Modeling Nonsaturated IEEE 802.11 DCF Networks Utilizing an Arbitrary Buffer Size

Qinglin Zhao, Danny H.K. Tsang, *Senior Member*, *IEEE*, and Taka Sakurai, *Member*, *IEEE*

Abstract—We propose an approximate model for a nonsaturated IEEE 802.11 DCF network. This model captures the significant influence of an arbitrary node transmit buffer size on the network performance. We find that increasing the buffer size can improve the throughput slightly but can lead to a dramatic increase in the packet delay without necessarily a corresponding reduction in the packet loss rate. This result suggests that there may be little benefit in provisioning very large buffers, even for loss-sensitive applications. Our model outperforms prior models in terms of simplicity, computation speed, and accuracy. The simplicity stems from using a renewal theory approach for the collision probability instead of the usual multidimensional Markov chain, and it makes our model easier to understand, manipulate and extend; for instance, we are able to use our model to investigate the important problem of convergence of the collision probability calculation. The remarkable improvement in the computation speed is due to the use of an efficient numerical transform inversion algorithm to invert generating functions of key parameters of the model. The accuracy is due to a carefully constructed model for the service time distribution. We verify our model using ns-2 simulation and show that our analytical results based on an M/G/1/K queuing model are able to accurately predict a wide range of performance metrics, including the packet loss rate and the waiting time distribution. In contradiction to claims by other authors, we show that 1) a nonsaturated DCF model like ours that makes use of decoupling assumptions for the collision probability and queuing dynamics can produce accurate predictions of metrics other than just the throughput, and 2) the actual service time and waiting time distributions for DCF networks have truncated heavy-tailed shapes (i.e., appear initially straight on a log-log plot) rather than exponential shapes. Our work will help developers select appropriate buffer sizes for 802.11 devices, and will help system administrators predict the performance of applications.

Index Terms—IEEE 802.11, fixed point analysis, nonsaturation, M/G/1/K.

1 Introduction

The popular IEEE 802.11 wireless LAN standard [1] uses a CSMA/CA mechanism called the Distributed Coordination Function (DCF) to regulate access to the shared medium. The operation of DCF is complicated due to its distributed, random-access nature, creating a need for accurate mathematical models to predict performance. Early performance evaluations [2], [3] of DCF made the simplifying assumption that each station always has a packet to transmit. While this saturation assumption is analytically appealing because it enables queuing dynamics to be ignored, real traffic flows are nonpersistent and do not give rise to true saturation. In recent years, performance evaluation under more realistic nonsaturated traffic (typically Poisson traffic) has attracted increasing attention [4], [5], [6], [7], [8], [9], [10], [11], [12]. In nonsaturated operation, the node transmit buffer size has a significant impact on the performance, notably in a transition regime from light to heavy traffic loads (see Fig. 1). However, previous work fails to adequately model this impact. In this paper, we develop an accurate model of a nonsaturated single-hop wireless LAN that yields a wide range of performance descriptors and correctly captures the influence of an arbitrary node buffer size. Furthermore, we devise computationally efficient methods to solve the model for the performance descriptors. The model is a nontrivial generalization of our previous work in [13] which considered small and infinite buffers. Our analysis is focussed on the basic access mode of DCF, but could be readily extended to the RTS-CTS mode.

At the heart of the analysis under saturation is a decoupling assumption that the collision probability is constant and independent between stations; this leads to a fixed-point formulation relating the per-station attempt rate with the collision probability of a packet [2], [14]. Several distinct approaches to arrive at a fixed-point formulation have been proposed, each leading to distinct defining equations. Bianchi [2] used an approach based on a bidimensional Markov chain representing backoff states, Tay and Chua [3] used mean-value arguments, while Kumar et al. [14] used a renewal theory approach.

Existing models for nonsaturated operation differ in approach and scope, but they are all in some way derived from a saturated fixed-point formulation. One simple approach is to assume that the number of active stations changes rather slowly and then to locally approximate each state using saturation results [8]. Another general approach is to modify a saturated fixed-point formulation to create a nonsaturated fixed-point formulation [4], [5], [6], [7], [9], [10], [11], [12]. In [4], a model for a short buffer is developed

- *Q. Zhao is with the Faculty of Information Technology, Macau University of Science and Technology, Avenida Wai Long, Taipa, Macau, China. E-mail: zqlict@hotmail.com.*
- *D.H.K. Tsang is with the Department of Electronic and Computer Engineering, Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong. E-mail: eetsang@ece.ust.hk.*
- *T. Sakurai is with the Department of Electrical and Electronic Engineering, The University of Melbourne, Melbourne, VIC 3010, Australia. E-mail: tsakurai@ieee.org.*

Manuscript received 28 Nov. 2009; revised 3 Aug. 2010; accepted 3 Sept. 2010; published online 20 Dec. 2010.
For information on obtaining reprints of this article, please send e-mail to: tmc@computer.org, and reference IEEECS Log Number TMC-2009-11-0519.
Digital Object Identifier no. 10.1109/TMC.2010.258.

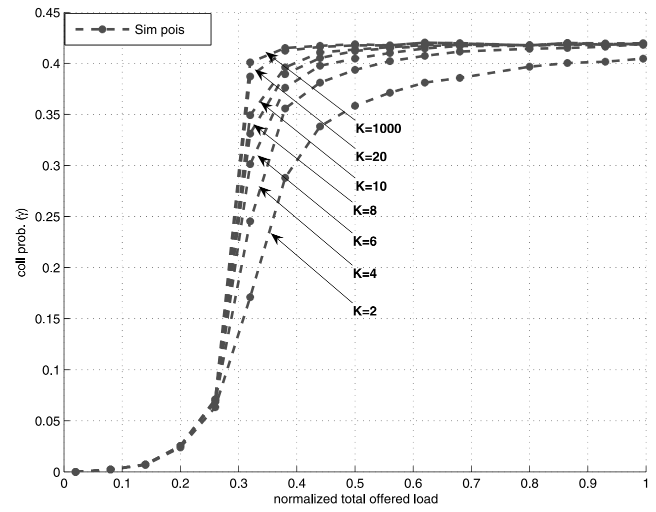


Fig. 1. The simulated collision probability versus the normalized total offered load when the number of contending nodes $n = 24$ and the buffer size $K = 2$, 4, 6, 8, 10, 20, and 1,000.

by augmenting the Bianchi Markov chain [2] with postbackoff states (where the station has transmitted a packet and begins a new backoff regardless of the presence of another packet to transmit). The analyses in [5], [6] treat Poisson traffic and an infinite buffer by modeling each station as an $M/G/1/\infty$ queue, and determine packet-level interactions on the channel using a Bianchi Markov chain extended with postbackoff states. In [7], each station is modeled as a discrete-time $G/G/1/\infty$ queue, and combined with a nonsaturated fixed-point formulation based on [3].

The models in [4], [5], [6], [7] are restricted to either small or infinite buffers. On the other hand, [9], [10], [11], [12] consider the more challenging case of a general buffer size. However, none of [9], [10], [11], [12] actually investigates the impact of different buffer sizes. For example, the buffer size in [9] is set to 40, the buffer size in [11] and [12] is set to 50, and while the authors in [10] consider different buffer sizes, they do not validate their analytical results with simulation. As we demonstrate in Fig. 1, the performance when the buffer size is larger than 20 is almost the same as that under the infinite buffer case.

The analysis in [9] uses a three-dimensional extension of the Bianchi Markov chain that explicitly tracks the buffer state of a station, as well as the number of other stations with a nonempty buffer. The models in [10], [11], [12] represent each station as an M/G/1/K queue, where K is the buffer size in packets, and model the backoff states of a station using a bidirectional Markov chain. The models in [9], [10] dispense with the decoupling assumption for the collision probability, which adds significantly to the modeling and computational complexity. Conversely, the model of [11] and its extension [12] preserve the collision probability decoupling assumption, and introduce a queue decoupling assumption whereby the degree of nonsaturation is approximated by the probability of a nonempty buffer in the M/G/1/K queue. The nonsaturated attempt rate is approximated by scaling the saturated attempt rate with the probability of a nonempty buffer (see (12)). The probability of a nonempty buffer depends on the packet arrival rate and the distribution of the packet service time.

In [13], we presented an accurate model for nonsaturated DCF networks that was restricted to small and infinite buffers. The current paper extends [13] to an arbitrary buffer size. Like [13], the nonsaturated model in this paper builds on the saturated renewal model of [14] and the scaling idea for the nonsaturated attempt rate. Since it bypasses the Bianchi Markov chain and its complicated interactions, our model is more compact than prior models, and is therefore easier to manipulate and extend. For instance, we show that it enables investigation of the problem of convergence of the collision probability calculation. The model can also be extended to characterize heterogeneous DCF networks and even 802.11e EDCA networks.

Extension to an arbitrary buffer size is not straightforward, because the extension raises a number of challenges, namely modeling accuracy, modeling complexity, computational complexity, and convergence of the fixed-point equation. In prior models, the latter three problems are never addressed, while the accuracy of these models is questionable, as we show in our numerical examples. Our main modeling contribution is a carefully constructed model for the service time that, in conjunction with the simple and intuitive expression for the saturated attempt rate from [14], leads to a fixed point approximation that yields accurate predictions for the nonsaturated collision probability. From this collision probability, we are able to obtain precise estimates of the throughput, and the mean and standard deviation of the MAC access delay, using an access delay model that we developed previously for the saturated setting [15]. Moreover, using standard results for the M/G/1/K queue, we show that we can obtain queuing statistics such as the mean total delay (the service time plus the waiting time), the packet loss rate due to buffer overflow, and the complementary cumulative distribution function (CCDF) of the waiting time. In fact, all performance descriptors that can be computed for an M/G/1/K queue can be computed for our system.

It is our belief that managing modeling and computational complexity in an analytical model is vital to maximise its utility and provide clear advantages over simulation. The complexity of nonsaturated DCF operation dictates that a certain degree of modeling complexity is unavoidable in any accurate performance model. Our M/G/1/K queue model is not simple, but it is decidedly more simple than [9], [10] because, unlike those models, it retains the decoupling assumption for the collision probability that is used for saturated modeling [2]. Furthermore, our model is simpler than [11], [12] because it uses the elegant fixed-point approach from [14] rather than the Bianchi Markov chain approach [2]. Nonetheless, to address the complexity issue, we also describe three model variants of our M/G/1/K model that trade accuracy for simplicity, namely 1) an M/M/1/K model for an arbitrary buffer size, 2) a simple model for the case of buffer size $K = 2$, and 3) a simple model for the case $K = \infty$. The model variants 2) and 3) originate from our paper [13] and are included in this paper for the purposes of comparison with our M/G/1/K and M/M/1/K models. These simpler models are only marginally less accurate than the M/G/1/K model for the collision probability, throughput, and mean delay statistics, but not so accurate for more subtle performance descriptors such as the distribution of the waiting time and the packet loss rate.

We address the issue of computational complexity by devising computationally efficient methods to solve for the collision probability in the M/G/1/K model. We use numerical inversion of the generating function to calculate the probability of a nonempty buffer, which is the main component of the computational cost. In prior work, there is no discussion on how to efficiently calculate the probability of a nonempty buffer. We prove that with our method, the time complexity of this main component is reduced to $O(K^2)$, or even $O(K)$ for low traffic loads, from $O((100n)^2 + K - 1)$ with the method advocated in [11], [12], where n is the number of contending nodes and K is the node buffer size. For large K, we also show that an acceleration technique can be used in the inversion algorithm to reduce the computational cost even further to $O(\widetilde{K}K)$, or $O(\widetilde{K})$ for low traffic loads, where $\widetilde{K} \ll K$. Thanks to the remarkable reduction in time complexity, when plotting an analytical curve for a given K according to our method, the computation time is reduced to several seconds from several hours using the method in [11], [12].

Under the M/M/1/K queuing analysis, we also consider a relaxed fixed point iteration to calculate the collision probability and find sufficient conditions for convergence. As far as we know, no paper in the literature discusses the convergence of the fixed point under nonsaturated operation.

To confirm the accuracy of our models, we focus on Poisson traffic and present extensive numerical examples comparing outputs of our models with ns-2 simulation, as well as with outputs of the models in [4], [5], [7], [8], [11], [12]. We show that our M/G/1/K model is more accurate than any of the existing models, and captures the effect of the buffer size. Our analytical results precisely predict the collision probability, the mean and variance of the MAC access delay, the mean total delay, the packet loss rate, and the CCDF of the waiting time. We find that increasing the buffer size can improve the throughput slightly, but can lead to a dramatic increase in the packet delay without necessarily a reduction in the packet loss rate. We also show that for a large buffer size, our much simpler M/M/1/K model has comparable accuracy to the M/G/1/K model for the collision probability, the mean and variance of the MAC access delay, and the mean total delay. The M/M/1/K model exhibits inaccuracy when the buffer is small and the offered traffic approaches saturation. Regarding non-Poisson traffic, prior work [4], [13] has illustrated that the 802.11 nonsaturated performance is largely insensitive to the input traffic arrival distribution. Therefore, Poisson traffic can serve as a proxy for other traffic types with the same mean arrival rate, including CBR and on-off traffic.

