, respectively.

2. PROBLEM AND CHALLENGES

In this section, we describe the in-cabin Wi-Fi systems, and present the problem definition and the challenges.

2.1 System Overview

2.1.1 In-cabin Wi-Fi System

An example of in-cabin Wi-Fi systems is shown in Fig. 1. The in-cabin Wi-Fi services are delivered via two kinds of wireless links - the long range cellular links and the short range Wi-Fi links. A long range cellular link is in charge of delivering data from cellular networks to in-cabin Wi-Fi APs, while a short range Wi-Fi link serves the in-vehicle Wi-Fi devices directly. An Ethernet is used to relay the data between these two links. While the cellular links are well controlled and scheduled by the cellular base station (BS), there is still much room for improvement on the performance of the Wi-Fi links. In the rest of this paper, we focus on the short range Wi-Fi transmissions, which are the bottleneck of the whole system. Note that the joint scheduling of the Wi-Fi links and the cellular links is also an important research topic, and will be considered in our future work.

The in-cabin Wi-Fi system is very attractive for multimedia streaming services due to the following unique advantages. First of all, an antenna array tailored for moving environments improves the cellular signal quality. Also, a constant energy source is offered by the vehicle to power the in-cabin Wi-Fi AP. In addition, off-the-shelf Wi-Fi devices can enjoy in-vehicle multimedia services without an expensive cellular module.

However, in-cabin Wi-Fi video streaming has several features, making it more difficult to schedule than conventional communications and multimedia services.

(1) The in-cabin Wi-Fi APs are deployed on private vehicles and moving rapidly. The cellular BS has no control of them. Also, there are few coordination message exchanges among the APs. Hence, these APs are fully distributed and non-cooperative with a frequently varying topology.

(2) Video packets require to be delivered before deadline with low delays and small jitters.

(3) Most client-side devices are off-the-shelf Wi-Fi devices. Any practical attempt to upgrade the system must be compatible with existing Wi-Fi protocols.

2.1.2 MAC Access in Wi-Fi

Wi-Fi communications are based on IEEE 802.11 standards, and adopt the Distributed Coordination Function (DCF) as its fundamental MAC technique. The DCF requires a Wi-Fi node to listen to the channel for a period of time equal to a Distributed Interframe Space (DIFS) before transmission. If the channel is idle, then the node can transmission. Otherwise, the node keeps listening until the channel is idle for a DIFS. Upon this, the node generates a random backoff interval and waits for it before transmission, so as to minimize collisions in the wireless channel. This backoff interval is a number randomly selected from a range called Contention Window (CW), of which the size determines the average waiting time. The CW size will be doubled (until a maximum value is reached), if the channel is sensed busy during the random backoff interval. By setting different CW sizes, one can adjust the priorities of different Wi-Fi packets.

2.2 Problem Statement and Metrics

In this paper, we aim to improve the performance of video streaming in the in-cabin Wi-Fi system by developing an advanced scheduling scheme. The performance metrics to be improved are defined as follows.

The average queueing delay and delay jitter: The queueing delay $D^{(p)}$ of a packet p is defined as

$$D^{(p)} = \begin{cases} t_{end} - t_{arr}, & p \text{ is received or dropped,} \\ t_{cur} - t_{arr}, & p \text{ is waiting for transmission,} \end{cases} \tag{1}$$

where t_{arr} is the time that packet p arrives at an in-cabin Wi-Fi AP, t_{end} is the time that packet p is received by a client or is dropped by the AP, and t_{cur} is the current time. Then the average queueing delay $\overline{D}$ is defined as the expected value of the queueing delay, i.e., $\overline{D} = \mathbf{E}\{D^{(p)}\}$.

There are different definitions of delay jitter (we call it jitter for short) in the literature. In this paper, we use the definition of jitter in the IP network [14], and define the jitter J as the average deviation of delays from the average delay, i.e.,

$$J = \Sigma_p |D^{(p)} - \overline{D}| / \Sigma_{i=1}^{N} \eta_i, \tag{2}$$

where N is the total number of clients, η_i is the number of video packets received by client i.

The deadline missing ratio: Suppose video buffer of a client device is going to drain at time t_{pre}, and the next packet p is going to arrive at t_{arr}. If $t_{arr} > t_{pre}$, then packet p misses its deadline. The deadline missing ratio ξ is then defined as the portion of packets that miss their deadlines, i.e.,

$$\xi = \Sigma_{i=1}^{N} \theta_i / \Sigma_{i=1}^{N} \eta_i, \tag{3}$$

where θ_i is the number of packets that arrive at client i and miss their deadlines.

The average goodput: The average goodput S is defined as the average streaming rate of all clients, i.e.,

$$S = \Sigma_{i=1}^{N}(\eta_i - \theta_i)\phi/NT, \quad (4)$$

where T is the time period we are interested in, and ϕ is the average size of video packets.

2.3 Challenges

The design of an advanced scheduling scheme for in-cabin Wi-Fi video streaming faces the following challenges.

Challenge 1*: The scheduling decisions must be made and executed in a fully distributed manner.* Due to the large communication delay, a cellular BS is not suitable to control the dynamic in-cabin Wi-Fi transmissions. Meanwhile, it is inefficient to select a centralized controller from all the constantly moving in-cabin Wi-Fi APs.

Challenge 2*: Meanwhile, the distributed scheduling must be delay-aware.* Existing delay-aware scheduling algorithms are mainly designed for centralized networks, and thus are not suitable for in-cabin Wi-Fi systems.

Challenge 3*: The scheduling scheme must be compatible with the commodity client and AP devices.* For the ease of deployment, the scheduling scheme should introduce little modification to the off-the-shelf devices.

3. RELATED WORK

Due to the limited space, we only discuss the most pertinent work on the distributed scheduling in a mobile scenario.

3.1 Scheduling of Video Streaming

A wealth of work exists on the scheduling of video streaming over wireless networks. Approaches for stationary networks (e.g., those proposed by Pahalawatta et al. [6] and Dua et al. [5]) are not suitable for the fully distributed in-cabin Wi-Fi APs. A number of distributed video scheduling algorithms have been proposed. For instance, Chuah et al. [9] designed an energy-efficient scheduling wireless mesh networks (WMNs). However, the assumption that no packet is lost due to delay is invalid in a vehicular scenario. Zhang et al. [10] jointly considered content priority, delay and channel condition in their model, which would consume much time in estimating the dynamic parameters. To sum up, these scheduling algorithms fail to address **Challenge 1**.

3.2 Adaptive CSMA

The basic idea of adaptive CSMA algorithms is to adjust the channel access probabilities of wireless nodes according to local and/or neighbouring information. A number of these algorithms, such as Q-SCHED proposed by Gupta et al. in [15], MIMO-pipe proposed by Qian et al. in [16] and DMDS proposed by Zhou et al. in [17], are based on message exchanges. However, message passing introduces large communication overhead, especially in a dense vehicular network. As a result, these adaptive CSMA algorithms cannot address **Challenge 1**.

