sensors

MDPI

Article

An Adaptive Channel Access Method for Dynamic Super Dense Wireless Sensor Networks

Chunyang Lei [1,*], Hongxia Bie [1], Gengfa Fang [2] and Xuekun Zhang [1]

Received: 15 October 2015; Accepted: 25 November 2015; Published: 3 December 2015
Academic Editor: Leonhard M. Reindl

[1] School of Information and Communication Engineering, Beijing University of Posts and Telecommunications, Beijing 100876, China; biehx@bupt.edu.cn (H.B.); zhangxuekun1990@gmail.com (X.Z.)
[2] Department of Engineering, Macquarie University, Sydney 2109, Australia; gengfa.fang@mq.edu.au
* Correspondence: leichunyang.2014@gmail.com; Tel.: +86-13810489874

Abstract: Super dense and distributed wireless sensor networks have become very popular with the development of small cell technology, Internet of Things (IoT), Machine-to-Machine (M2M) communications, Vehicular-to-Vehicular (V2V) communications and public safety networks. While densely deployed wireless networks provide one of the most important and sustainable solutions to improve the accuracy of sensing and spectral efficiency, a new channel access scheme needs to be designed to solve the channel congestion problem introduced by the high dynamics of competing nodes accessing the channel simultaneously. In this paper, we firstly analyzed the channel contention problem using a novel normalized channel contention analysis model which provides information on how to tune the contention window according to the state of channel contention. We then proposed an adaptive channel contention window tuning algorithm in which the contention window tuning rate is set dynamically based on the estimated channel contention level. Simulation results show that our proposed adaptive channel access algorithm based on fast contention window tuning can achieve more than 95% of the theoretical optimal throughput and 0.97 of fairness index especially in dynamic and dense networks.

Keywords: backoff algorithm; MAC Protocol; Wireless Sensor Network

1. Introduction

Distributed Wireless Sensor Networks (WSNs) play an important role of monitoring and sensing wide-range of environmental parameters in the current and future surveillance systems. Thanks to the tremendous range of applications that they can enable, distributed WSNs become an essential part of 5G networks where Internet of Things (IoT), Machine-to-Machine (M2M) and Vehicular-to-Vehicular (V2V) networks are the promising new applications. As a result, distributed WSNs will become extremely dense in the future. Highly dense wireless sensors need to be deployed in order to sense environmental parameters as a part of the smart cities [1,2]. Highly dense development of sensors is also essential to support applications, such as the high precise seismic exploration and pollution monitoring [3,4], where there are up to thousands of wireless sensors accessing the channel dynamically within an area of 1000 square meters.

How to design such highly dense and distributed wireless sensor network and more specifically the channel access algorithm to support extremely large contending nodes accessing the channel are new challenges. While the wireless sensors have become much denser, contention of accessing wireless channels among neighboring nodes has increased exponentially and become very dynamic depending on whether the neighboring nodes have data for transmission or not. The number of contending nodes can vary from tens to hundreds. Furthermore, the number of contending nodes changes more quickly

and frequently than the classic sensor networks. For example, a sudden change of contending nodes from several to several hundreds may happen when there is an event happening in an area where all the nodes have data to report. In super dense WSNs, new adaptive and efficient channel access control protocol and algorithm are required to achieve efficient wireless channel access by avoiding high congestion among the nodes.

In WSNs, contention based carrier sense multiple access with collision avoidance (CSMA/CA) protocol is applied to enable the channel accessing by multiple nodes in a distributed way. According to CSMA/CA, WSN nodes generate random backoff time based on the predefined contention window size before accessing the channel. A well designed scheme to control the contention window size based on the contending nodes nearby can efficiently solve the channel contention problem [5]. In this paper, we focus on analyzing CSMA/CA based channel access and contention problem, and proposing a new channel accessing algorithm which is adaptive to the dynamics of the contending nodes.

Binary Exponential Backoff (BEB) [6,7] has been widely used as the channel access protocol for WSNs based on IEEE 802.15.4 Zigbee [8] or IEEE 802.11x WiFi [9,10]. By increasing the contention window size exponentially after each successful channel access, BEB tries to schedule the nodes accessing the channel in a distributed and efficient way. However, its performance decreases rapidly when the number of contending nodes becomes very large. Previous work in [11] indicates that more than 40% throughput is lost when the number of contending nodes reaches 60 for Zigbee/WiFi based sensor networks. Others [12–15] proposed new backoff algorithms such as EIED and MILD based on BEB in which contention window is tuned exponentially or linearly after each failure of channel access attempt. The above proposed schemes have poor overall network performance in dense networks for lack of the knowledge of channel congestion level when the number of contending nodes becomes large.