The accuracy of our model for a wide range of performance indicators contradicts the view put forward in [16] that a model like ours that uses a queue decoupling assumption can only be expected to be accurate for throughput estimates, and the conclusion in [9] that maintaining the collision probability decoupling assumption cannot produce accurate results for the nonsaturated analysis. Furthermore, our results show that the service time and waiting time distributions of DCF networks have truncated heavy-tailed shapes rather than exponential shapes, as claimed in [11].

The rest of this paper is organized as follows: In Section 2, we first outline the backoff procedure in the IEEE 802.11 DCF and then analyze the MAC service time under saturation. Section 3 presents a general fixed-point formulation in terms of an arbitrary buffer size. Section 4 presents the formulas for the throughput, the mean and variance of the MAC access delay, the Laplace transform of the waiting time distribution, the mean total delay, and the packet loss rate. In Section 5, the theoretical results are verified via ns2 simulation and compared with related work. Section 6 concludes the paper. Finally, Appendix A presents the proofs of Theorem 2 and Corollary 1, whereas Appendix B gives a table of main notation and associated equation numbers.

2 The 802.11 MAC Protocol and MAC Service Time for a Nonempty Buffer

Before developing the nonsaturated model in Section 3, we briefly review the operation of DCF and then derive the MAC service time under the condition that the buffer is nonempty. This result plays a critical role in the accurate modeling of nonsaturated operation.

We first introduce the terminology and assumptions that will be used in our analysis: a packet transmission is said to be *finished* when the packet is either successfully received at the destination node or dropped due to reaching a retransmission limit; the *service time* of a packet is defined as the interval between when a packet becomes the head-of-line packet in the buffer and when the packet transmission is finished; the *MAC access delay* is defined as the interval between when a packet becomes the head-of-line packet in the buffer and when the packet is successfully received at the destination node; time is measured in slots unless explicitly indicated. Similar to [2], [14] and most other saturated and nonsaturated analyses, we assume that 1) all nodes reside in a single-cell network (i.e., all stations are in sensing range of each other), 2) the collision processes of the nodes can be decoupled, such that the collision probability experienced by each node is constant and identical, and 3) channel conditions are ideal so that transmission errors are a result of packet collision only. In common with other nonsaturated analyses, we assume that the interarrival time (in slots) of packets from the upper layers into the transmit buffer of each station is independent and identically distributed, and we focus on homogeneous DCF networks, where all stations have the same protocol parameter values (such as the initial window size and the multiplier of the exponential backoff), the same traffic parameter values (namely, traffic arrival rate and packet size) and the same buffer size. We also assume that the buffer size is measured in units of packets.

2.1 IEEE 802.11 DCF

The IEEE 802.11 DCF [1] is a binary exponential backoff protocol. The event of finishing a packet transmission triggers slightly different protocol actions depending on whether or not the buffer is empty.

If the buffer is not empty (e.g., in saturated operation), the node immediately enters the initial backoff stage, wherein the node generates a random backoff count uniformly distributed in $[0, \mathrm{CW}_0 - 1]$, where CW_0 is a given minimum contention window (CW) size. Thereafter, the backoff counter decreases by one for each idle time slot and is suspended when the channel is busy. The suspended backoff counter resumes after the channel is sensed idle for a DCF interframe space (DIFS). When the backoff counter reaches zero, the node starts the transmission of the next data packet at the beginning of the next time slot. For each successful transmission, the sender will receive an acknowledgement (ACK) frame after a short interframe space (SIFS). If the node does not receive the ACK within a certain time (i.e., ACK timeout), it assumes that the data packet were not successfully received at the target node and doubles the CW and repeats the above procedure. Doubling of the CW stops after the maximum window size $\mathrm{CW}_{\max}$ is reached. When a retransmission limit M is reached, the sender drops the data packet.

If the buffer is empty, the node also enters the initial backoff stage. At the end of the initial backoff stage, if the buffer is not empty, the node will immediately transmit the next data packet; otherwise, the node will remain idle. During the idle duration, when a new packet arrives, the node will immediately transmit the new packet if it senses an idle channel for a DIFS period. If the channel is sensed busy or any transmission fails, the node doubles the CW and executes the nonempty buffer procedure described above.

2.2 MAC Service Time Conditioned on a Nonempty Buffer

In this section, we develop an approximate expression for the conditional MAC service time and calculate its mean, generating function, and probability mass function. Unless otherwise specified, we adopt the following notational conventions in our analysis: if ξ is a random variable, we let $\overline{\xi}$ denote the mean of ξ; if ξ is a nonnegative, integer-valued random variable, we denote the probability mass function of ξ by $P(\xi = x)$ and the generating function of ξ by

$$\widehat{\xi}(z) = \sum_{x=0}^{\infty} P(\xi = x) z^x, \text{ for } z \in \mathbb{C}.$$

Note that if ξ is not integer-valued, it can be easily discretized to an integer-valued random variable by defining a lattice, with spacing τ say, such that the values of ξ fall on the lattice points $(0, \tau, 2\tau, \dots)$. The computational cost of inverting the generating function increases with decreasing τ, so it is sometimes prudent to choose a larger τ and tolerate some rounding for the values of ξ. In IEEE 802.11 DCF, time is measured in slots, where one slot $= 20\ \mu$s. Therefore, it is a natural choice to set $\tau = 20\ \mu$s.

First, we summarize the model used for the saturated collision probability and attempt rate, which is from [14]. Let γ denote the collision probability experienced by a tagged node on the condition that the buffer is not empty. Let β^c denote the attempt rate per slot for each node (i.e., the ratio of the number of attempts to the time spent in backoff measured in slots) on the condition that the buffer is not empty. Applying the result in [14], we can express $\beta^c \triangleq \beta^c(\gamma)$ as follows:

$$\beta^c(\gamma) = \frac{1 + \gamma + \gamma^2 + \cdots + \gamma^{M-1}}{b_0 + \gamma b_1 + \gamma^2 b_2 + \cdots + \gamma^{M-1} b_{M-1}}, \qquad (1)$$

where M is the retransmission limit and b_k is the mean backoff time of stage k for each node. On the other hand, according to the decoupling assumption [2], the collision probability γ can be expressed as follows in terms of β^c:

$$\gamma = \Gamma(\beta^c(\gamma)) \triangleq 1 - (1 - \beta^c(\gamma))^{n-1}, \qquad (2)$$

where n $(n \geq 2)$ is the number of contending nodes. Substituting (1) into (2), we obtain a fixed-point equation in terms of γ. We call the solution of the fixed-point equation the saturated collision probability. Thereafter, we let γ^s denote the saturated collision probability and let β^s denote the saturated attempt rate, where $\beta^s = \beta^c(\gamma^s)$.

Let Y^c be the service time (in slots) of a packet of a tagged node on the condition that the buffer is not empty. We now express the MAC service time in terms of γ and β^c.

The expression for Y^c. Let X be a random variable representing the backoff count (measured in decrements of the backoff counter) that elapses before a packet transmission of the tagged node is finished. Let Ω be a random variable representing the time (in slots) that elapses for one decrement of the backoff counter. Since the tagged node waits for a duration of Ω per backoff decrement and it must observe a backoff count of X before its packet transmission finishes, Y^c is given by

$$Y^c = \sum_{i=1}^{X} \Omega. \qquad (3)$$

We now find an expression for the backoff count X. Depending on the collision probability γ, a packet may undergo up to M backoff stages before its transmission is finished. Therefore, the random variable X is equal to the sum of the total backoff count spent by the tagged node in different possible subsets of the M backoff stages, and the probability of each subset can be expressed in terms of the collision probability γ. We define X as

$$X = \sum_{k=0}^{j} \eta_k, \text{ w.p. } \delta(\gamma, j), 0 \leq j \leq M-1,$$
$$\text{where } \delta(\gamma, j) = \begin{cases} (1-\gamma)\gamma^j, & j = 0, \dots, M-2, \\ \gamma^{M-1}, & j = M-1, \end{cases} \qquad (4)$$

where "w.p." means "X is equal to $\sum_{k=0}^{j} \eta_k$ with probability $\delta(\gamma, j)$." In (4), η_k is uniformly distributed in $[0, CW_k - 1]$ with mean b_k, where $CW_k = \nu^k CW_0$ for $0 \leq k \leq m-1$ and $CW_k = \nu^m CW_0$ for $m \leq k \leq M-1$; CW_0 is the minimum window size (in slots); ν (>1) is the multiplier of the exponential backoff; m determines the maximum backoff window size $\mathrm{CW}_{\max}$ (i.e., $\mathrm{CW}_{\max} = \nu^m \mathrm{CW}_0$ and $\nu = 2$ in the standard); M is the retransmission limit; and $\delta(\gamma, j)$ is the probability that the packet transmission finishes at the jth backoff stage.

The generic slot duration Ω depends on whether a slot is idle or interrupted by a successful transmission or a collision. We define Ω as

$$\Omega = \begin{cases} \sigma, & \text{w.p. } 1 - P_b, \\ T_s + \sigma, & \text{w.p. } P_s, \\ T_{\bar{s}} + \sigma, & \text{w.p. } P_{\bar{s}}, \end{cases} \qquad (5)$$

where

$$\begin{aligned} P_b &= 1-(1-\beta^c)^n = 1-(1-\gamma)^{\frac{n}{n-1}}, \\ P_s &= n\beta^c(1-\beta^c)^{n-1} = n(1-(1-\gamma)^{\frac{1}{n-1}})(1-\gamma), \\ P_{\bar{s}} &= P_b - P_s, \end{aligned} \tag{6}$$

denote the probability of a busy slot, the probability of a successful transmission from any of the n contending nodes, and the probability of an unsuccessful transmission from any of the n contending nodes, respectively; $\sigma = 1$ slots $= 20\,\mu s$; and T_s and $T_{\bar{s}}$ are the mean time (in slots) for a successful transmission and an unsuccessful transmission, respectively. The parameters T_s and $T_{\bar{s}}$ depend on packet payload length, SIFS, DIFS, and other protocol parameters. Note we have explicitly expressed Ω in terms of γ. Since the backoff counter must decrease one slot before the next decrease, to be strictly correct, we add one slot in each of the last two terms of Ω.

The expression for $\overline{Y}^c$. Since X and Ω depend on γ, Y^c also depends on γ. Let $\overline{X} \triangleq \overline{X}(\gamma), \overline{\Omega} \triangleq \overline{\Omega}(\gamma)$, and $\overline{Y}^c \triangleq \overline{Y}^c(\gamma)$ denote the mean of X, Ω, and Y^c, respectively. From (3), we have

$$\overline{Y}^c(\gamma) = \overline{X}(\gamma) \cdot \overline{\Omega}(\gamma). \tag{7}$$

In (7), $\overline{X}(\gamma)$ is calculated by

$$\overline{X}(\gamma) = b_0 + \gamma b_1 + \gamma^2 b_2 + \cdots + \gamma^{M-1} b_{M-1}. \tag{8}$$

Note that the denominator in (1) is just the mean backoff time $\overline{X}(\gamma)$ in (8). On the other hand, $\overline{\Omega}(\gamma)$ is calculated by

$$\overline{\Omega}(\gamma) = \sigma + P_s T_s + P_{\bar{s}} T_{\bar{s}}. \tag{9}$$

The expression for $\widehat{Y^c}$. Since Y^c depends on γ, we write $\widehat{Y^c} \triangleq \widehat{Y^c}(\gamma, z)$ to denote the generating function of Y^c. From (3) and [17], $\widehat{Y^c}(\gamma, z)$ is calculated by

$$\widehat{Y^c}(\gamma, z) = \widehat{X}(\gamma, \widehat{\Omega}(\gamma, z)), \tag{10}$$

where from (4) and (5), $\widehat{X}(\gamma, z)$ and $\widehat{\Omega}(\gamma, z)$ are given by

$$\begin{aligned} \widehat{X}(\gamma, z) &= \sum_{i=0}^{M-1}\left[\delta(\gamma, i)\prod_{k=0}^{i}\widehat{\eta}_k(z)\right], \\ \widehat{\Omega}(\gamma, z) &= (1-P_b)z^{\sigma} + P_s z^{T_s+\sigma} + P_{\bar{s}} z^{T_{\bar{s}}+\sigma}, \\ \widehat{\eta}_k(z) &= \begin{cases} \frac{1}{CW_k}\frac{1-z^{CW_k}}{1-z}, & k = 0, \ldots, m-1, \\ \frac{1}{CW_m}\frac{1-z^{CW_m}}{1-z}, & k = m, \ldots, M-1. \end{cases} \end{aligned}$$

The calculation of $\Pr\{Y^c = h\}$ for $h = 1, 2, \ldots$. To get the probability mass function of Y^c from $\widehat{Y^c}$, we use the lattice-Poisson numerical inversion algorithm developed in [18]. The inversion formula used in this algorithm is

$$\Pr\{Y^c = h\} \approx \frac{1}{2hlr^h}\sum_{j=-hl}^{hl-1}\widehat{Y^c}(\gamma, re^{-i\pi j/(hl)})e^{i\pi j/l}, \tag{11}$$

where h is in slots, and real r and integer l are parameters of the inversion algorithm. The results we present in Section 5 are calculated using $l = 1$ and $r = 10^{-4/h}$. From [18], these settings for l and r result in a numerical inversion error less than 10^{-8}.