Another line of research develops adaptive CSMA without message exchanges. Examples include, but are not limited to, the transmission length control algorithm proposed by Jiang et al. in [12], the Network Utility Maximization (NUM) based algorithm proposed by Rad et al. in [18] and the Q-CSMA algorithms proposed by Ni et al. in [11]. However, the delay and deadline of video streaming have not yet been considered explicitly, and thus fail to address **Challenge 2**.

3.3 Delay/Deadline-Aware Scheduling

To support delay sensitive services, such as video streaming in VANETs, a delay/deadline-aware scheduling algorithm is a must. A number of delay/deadline-aware scheduling algorithms have been proposed in the literature. Centralized algorithms, such as those proposed by Hou et al. in [3] and by Wu et al. in [4], are not suitable for a fully distributed vehicular network. Caccamo et al. in [7] proposed a distributed implicit Earliest Deadline First (EDF) scheduling, which establishes the scheduling pattern with a long initiation stage. Kanodia et al. in [8] designed a distributed algorithm, which exchanges the EDF priorities among nodes. These algorithms require either an initiation stage, message exchanges or prior knowledge of topology. They are impractical in highly dynamic scenarios, and are not able to tackle **Challenge 1**.

3.4 Distributed TDMA

A number of distributed TDMA algorithms, such as VeMAC [19] and SUV [20], have been proposed for 802.11p based communications. A considerable portion of time resource could be wasted when the vehicle density is low. Moreover, most of today's commercial Wi-Fi devices (e.g. phones and laptops) do not support the 802.11p protocol. Consequently, these distributed TDMA algorithms cannot address **Challenge 3**.

4. THE DRIVING FRAMEWORK

To address all the challenges, we propose the delay-aware distributed video scheduling framework.

4.1 Overall Design

The core idea of DRIVING is to assign high transmission priorities to packets with large queueing delays. DRIVING supports K levels of priorities. Level 0 represents the highest priority while level $K-1$ denotes the lowest priority. To realize the priorities, a small Contention Window (CW) size is assigned to a high-priority packet. DRIVING consists of three modules, and is summarized as follows.

The DRIVING framework

1. **The delay measurement** module measures the queueing delay of the head-of-line packet, and saves this delay as D_{head}. D_{head} is set to 0 for an empty queue.
2. **The priority evaluation** module evaluates the priority level of the head-of-line packet according to D_{head}, and set the corresponding priority as k, where $k = 0, 1, \cdots, K-1$.
3. **The scheduling decision** module sets the CW size of the upcoming backoff procedure as $W_{i,k}$, according to the priority level k and a number i. This number i denotes the number of transmission attempts that have been made for the current head-of-line packet. The scheduling decision module then instructs the backoff handler of the selected CW size $W_{i,k}$.

Note that the priority level k only determines the minimum CW size $W_{0,k}$. The final CW size is then determined by $W_{0,k}$ and retransmission times as $W_{i,k} = (W_{0,k}+1)2^i$, where i is the number of transmission attempts (and $i-1$ is the retransmission times).

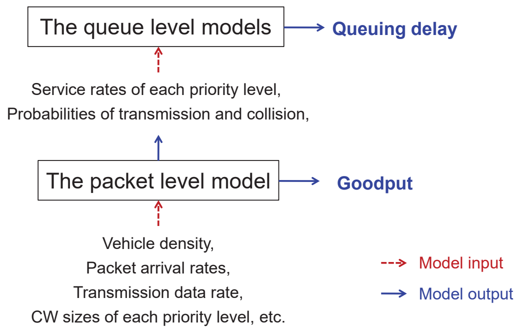


Figure 2: The two-level models.

4.2 Solving the Challenges with DRIVING

The delay measurement module measures the queueing delay D_{head} of the head-of-line packet. This information is measured locally by APs (while acquiring the deadline information requires extra message exchanges). This delay serves as a fully distributed criterion of scheduling. In this way, DRIVING solves **Challenge 1**.

The priority evaluation module evaluates the transmission priority level of each head-of-line packet, according to the queueing delay D_{head}. A large queueing delay indicates an approaching deadline. Thus, a high transmission priority is assigned to the packet. In this way, transmissions are scheduled by delay-aware priorities. Hence, DRIVING is delay-aware and solves **Challenge 2**.

The detailed implementation of the priority evaluation module is summarized as follows. Each priority level k corresponds to a continuous set β_k of the queueing delay. The queueing delay of a packet with a higher priority is always larger than the queueing delay of a packet with a lower priority. Let D_k denotes the queueing delay of a packet in priority level k. For two priority levels i and j, we have

$$\max(D_j|D_j \in \beta_j) < \min(D_i|D_i \in \beta_i), 0 \le i < j \le K-1. \tag{5}$$

Further, the union of all the delay sets covers all the non-negative real numbers, i.e., $\bigcup_{k=0}^{K} \beta_k = \mathbf{R}^+ \cup \{0\}$. Note that the DRIVING framework generalize different designs of the delay sets.

The scheduling decision module realizes the scheduling decision based on the priority passed down from the priority evaluation module. To achieve this, this module sets the minimum CW size of the upcoming backoff procedure as $W_{0,k}$ according to the priority k. A packet with a high priority has a small minimum CW size, i.e.,

$$W_{0,0} \le W_{0,1} \le \cdots \le W_{0,K-1}. \tag{6}$$

For a packet that has been transmitted i times, this module further sets the CW size as $W_{0,k}2^i$, and informs the backoff handler of this number. In this way, DRIVING utilizes the CSMA backoff procedures to realize its scheduling decisions. It only requires a software upgrade to integrate the updated backoff procedure into commodity APs, and thus solves **Challenge 3**.

5. MODELING

To apply the DRIVING framework in practice, the number of priority levels K and the design of each delay set β_k should determined. This requires us to theoretically analyze the DRIVING framework.

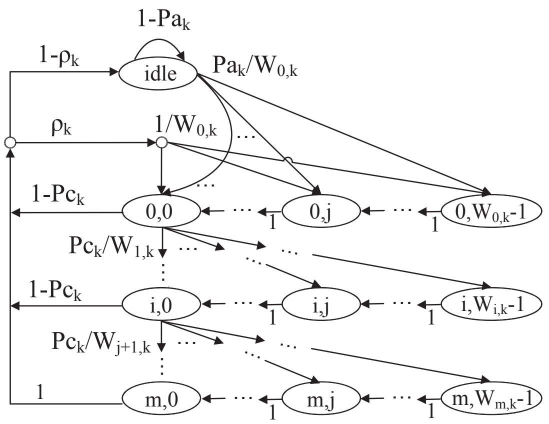


Figure 3: The DTMC for the backoff process of a node in priority level k, extended from [21][22].