Authors in [16,17] analyzed the relationship between different number of contending nodes and their corresponding optimal contention window size, based on which a series of contention based backoff algorithms were proposed in [18–20]. These algorithms firstly try to estimate the number of nodes trying to access the channel simultaneously by continuously monitoring the channel state, and then tune the contention window accordingly. It is not easy to estimate the exact number of contending nodes on-the-fly especially in the super dense and dynamic networks where the number of contending nodes changes dramatically from time to time. What is more, an accurate estimation always takes longer time and ultra higher computation complexity, which is not affordable for low power and low computing power nodes in WSNs.

Considering the problems above, the authors in [21] proved that the probability of an idle state keeps constant when the network throughput approaches the theoretical upper bound of the channel. As a result, lots of idle state sensing based algorithms were proposed [22–24]. Without estimating the exact number of contending nodes, the proposed algorithms try to tune the contention window according to whether the estimated channel state parameter is optimal or not. These methods can effectively improve the network throughput when the number of contending nodes is very large. However, the contention window tuning speed is not fast enough to cope with a situation where there is a fast and dramatic change of the contending nodes. What is more, as the estimation results are obtained based on a limited number of samples, large estimation error is inevitable. In idle state sensing based algorithms, the contention window tuning rate is limited to small value only, otherwise the contention window size will not be able to reach its optimum because of the inaccurate estimation. A small tuning rate takes much longer period of time before the contention window can reach its optimal. If the number of contending nodes changes faster than the contention window tuning rate, network throughput will decline severely because the contention window is not always at its optimal size. Considering the issues above, the following parameters are used in this paper to evaluate the performance of backoff algorithms:

- **Tuning Accuracy:** defines how well the contention window of the backoff algorithm fits the channel contention based on the contention levels and number of contending nodes.

- **Tuning Speed:** defines how fast the backoff algorithm can tune the contention window to the optimal size whenever there is a change of the number of contending nodes.

In this paper, a normalized channel contention analysis model is proposed considering the property of dynamic channel contention. In this model, we introduce a series of transformations to map channel conditions onto pre-defined reference cases. For each reference case, we can easily calculate accurate indications about the contention window and its tuning strategy. We also prove the accuracy of the model. In order to speed up the contention window tuning process without sacrificing the tuning accuracy, we propose a contention window tuning scheme through adaptive tuning rate. In this method, the tuning rate is calculated based on the gap between the current contention window size and the channel contention level. When the gap is big, a large tuning step is applied to make sure the contention window can quickly reach to its optimal to improve the turning speed, otherwise a small tuning step is chosen to improve the tuning accuracy.

The rest of this paper is organized as follows. In Section 2 we setup and verify a normalized channel contention analysis model. An adaptive contention window tuning scheme is proposed and studied in Section 3. In Section 4, we study the performance of the proposed algorithm both in static and dynamic settings of networks through simulations. In Section 5 we conclude this paper.

2. Normalized Channel Contention Analysis

According to the CSMA/CA protocol, a contending node can not access the channel directly by sending data even when it senses a free channel. Instead, it will firstly need to wait for a random interval decided by the contention window before it starts accessing the channel which is the key part of the backoff algorithms. Channel contention analysis plays a key role in the design of backoff algorithms since it helps to understand how to do the tuning of contention window and its impact on the channel access in terms of throughput and fairness is huge. As specified by the previous research, channel contention level indicates the average number of contending nodes which try to send data through the wireless channel simultaneously in a given time slot (time slot: the unit time in Zigbee/WiFi networks [9]). Therefore, the channel contention level can be represented by the number of nodes that are contending to access the channel and their corresponding probability of accessing the channel successfully in a given slot. By modeling the backoff process as a Markov chain, authors in [25] defined the probability of successfully accessing the channel in the current slot as below Equation (1),

$$\tau = \frac{2}{cw + 1} \tag{1}$$

where cw is a node's contention window size. The contention level then can be defined as a function of the number of contending nodes n and the contention window size cw. We denote the contention level as $C(cw, n)$. (The detailed definition of the function $C(cw, n)$ will be given in next subsection.)

We divide the process of backoff into two steps: the first step is about estimating the contention level in current channel indicated by $C(cw, n)$, and the second step is to calculate the contention window so as to optimize the contention level where we model the selection of the contention window as a matching process problem between cw, n and $C(cw, n)$. Previous research [26] focuses on more ideal settings where n and cw are perfectly matched without considering cases in highly dense networks where $C(cw, n)$ is far more complicated. The mismatched cases are very important in determining the contention window tuning proceeding in highly dense networks in practice.