Remarks. The authors in [11] suggest that the service time distribution in DCF can be approximated by an exponential distribution. However, our extensive numerical experiments indicate that the service time distribution actually has a truncated heavy-tailed shape. For details, please refer to the explanations of Fig. 11 in Section 5.

3 The Collision Probability for an Arbitrary Buffer Size Under Nonsaturated Operation

Armed with the results for the conditional MAC service time derived in the previous section, we are ready to model the backoff procedure for nonsaturated operation for an arbitrary buffer size (i.e., to model the attempt rate and the collision probability for an arbitrary buffer size).

The node buffer size has a significant impact on the performance of nonsaturated DCF networks. Fig. 1 plots the simulated collision probability versus the normalized total offered load for different node buffer sizes. We observe that, in a transition regime between light and heavy traffic loads, the collision probability increases significantly with the buffer size. The reason is that when the buffer size increases, more packets are backlogged, and these backlogged packets will contend for the shared channel and thereby cause more collisions. In turn, these collisions dramatically affect the throughput and the delay. Therefore, a precise characterization of the performance for an arbitrary buffer size requires an accurate model for the collision probability. The key to modeling the collision probability lies in devising a model for the attempt rate. To do so, we adopt the following approximation idea.

The key approximation. As summarized in Section 2.1, the backoff procedure executed by a DCF node is the same irrespective of whether it is saturated or nonsaturated. If we ignore the time when the backoff counter is idle due to the absence of a packet, the only difference between the nonsaturated case and the saturated case is that the mean number of backoff stages in the former is less than the mean number in the latter. This inspires us to approximate the attempt rate of the nonsaturated case by scaling the attempt rate of the saturated case with the probability of a nonempty buffer (see (12)). As a consequence, we disregard the details of the postbackoff state interactions in nonsaturated operation.

The same approximation idea has been adopted in [11] and its extension [12]. Our main modeling contribution is a carefully constructed model for the service time derived in (3) that, in conjunction with the simple and intuitive expression for the saturated attempt rate from [14], leads to a fixed point approximation that yields accurate predictions for the nonsaturated collision probability. More detailed comparisons between our model and the model in [11], [12] are given in Section 3.3.

3.1 The General Fixed-Point Equation

This section develops a simple general fixed-point equation that governs the collision probability under nonsaturated operation. Recall that β^c is the average attempt rate per slot for each node (i.e., the ratio of the number of attempts to the time spent in backoff measured in slots) on the condition that the buffer is not empty. Let p_0 be the probability of an empty buffer. Let β ($0 \leq \beta \leq 1$) be the general (or average) attempt rate per slot for each node. Noting that the attempt rate is equal to zero when the buffer is empty, we propose that

$$\beta = (1 - p_0)\beta^c. \tag{12}$$

In other words, we assume that the general attempt rate β is equal to the product of the system utilization $(1 - p_0)$ and the conditional attempt rate β^c.

We assume that the decoupling assumption commonly used for saturated performance analysis [2], [14] can be extended to the nonsaturated setting. We can then express the general collision probability γ experienced by a tagged node under nonsaturated operation using the same mathematical form as for the saturated case (2), namely:

$$\gamma = \Gamma(\beta) = 1 - (1 - \beta)^{n-1}. \tag{13}$$

Note, however, the subtle difference between how the general collision probability γ and the saturated collision probability γ^s are computed; γ is the solution of the fixed-point equation $\gamma = \Gamma(\beta)$, while γ^s is the solution of the fixed-point equation $\gamma = \Gamma(\beta^c)$.

To complete the fixed-point equation governed by (12) and (13), we need to express β (that is, β^c and p_0) in terms of the general collision probability γ. Since $\beta^c \triangleq \beta^c(\gamma)$ is already given in (1), we only need to express p_0 in terms of γ.

In queuing theory, p_0 is naturally connected to the traffic intensity ρ and can be expressed in terms of ρ. In our nonsaturated model, we define $\rho \triangleq \rho(\gamma)$ as

$$\rho(\gamma) = \lambda\overline{Y}^c(\gamma) = \lambda\overline{X}(\gamma)\overline{\Omega}(\gamma), \tag{14}$$

where $\overline{Y}^c(\cdot)$ is given by (7) and λ is the parameter of the Poisson arrival process and, in our time-slotted setting, represents the packet arrival rate per slot. Then, to complete the fixed-point equation governed by (12) and (13), what we do next is to express p_0 in term of ρ using well-established results for the M/G/1/K queue.

3.1.1 *Computation of p_0 Based on the M/G/1/K Queue*

The parameters introduced above for a DCF node translate in a natural way to parameters of an M/G/1/K queue. In particular, packets awaiting transmission represent jobs in the queue, the Poisson distribution with parameter λ describes the packet arrival process to the queue, the MAC service time Y^c represents the service process of a job in the server, and $\rho(\gamma) = \lambda\overline{Y}^c(\gamma)$ describes the traffic intensity. As explained previously, Zhai et al. [11] and Zheng et al. [12] also use an M/G/1/K queue model, and we first describe their method of computation before describing a simpler method that we employ.

Method used in [11] and [12]. Let p_k be the steady-state probability that there are k packets present in the transmit buffer of a node at an arbitrary time, where $k = 0, 1, 2, \ldots, K$. Let π_k be the probability that there are k packets present in the buffer after a packet's transmission is finished, where $k = 0, 1, 2, \ldots, K - 1$. From [19], p_k can be calculated using

$$p_k = \begin{cases} \frac{\pi_k}{\pi_0+\rho}, & k = 0, 1, \ldots, K-1, \\ 1 - \frac{1}{\pi_0+\rho}, & k = K. \end{cases} \tag{15}$$

In (15), $\pi = \{\pi_k; 0 \le k \le K-1\}$ can be calculated by solving the system of linear equations below:

$$\pi = \pi \begin{pmatrix} a_0 & a_1 & a_2 & \cdots & a_{K-2} & 1 - \sum_{j=0}^{K-2} a_j \\ a_0 & a_1 & a_2 & \cdots & a_{K-2} & 1 - \sum_{j=0}^{K-2} a_j \\ 0 & a_0 & a_1 & \cdots & a_{K-3} & 1 - \sum_{j=0}^{K-3} a_j \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ \cdots & \cdots & \cdots & \cdots & a_0 & 1 - a_0 \end{pmatrix}, \tag{16}$$

where $a_k \triangleq a_k(\lambda, Y^c)$ is given by

$$\begin{aligned} a_k &= \Pr\{k \text{ packets arrive during MAC service time } Y^c\} \\ &= \sum_{h=0}^{\infty} \frac{(\lambda h)^k}{k!} e^{-\lambda h} \Pr\{Y^c = h\} \\ &\approx \sum_{h=0}^{I_{\max}} \frac{(\lambda h)^k}{k!} e^{-\lambda h} \Pr\{Y^c = h\}. \end{aligned} \tag{17}$$

In (17), h is measured in slots, $\Pr\{Y^c = h\}$ is given by (11), and $I_{\max}$ is set to some appropriately large number. Note that from (17), we have

$$a_0 = \widehat{Y}^c(\gamma, e^{-\lambda}), \tag{18}$$

which will be useful later.

In [11] and [12], the expressions for β^c, Ω, and Y^c differ from ours. Furthermore, the authors use (17) to calculate a_k but provide no explanation as to how to obtain $\Pr\{Y^c = h\}$ from $\widehat{Y}^c$. For the comparative results presented in Section 5 for the model of [11], [12], we use numerical inversion with the lattice-Poisson algorithm to find $\Pr\{Y^c = h\}$.

Method used in this paper. Let $p_k \triangleq p_k(\lambda, Y^c)$. Tijms [20] introduces a simpler approach to calculate p_k for the M/G/1/K queue. From [20], p_k can be directly calculated using the recursive equations below:

$$\begin{aligned} p_k &= \frac{p'_k}{p'_0 + p'_1 + \cdots + p'_K}, \quad k = 0, 1, \ldots, K, \\ p'_k &= \begin{cases} 1, & k = 0, \\ \frac{\lambda a'_{k-1} + \lambda \sum_{j=1}^{k-1} p'_j a'_{k-j}}{1 - \lambda a'_0}, & k = 1, \ldots, K-1, \\ \rho - (1-\rho)\sum_{j=1}^{k-1} p'_j, & k = K, \end{cases} \end{aligned} \tag{19}$$

where a'_k is given by

$$a'_k = \sum_{h=0}^{\infty} \frac{(\lambda h)^k}{k!} e^{-\lambda h} \Pr\{Y^c \ge h\}. \tag{20}$$

We construct the generating function $\widehat{a'} \triangleq \widehat{a'}(\lambda, z)$ of the sequence $\{a'_k : k = 0, 1, 2, \ldots\}$:

$$\begin{aligned} \widehat{a'} &= \sum_{k=0}^{\infty} a'_k z^k \\ &= \sum_{k=0}^{\infty} \left[\sum_{h=0}^{\infty} \frac{(\lambda h)^k}{k!} e^{-\lambda h} \Pr\{Y^c \ge h\} \right] z^k \\ &= \sum_{h=0}^{\infty} \Pr\{Y^c \ge h\} e^{-\lambda h} \sum_{k=0}^{\infty} \frac{(\lambda z h)^k}{k!} \\ &= \sum_{h=0}^{\infty} \Pr\{Y^c \ge h\} e^{-(1-z)\lambda h} \\ &= \frac{1 - e^{-(1-z)\lambda} \widehat{Y}^c(\gamma, e^{-(1-z)\lambda})}{1 - e^{-(1-z)\lambda}}. \end{aligned}$$

Finally, we calculate a'_k in (20) by inverting $\widehat{a'}$ using the lattice-Poisson algorithm:

$$a'_0 = \widehat{a'}(\lambda, 0),$$
$$a'_k \approx \frac{1}{2klr^k}\sum_{j=-kl}^{kl-1} \widehat{a'}(\lambda, re^{-i\pi j/(kl)})e^{i\pi j/l}, \quad k = 1, 2, \ldots, K-1. \tag{21}$$

3.1.2 Computation of p_0 Based on the M/G/1/∞ Analysis

When $\rho = \lambda\overline{Y}^c < 1, p_0$ can be calculated using an approach far simpler than that in Section 3.1.1 without introducing any additional approximations nor sacrificing any accuracy. The basic idea is to exploit the fact that p_k in the M/G/1/K queue can be calculated exactly using results for the M/G/1/∞ queue [20]. Let π_k^∞ be the probability that there are k packets present in the infinite buffer of a tagged node after it finishes a packet transmission. From [20], p_k can be calculated as follows:

$$p_k = \begin{cases} \frac{\pi_k^\infty}{1-\rho q_K}, & k = 0, 1, \ldots, K-1, \\ \frac{(1-\rho)q_K}{1-\rho q_K}, & k = K. \end{cases} \tag{22}$$

In (22), q_K and π_k^∞ are further calculated using

$$q_K = \sum_{j=K}^{\infty} \pi_j^\infty = 1 - \sum_{j=0}^{K-1} \pi_j^\infty,$$
$$\pi_k^\infty = \begin{cases} 1-\rho, & k = 0, \\ \frac{a'_{k-1}\pi_0^\infty + \sum_{j=1}^{k-1} a'_{k-j}\pi_j^\infty}{1-a'_0}, & k = 1, \ldots, K-1, \end{cases} \tag{23}$$

where a'_k is shown in (20).