5.1 Analytical Models in Two Levels

We establish one packet level model for the backoff process and two queue level models for the queueing process. These two levels of models are correlated through their inputs and outputs, as presented in Fig. 2.

5.2 The Packet Level Model

5.2.1 *The Markov Chain of Priority Level k*

A discrete-time Markov chain (DTMC) is used to model the backoff process of each node in priority level k. This DTMC is presented in Fig. 3. This DTMC is extended from the models in [21][22]. Denote each state in the DTMC as $\{s[t], b[t]\}$. Here, $s[t]$ denotes the backoff stage, i.e., the number of re-transmissions made by the head-of-line packet. Each packet will be dropped after m unsuccessful re-transmission attempts. $b[t]$ denotes the backoff time counter, which decrements by one if the channel is idle. To characterize unsaturated packet arrival, we introduce an idle state $\{idle\}$. We divide time into **virtual slots** of length E_S, which denotes the average time that the backoff process spends in each state. Such a time division allows us to achieve more accurate results than those in [21][22].

Denote the average packet arrival rate and the average service rate of a node in priority level k as λ_k and μ_k, respectively. Then at current virtual slot, the probability ρ_k that the node has at least one packet to transmit is [21]

$$\rho_k = \min[1, \lambda_k/\mu_k]. \tag{7}$$

Assume packets arrive according to a Poisson process, then the probability Pa_k that at least one packet arrives during the current virtual slot is calculated as

$$Pa_k = 1 - e^{-\lambda_k E_S}. \tag{8}$$

5.2.2 *Interactions among Markov Chains in the Packet Level Model*

Denote the number of nodes with priority k as N_k. In real deployment, an indicator can be appended to the packet header to indicate the priority k of the packet. Then, N_k can be estimated by monitoring this indicator and the distinct MAC addresses encapsulated in headers of received packets [23]. The total number of nodes is then $N = \sum_{k=0}^{K-1} N_k$. We assume that all these N nodes are within the carrier sensing range of each other. For a node in priority level k, it not only contends with other $N_k - 1$ nodes in the same priority level, but also experiences collisions from $N - N_k$ nodes in other priority levels. Let τ_k denote the probability that a node in priority level k attempts to transmit in the current virtual slot. Then, the probability Pc_k that the channel is busy is

$$Pc_k = 1 - (1-\tau_k)^{N_k - 1} \prod_{i \neq k} (1-\tau_i)^{N_i}, k = 0, \cdots, K-1. \quad (9)$$

The probability Ps_k that a node in priority level k successfully transmits a packet in the current virtual slot is [24]

$$Ps_k = \tau_k (1 - Pc_k), k = 0, 1, \cdots, K-1. \quad (10)$$

Further, the probability P_{tr} that at least one of the nodes is transmitting at current virtual slot is defined as [24]

$$P_{tr} = 1 - \prod_{k=0}^{K} (1-\tau_k)^{N_k}. \quad (11)$$

5.3 Average Service Rate and Goodput

In this section, we derive the average service rate μ_k and the average goodput S_k based on the packet level model. We denote the average time the channel is sensed busy because of a successful transmission as T_S, and denote the average time the channel is sensed busy by each node during a collision as T_C. As discussed in Section 2.1.2, Wi-Fi adopts the DCF as the default MAC access technique. For transmissions with the DCF mechanism, we have

$$T_S = T_{DIFS} + T_H + T_P + \delta + T_{SIFS} + T_{ACK} + \delta, \quad (12)$$
$$T_C = T_{DIFS} + T_H + T_P + \delta, \quad (13)$$

where T_H is the duration of a packet header, T_P is the duration of data section of each packet, T_{DIFS} is the duration of DIFS, T_{SIFS} is the duration of Short Interframe Space (SIFS), T_{ACK} is the duration of an ACK packet, and δ is the prorogation delay. To capture the RTS/CTS mechanism, T_S and T_C can be further modified as those in [24]. Then the mean state duration E_S is estimated as

$$E_S = \sigma(1 - P_{tr}) + T_S \sum_{k=0}^{k} N_k Ps_k + T_C (P_{tr} - \sum_{k=0}^{k} N_k Ps_k), \quad (14)$$

where σ is the time unit defined by the IEEE 802.11 g/n standards.

We then employ transfer-function approach to evaluate the Probability Generating Function (PGF) of the MAC access delay. The average MAC access delay d_k is then estimated by Eq. (15). Then the average service rate μ_k is

$$\mu_k = 1/d_k. \quad (16)$$

The average goodput of a node in priority level k is

$$S_k = \min[Ps_k T_P R / E_S, \ \lambda_k], \quad (17)$$

where R is the Wi-Fi data rate selected for transmission.

5.4 The Queue Level Models

The queue level model consists of another set of DTMCs. We establish one DTMC to model the queueing process of one node in each priority level. We notice that queues with $\lambda_k \leq \mu_k$ evolve differently from queues with $\lambda_k > \mu_k$.

5.4.1 *The Queue Level Model of* $\lambda_k \leq \mu_k$

The DTMC for this case is presented in Fig. 4. Each state represents the queueing delay of the current head-of-line packet, which is measured by number of **time units** of length σ. We denote the expected interval between two consecutive packet arrivals as T_a, where $T_a = 1/\lambda_k$. We further define $T_1 = T_C/\sigma$ and $T_2 = (T_a - T_S)/\sigma$. When $\lambda_k \leq \mu_k$, we have $T_2 = (T_a - T_S)/\sigma \geq 0$.

There are three one-step transition probabilities for each state except state 0. (a) Let $P1_k$ denote the probability that the channel is busy and the head-of-line packet does not reach its re-transmission limit (thus is not dropped). In this case, the queueing delay increases by T_1 time units. (b) Let $P2_k$ denote the probability that the channel is idle. In this case, the queueing delay increases by 1 time unit. (c) Let $P3_k$ denote the probability that the head packet is successfully transmitted or is dropped after $m+1$ transmission attempts. In this case, the queueing delay of the head packet decreases by $\min[D, T_2]$ time units, where D is the current queueing delay.

For the special case of state 0, we have the following rules. From state 0, the DTMC stays in state 0 with probability $(1 - Pa_k) + Pa_k P3_k$, transits to state 1 with probability $Pa_k P2_k$, and transits to state T_1 with probability $Pa_k P1_k$.