2.1. Model Construction

In order to deal with the mismatched cases, we introduce a correlation coefficient θ and define it as $\theta = n/cw$. Thus we have $C(cw, n) = C(cw, \theta \cdot cw)$. We define $C_1(cw, \theta) = C(cw, \theta \cdot cw)$. The mappings between $C_1(cw, \theta)$ and $C(cw, n)$ are presented in Equation (2), where θ represents a matching degree between the contention window size and the channel contention state decided by

the number of contending nodes. In Equation (2), a is an arbitrary positive number, which indicates the linear variety of cw and θ.

$$\begin{cases} C_1(cw, a \cdot \theta) = C(cw, a \cdot \theta \cdot cw) = C(cw, a \cdot n) \\ C_1(a \cdot cw, \theta) = C(a \cdot cw, \theta \cdot (a \cdot cw)) = C(a \cdot cw, a \cdot n) \end{cases} \tag{2}$$

By combining the function of cw and θ in $C_1(cw, \theta)$, we can notice that although channel contention level changes as a function of θ, it may remain static if cw and n are increased and decreased accordingly. We introduce a constant contention window size cw_{ref}, so that $C_1(cw, \theta)$ can be expressed as follows Equation (3),

$$C_1(cw, \theta) = C_1(cw_{ref}, \theta) + N(cw) \tag{3}$$

In Equation (3), $N(cw)$ was defined as the difference between $C_1(cw, \theta)$ and $C_1(cw_{ref}, \theta)$. If we can prove that $N(cw)$ is small enough compared to the value of $C_1(cw_{ref}, \theta)$, C_1 can be further simplified as a function of one variable plus the noise. Moreover, if the noise is small enough, the expression of C_1 can be further simplified. We denote the simplified $C_1(cw, \theta)$ as $C_2(\theta)$. Mapping $C(cw, n)$ onto $C_2(\theta)$ brings significant difference to the design of backoff algorithms.

Although the expression of $C(cw, n)$ has been simplified, original $C(cw, n)$ is still embedded in $C_2(\theta)$ which is important to improve the backoff algorithm. If the reference cases are properly designed, we can significantly simplify the algorithm. The change of channel contention level will be indicated by a deviation of θ. By utilizing its deviation degree, the contention window can be tuned more efficiently in terms of throughput and speed. As some of the cases with different cw have the same θ, we can dramatically reduce the complexity of the proposed algorithm which is important in practice.

2.2. Model Verification

The validity of the proposed model can be evaluated through $|N(cw)|$ which is the absolute deviation of the mapping as in Equation (3). The upper bound of $|N(cw)|$ can be calculated as Equation (4)

$$\begin{aligned} |N(cw)| &= |C_1(cw, \theta) - C_1(cw_{ref}, \theta)| \\ &= |\int_{cw_{ref}}^{cw} \frac{\partial C_1(t, \theta)}{\partial t} dt| \leq |cw - cw_{ref}| \cdot max\{|\frac{\partial C_1(t, \theta)}{\partial t}|, t \in [cw_{ref}, cw]\} \end{aligned} \tag{4}$$

According to work in [11,27,28], the contention of accessing the channel can be modeled as a discrete time stochastic process that includes three states: idle slot state, successful packet transmission state including RTS/CTS/DATA/ACK and failed transmission state indicated by a failed RTS. The probabilities of the above states are three basic parameters in CSMA/CA based channel access schemes. They can be used to indicate the contention level in channel. Other channel parameters can be calculated based on the three parameters above. Thus evaluating $|N(cw)|$ based on $C_1(cw, \theta)$ equals to evaluating $|N(cw)|$ based on the three basic states.

Next we will focus on calculating the upper bound of $|N(cw)|$ for the three cases where channel contention level is determined by the probability of idle slot state, the probability of successful transmission state, and the probability of failed transmission state accordingly.

2.2.1. Case 1: Channel Contention Level Depends on the Probability of Idle Slot State

An idle slot state will appear if there is no node having data for transmission. Thus the probability of an idle slot state can be expressed as Equation (5) below.

$$P_{Idle}(n, \tau) = (1 - \tau)^n \tag{5}$$

By substituting $n = \theta \cdot cw$ and Equation (1) to Equation (5), the probability of an idle channel becomes:

$$P_I(cw,\theta) = (\frac{cw-1}{cw+1})^{\theta \cdot cw} \tag{6}$$

We can have the first-order partial derivative of $P_I(cw,\theta)$:

$$\frac{\partial P_I(cw,\theta)}{\partial cw} = P_I(cw,\theta) \cdot (\theta \cdot \ln \frac{cw-1}{cw+1} + \frac{2\theta \cdot cw}{cw^2-1}) \tag{7}$$

In order to estimate $\partial P_I(cw,\theta)/\partial cw$, we continue to calculate the second-order partial derivative of $P_I(cw,\theta)$.