Now, from (22), p_0 is given by

$$p_0 = \frac{1-\rho}{1-q_K\rho}. \tag{24}$$

To obtain p_0, we only need to calculate q_K. To this end, we find the generating function of the sequence $\{\pi_k^\infty : k = 0, 1, 2, \ldots\}$ [20]:

$$\widehat{\pi}^\infty \triangleq \widehat{\pi}^\infty(\lambda, z) = \frac{(1-\rho)(1-z)\widehat{Y}^c(\gamma, e^{-(1-z)\lambda})}{\widehat{Y}^c(\gamma, e^{-(1-z)\lambda}) - z}.$$

Then, applying the lattice-Poisson algorithm, q_K can be calculated by

$$q_K \approx \frac{1}{2Klr^K}\sum_{j=-Kl}^{Kl-1} \widehat{q}(\lambda, re^{-i\pi j/(Kl)})e^{i\pi j/l}, \tag{25}$$

where $\widehat{q}(\lambda, z) = \frac{1-z\widehat{\pi}^\infty(\lambda,z)}{1-z}$.

In the sequel, we will refer to the method described above as the M/G/1/∞ method. However, we reiterate that in this method, we still cast each station as an M/G/1/K queue but use results for the M/G/1/∞ queue to solve for p_0.

3.1.3 Computation of p_0 Based on Three Simplifications

Our M/G/1/K queue model detailed above is accurate but admittedly complex. To address the complexity issue, we now describe three model variants that trade accuracy for simplicity.

1. **The $\mathbf{K = 2}$ case**. Under this case, we consider the M/G/1/2 queue. From (15), (16), and (18), we have

$$p_0 = \frac{\widehat{Y}^c(\gamma, e^{-\lambda})}{\widehat{Y}^c(\gamma, e^{-\lambda}) + \rho}. \tag{26}$$

   It is worth pointing out that for $K = 2, p_0$ can be approximated by an even simpler expression [13], namely,

$$p_0 = e^{-\rho}. \tag{27}$$

   It has been shown in [13] that the approximate model based on (27) can achieve comparable accuracy to the model in [4]. However, the accuracy of (27) is slightly worse than that of (26).
2. **The $\mathbf{K = \infty}$ case.** For this case, in [13], using the well-known fact that the steady-state probability of a nonempty buffer is given by ρ, we define

$$p_0 = 1 - \min(1, \rho), \tag{28}$$

   where we use the min function to prevent p_0 from becoming negative since $\rho < 1$ is required to maintain queue stability.
3. **The arbitrary buffer size case.** For this case, we propose the use of an M/M/1/K queue to calculate p_0, where the service time is assumed to be an exponential distribution with mean $\overline{Y}^c(\gamma, \beta)$. From [19], p_0 can be calculated using

$$p_0 = \frac{1}{1 + \rho + \rho^2 \ldots + \rho^K}. \tag{29}$$

   As evident from (29), this method has a much lower computational cost than our other methods for an arbitrary buffer size.

3.1.4 Time Complexity of Calculating p_0

The lattice-Poisson algorithm is an accurate and efficient method to invert generating functions. However, when K is large, it is possible to use a variant of the lattice-Poisson algorithm that employs Euler summation [18] to dramatically accelerate the computation in (21) and (25); we shall refer to this as the lattice-Poisson-Euler algorithm. The idea behind the acceleration algorithm is to use Euler summation to avoid the need to sum all the terms of the Fourier series. As well as the parameters r and l from the lattice-Poisson algorithm that control the inversion error, the lattice-Poisson-Euler algorithm requires two additional parameters: $\widetilde{K}$, which specifies how many initial terms of the Fourier series to use, and E, which specifies how many terms to use after the initial $\widetilde{K}$ terms; the typical value of E is 11. Theorem 1 below summarizes the time complexity of calculating p_0 using the two algorithms. In this theorem, if we adopt the lattice-Poisson-Euler algorithm, we set $E = 11$ and $\widetilde{K} \leq K - 1 - E$. The lattice-Poisson-Euler algorithm is not guaranteed to work in all applications (it requires the inverse function to satisfy certain regularity properties), but we have found that it works very well in our application.

Theorem 1. *Suppose $l = 1$.*

1. *Assume $\tau = 1$ slot. For the M/G/1/K method used in [11] and [12] (i.e., using (11), (15), (16), and (17)), at least $O((100n)^2 + K - 1)$ operations are required to calculate p_0.*
2. *For the M/G/1/K method used in this paper (i.e., using (19) and (21)), a) if we adopt the lattice-Poisson algorithm, $O(K^2)$ operations are required to calculate p_0; b) if we adopt the lattice-Poisson-Euler algorithm, $O(\widetilde{K}K)$ operations are required to calculate p_0.*
3. *For the M/G/1/∞ method used in this paper (i.e., using (24) and (25)), a) if we adopt the lattice-Poisson algorithm, $O(K)$ operations are required to calculate p_0; b) if we use the lattice-Poisson-Euler algorithm, $O(\widetilde{K})$ operations are required to calculate p_0.*

Proof.

1. The main computational load to calculate p_0 lies in calculating $a_k, k = 0, 1, \ldots, K-2$. We now evaluate the time complexity of calculating these a_ks in the three steps below.

   First, we estimate $I_{\max} \geq 100n$. To elaborate, we have proved in [21] that the mean of the saturated service time is $O(n)$. That is, $\overline{Y^c} \approx n$ ms $= n\frac{1000}{20}$ slots $= 50n$ slots. So, to accurately estimate a_k in (17), we should calculate the probability mass function of Y^c to at least $2 \times (50n)$ terms (i.e., from $\Pr\{Y^c = 1\}$ to $\Pr\{Y^c = 100n\}$). Hence, we should set $I_{\max} \geq 100n$.

   Second, we need at least $O((100n)^2)$ operations to calculate $\Pr\{Y^c = h\}, h = 1, \ldots, I_{\max}$. From (11), calculating each $\Pr\{Y^c = h\}$ needs $2h$ operations and thus we need at least $\sum_{h=1}^{I_{\max}} 2h \geq \sum_{h=1}^{100n} 2h > (100n)^2$ operations to calculate these $I_{\max}$ probabilities.

   Third, given $\Pr\{Y^c = h\}, h = 1, \ldots, I_{\max}$, we need $O(K-1)$ operations to calculate these a_ks ($k = 0, 1, \ldots, K-2$) since the time complexity of calculating each a_k can be regarded as constant.

   In short, we need at least $O((100n)^2 + K - 1)$ operations to calculate $a_k, k = 0, 1, \ldots, K-2$.
2.
   a) From (21), we need $2k$ operations to calculate $a_k, k = 1, \ldots, K-1$. Therefore, we need $O(K^2)$ operations to compute p_0:

   $$\sum_{k=1}^{K-1} 2k = K^2 - K = O(K^2).$$

   b) We now consider using the lattice-Poisson-Euler algorithm to calculate $a_k, k = 1, \ldots, K-1$. When $k \leq E + \widetilde{K}$, the lattice-Poisson-Euler algorithm reverts to the lattice-Poisson algorithm and hence needs $2k$ operations for each k; when $k > E + \widetilde{K}$, it performs $E + \widetilde{K} + 1$ operations for each k. Therefore, we need $O(\widetilde{K}K)$ operations to calculate p_0:

   $$\begin{aligned} &\sum_{k=1}^{E+\widetilde{K}} 2k + \sum_{k=E+\widetilde{K}+1}^{K-1} (E + \widetilde{K} + 1) \\ &= [(E + \widetilde{K} + 1)^2 - (E + \widetilde{K} + 1)] \\ &\quad + [(E + \widetilde{K} + 1)(K - (E + \widetilde{K} + 1))] \\ &= (E + \widetilde{K} + 1)(K - 1) \\ &= O(\widetilde{K}K). \end{aligned}$$
3.
   a) From (24) and (25), we need $2K$ operations to calculate p_0.

   b) The lattice-Poisson-Euler algorithm needs $E + \widetilde{K} + 1 = O(\widetilde{K})$ operations to calculate p_0. □

Remark. Owing to the remarkable reduction in time complexity, when plotting an analytical curve for a given K according to our methods, the computation time is reduced to several seconds compared to several hours when the method in [11] and [12] is used, which is shown in Table 1. The lattice-Poisson-Euler algorithm is highly effective in reducing the computational cost when K is large. We find that a setting of $\widetilde{K} = 50$ is sufficient to obtain the same accuracy as the lattice-Poisson algorithm, which implies that there is sufficient smoothness in the inverse function to give rise to a rapidly decaying Fourier series [18].

According to Theorem 1, item 2, the time complexity using the lattice-Poisson-Euler algorithm when $K = 1{,}000$ is reduced to $O(5 \times 10^4)$ from $O(10^6)$ when the lattice-Poisson algorithm is used.

3.2 Convergence Analysis of the Fixed Point

The general fixed point governed by (12) and (13) is the key to calculating the collision probability and all other performance metrics. In this section, we investigate the convergence of the general fixed point. Given λ, the general fixed point γ is a solution to

$$\gamma = \Gamma(\beta(\lambda, \gamma)), \tag{30}$$

where $\beta \triangleq \beta(\lambda, \gamma)$ is given by (12).

We now consider the following relaxed fixed point iteration for calculating the general fixed point γ:

$$\gamma_{k+1} = (1 - \alpha)\Gamma(\beta(\lambda, \gamma_k)) + \alpha\gamma_k, \quad 0 < \alpha < 1. \tag{31}$$

Note that $\alpha = 0$ corresponds to the usual fixed point iteration by repeated substitution. Using the relaxed iteration, Kumar et al. [14] proved the convergence of the sequence for the special case of $p_0 = 0$ and $M = \infty$. Here, we carry out the convergence analysis for the general case (i.e., the case of $p_0 \neq 0$ and $M < \infty$).

Theorem 2 below presents conditions (related to the parameter α and the initial iteration value γ_1) under which the sequence $\{\gamma_k; k \geq 1\}$ converges to a general fixed point γ. Note that in this section and the proofs of Theorem 2 and Corollary 1, g' denotes the first derivative of function g with respect to γ.

TABLE 1
Comparison among Different Queuing Models

		papers [11][12]	this paper		
		M/G/1/K	M/G/1/K	M/G/1/∞	M/M/1/K
fixed point formulation		Bianchi Markov chain approach extension	Kumar renewal approach extension		
calculation of $\overline{Y}^c$		by differentiation of generating function	by simple and explicit formula (7)		
derivation of $\widehat{Y}^c$		by complicated transfer function method [11]	by straightforward generating function method		
calculation of p_0		>((100n)^2+K-1)	O(K^2)	O(K)	—
time (in seconds) to plot one curve (16 points) in Fig. 4 when n=24	K=2	1.61$\times$10^4=4.48 hours	0.81	0.62	0.09
	K=6	1.63$\times$10^4=4.52 hours	2.43	0.75	0.09
	K=20	1.69$\times$10^4=4.68 hours	9.78	1.61	0.09
convergence of γ		not treated	not treated	not treated	proved
accuracy		only accurate near saturation	always accurate	accurate when $\rho < 1$	slightly inaccurate near saturation

In lines 4 to 6, we compare the calculation methods of $\overline{Y}^c$, $\widehat{Y}^c$, and p_0, respectively. In line 7, we compare the convergence of γ.

Theorem 2. *Given ν, λ, and n, if $|p_0'| \leq B$, where B is a positive constant, the sequence $\{\gamma_k; k \geq 1\}$ converges from above to a general fixed point γ if the following conditions hold:*

$$\begin{array}{ll} 1. & \alpha \geq \frac{|\Gamma'(\beta)|_{\max}}{1+|\Gamma'(\beta)|_{\max}}, \\ 2. & \gamma_1 > \Gamma(\beta(\lambda, \gamma_1)). \end{array} \tag{32}$$

In (32), $|\Gamma'(\beta)|_{\max}$ is defined as follows:

$$|\Gamma'(\beta)|_{\max} \triangleq \frac{(n-1)(B+2\nu)}{b_0}.$$

Proof. Please refer to the Appendix A. □

Remarks. If the fixed point is unique, the sequence $\{\gamma_k; k \geq 1\}$ must converge to the unique fixed point. If the fixed point is not unique, the sequence must converge to the fixed point that is nearest to γ_1.