Taking the outputs from the packet level model, we determine $P1_k$, $P2_k$, and $P3_k$ as follows

$$P1_k = Pc_k (1 - Pc_k^m \pi_{0,0,k}), \quad (18)$$
$$P2_k = (1-\tau_k)(1 - Pc_k), \quad (19)$$
$$P3_k = \tau_k (1 - Pc_k) + Pc_k \cdot Pc_k^m \pi_{0,0,k}, \quad (20)$$

where $\pi_{0,0,k}$ is the stationary probability of the state (0,0) in the DTMC of priority k from the packet level.

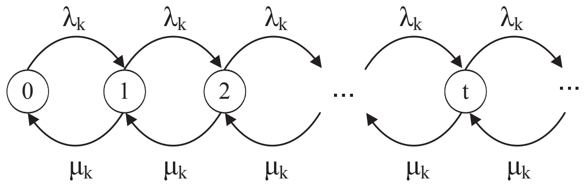


Figure 5: The DTMC for the queueing process of each node in priority level k when $\lambda_k > \mu_k$.

5.4.2 *The Queue Level Model of* $\lambda_k > \mu_k$

To model the queueing process of this case, we use a M/M/1 queue with an arrival rate of λ_k and a service rate of μ_k, as shown in Fig. 5. Each state in the DTMC in Fig. 5 represents a queue length. M/M/1 queues have been extensively studied. However, we still need a ready-to-use queueing delay analytical model for the case of $\lambda_k > \mu_k$.

5.5 Average Queueing Delay

Given the queue level model, λ_k and μ_k, we next derive the average queueing delay $\overline{D}$. Note that μ_k is derived in Section 5.3.

5.5.1 *The Average Queueing Delay of* $\lambda_k \leq \mu_k$

We focus on the DTMC shown in Fig. 4. Let $e_k(t)$ denote the stationary distribution of this DTMC, where t is the index of state. Then we have the following recurrence

$$\begin{aligned}d_k =&T_S[1-(Pc_k)^{m+1}]+T_C(Pc_k)^{m+1}+\frac{E_S(Pc_k)^{m+1}}{2}[W_{0,k}(2^{m+1}-1)-(m+1)]\\&+E_S\left\{W_{0,k}(1-Pc_k)\left[\frac{1-(2Pc_k)^{m+1}}{1-2Pc_k}-\frac{1-(Pc_k)^{m+1}}{2(1-Pc_k)}\right]-\frac{1-(m+2)(Pc_k)^{m+1}+(m+1)(Pc_k)^{m+2}}{2(1-Pc_k)}\right\}.\end{aligned}\tag{15}$$

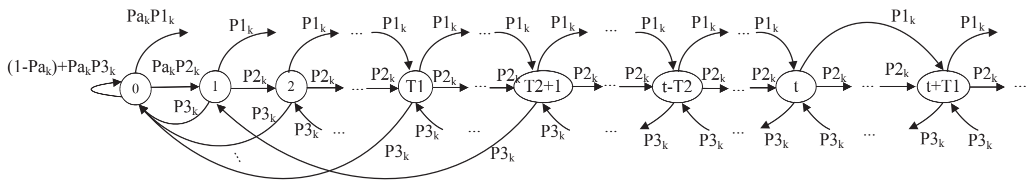


Figure 4: The DTMC for the queueing process of each node in priority level k when $\lambda_k \leq \mu_k$

equations

$$e_k(t)=\begin{cases}P1_k e_k(t-T_1)+P2_k e_k(t-1)\\ \quad +P3_k e_k(t+T_2), t\geq T_1+1,\\ P_a P1_k e_k(t-T_1)+P2_k e_k(t-1)\\ \quad +P3_k e_k(t+T_2), t=T_1,\\ P2_k e_k(t-1)++P3_k e_k(t+T_2), 2\leq t\leq T_1-1,\\ P_a P2_k e_k(t-1)++P3_k e_k(t+T_2), t=1,\\ \dfrac{P3_k}{Pa_k(1-P3_k)}\displaystyle\sum_{i=1}^{T_2} e_k(i), t=0.\end{cases}\tag{21}$$

The next lemma characterizes $e_k(t)$ when $t \geq T_1+1$.

Lemma 1. *The stationary distribution of the DTMC shown in Fig. 4 satisfies*

$$e_k(t)=A_1 r^t,\ t\geq T_1+1,\tag{22}$$

where A_1 is a constant, $0<r<1$, and r is a root of the following characteristic function

$$P1_k u+P2_k u^{T_1}+P3_k u^{T_1+T_2+1}-u^{T_1+1}=0.\tag{23}$$

We skip the detailed proof of this lemma, due to limited space.

With *Lemma 1* and Eq. 21, we can further derive equations (24), (25) and (26). By solving Eq. (24), (25) and (26), we obtain the values of $e_k(0)$, $e_k(T_1)$, and the constant A_1. All the $e_k(t), t=0,1,\cdots$, can be explicitly expressed by $e_k(0)$, $e_k(T_1)$, and A_1. Thus, the stationary distribution of the DTMC in Fig. 4 is achieved.

Denote the average queueing delay of packets transmitted by any node in priority level k as $\overline{D}_k$. Given the stationary distribution, $\overline{D}_k$ is estimated as

$$\overline{D}_k=\sigma\cdot\sum_{t=0}^{\infty} t\cdot e_k(t).\tag{27}$$

5.5.2 The Average Queueing Delay of $\lambda_k > \mu_k$

We focus on the DTMC shown in Fig. 5. We number the arrival packets by $1,2,\cdots$, in the order of their arrival time. Denote the arrival time of the nth packet as a_n, and the departure time of the nth packet as c_n. We focus on the case where the queue begins to evolve with no packet. (An initially non-empty queue can be easily modeled by changing the initial conditions of our model.) Then we have the following lemma on the expectations of a_n and c_n.

Lemma 2. *Let $\mathbf{E}(a_n)$ and $\mathbf{E}(c_n)$ be the expectations of arrival and departure time of the nth packet, respectively. When $\lambda_k > \mu_k$, we have the following equations for the queueing process presented in Fig. 5*

$$\lim_{n\to\infty}\mathbf{E}(a_n)/n = (\lambda_k+\mu_k)/\lambda_k,\tag{28}$$
$$\lim_{n\to\infty}\mathbf{E}(c_n)/n = (\lambda_k+\mu_k)/\mu_k.\tag{29}$$

The proof of this lemma is skipped due to limited space.

When $\lambda_k > \mu_k$, the average queueing delay keeps increasing as time goes by. Therefore, we focus on the average queueing delay of packets that are transmitted before a constant time unit T ($T<\infty$). To capture the delay up to T, we first need to calculate the number of clients transmitted before T. We have the following lemma for the average number of transmitted packets before time unit T.

Lemma 3. *Let N_T be the number of packets that are transmitted before T. When $\lambda_k > \mu_k$, we have the following equation for the queueing process presented in Fig. 5*

$$\lim_{T\to\infty} N_T/T=\mu_k/(\lambda_k+\mu_k).\tag{30}$$

We skip the proof because of the limited space.