Let

$$X(cw,\theta) = \theta \cdot \ln \frac{cw-1}{cw+1} + \frac{2\theta \cdot cw}{cw^2-1} \tag{8}$$

and

$$\frac{\partial X(cw,\theta)}{\partial cw} = \frac{-4\theta}{(cw^2-1)^2} \tag{9}$$

Thus

$$\frac{\partial P_I(cw,\theta)}{\partial cw} = P_I(cw,\theta) \cdot X(cw,\theta) \tag{10}$$

Then we can have

$$\frac{\partial^2 P_I(cw,\theta)}{\partial cw^2} = P_I(cw,\theta) \cdot [X(cw,\theta)^2 + \frac{-4\theta}{(cw^2-1)^2}] \tag{11}$$

Obviously, $\partial X(cw,\theta)/\partial cw < 0$ which means that $X(cw,\theta)$ decreases as a function of cw. As $\lim_{cw\to+\infty} X(cw,\theta) = 0$ for every θ, we have $X(cw,\theta) > 0$. Since $P_I(cw,\theta) \in [0,1]$, we can have $\partial P_I(cw,\theta)/\partial cw > 0$.

We define

$$X_a(cw,\theta) = \sqrt{-\frac{\partial X(cw,\theta)}{\partial cw}} = \frac{2\sqrt{\theta}}{cw^2-1} \tag{12}$$

We then have

$$\frac{\partial X_a(cw,\theta)}{\partial cw} = \frac{-4\sqrt{\theta} \cdot cw}{(cw^2-1)^2} \tag{13}$$

Now we can combine Equation (11) with Equation (12) and we can get

$$\frac{\partial^2 P_I(cw,\theta)}{\partial cw^2} = P_I(cw,\theta) \cdot [X(cw,\theta) + X_a(cw,\theta)] \cdot [X(cw,\theta) - X_a(cw,\theta)] \tag{14}$$

From Equations (9) and (13), we can get $\partial[X(cw,\theta) + X_a(cw,\theta)]/\partial cw < 0$ and $\partial[X(cw,\theta) - X_a(cw,\theta)]/\partial cw > 0$. By combining the conditions $\lim_{cw\to+\infty}[X(cw,\theta) \pm X_a(cw,\theta)] = 0$, we have $X(cw,\theta) + X_a(cw,\theta) > 0$ and $X(cw,\theta) - X_a(cw,\theta) < 0$. For $P_I(cw,\theta) \in [0,1]$, we can get $\partial^2 P_I(cw,\theta)/\partial cw^2 < 0$.

From the above we can see that $\partial P_I(cw,\theta)/\partial cw$ monotonously approaches 0 with respect to cw. Therefore, for any $cw > cw_{ref}$ we have

$$\max\{|\frac{\partial P_I(t,\theta)}{\partial t}|, t \in [cw_{ref}, cw]\} = |\frac{\partial P_I(cw_{ref},\theta)}{\partial cw_{ref}}| \tag{15}$$

Thus according to Equation (4), we can have the upper bound of $N(cw)$ in Equation (16) for any $cw > cw_{ref}$ as below,

$$|N(cw)| \leq (cw - cw_{ref}) \cdot |\frac{\partial P_I(cw_{ref},\theta)}{\partial cw_{ref}}| \tag{16}$$

2.2.2. Case 2: Channel Contention Level in the Successful Transmission State

For this case, the successful transmission in a given slot happens when there is only one node having data for transmission and the rest nodes do not have data. Thus the probability of successful transmission state can be expressed as:

$$P_{success}(n,\tau) = n\tau(1-\tau)^{n-1} \tag{17}$$

By substituting $n = \theta \cdot cw$ and Equation (1) to Equation (17), we can have the next expression:

$$P_S(cw,\theta) = \frac{2\theta \cdot cw}{cw-1} \cdot (\frac{cw-1}{cw+1})^{\theta \cdot cw} \tag{18}$$

By defining

$$Y(cw,\theta) = \frac{-1}{cw \cdot (cw-1)} \tag{19}$$

We can have

$$\frac{\partial Y(cw,\theta)}{\partial cw} = \frac{2cw-1}{cw^2 \cdot (cw-1)^2} \tag{20}$$

and then we have

$$\frac{\partial P_S(cw,\theta)}{\partial cw} = P_S(cw,\theta) \cdot [X(cw,\theta) + Y(cw,\theta)] \tag{21}$$