Corollary 1. *For the M/M/1/K method, the sequence $\{\gamma_k; k \geq 1\}$ converges from above to a general fixed point γ if we let α and γ_1 satisfy (32) and let $B = K|\rho'|_{\max}$, where*

$$|\rho'|_{\max} \triangleq \lambda[\overline{X}'(1)(\sigma + T_s + T_{\bar{s}}) + \overline{X}(1)(2T_{\bar{s}} + n|T_s - T_{\bar{s}}|)].$$

Proof. Please refer to the Appendix A. □

Choices of γ_1 and α. We can set $\alpha = \frac{|\Gamma'(\beta)|_{\max}}{1+|\Gamma'(\beta)|_{\max}}$ based on condition 1 in Theorem 2, and $\gamma_1 = \Gamma(\frac{1}{b_0}) + \varepsilon$, where $0 < \varepsilon < 1 - \Gamma(\frac{1}{b_0})$. To justify the latter, we note that $\beta^c(\gamma)$ is a decreasing function of γ [22], and so we have $\beta^c(\gamma) \leq \beta^c(0) = \frac{1}{b_0}$ for $\gamma \in [0,1]$ and, hence,

$$\beta(\lambda, \gamma_1) \leq \beta^c(\gamma_1) \leq \frac{1}{b_0}.$$

Noting that $\Gamma(\beta)$ is an increasing function of β, we have

$$\Gamma(\beta(\lambda, \gamma_1)) \leq \Gamma\left(\frac{1}{b_0}\right) < \Gamma\left(\frac{1}{b_0}\right) + \varepsilon = \gamma_1,$$

which satisfies condition 2. in Theorem 2.

In Section 4, we will use the relaxed iteration algorithm with the above choices of γ_1 and α to calculate the collision probability.

3.3 Comparison between Models in [11] and [12] and Our Models

The models in [11] and [12] are the most closely related existing models to our work. Note that [12] uses the same method as [11] but improves on the accuracy by revising the expression for Ω used in [11]. Like our models, Zhai et al. [11] and Zheng et al. [12] permit an arbitrary buffer size and invoke the key approximation (12) and the M/G/1/K and M/M/1/K queue assumptions. However, as already discussed, we go much further and the result is that our models outperform those of [11], [12] in terms of modeling accuracy, modeling complexity, computational complexity, and convergence of the fixed-point equation. To be more explicit, we present a comparison in Table 1 of the attributes of the M/G/1/K method of [11], [12] with our M/G/1/K, M/G/1/∞, and M/M/1/K methods. In this table, lines 3 to 5 compare modeling complexity, line 6 compares computational complexity, line 7 compares convergence of the fixed-point equation, and line 8 compares modeling accuracy (demonstrated in Figs. 2, 3, and 4).

Note that in [12], p_0 is calculated according to (15), (16), and (17). However, for the calculation of a_k in (17), there is no discussion of the nontrivial task of deriving $\Pr\{Y^c = h\}$ from $\widehat{Y}^c$. To analyze the time complexity of calculating p_0, we assume that $\Pr\{Y^c = h\}$ is derived from $\widehat{Y}^c$ according to (11). Also, since there is no need to invert the generating function for the M/M/1/K method, we insert "-" in the corresponding field. In addition, we also show in Table 1 the practical time to plot one theoretical curve in Fig. 4 when n = 24 and $K = 2$, 6, and 20 using different methods. The time is calculated after we finish running the Matlab codes on an X86-based PC with Pentium(R) Dual-Core CPU E5300 @ 2.60 GHz, 2.00 GB Physical Memory, and Windows 7 Ultimate operating system.

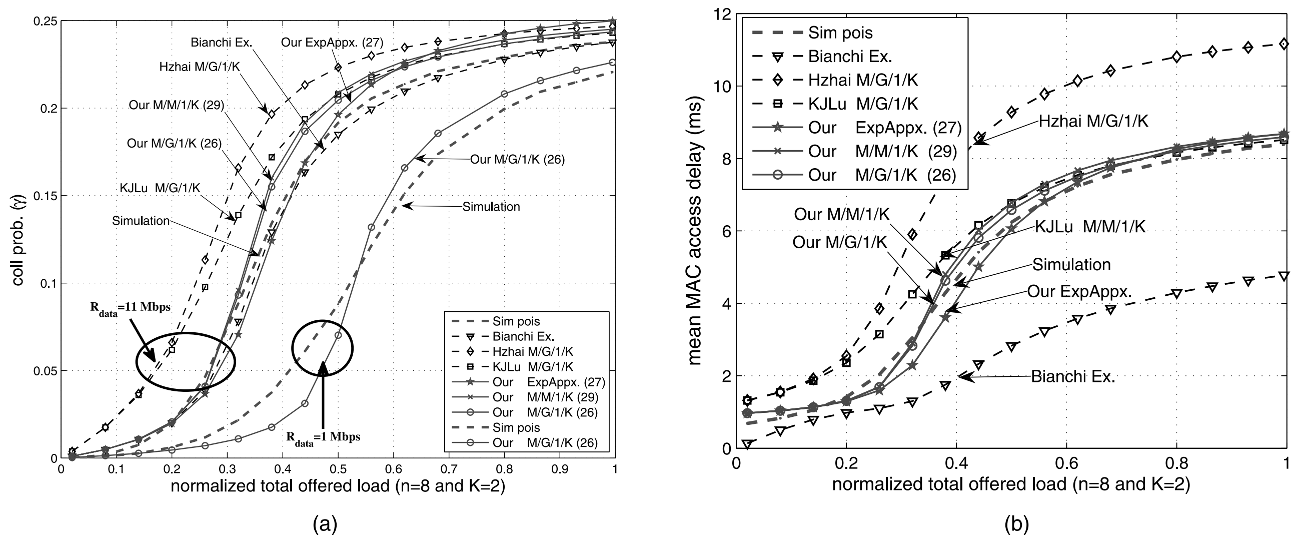


Fig. 2. (a) The collision probability and (b) the mean MAC access delay versus the normalized total offered load when $n = 8$ and $K = 2$.

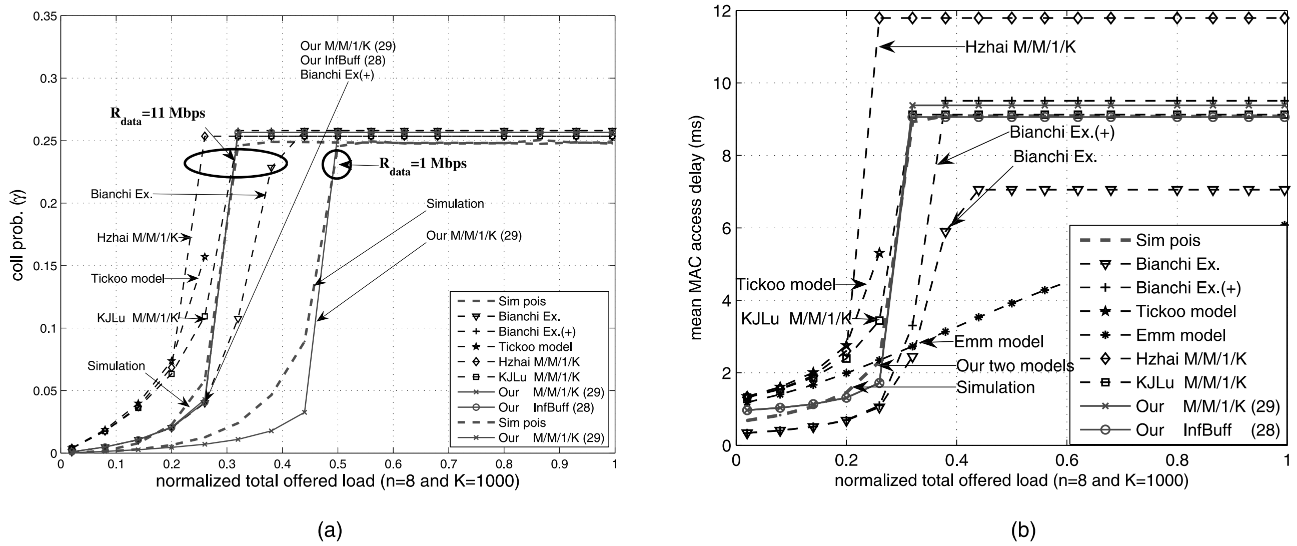


Fig. 3. (a) The collision probability and (b) the mean MAC access delay versus the normalized total offered load when $n = 8$ and $K = 1{,}000$. Note that the MAC access delay formulas from others' model such as the Emm model are inaccurate even for the saturated networks when the total offered load > 0.5.

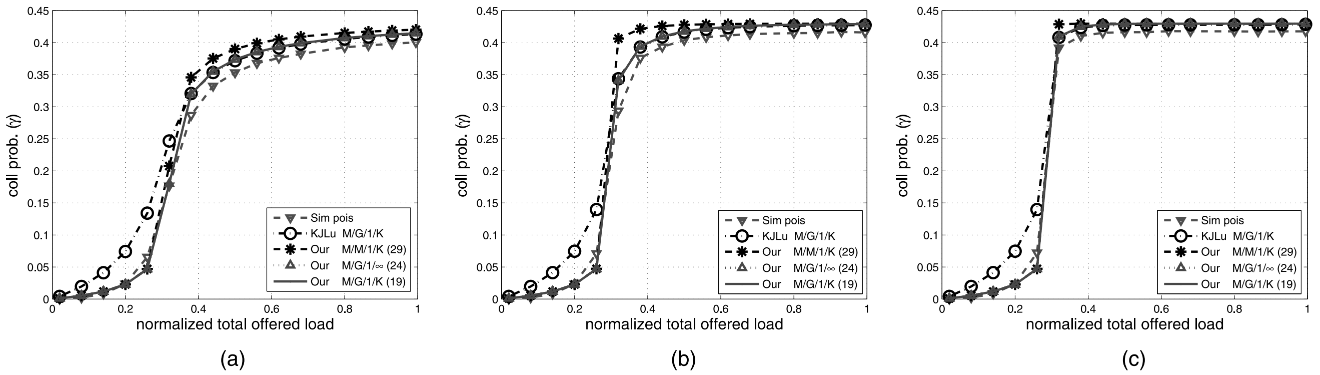


Fig. 4. The collision probability versus the normalized total offered load when $n = 24$ and (a) $K = 2$, (b) $K = 6$, and (c) $K = 20$.

4 Computation of Throughput, Delay, and Packet Loss Rate

This section presents formulas for the throughput, the mean and variance of the MAC access delay, the total delay, and the packet loss rate.

Throughput. For the throughput per node, s, we adopt the expression derived in [2], [14], namely

$$s = \frac{P_s}{n}\frac{L}{\overline{\overline{\Omega}}}, \tag{33}$$

where L is the packet size in bits, $\frac{P_s}{n}$ is the per node probability of successful packet transmission.

Mean and variance of MAC access delay. Let D denote the MAC access delay. From [21, (18)], the mean of the MAC access delay $\overline{D}$ can be calculated as follows:

$$\begin{aligned}
\overline{D} &= A_1 + B_1,\\
\text{where } A_1 &= \frac{1-\gamma}{(1-\gamma^M)} \sum_{i=0}^{M-1} \gamma^i \left\{ \theta_1 \sum_{k=0}^{i} \overline{\eta}_k + iT_{\overline{s}} \right\},\\
B_1 &= T_s - T_{ACK},\\
T_{ACK} &= \text{the transmission time of an ACK packet,}
\end{aligned}$$

where θ_1 is defined in (34).

From [21, (19)], the variance of the MAC access delay $Var[D]$ can be calculated as follows:

$$\begin{aligned}
Var[D] &= \frac{1-\gamma}{(1-\gamma^M)} \sum_{i=0}^{M-1} \gamma^i \{A_2^i + B_2^i\},\\
\text{where } A_2^i &= \sum_{k=0}^{i} (\overline{\eta}_k \theta_3 + (\theta_1)^2 Var(\eta_k)),\\
B_2^i &= \left(\theta_1 \sum_{k=0}^{i} \overline{\eta}_k + iT_{\overline{s}} - A_1 \right)^2,
\end{aligned}$$

where θ_1 and θ_3 are defined in (34).

$$\begin{aligned}
q &= (n-1)\beta(1-\beta)^{n-2},\\
\theta_1 &= \sigma + \theta_2,\\
\theta_2 &= (qT_s + (\gamma - q)T_{\overline{s}})(1-\beta),\\
\theta_3 &= (q(T_s - \theta_2)^2 + (\gamma - q)(T_{\overline{s}} - \theta_2)^2)(1-\beta)\\
&\quad + (1 - \gamma(1-\beta))(\theta_2)^2.
\end{aligned} \tag{34}$$

More general expressions for the mean and variance of the MAC access delay can be found in [21, (18) and (19)]. Each of the related works [4], [5], [7], [8], [11], and [12] develop their own delay models and obtain mean delay expressions that differ from ours; however, none of them derive the variance.