Given *Lemma 2* and *Lemma 3*, we can characterize the average queueing delay for the case of $\lambda_k > \mu_k$ with the following theorem.

Theorem 1. *Let $\overline{D}_k(T)$ denote the average queueing delay (in seconds) of packets transmitted by a priority k node before time slot T. When $\lambda_k > \mu_k$, we have the following equation for the queueing process presented in Fig. 5*

$$\overline{D}_k(T)\xrightarrow{a.s.}\frac{\sigma T}{2}\cdot\frac{\lambda_k-\mu_k}{\lambda_k},\tag{31}$$

where $\xrightarrow{a.s.}$ means "converges almost surely".

Proof. By *Lemma 2*, we have the following equation for the queueing delay c_n-a_n of the nth packet

$$\begin{aligned}\mathbf{E}[c_n-a_n]&=\mathbf{E}[c_n]-\mathbf{E}[a_n]\\&\xrightarrow{a.s.}\frac{\lambda_k+\mu_k}{\mu_k}n-\frac{\lambda_k+\mu_k}{\lambda_k}n=\frac{\lambda_k^2-\mu_k^2}{\lambda_k\mu_k}n.\end{aligned}\tag{32}$$

And by *Lemma 3*, we have the following equation for the number of packets transmitted before T

$$\mathbf{E}[N_T]\xrightarrow{a.s.}\frac{\mu_k}{\lambda_k+\mu_k}T.\tag{33}$$

Take the length of each time slot as σ. Then combining Eq.

$$e_k(0) = \frac{P3_k}{P_a(1-P3_k)}\left\{\frac{1}{(1-P2_k)}\left[A_1 P3_k - kr^{T_2+1}\left(\frac{1-r^{T_1}}{1-r}\right) + P_a(P1_k+P2_k)e_k(0) - e_k(T_1)\right] + A_1 r^{T_1+1}\left(\frac{1-r^{T_2-T_1}}{1-r}\right)\right\}, \quad (24)$$

$$e_k(T_1) = \frac{A_1}{P2_k} r^{T_1+1} - P1_k P_a e_k(0) - \frac{P1_k P3_k}{P2_k} A_1 r^{T_2+1} - \frac{P3_k}{P2_k} A_1 r^{T_1+T_2+1}, \quad (25)$$

$$1 = \left[1 + \frac{P_a(1-P3_k)}{P3_k}\right] e_k(0) + \frac{Ar^{T_2+1}}{1-r}. \quad (26)$$

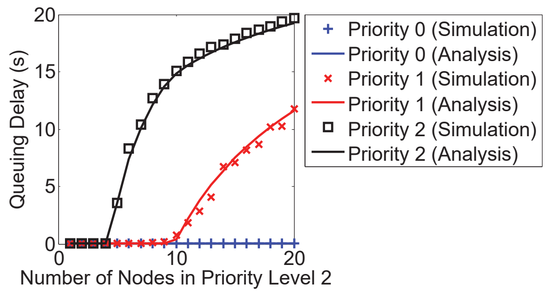


Figure 6: The model validation result.

(32) and Eq. (33), we have

$$\overline{D}_k(T) = \sigma \mathbf{E}\left[\frac{\sum_{i=1}^{N_T} i \cdot \frac{\lambda_k^2 - \mu_k^2}{\lambda_k \mu_k}}{N_T}\right]$$
$$\xrightarrow{a.s.} \sigma \cdot \frac{\mathbf{E}[N_T]+1}{2} \cdot \frac{\lambda_k^2 - \mu_k^2}{\lambda_k \mu_k}$$
$$\xrightarrow{a.s.} \frac{\sigma T}{2} \cdot \frac{\lambda_k - \mu_k}{\lambda_k}.$$

$\square$

5.6 Model Validation

In our model validation, there are 3 priority levels, i.e., $k = 0, 1, 2$. The minimum CW sizes are $W_{0,0} = 3$, $W_{0,1} = 7$, and $W_{0,2} = 15$, respectively. To present the validation results clearly, we fix the numbers of nodes in priority levels 0 and 1, and change the number of nodes in priority level 2. Concretely, we fix $N_0 = 2$ and $N_1 = 2$, and change N_2 from 1 to 20. The packet size is set to $500B$. For all the nodes, the packet arrival rate is $1Mbps$, which matches the standard bit rate of YouTube 360p videos. Each run of simulations lasts for a duration of 50 seconds, i.e., $T = 50$. All the queues are empty at the beginning of this 50-second duration. We use a date rate of $24Mbps$ as an example.

We validate our models with queueing delay in Fig. 6. Generally, it is shown that the queueing delays are well captured by our two-level models. One interesting observation is that the average queueing delay of nodes in priority level 2 increases suddenly at the point of $N_2 = 5$. This sudden increase of queueing delay also happens for nodes in priority level 1 at the point of $N_2 = 11$. When $\lambda_k < \mu_k$, packets are transmitted rapidly by nodes in priority level k. The queue length stays as a constant even if time goes to infinity. When $\lambda_k > \mu_k$, packets begin to accumulate in queues of nodes in priority level k, and the lengths of these queues keep increasing as time goes by. Therefore, instead of having a constant value, the average queueing delay becomes an increasing function of time, and will approach infinity if time approaches infinity. Note that this is captured by our queue level models.

Table 1: Candidate delay sets for DRIVING.

Setting	β_2	β_1	β_0
1	$[0,1)$	$[1,2)$	$[2,+\infty)$
2	$[0,1)$	$[1,4)$	$[4,+\infty)$
3	$[0,2)$	$[2,4)$	$[4,+\infty)$
4	$[0,2)$	$[2,10)$	$[10,+\infty)$
5	$[0,4)$	$[4,10)$	$[10,+\infty)$

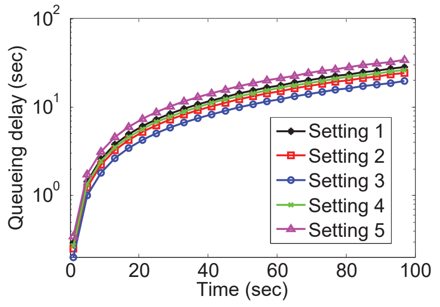


Figure 7: The evolution of queueing delays under different settings of DRIVING.

5.7 Tuning DRIVING with the Models

One important application of our analytical models is to tune the DRIVING for better performance before real deployment. There are three kinds of parameters to be tuned, i.e., the number of priority levels, their corresponding delay sets, and their corresponding minimum CW sizes. Due to the limited space, we only present the tuning of the delay sets while fixing the number of priority levels and the corresponding minimum CW sizes. Concretely, we use 3 priority levels, and set their minimum CW sizes as 3, 7, and 15, respectively. We compare five candidate delay sets as presented in Table 1, and select the best one from them.