Thus we can have

$$\frac{\partial^2 P_S(cw,\theta)}{\partial cw^2} = P_S(cw,\theta) \cdot \{[X(cw,\theta) + Y(cw,\theta)]^2 + \frac{\partial X(cw,\theta)}{\partial cw} + \frac{\partial Y(cw,\theta)}{\partial cw}\} \tag{22}$$

It is obvious that $\partial X(cw,\theta)/\partial cw + \partial Y(cw,\theta)/\partial cw > 0$. Since $P_S(cw,\theta) \in [0,1]$, we have $\partial^2 P_S(cw,\theta)/\partial cw^2 > 0$. As $\partial X(cw,\theta)/\partial cw + \partial Y(cw,\theta)/\partial cw > 0$, and $\lim_{cw\to+\infty}[X(cw,\theta) + Y(cw,\theta)] = 0$, we have $\partial P_S(cw,\theta)/\partial cw < 0$.

Based on the above, we can conclude that $P_S(cw,\theta)$ monotonously approaches 0 with respect to cw. Therefore, for any $cw > cw_{ref}$, we have

$$\max\{|\frac{\partial P_S(t,\theta)}{\partial t}|, t \in [cw_{ref}, cw]\} = |\frac{\partial P_S(cw_{ref},\theta)}{\partial cw_{ref}}| \tag{23}$$

According to Equation (4), for any $cw > cw_{ref}$ we can get the upper bound of $N(cw)$ according to Equation (24) below,

$$|N(cw)| \leq (cw - cw_{ref}) \cdot |\frac{\partial P_S(cw_{ref}, \theta)}{\partial cw_{ref}}| \tag{24}$$

2.2.3. Case 3: Channel Contention Level in the Conflicting Transmission State

Since the idle slot state, successful packet transmission state and conflicting packet transmission state are the only three possible states while accessing the channel, the probability of conflicting packet transmission state $P_C(cw, \theta)$ and its corresponding first-order partial derivative can be calculated as Equations (25) and (26) as below accordingly:

$$P_C(cw, \theta) = 1 - P_I(cw, \theta) - P_S(cw, \theta) \tag{25}$$

$$\frac{\partial P_C(cw, \theta)}{\partial cw} = -\frac{\partial P_I(cw, \theta)}{\partial cw} - \frac{\partial P_S(cw, \theta)}{\partial cw} \tag{26}$$

According to Sections 2.2.1 and 2.2.2, we can have

$$\frac{\partial P_I(cw, \theta)}{\partial cw} \cdot \frac{\partial P_S(cw, \theta)}{\partial cw} < 0 \tag{27}$$

Therefore, for any $cw < cw_{ref}$ we have

$$\max\{|\frac{\partial P_C(t, \theta)}{\partial t}|, t \in [cw_{ref}, cw]\} = \max\{|\frac{\partial P_S(cw, \theta)}{\partial cw}|, |\frac{\partial P_I(cw, \theta)}{\partial cw}|\} \tag{28}$$

Thus according to Equation (4), we can calculate the upper bound of $N(cw)$ as in Equation (29) for any $cw > cw_{ref}$.

$$|N(cw)| \leq (cw - cw_{ref}) \cdot \max\{|\frac{\partial P_I(cw_{ref}, \theta)}{\partial cw_{ref}}|, |\frac{\partial P_S(cw_{ref}, \theta)}{\partial cw_{ref}}|\} \tag{29}$$

2.2.4. The Analysis of System Throughput

According to the above analysis of $P_I(cw, \theta)$, $P_S(cw, \theta)$ and $P_C(cw, \theta)$, the system throughput $S(cw, \theta)$ can be expressed as Equation (30) [25] as follows,

$$S(cw, \theta) = \frac{L_{data} \cdot P_S(cw, \theta)}{T_S \cdot P_S(cw, \theta) + T_C \cdot P_C(cw, \theta) + T_I \cdot P_I(cw, \theta)} \tag{30}$$

where L_{data} is the average frame size at the physical layer, and T_I, T_S, T_C are the average time durations of the three states accordingly which can be estimated by PHY and MAC layers [9].

By separating the variable cw and θ and substituting Equation (25) to Equation (30), we have:

$$S_r(cw, \theta) = \frac{L_{data}}{T_S - T_C + T_C \cdot [P_S^{-1}(cw, \theta) - \eta \cdot P_I(cw, \theta) \cdot P_S^{-1}(cw, \theta)]}, \quad \eta = 1 - \frac{T_I}{T_C} \tag{31}$$

We then define

$$\Lambda(cw, \theta) = P_S^{-1}(cw, \theta) - \eta \cdot P_I(cw, \theta) \cdot P_S^{-1}(cw, \theta) \tag{32}$$

According to Equations (6)–(9) and (18)–(20), we can get the first-order and second-order partial derivatives of $\Lambda(cw, \theta)$ as shown in Equations (33) and (34) below.