Waiting time. Let W be the waiting time of a packet in the transmit buffer of a node at an arbitrary instant. Let $\overline{W}$ be the mean of W and $W^*(s)$ be the Laplace transform of W. For the M/G/1/K queue, we have [19]

$$\begin{aligned}
W^*(s) &= \frac{\pi_0 \left(1 - \left(\frac{\lambda \widehat{Y}^c(\gamma, e^{-s})}{\lambda - s} \right)^K \right) s}{s - \lambda + \lambda \widehat{Y}^c(\gamma, e^{-s})}\\
&\quad + [\widehat{Y}^c(\gamma, e^{-s})]^{K-1} \sum_{k=0}^{K-1} \pi_k \left(\frac{\lambda}{\lambda - s} \right)^{K-k},\\
\overline{W} &= \frac{\pi_0 + \rho}{\lambda} \sum_{k=1}^{K} k p_k - \overline{Y}^c,
\end{aligned} \tag{35}$$

where p_k can be calculated by (19) and π_k can be calculated by p_k and (15).

Let W_c be the complementary cumulative distribution function of W, and $W_c^*(s)$ be the Laplace transform of W_c. We have

TABLE 2
Default Parameter Values Used in [4] and This Paper

CW_0	32	Header	227 µs	=	Mheader + Pheader + RouteHeader
m/M	5/7	T_s	48 slot	=	Header + L_{tm} + SIFS + δ + ACK + δ + DIFS
σ	1 slot	$T_{\overline{s}}$		=	T_s
δ	0 µs	L_{tm}	364 µs	=	500 bytes @ R_{data}
SIFS	10 µs	ACK	304 µs	=	24 bytes @ R_{basic} + 14 bytes @ R_{basic}
DIFS	50 µs	Mheader	20 µs	=	24 bytes @ R_{data} + 4 bytes @ R_{data}
R_{data}	11 Mbps	Pheader	192 µs	=	24 bytes @ R_{basic}
R_{basic}	1 Mbps	RouteHeader	15 µs	=	20 bytes @ R_{data}

$$W_c^*(s) = \frac{1 - W^*(s)}{s}. \tag{36}$$

Then, we can calculate W_c from (36) using the Euler numerical transform inversion algorithm in [18].

Total delay. Let $\overline{T}_t$ be the mean total delay, which equals the mean MAC service time plus mean waiting time of a packet. For the M/G/1/K queue, Takagi [19] expresses $\overline{T}_t$ as follows:

$$\overline{T}_t = \overline{Y}^c + \overline{W} = \frac{\sum_{k=1}^{K} k p_k}{\lambda(1 - p_K)}.$$

Packet loss rate. The packet loss rate due to buffer overflow is given by the probability p_K of encountering K packets in the buffer. For the M/G/1/K model, the packet loss rate is equal to p_K in (15); for the M/M/1/K method, the packet loss rate is equal to p_K, which is similar to (29) in the form of $p_K = \frac{\rho^K}{1+\rho+\rho^2+\cdots+\rho^K}$.

5 Model Verification

We verify our model using the 802.11 simulator in ns2 version 2.28 [23]. The default parameter values shown in Table 2 are set in accordance with 802.11b and [4], where one slot is equal to 20 μs and δ denotes the propagation delay. In our simulation, we used the NOAH routing protocol [14] and removed some bugs in the 802.11 simulator. These bugs, which are reported in [21], significantly affected the MAC access delay and the total delay. Each simulation value is an average over five simulation runs, where each run was for 100 seconds when the buffer size < 30 and for 500 seconds when the buffer $\text{size} = 1{,}000$. We present the theoretical results under the assumption of $T_s = T_{\overline{s}}$. For practical networks, this assumption does not necessarily hold. However, many analytical studies, including [4], [2], and [14], adopt this assumption, and it has also been adopted by the developers of the ns2 simulation tool [23]. The assumption is equivalent to assuming that ACK packets are transmitted at the basic rate and the ACK timeout after a collision matches the guard time observed by noncolliding nodes.

We assume Poisson arrivals and run two experiments. In the first experiment, we study the collision probability, the mean and standard deviation of the MAC access delay, the mean total delay, the packet loss rate, and the CCDF of the waiting time. In the second experiment, we study the distribution of the service time and the waiting time. We explore the impact on these performance metrics of different node populations n and different buffer sizes K. The abscissa of most graphs is the total offered load $\overline{\rho} \triangleq \frac{n\lambda L}{R_{data}}$, where R_{data} is the data rate.

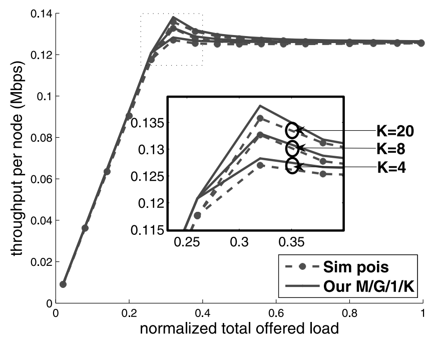


Fig. 5. The throughput versus the normalized total offered load when $n = 24$ and $K = 4$, 8, and 20.

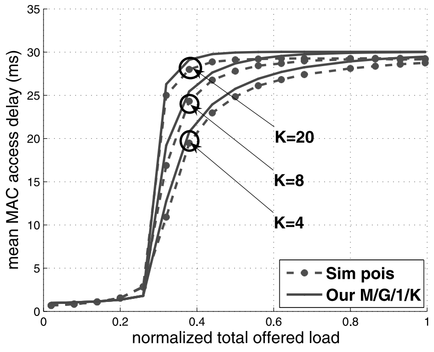


Fig. 6. The mean MAC access delay versus the normalized total offered load when $n = 24$ and $K = 4$, 8, and 20.

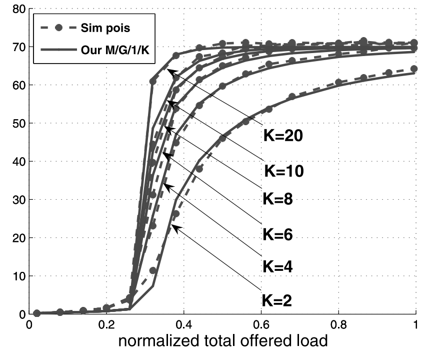


Fig. 7. The standard deviation of the MAC access delay versus the normalized total offered load when $n = 24$ and $K = 2$, 4, 6, 8, 10, and 20.

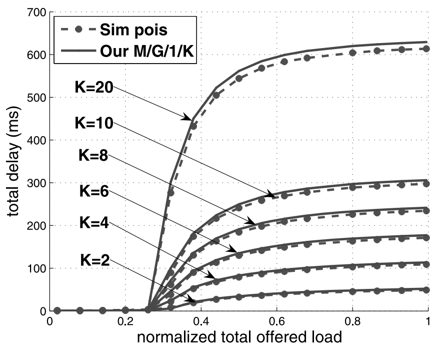


Fig. 8. The mean total delay versus the normalized total offered load when $n = 24$ and $K = 2$, 4, 6, 8, 10, and 20.

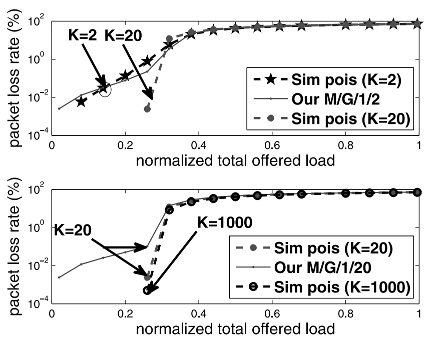


Fig. 9. The packet loss rate when $n = 24$, $K = 2$, 20, and 1,000.

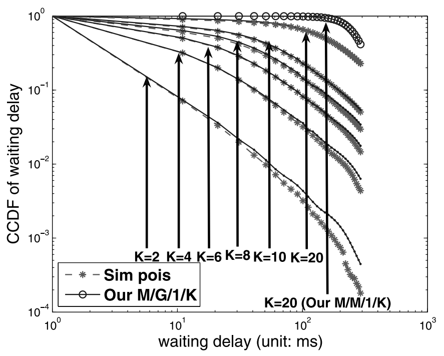


Fig. 10. The CCDF of waiting time when $n = 24$, $\overline{\rho} = 0.38$, and $K = 2$, 4, 6, 8, 10, and 20.

We provide comparisons of our work with results from existing models, namely [4] (which we call the Bianchi extension), [11] (the Hzhai model), [12] (the KJLu model), [8] (the Emm model), and [7] (the Tickoo model). We consider the Bianchi extension to be the most accurate among those works that model the case of $K = 2$, while we consider the Hzhai and KJLu models to be the most accurate among those works that model an arbitrary buffer size. When presenting the results from the M/M/1/K method, we use the relaxed iteration algorithm (31) to calculate the collision probability, where the algorithm stops when $|\gamma_k - \gamma_{k+1}| \leq 10^{-8}$. We find that the collision probability from our iteration algorithm is the same as that returned by the fzero function of Matlab.

We now detail the first experiment, shown in Figs. 2, 3, 4, 5, 6, 7, 8, 9, and 10. Figs. 2a and 2b, respectively, plot the collision probability and the mean MAC access delay for the case of $n = 8$ and $K = 2$ when the normalized total offered load is varied. In this figure, the curves labeled "Bianchi Ex.," "Hzhai M/G/1/K Ex.," "KJLu M/G/1/K," "Our ExpAppx.," "Our M/M/1/K," and "Our M/G/1/K" are calculated according to [4], [11], [12], (27), (29), and (26), respectively. Fig. 2a shows that our two models and the model in [4] closely match the simulation results; the Hzhai model is very inaccurate; the KJLu model is not accurate when the offered traffic is light but accurate when the offered traffic approaches saturation. Fig. 2a also shows that our M/G/1/K model works for small data rate, where $R_{data} = 1$ Mbps and $L = 150$ bytes. Fig. 2b shows that our MAC access delay model is the most accurate among the existing models, and shows that the MAC access delay models of [4] and [11] are quite inaccurate.

Figs. 3a and 3b, respectively, plot the collision probability and the mean MAC access delay for the case of $n = 8$ and $K = 1{,}000$ when the normalized total offered load is varied. In this figure, the curves labeled "Bianchi Ex.," "Tickoo model," "Emm model," "Hzhai M/M/1/K Ex.," "KJLu M/M/1/K," "Our M/M/1/K," and "Our infBuff" are calculated according to [5], [7], [8], [11], [12], (29), and (28), respectively. We observe similar results as in Fig. 2. The difference is that our two models are more accurate than that of [5], and the Emm model and the Tickoo model are wildly inaccurate. In addition, since the authors in the Emm model do not analyze the nonsaturated collision probability, we cannot show the corresponding results; the Tickoo model fails for a total offered load > 0.3 (the fixed-point iteration in their model converges to a value outside [0, 1]) and we omit these results; the MAC access delay formulas from other models such as the Emm model are inaccurate even for the saturated scenario for a total offered load > 0.5. In addition, Fig. 3a also shows that our larger buffer model works for small data rate, where $R_{data} = 1$ Mbps and $\mathrm{L} = 150$ bytes.

We now explain the curves "Bianchi Ex." and "Bianchi Ex. (+)" in Fig. 3. In the Bianchi Markov-chain extension [5], the mean backoff count, which we denote by $\overline{X}^*$, is a key variable. Following the derivation arguments in [5], it appears that $\overline{X}^*$ should be given by

$$\overline{X}^* = \frac{1}{(1-\gamma)} \frac{1-\gamma-\gamma(2\gamma)^m}{(1-2\gamma)} \frac{CW_0}{2}, \tag{37}$$

but in their presented expression (i.e., [5, (5)]), the factor $\frac{1}{(1-\gamma)}$ in $\overline{X}^*$ is missing. Note that our estimate of the mean backoff count, $\overline{X}$ in (8), can also be written with a factor $\frac{1}{(1-\gamma)}$, and note also that $\lim_{M\to\infty} \overline{X}(\gamma) = \overline{X}^*$ in (37). To demonstrate the error due to the absence of $\frac{1}{(1-\gamma)}$ in the model in [5], we plot the curves "Bianchi Ex." according to [5, (5)] and the curves "Bianchi Ex. (+)" after replacing [5, (5)] by (8). Note that the results after replacing [5, (5)] by (37) are almost as accurate as those after replacing [5, (5)] by (8). From Fig. 3a, the curve titled "Bianchi Ex. (+)" closely matches the simulated results for the collision probability, just like our model. From Fig. 3b, the new analytical results greatly improve the accuracy for the MAC access delay but they are still less accurate than ours. The reason is that the MAC access delay model in [5] is not as accurate as ours.