Based on our two-level models, we calculate the evolution of queueing delays under these five settings. As an example, we set set packet size as $500B$, and the packet arrival rate as $1Mbps$. Fig. 7 illustrate how the queueing delays increase with time. It is shown that Setting 3, which is neither the most aggressive nor the most conservative, achieves the lowest queueing delay at all time. The reason is two-fold. On ond hand, an aggressive setting tends to push all nodes to the highest priority level quickly. As a result, the CW sizes of all nodes become too small to avoid packet collisions. On the other hand, a conservative setting tends to retain all nodes in at the lowest priority level. Consequently, the CW sizes of all nodes remain too large, and thus the channel utilization remains low.

6. PERFORMANCE EVALUATION

In this section, we conduct comprehensive simulation studies to illustrate the improvements brought by DRIVING. We first present the setup of our ns-2 evaluation platform. We then evaluate the performance of DRIVING in terms of delay, jitter, deadline missing ratio and goodput.

6.1 Evaluation Setup

The evaluation is conducted in our ns-2 based platform, and its setup is described as follows.

6.1.1 *The Highway Scenario*

We conduct our simulations in an urban bidirectional highway section. This highway section is of $3000m$ long. It has 2 lanes in each direction and an isolation zone in the middle. The vehicle number varies from 4 to 80. The movement traces of vehicles are generated using the tool of Simulation of Urban MObility (SUMO).

6.1.2 *In-cabin Wi-Fi Settings*

Each vehicle on the highway is equipped with an in-cabin Wi-Fi AP, which runs on the 2.4GHz band. We assume that all APs are working in the same 20MHz Wi-Fi channel. The transmission power of the APs is set to 20dBm. Each in-cabin Wi-Fi AP has a $16MB$ buffer for video streaming. We focus on the case where each Wi-Fi AP serves one client. We apply the two-slope vehicular prorogation model presented in [25]. The path loss factor introduced by the vehicle cabin is $-10dB$.

6.1.3 *Wireless Traffic Settings*

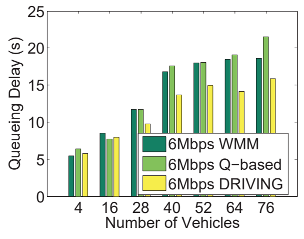
The video bit-rate in our simulations is $3.3Mbps$ (High-Definition video streaming). Each video packet carries 500 bytes of video data. Multiple packets are grouped into one large video chunk [26], which represents around 10 seconds of video. Video chunks arrive at in-cabin Wi-Fi APs periodically [27]. Besides the video streaming traffic, there are other kinds of data traffic in the wireless channel, including web browsing traffic, audio traffic and file sharing traffic. The percentages of video, web browsing, audio and file sharing in the whole traffic are 75%, 17%, 6% and 2%, respectively [1].

6.1.4 *Scheduling Algorithms*

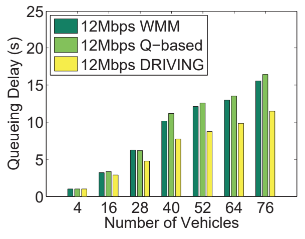
In the simulations, we compare the following three distributed algorithms.

The **DRIVING** algorithm is an implementation of the DRIVING framework, which adopts the tuning result from Section 5.7. It supports 3 levels of priorities (in consistence with the industrial practice, e.g., 802.11 EDCA.). It assigns priority 0 to the head-of-line packets with delays larger than 4 seconds, and allocate priority 2 to the head-of-line packets with delays smaller than 2 seconds. The rest of the head-of-line packets are with priority 1. The minimum CW sizes corresponding to priorities 0, 1 and 2 are $W_{0,0} = 3$, $W_{0,1} = 7$ and $W_{0,2} = 15$, respectively.

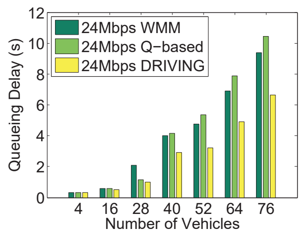
The **Q-based** algorithm is an extension from the queue-length based CSMA algorithms in [12, 18, 11]. To compare with the DRIVING algorithms, we update existing queue-length based CSMA algorithms into the Q-based algorithm as follows. The Q-based algorithm also supports 3 levels of priorities. The minimum CW sizes of transmissions in priorities 0, 1 and 2 are $W_{0,0} = 3$, $W_{0,1} = 7$ and $W_{0,2} = 15$, respectively. If the length of a queue is larger than $8MB$ (i.e., 50% of the AP's buffer size), the Q-based algorithm assigns the highest priority (i.e., priority 0) to the head-of-line packet in this queue. If the length of a queue is between $3.2MB$ and $8MB$ (i.e., $20-50\%$ of the AP's buffer size), the Q-based algorithm assigns priority 1 to the head-of-line packet in this queue. Otherwise, the head-of-line packet is with priority 2.



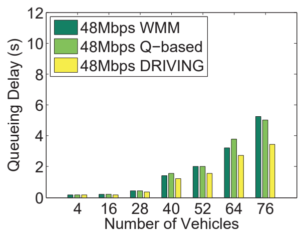
(a) Data rate: $6Mbps$



(b) Data rate: $12Mbps$



(c) Data rate: $24Mbps$



(d) Data rate: $48Mbps$

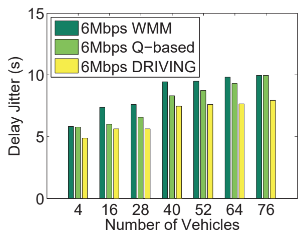
Figure 8: The average queueing delays.

The **WMM** algorithm follows the Wi-Fi Multimedia protocol, which is a Wi-Fi Alliance certification based on IEEE 802.11e standard. This algorithm prioritizes traffic according to four Access Categories (AC) - voice, video, best effort, and background.

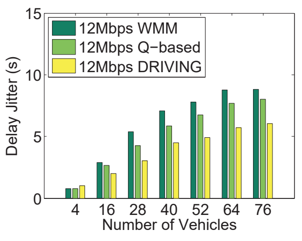
6.2 Delay and Jitter Performance

Fig. 8 compares the average queueing delays of the three algorithms. Compared to the second best algorithm, the DRIVING algorithm reduces the average queueing delay by up to 26.1%, 31.1%, 40.0% and 31.8% for transmission data rates of $6Mbps$, $12Mbps$, $24Mbps$ and $48Mbps$, respectively. Furthermore, even with the highest traffic density, DRIVING still results in the lowest queueing delay. This demonstrates that DRIVING is robust to the changes of traffic.