$$\frac{\partial \Lambda(cw, \theta)}{\partial cw} = -\frac{1}{P_S(cw, \theta)} \cdot (X(cw, \theta) + Y(cw, \theta) - \eta \cdot P_I(cw, \theta) \cdot Y(cw, \theta)) > 0 \tag{33}$$

$$\begin{aligned}\frac{\partial^2\Lambda(cw,\theta)}{\partial cw^2} =& \frac{1}{P_S(cw,\theta)}\cdot[(X(cw,\theta)+Y(cw,\theta))^2-(\frac{\partial X(cw,\theta)}{\partial cw}+\frac{\partial Y(cw,\theta)}{\partial cw})\\ &-\eta\cdot P_I(cw,\theta)\cdot(Y(cw,\theta)^2-\frac{\partial Y(cw,\theta)}{\partial cw})]<0\end{aligned} \tag{34}$$

Therefore we can have $\partial S(cw,\theta)/\partial cw < 0$ and $\partial^2 S(cw,\theta)/\partial cw^2 > 0$. We can conclude that $\partial S(cw,\theta)/\partial cw$ monotonously approaches 0 with respect to cw. Therefore, for any $cw > cw_{ref}$ we can have

$$max\{|\frac{\partial S(t,\theta)}{\partial t}|, t\in[cw_{ref},cw]\} = |\frac{\partial S(cw_{ref},\theta)}{\partial cw_{ref}}| \tag{35}$$

According to Equation (4), we can calculate the upper bound of $N(cw)$ in Equation (36) for any $cw > cw_{ref}$.

$$|N(cw)| \le (cw-cw_{ref})\cdot|\frac{\partial S(cw_{ref},\theta)}{\partial cw_{ref}}| \tag{36}$$

From the definition of $\Lambda(cw,\theta)$, we can see that L_{data} and T_S are independent of $S(cw,\theta)$. Therefore, we set $L_{data} = 1KB$ for simplicity in the next section.

2.2.5. Model Evaluation

According to the above analysis, the upper bound of $|N(cw)|$ is controlled by Equations (16), (24), (29) and (36). We set the maximum contention window size $cw_{max} = 10000$ in this paper since this size is large enough even for super dense networks. According to the monotone property of $P_I(cw,\theta)$, $P_S(cw,\theta)$, $P_C(cw,\theta)$ and $S(cw,\theta)$, we have:

$$|N(cw)| \le |N(cw_{max})| \tag{37}$$

We can calculate the upper bound of $|N(cw)|$ for $cw \in [cw_{ref}, cw_{max}]$ through Equations (16), (24), (29) and (36) respectively.

In order to improve the estimation accuracy, we introduce sub-cases by $cw = cw_i, i = 0,1,2,...,I_n$ (I_n is a counter of the number of target sub-cases), and we have $cw_{ref} = cw_0 < cw_1 < cw_2 < \cdots < cw_{I_n} < cw_{max}$. Then $|N(cw)|, cw \in [cw_{ref}, cw_{max}]$ can be calculated according to Equation (38) below,

$$|N(cw)| \le |N(cw_{max})| < \sum_{i=0}^{I_n-1}(cw_{i+1}-cw_i)\cdot|\frac{\partial C_1(cw_i,\theta)}{\partial cw_i}| + (cw_{max}-cw_{I_n})\cdot|\frac{\partial C_1(cw_{I_n},\theta)}{\partial cw_{I_n}}| \tag{38}$$

Figure 1 shows the logarithmic variance of $\partial P_I(cw,\theta)/\partial cw$, $\partial P_S(cw,\theta)/\partial cw$ and $\partial S(cw,\theta)/\partial cw$ when cw changes from 32 to 700 and θ changes from 0 to 1. In the figure, we plotted the contours of different orders of magnitude. The maximum contention window size on each contour has been marked.

Based on both Figure 1 and Equation (38), we can calculate $|N(cw)|$. We take an example where the channel contention level is indicated by the probability of an idle slot. By letting $[cw_0 = 32, cw_1 = 59, cw_2 = 132, cw_3 = 286, cw_4 = 620]$, we can get $|N(cw)| < 2.48\times10^{-4}$ from Equation (38) as shown in Figure 1. Table 1 shows $|N(cw)|$ in the case where we have $cw_{ref} = 32, 64, 128, 256$. As for the throughput, $|N(cw)|$ is normalized by being divided by $S(cw_{ref},\theta_{opt})$ which is the optimal throughput in this case.