Figs. 4a, 4b, and 4c, respectively, plot the collision probability versus the normalized total offered load when $K = 2, 6, 20$, and $n = 24$. We compare results for the KJLu M/G/1/K model, and our M/G/1/K, M/G/1/∞, and M/M/1/K methods. The KJLu M/G/1/K model is not accurate when the offered traffic is light but is more accurate when the offered traffic approaches saturation. Our M/G/1/K and M/G/1/∞ methods (which give identical results) are the most accurate. On the other hand, the collision probability based on our M/M/1/K method is less accurate than that based on our M/G/1/K method. Nevertheless, the collision probability based on our M/M/1/K method only exhibits inaccuracy when the offered traffic approaches saturation but still closely matches the simulated result in other cases. As a result, it will only lead to slight inaccuracy of other performance metrics when the offered traffic approaches saturation. Because of space limitations, we do not plot further results obtained from our M/M/1/K method. Note that as the normalized total offered load increases, ρ can increase beyond 1, which should invalidate our M/G/1/∞ method (22). However, a large number of numerical examples show that (22) usually still works well for $\rho \geq 1$.

Figs. 5, 6, 7, and 8 plot the throughput, the mean MAC access delay, the standard deviation of the MAC access delay, and the total delay for $n = 24$ and different buffer sizes. Note that in the transition regime from light to heavy traffic loads, the buffer size dramatically affects the system performance, particularly the total delay. Since the collision probability derived using our M/G/1/K method is quite accurate (as demonstrated in Fig. 4), the other performance metrics derived using our M/G/1/K method also closely match the corresponding simulated results.

Fig. 9 plots the packet loss rate (due to buffer overflow) versus the normalized total offered load when $n = 24$ and $K = 2, 20$, and $1{,}000$. It can be seen that when $\overline{\rho} < 0.3$, there is virtually no packet loss, even for the very small buffer size $K = 2$. This is because in very light traffic, arrived packets are quickly transmitted and do not cause buildup of the queue. However, when $\overline{\rho}$ increases beyond 0.3, the packet loss rate is significant and is influenced by the buffer size. Surprisingly, we find in this example that for a given $\overline{\rho}$, the packet loss rate does not decrease monotonically with increasing K, but rather, increases as K is increased from 2 to 20, and then decreases as K is increased above 20. We believe the reason for this is the interplay between two opposing effects when the buffer size of all nodes is increased; the first is the obvious one of more buffer positions, which should reduce the buffer loss probability; the second is that increasing buffer backlog increases contention which increases the service time, and tends to increase the buffer loss probability. Note that when $K = 2$ and the normalized total offered load <0.08 and when $K = 20,\ 1{,}000$, and the normalized total offered load <0.26, the simulated collision probabilities are equal to zero and therefore are not shown in Fig. 9.

Fig. 10 plots the CCDF of the waiting time when $n = 24, \overline{\rho} = 0.38$, and $K = 2, 4, 6, 8, 10$, and 20. From this figure, we see that the waiting time drastically increases as K increases. For example, when $K = 2$, less than 1 percent packets experience a waiting time larger than 100 ms. In contrast, when $K = 20$, more than 65 percent packets experience a waiting time larger than 100 ms. In this figure, we also plot the CCDF of the waiting time obtained from the M/M/1/K method when $K = 20$ (see the solid curve with circular symbols). Observe that it exhibits a large error compared with our M/G/1/K method.

The results in Figs. 5, 6, 7, 8, 9, and 10 show that increasing the buffer size can slightly improve the throughput but dramatically increase the waiting time. Moreover, it does not necessarily reduce the packet loss rate. Therefore, there may be little benefit in provisioning very large buffers, even for loss-sensitive applications.

The second experiment demonstrates that the service time and waiting time distributions exhibit truncated heavy-tailed shapes (i.e., the distributions initially approximate straight lines when plotted on loglog coordinates), rather than exponential-like shapes. Fig. 11 compares the theoretical models and the simulated results for the service time distribution. In Fig. 11, the curve titled "Our M/G/1/K model" is plotted using (11), and the curve titled "Our

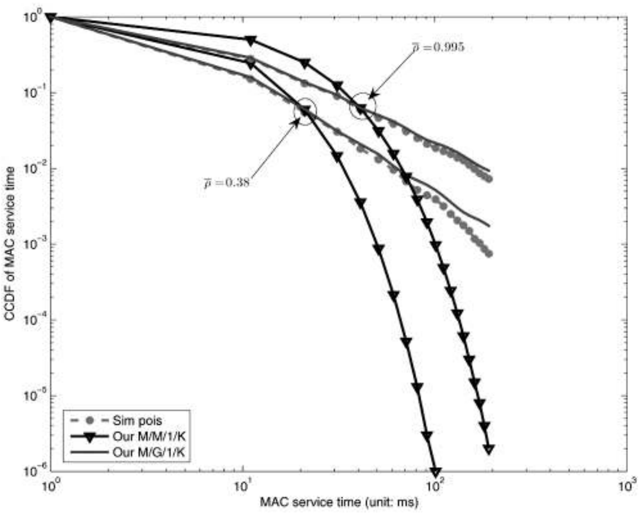


Fig. 11. The CCDF of MAC service time when $n = 24$, $K = 2$, and $\overline{\rho} = 0.38, 0.995$.

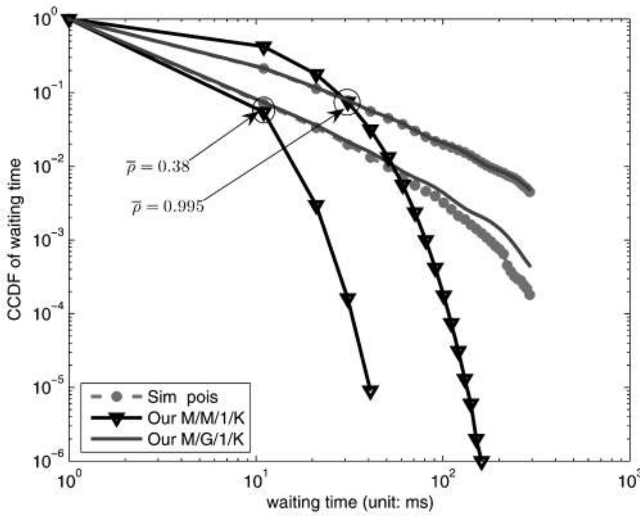


Fig. 12. The CCDF of waiting time when $n = 24$, $K = 2$, and $\overline{\rho} = 0.38, 0.995$.

M/M/1/K model" is plotted assuming that the service time is a geometric distribution, i.e., $\Pr(Y^c = k\sigma) = (1-q)^{k-1}q$, where $k = 1, 2, \ldots, q = 1/\overline{Y}^c$, and $\overline{Y}^c$ is given in (7). Fig. 12 compares the theoretical models and the simulated results for the waiting time distribution. In Fig. 12, the curve titled "Our M/G/1/K model" is plotted using W_c calculated by (36), and the curve titled "Our M/M/1/K model" is plotted assuming that the waiting time is a geometric distribution, i.e., $\Pr(W = k\sigma) = (1-q)^{k-1}q$, where $k = 1, 2, \ldots, q = 1/\overline{W}$, and $\overline{W}$ is given in (35). From Figs. 11 and 12, "Our M/G/1/K model" closely matches the actual distribution while the "Our M/M/1/K model" is far away from the actual distribution. Therefore, assuming the service time is an exponential-like distribution leads to a large error for subtle performance descriptors, such as the distribution of the service time, although it can be accurate for coarser performance descriptors, such as the mean of the service time. Note that the inaccuracy in the M/G/1/K queuing model in [11], as shown in Figs. 2 and 3, suggests that the corresponding exponential service time model used in their M/M/1/K queuing analysis is even more inaccurate. Due to this reason, we do not plot the exponential service time distribution in [11].

6 Conclusion

This paper presents a simple approximate model to characterize the impact of an arbitrary buffer size on the performance of nonsaturated 802.11 DCF networks. We devise different methods to address the modeling and computational complexity. Compared to related work, our model is the simplest, the most accurate, has the quickest computation speed, and yields the widest range of performance descriptors. The work in this paper will provide guidance to system developers to correctly size the transmit buffer in 802.11 devices, and to system administrators to predict the performance of applications. It also provides a solid foundation for extensions to more complex modeling scenarios, such as heterogeneous DCF networks, where stations could have different packet arrival rates, packet sizes and buffer sizes, and 802.11e EDCA networks, where access categories are differentiated by their MAC parameters. The key steps required for these extensions are generalizations of pivotal parameters to multiple classes, such as the attempt rate, collision probability, and service time.

Appendix A
Proofs of Theorem 2 and Corollary 1

In this appendix, we provide the proofs of Theorem 2 and Corollary 1.

Proof of Theorem 2. The key of the proof lies in the fact that $|\Gamma'(\beta)|$ is bounded. Note that $\Gamma'(\beta)$ denotes the first derivative of the function $\Gamma(\beta)$ with respect to γ. We now prove the convergence in two steps.

Step 1. $|\Gamma'(\beta)| \leq |\Gamma'(\beta)|_{\max}$ holds. Ramaiyan et al. [22] analyzes the properties of the saturated attempt rate (i.e., β^c) and proves that

$$|\beta^c| \leq \frac{1}{b_0} \text{ and } |(\beta^c)'| \leq \frac{2\nu}{b_0}.$$

On the other hand, from (12), we have

$$\begin{aligned} \beta' &= -p_0'\beta^c + (1-p_0)(\beta^c)', \\ |\beta'| &\leq |p_0'||\beta^c| + |(\beta^c)'|. \end{aligned}$$

Also from (13), noting $|p_0'| \leq B$, we have

$$\begin{aligned} \Gamma'(\beta) &= (n-1)(1-\beta)^{n-2}\beta', \\ |\Gamma'(\beta)| &\leq (n-1)|\beta'| \\ &\leq (n-1)(|p_0'||\beta^c| + |(\beta^c)'|) \\ &\leq (n-1)\left(B\frac{1}{b_0} + \frac{2\nu}{b_0}\right) \\ &\leq |\Gamma'(\beta)|_{\max}. \end{aligned}$$

Step 2. The sequence $\{\gamma_k; k \geq 1\}$ must converge from above to a general fixed point γ. We first prove $\Gamma(\beta(\lambda, \gamma_k)) \leq \gamma_k$ for $k \geq 1$. In fact, due to condition 2, we only need to provide an inductive proof that if $\Gamma(\beta(\lambda, \gamma_k)) \leq \gamma_k$, we must have $\Gamma(\beta(\lambda, \gamma_{k+1})) \leq \gamma_{k+1}$. Note that $|\Gamma'(\beta)| \leq |\Gamma'(\beta)|_{\max}$ and condition 1. We have

$$\begin{aligned}
&\Gamma(\beta(\lambda, \gamma_{k+1})) - \gamma_{k+1} \\
&= \Gamma(\beta(\lambda, \gamma_{k+1})) - [(1-\alpha)\Gamma(\beta(\lambda, \gamma_k)) + \alpha\gamma_k] \\
&= [\Gamma(\beta(\lambda, \gamma_{k+1})) - \Gamma(\beta(\lambda, \gamma_k))] + \alpha(\Gamma(\beta(\lambda, \gamma_k) - \gamma_k) \\
&= \Gamma'(\beta(\lambda, \xi_k^*))(\gamma_{k+1} - \gamma_k) + \alpha(\Gamma(\beta(\lambda, \gamma_k) - \gamma_k) \\
&\leq |\Gamma'(\beta)|_{\max}\left|\gamma_{k+1} - \gamma_k\right| + \alpha(\Gamma(\beta(\lambda, \gamma_k) - \gamma_k) \\
&= |\Gamma'(\beta)|_{\max}(1-\alpha)|\Gamma(\beta(\lambda, \gamma_k)) - \gamma_k| \\
&\quad + \alpha(\Gamma(\beta(\lambda, \gamma_k) - \gamma_k) \\
&= [\alpha - |\Gamma'(\beta)|_{\max}(1-\alpha)](\Gamma(\beta(\lambda, \gamma_k)) - \gamma_k) \\
&\leq 0.
\end{aligned} \tag{38}$$

where ξ_k^* is between γ_{k+1} and γ_k; in the last third equality, we use (31).