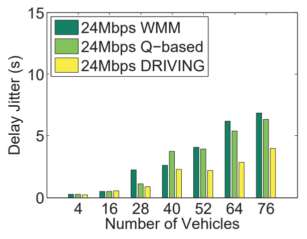
Another observation from Fig. 8 (b) and Fig. 8 (c) is worth noticing. The Q-based algorithm yields the largest average queueing delay when the number of vehicles is larger than 40. This indicates that the Q-based algorithm is not suitable for the delay sensitive service of video streaming in high traffic scenarios. The reason is as follows. When the scheduling decision is made based on the queue length, the packets in a large queue always has a higher priority than those in a small queue. Upon the arrival of a new video chunk to an AP Z_1, the queue length of this AP soars. Suppose there is another AP Z_2, who has only one largely delayed packet p. By the Q-based algorithm, packet p in AP Z_2 has a lower priority than those of the newly arrived packets in AP Z_1. Therefore, the largely delayed packet p has to wait for almost all the newly arrived packets in AP Z_1. Consequently, the delay of packet p becomes really large. On the contrary, the DRIVING algorithm avoids this problem by assigning higher priorities to packets with larger delays, and thus outperforms the other two algorithms, especially in high-density scenarios.



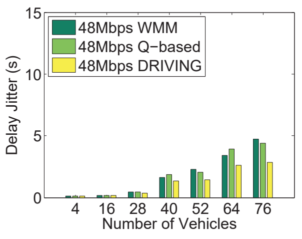
(a) Data rate: $6Mbps$



(b) Data rate: $12Mbps$



(c) Data rate: $24Mbps$



(d) Data rate: $48Mbps$

Figure 9: The delay jitters.

Fig. 9 compares the three algorithms in terms of the jitter. A smaller jitter of the queueing delay indicates a smoother video streaming, thus a better video viewing experience. Compared to the second best algorithm, the DRIVING algorithm reduces jitter by up to 20.3%, 28.8%, 47.4% and 35.6% for the transmission data rates of $6Mbps$, $12Mbps$, $24Mbps$ and $48Mbps$, respectively. Therefore, the DRIVING algorithm provides the most stable video streaming.

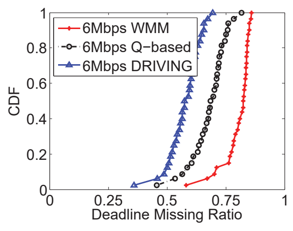
6.3 Deadline Missing Ratio Performance

In this section, we evaluate the performance of in-cabin Wi-Fi video streaming in terms of the deadline missing ratio.

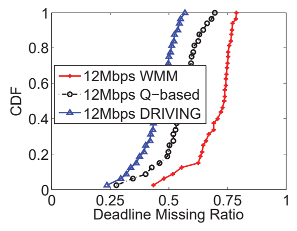
Fig. 10 compares the Cumulative Distribution Functions (CDFs) of the deadline missing ratios, when there are 80 vehicles. It is illustrated that, with the support of the DRIVING algorithm, the deadline missing ratio of in-cabin Wi-Fi video streaming is largely reduced. Compared to the second best algorithm, the DRIVING algorithm reduces the average deadline missing ratio by 19.0%, 29.4%, 38.4% and 37.9% for transmission data rates of $6Mbps$, $12Mbps$, $24Mbps$ and $48Mbps$, respectively. By using the queueing delay as the criterion of scheduling, the DRIVING algorithm is able to assign smaller CW sizes to packets that are approaching their deadlines. On average, these deadline-approaching packets are transmitted faster than others. In this way, the DRIVING algorithm reduces the deadline missing ratio.

6.4 Goodput Performance

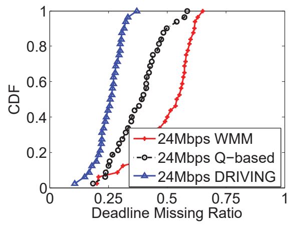
Fig. 11 compares the average goodputs of the three algorithms. It is shown that the DRIVING algorithm achieves the highest average goodput. Compared to the default algorithm, the DRIVING algorithm enlarges the average goodput by up to 27.0%, 25.1%, 26.1% and 22.1% (for the transmission data rates of $6Mbps$, $12Mbps$, $24Mbps$ and $48Mbps$, respectively. More importantly, the gap between DRIVING's average goodput and those of the other two algorithms increases with traffic density. Take the data rate of $24Mbps$ for example: DRIVING increases the goodput by 13.5% when there are 16 vehicles. The improvement increases to 26.1% when there are 76 vehicles. This suggests that, com-



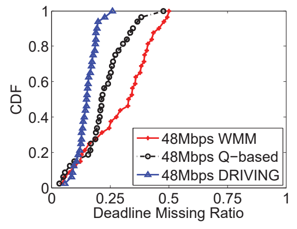
(a) Data rate: $6Mbps$



(b) Data rate: $12Mbps$

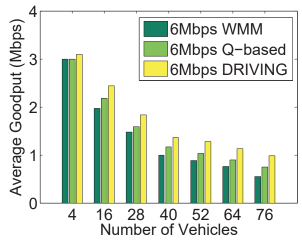


(c) Data rate: $24Mbps$

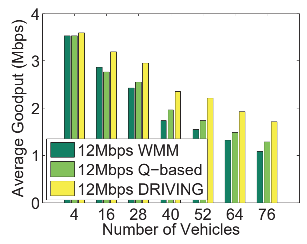


(d) Data rate: $48Mbps$

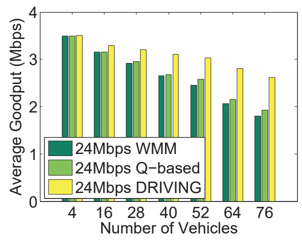
Figure 10: The CDFs of deadline missing ratios when the vehicle number is 80.



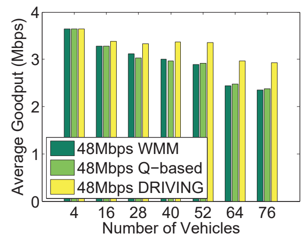
(a) Data rate: $6Mbps$



(b) Data rate: $12Mbps$



(c) Data rate: $24Mbps$



(d) Data rate: $48Mbps$

Figure 11: The average goodputs.

pared to the other two algorithms, the DRIVING algorithm is more capable of effective scheduling video streaming service in high-density traffic conditions.

7. CONCLUSION

This paper studies the problem of scheduling video streaming in the new in-cabin Wi-Fi systems. Considering the completely decentralized in-cabin Wi-Fi APs and the delay-sensitive video packets, we propose the fully distributed and delay-aware DRIVING framework. To reduce latency and minimize deadline missing ratio, DRIVING prioritizes packets with large queueing delays. We conduct extensive theoretical and simulation studies to evaluate and optimize DRIVING. In a typical scenario, DRIVING enlarges the average video streaming goodput by up to 27.0%, and at the same time lowers the average queueing delay and deadline missing ratio by up to 40.0% and 38.4%, respectively.