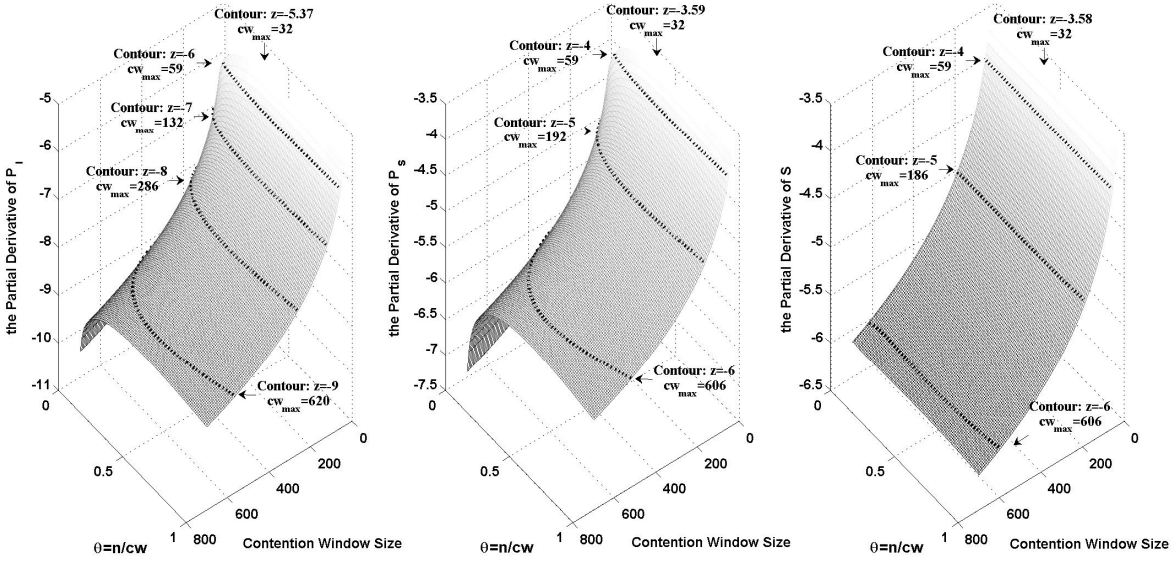

Figure 1. The logarithmic variance of $\partial P_I(cw,\theta)/\partial cw$, $\partial P_S(cw,\theta)/\partial cw$ and $\partial S(cw,\theta)/\partial cw$, when cw changes from 32 to 700 and θ changes from 0 to 1.

Table 1. The certification results of $|N(cw)|$ in the cases where channel contention level is indicated by different channel parameters.

	$cw_{ref} = 32$	$cw_{ref} = 64$	$cw_{ref} = 128$	$cw_{ref} = 256$
$C_1 = P_i$	2.48×10^{-4}	9.20×10^{-5}	2.88×10^{-5}	1.32×10^{-5}
$C_1 = P_s$	2.77×10^{-2}	1.80×10^{-2}	7.70×10^{-3}	4.10×10^{-3}
$C_1 = P_c$	2.75×10^{-2}	1.79×10^{-2}	7.70×10^{-3}	4.10×10^{-3}
$C_1 = S$	3.08%	2.06%	0.87%	0.46%

3. Adaptive Multi-Level Contention Window Tuning Algorithm

In the previous sections, we introduced normalized contention analysis model through which every possible channel state is effectively mapped onto a reference case. In this section, we propose a multi-level contention window tuning scheme with contention window tuning rate adaptively adjusted based on the channel contention information.

3.1. Basic Contention Window Tuning under Normalized Model

As shown in Table 1, $|N(cw)|$ in $P_I(cw_{ref},\theta)$ is smaller than that in $P_s(cw_{ref},\theta)$ and $P_c(cw_{ref},\theta)$ in the normalized model. Therefore, we estimate the probability of idle slot state to control the channel contention in our proposed algorithm. As the estimation module with the sample size of hundreds can only effectively distinguish the deviations which are larger than 10^{-3}, $|N(cw)| \leq 2.48 \times 10^{-4}$ can hardly be detected. Therefore, $cw_{ref} = 32$ is applied in the following analysis.

We make $CW_{MAX} = 10,000$ as the maximum contention window size and $CW_{MIN} = 32$ as the minimum contention window size. The probability of the idle slot state of the channel at all different contention level can be calculated through $P_I(cw_{ref},\theta) = P_I(cw_{ref},n/cw)$. Figure 2 shows the relationship between graph of $P_I(cw_{ref},\theta)$ and the normalized $S(cw_{ref},\theta)$ which is obtained by $S(cw_{ref},\theta)/\max\{S(cw_{ref},\theta)\}$. As shown by the figure, a one-one mapping can be obtained between the probability of an idle slot in channel and the normalized throughput.