From (31) and $\Gamma(\beta(\lambda, \gamma_k)) \leq \gamma_k$, we now have $0 \leq \gamma_{k+1} \leq \gamma_k \leq 1$. Note that $\Gamma(\beta(\lambda, \gamma))$ is continuous with respect to γ. Then the bounded and nonincreasing sequence must converge to a fixed point of $\Gamma(\beta)$. □

Proof of Corollary 1. According to Theorem 2, we only need to prove $\left|p_0'\right| \leq B$. The conclusion holds following Claims 1 and 2 below.

Claim 1. $|\rho'| \leq |\rho'|_{\max}$. From (6), we have

$$\begin{aligned}
0 \leq P_b' &= \frac{n}{n-1}(1-\gamma)^{\frac{1}{n-1}} \leq 2, \\
P_s' &= n\Big[\frac{n}{n-1}(1-\gamma)^{\frac{1}{n-1}} - 1\Big] \\
&= n(P_b' - 1).
\end{aligned}$$

From (9), setting $P_b = P_s = 1$, we can bound $\overline{\Omega}$.

$$\overline{\Omega}(\gamma) = \sigma + P_s T_s + P_{\overline{s}} T_{\overline{s}} \leq \sigma + T_s + T_{\overline{s}}.$$

Further, we have

$$\begin{aligned}
\overline{\Omega} &= \sigma + P_b T_{\overline{s}} + P_s(T_s - T_{\overline{s}}), \\
\overline{\Omega}' &= P_b' T_{\overline{s}} + n(P_b' - 1)(T_s - T_{\overline{s}}), \\
\left|\overline{\Omega}'\right| &\leq 2T_{\overline{s}} + n|T_s - T_{\overline{s}}|.
\end{aligned}$$

On the other hand, noting $\overline{X}$ and $\overline{X}'$ are increasing with respect to γ, we have

$$\overline{X} \leq \overline{X}(1) \text{ and } \overline{X}' \leq \overline{X}'(1).$$

Since $\rho = \lambda\overline{Y}^c$, where $\overline{Y}^c = \overline{X} \cdot \overline{\Omega}$, we have

$$\rho' = \lambda(\overline{X}' \cdot \overline{\Omega} + \overline{X} \cdot \overline{\Omega}') \text{ and } |\rho'| \leq |\rho'|_{\max}.$$

Claim 2. $\left|p_0'\right| \leq K|\rho'|_{\max}$. Let $\rho^* = \frac{(1+2\rho+\cdots+K\rho^{K-1})}{(1+\rho+\rho^2\cdots+\rho^K)^2}$, we have

$$\begin{aligned}
\rho^* &= \frac{(1+2\rho+\cdots+K\rho^{K-1})}{(1+\rho+\rho^2+\cdots+\rho^K)^2} \\
&\leq \frac{K(1+\rho+\rho^2+\cdots+\rho^K)}{(1+\rho+\rho^2+\cdots+\rho^K)^2} \\
&= \frac{K}{(1+\rho+\rho^2+\cdots+\rho^K)} \\
&\leq K.
\end{aligned}$$

TABLE 3
Main Notation and Associated Equation Numbers

Eq.No.	notation	Eq.No.	notation
(1)	β^c, γ, b_k, M	(14)	ρ, λ
(2)	Γ, n	(15)	p_k, π_k, K
(3)	Y^c, X, Ω	(16)	π, a_k
(4)	$\eta_k, \delta(\cdot), CW_k, \nu$	(17)	$I_{\max}$
(5)	$\sigma, T_s, T_{\overline{s}}, P_b, P_s, P_{\overline{s}}$	(19)	p_k', a_k'
(7)	$\overline{Y}^c, \overline{X}, \overline{\Omega}$	(21)	$\widehat{a'}$
(10)	$\widehat{Y}^c, \widehat{X}, \widehat{\Omega}, \widehat{\eta}_k$	(22)	π_k^∞, q_K
(11)	$\Pr\{Y^c = h\}, h, l, r$	(25)	$\widehat{q}, \widehat{\pi}^\infty$
(12)	β, p_0	(37)	$\overline{X}^*$

Under the M/M/1/K queue assumption, noting $p_0 = \frac{1}{1+\rho+\rho^2+\cdots+\rho^K}$ from (29), we have

$$\begin{aligned}
p_0' &= -\rho'\rho^*, \\
\left|p_0'\right| &= |-\rho'\rho^*| \leq K|\rho'|_{\max}.
\end{aligned}$$

□

Appendix B

Main Notation and Associated Equation Numbers

Table 3 lists the main mathematical notation used in this paper and associated equation numbers where the notation first appears. Note that $\overline{(\cdot)}$ denotes the mean of $(\cdot)$ and $\widehat{(\cdot)}$ denotes the generating function of $(\cdot)$. In addition, we define α, γ_k, B, p_0', Γ', and $|\Gamma'(\beta)|_{\max}$ in Theorem 2, and $|\rho'|_{\max}$ in Corollary 1, and $\widetilde{K}$ and E in Section 3.1.4, and s, L, $\overline{D}$, T_{ACK}, θ_1, $\overline{\eta}_k$, $Var[D]$, W, $\overline{W}$, W^*, W_c, W_c^*, and $\overline{T}_t$ in Section 4, and $\overline{\rho}$ and R_{data} in Section 5, respectively.

Acknowledgments

This work is supported by the RGC Earmarked Research Grant HKUST 6160/04E.

References

[1] *ANSI/IEEE Std 802.11, Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications, 1999 ed. (R2003)*, IEEE, 1999.

[2] G. Bianchi, "Performance Analysis of the IEEE 802.11 Distributed Coordination Function," *IEEE J. Selected Areas in Comm.*, vol. 18, no. 3, pp. 535-547, Mar. 2000.

[3] Y.C. Tay and K.C. Chua, "A Capacity Analysis for the IEEE 802.11 MAC Protocol," *Wireless Networks*, vol. 7, no. 2, pp. 159-171, July 2001.

[4] D. Malone, K. Duffy, and D. Leith, "Modeling the 802.11 Distributed Coordination Function in Non-Saturated Heterogeneous Conditions," *IEEE/ACM Trans. Networking*, vol. 15, no. 1, pp. 159-172, Feb. 2007.

[5] K. Duffy and A. Ganesh, "Modeling the Impact of Buffering on 802.11," *IEEE Comm. Letters*, vol. 11, no. 2, pp. 219-221, Feb. 2007.

[6] G.R. Cantieni, C.B.Q. Ni, and T. Turletti, "Performance Analysis under Finite Load and Improvements for Multirate 802.11," *Computer Comm.*, vol. 28, no. 10, pp. 1095-1109, June 2005.

[7] O. Tickoo and B. Sikdar, "A Queuing Model for Finite Load IEEE 802.11 Random Access MAC," *Proc. IEEE Int'l Conf. Comm. (ICC '04)*, pp. 175-179, 2004.

[8] E. Winands, T. Denteneer, J. Resing, and R. Rietman, "A Finite-Source Feedback Queueing Network as a Model for the IEEE 802.11 DCF," *European Trans. Telecomm.,* vol. 16, no. 1, pp. 77-89, 2005.

[9] M. Garetto and C. Chiasserini, "Performance Analysis of 802.11 WLANs under Sporadic Traffic," *Proc. Networking,* pp. 1343-1347, July 2005.

[10] M. Ozdemir and A.B. McDonald, "On the Performance of Ad Hoc Wireless LANs: A Practical Queuing Theoretic Model," *Performance Evaluation,* vol. 63, no. 11, pp. 1127-1156, Nov. 2006.

[11] H. Zhai, Y. Kwon, and Y. Fang, "Performance Analysis of IEEE 802.11 MAC Protocols in Wireless LANs," *Wireless Comm. and Mobile Computing,* vol. 4, no. 8, pp. 917-931, Dec. 2004.

[12] Y. Zheng, K.J. Lu, D. Wu, and Y. Fang, "Performance Analysis of IEEE 802.11 DCF in Imperfect Channels," *IEEE Trans. Vehicular Technology,* vol. 55, no. 5, pp. 1648-1656, Sept. 2006.

[13] Q.L. Zhao, D.H.K. Tsang, and T. Sakurai, "A Simple and Accurate Model for Nonsaturated IEEE 802.11 DCF," *IEEE Trans. Mobile Computing,* vol. 8, no. 11, pp. 1539-1553, Nov. 2009.

[14] A. Kumar, E. Altman, D. Miorandi, and M. Goyal, "New Insights from a Fixed Point Analysis of Single Cell IEEE 802.11 WLANs," *IEEE/ACM Trans. Networking,* vol. 15, no. 3, pp. 588-601, Mar. 2007.

[15] H.L. Vu and T. Sakurai, "Accurate Delay Distribution for IEEE 802.11. DCF," *IEEE Comm. Letters,* vol. 10, no. 4, pp. 317-319, Apr. 2006.

[16] K.D. Huang, K.R. Duffy, D. Malone, and D.J. Leith, "Investigating the Validity of IEEE 802.11 MAC Modeling Hypotheses," *Proc. IEEE 19th Int'l Symp. Personal, Indoor and Mobile Radio Comm. (PIMRC '08),* Sept. 2008.

[17] Probability-Generating Function, http://en.wikipedia.org/wiki/Probability-generating_function, 2011.

[18] J. Abate, G.L. Choudhury, and W. Whitt, "An Introduction to Numerical Transform Inversion and Its Application to Probability Models," *Computational Probability,* W.K. Grassman, ed., Kluwer, pp. 257-323, 2000.

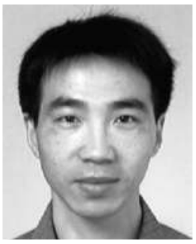
[19] H. Takagi, *Queueing Analysis: A Foundation of Performance Evaluation, Volume 2: Finite Systems.* North-Holland, pp. 197-203, 1993.

[20] H.C. Tijms, *A First Course in Stochastic Models.* Wiley, pp. 408-417, 2003.

[21] T. Sakurai and H.L. Vu, "Access Delay of the IEEE 802.11 MAC Protocol under Saturation," *IEEE Trans. Wireless Comm.,* vol. 6, no. 5, pp. 1702-1710, May 2007.

[22] V. Ramaiyan, A. Kumar, and E. Altman, "Fixed Point Analysis of Single Cell IEEE 802.11e WLANs: Uniqueness and Multistability," *IEEE/ACM Trans. Networking,* vol. 16, no. 5, pp. 1080-1093, Oct. 2008.

[23] The Network Simulator: Building Ns, http://www.isi.edu/nsnam/ns/ns-build.html, 2011.



Qinglin Zhao received the PhD degree from the Institute of Computing Technology, Chinese Academy of Sciences, Beijing, China, in 2005. From October 2006 to August 2009, he worked as a postdoctoral researcher in the Department of Electronic and Computer Engineering, the Hong Kong University of Science and Technology (HKUST), HKSAR. Since September 2009, he has been with the faculty of Information Technology, Macau University of Science and Technology, Avenida Wei Long, Taipa, Macau, China. His research interests include wireless and mobile networks, mobility management, and performance analysis.

Danny H.K. Tsang received the PhD degree in electrical engineering from the Moore School of Electrical Engineering at the University of Pennsylvania in 1989. He joined the Department of Electronic and Computer Engineering at the Hong Kong University of Science and Technology in summer 1992 and is now a professor in there. He was a guest editor for the *IEEE Journal of Selected Areas in Communications'* Special Issue on Advances in P2P Streaming Systems and an associate editor for the *Journal of Optical Networking* published by the Optical Society of America. He currently serves as an associate technical editor for the *IEEE Communications Magazine*. During his leave from HKUST during 2000-2001, he assumed the role of principal architect at Sycamore Networks in the United States. He was responsible for the network architecture design of Ethernet MAN/WAN over SONET/DWDM networks. He invented the 64B/65B encoding scheme (US Patent number 6,952,405) for Transparent GFP and it has been adopted by the International Telecommunication Union's Generic Framing Procedure recommendation GFP-T (ITU-T G.7041/Y.1303). His current research interests include Internet quality of service, P2P video streaming over the Internet, and wireless multimedia networks. He is a senior member of the IEEE.

Taka Sakurai received the BSc degree in applied mathematics and the BE degree in electrical engineering from the University of Adelaide in 1988 and 1989, respectively. He received the PhD degree in electrical engineering from the University of Melbourne in 2003. From 1991 to 1997, he was a research engineer at Telstra Research Laboratories. Subsequently, he held research and development roles at NEC and Lucent Technologies. From 2003 to 2005, he was with the Department of Electrical and Electronic Engineering, University of Melbourne. Currently, he is with the Chief Technology Office of Telstra Corporation and an honorary fellow of the University of Melbourne. His research interests are in the areas of performance analysis of next-generation wireless networks, design of MAC protocols for wireless LANS and sensor networks, and computational probability. He is a member of the IEEE.