Acknowledgements

This work was supported in part by the NSERC Collaborative Research and Development Grant (CRDJP 418713). We also thank the anonymous reviewers and editors for their constructive comments.

8. REFERENCES

[1] *Cisco Visual Networking Index: Global Mobile Data Traffic Forecast Update, 2015 - 2020.* Cisco Systems Inc, Feb, 2016.

[2] X. Chen, L. Rao, Y. Yao, X. Liu, and F. Bai, "The answer is rolling on wheels: Modeling and performance evaluation of in-cabin Wi-Fi communications," *Vehicular Communications*, vol. 2, no. 1, pp. 13 – 26, 2015.

[3] I.-H. Hou and P. R. Kumar, "Utility-optimal scheduling in time-varying wireless networks with delay constraints," in *Proc. of ACM MobiHoc*, 2010.

[4] H. Wu, X. Lin, X. Liuz, and Y. Zhang, "Application-level scheduling with deadline constraints," *in Proc. of IEEE INFOCOM*, pp. 1–9, 2014.

[5] A. Dua, C. Chan, N. Bambos, and J. Apostolopoulos, "Channel, deadline, and distortion (CD^2) aware scheduling for video streams over wireless," *IEEE Trans. on Wireless Communications*, vol. 9, no. 3, pp. 1001–1011, 2010.

[6] P. Pahalawatta, R. Berry, T. Pappas, and A. Katsaggelos, "Content-aware resource allocation and packet scheduling for video transmission over wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 4, pp. 749–759, 2007.

[7] M. Caccamo, L. Zhang, L. Sha, and G. Buttazzo, "An implicit prioritized access protocol for wireless sensor networks," in *Proc. of IEEE RTSS*, 2002, pp. 39–48.

[8] V. Kanodia, C. Li, A. Sabharwal, B. Sadeghi, and E. Knightly, "Distributed priority scheduling and medium access in ad hoc networks," *Springer Journal of Wireless Networks*, vol. 8, no. 5, pp. 455–466, 2002.

[9] S.-P. Chuah, Z. Chen, and Y.-P. Tan, "Energy-efficient resource allocation and scheduling for multicast of scalable video over wireless networks," *IEEE Trans. on Multimedia*, vol. 14, no. 4, pp. 1324–1336, 2012.

[10] Y. Zhang, S. Qin, and Z. He, "Fine-granularity transmission distortion modeling for video packet scheduling over mesh networks," *IEEE Trans. on Multimedia*, vol. 12, no. 1, pp. 1–12, 2010.

[11] J. Ni, B. Tan, and R. Srikant, "Q-CSMA: Queue-length based CSMA/CA algorithms for achieving maximum throughput and low delay in wireless networks," in *Proc. of IEEE INFOCOM*, 2010.

[12] L. Jiang and J. Walrand, "Approaching throughput-optimality in distributed CSMA scheduling algorithms with collisions," *IEEE/ACM Trans. on Networking*, vol. 19, no. 3, pp. 816–829, 2011.

[13] P. Si, F. Yu, H. Ji, and V. Leung, "Distributed sender scheduling for multimedia transmission in wireless mobile peer-to-peer networks," *IEEE Trans. on Wireless Communications*, vol. 8, no. 9, pp. 4594–4603, 2009.

[14] C. Demichelis and P. Chimento, "IP packet delay variation metric for IP performance metrics (IPPM)," 2002.

[15] A. Gupta, X. Lin, and R. Srikant, "Low-complexity distributed scheduling algorithms for wireless networks," in *Proc. of IEEE INFOCOM*, 2007, pp. 1631–1639.

[16] D. Qian, D. Zheng, J. Zhang, N. Shroff, and C. Joo, "Distributed csma algorithms for link scheduling in multihop mimo networks under sinr model," *IEEE/ACM Trans. on Networking*, vol. 21, no. 3, pp. 746–759, 2013.

[17] L. Zhou, X. Wang, W. Tu, G. Muntean, and B. Geller, "Distributed scheduling scheme for video streaming over multi-channel multi-radio multi-hop wireless networks," *IEEE Journal on Selected Areas in Communications*, vol. 28, no. 3, pp. 409–419, 2010.

[18] A. Rad, J. Huang, M. Chiang, and V. Wong, "Utility-optimal random access without message passing," *IEEE Trans. on Wireless Communications*, vol. 8, no. 3, pp. 1073–1079, 2009.

[19] H. Omar, W. Zhuang, and L. Li, "Vemac: A tdma-based mac protocol for reliable broadcast in VANETs," *IEEE Trans. on Mobile Computing*, vol. 12, no. 9, pp. 1724–1736, 2013.

[20] F. Soldo, C. Casetti, C. Chiasserini, and P. Chaparro, "Video streaming distribution in VANETs," *IEEE Trans. on Parallel and Distributed Systems*, vol. 22, no. 7, pp. 1085–1091, 2011.

[21] Y. Yao, L. Rao, X. Liu, and X. Zhou, "Delay analysis and study of IEEE 802.11p based DSRC safety communication in a highway environment," in *Proc. of IEEE INFOCOM*, 2013.

[22] D. Malone, K. Duffy, and D. Leith, "Modeling the 802.11 distributed coordination function in nonsaturated heterogeneous conditions," *IEEE/ACM Trans. on Networking*, vol. 15, no. 1, pp. 159–172, Feb 2007.

[23] D. Rawat, D. Popescu, G. Yan, and S. Olariu, "Enhancing vanet performance by joint adaptation of transmission power and contention window size," *IEEE Trans. on Parallel and Distributed Systems*, vol. 22, no. 9, pp. 1528–1535, 2011.

[24] G. Bianchi, "Performance analysis of the IEEE 802.11 distributed coordination function," *IEEE Journal on Selected Areas in Communications*, vol. 18, no. 3, pp. 535–547, 2000.

[25] L. Cheng, B. Henty, D. Stancil, F. Bai, and P. Mudalige, "Mobile vehicle-to-vehicle narrow-band channel measurement and characterization of the 5.9 GHz Dedicated Short Range Communication (DSRC) frequency band," *IEEE Journal on Selected Areas in Communications*, vol. 25, no. 8, pp. 1501–1516, 2007.

[26] *ISO/IEC 23009-1: Information technology âĂŤ Dynamic adaptive streaming over HTTP (DASH)*, 2014.

[27] P. Ameigeiras, J. J. Ramos-Munoz, J. Navarro-Ortiz, and J. Lopez-Soler, "Analysis and modelling of YouTube traffic," *Trans. on Emerging Telecommunications Technologies*, vol. 23, no. 4, pp. 360–377, 2012.