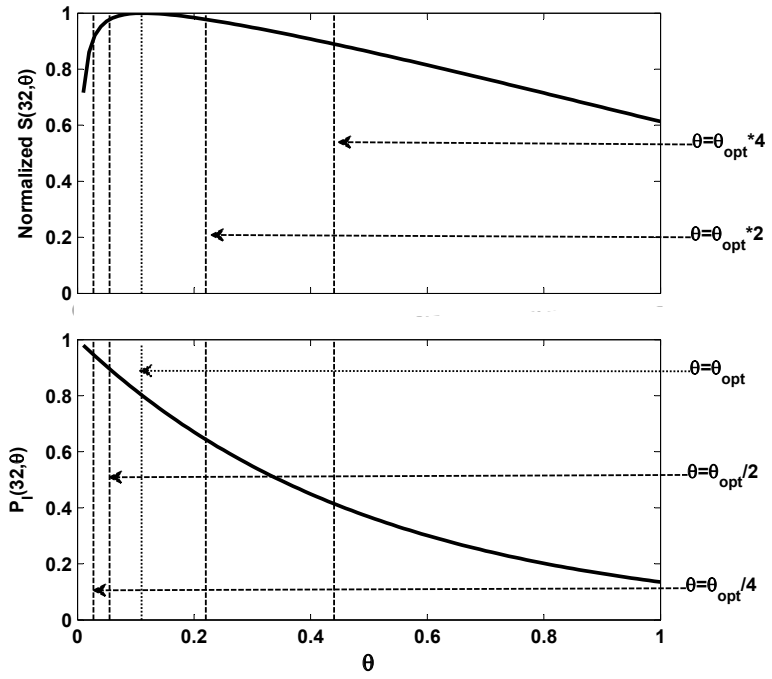

Figure 2. The $P_I(32,\theta)$ and the normalized $S(32,\theta)$ vary with different θ.

From the figure above, we can easily get θ_{opt} corresponding to the optimal overall throughput. The probability of the idle slot state can be calculated using $P_I(cw_{ref},\theta_{opt})$. In order to achieve maximum throughput, θ should stay at its optimal value, which can be fully indicated by $P_I(cw_{ref},\theta)$. We denote $\hat{P}_I$ as the estimated $P_I(cw_{ref},\theta)$. If we detect $\hat{P}_i > P_I(cw_{ref},\theta_{opt})$, we know a smaller cw is required to pull the θ back to θ_{opt}, and likewise if we detect $\hat{P}_I < P_I(cw_{ref},\theta_{opt})$, then cw should be increased.

3.2. Adaptive Contention Window Tuning

As we know, the number of contending nodes may vary from tens to hundreds within a short time period depending on if nodes have data for transmission. The dynamics of number of contending nodes will deviate $P_I(cw_{ref},\theta)$ from its optimal value. The contention window tuning rate depends on the specific number of contending nodes and how fast it changes. If the number of contending nodes changes slowly, we have enough time to tune the contention window to match the current number of contending nodes. If a large number of contending nodes join the contention, a much faster tuning process is required to make sure the contention window reaches the optimal value on time. Unfortunately, these two situations can hardly be solved using one single contention window tuning rate because a more refined tuning requires a smaller tuning rate, while a faster tuning needs a larger turning rate. In order to solve the problem above, we propose an adaptive contention window tuning strategy in which the contention window tuning rate can be set dynamically according to the real-time channel contention level.

3.2.1. Algorithm Design

In the normalized model, a sudden increase of contending nodes can be represented by the increase of θ through the mapping of $\theta = n/cw$. Based on the one-to-one mapping in Figure 2, we can see that the probability of an idle slot state is exactly determined by θ. It means that the amplitude of changing in the number of contending nodes can be indicated by the variance of $P_I(cw_{ref},\theta)$. Therefore, contention window tuning rates can be determined in real-time adaptively based on $\hat{P}_I$.

In Figure 2, we can see the relationship between $P_I(cw_{ref},\theta)$ and the corresponding overall throughput by dotted lines in the cases where θ suddenly increases to 2 and 4 times of θ_{opt}, and then

decreases to half and quarter of θ_{opt}. Just the same as the basic contention window tuning processing during which we believe θ has increased when $\hat{P}_I$ is smaller than $P_i(cw_{ref}, \theta_{opt})$, and θ has at least doubled when the $\hat{P}_I$ is smaller than $P_I(cw_{ref}, 2 \cdot \theta_{opt})$. Therefore, the contention window can be increased again to make sure θ is approaching θ_{opt} faster. Similarly the contention window size should be increased again if $\hat{P}_I < P_I(cw_{ref}, 4 \cdot \theta_{opt})$ happens. The similar rules will be applied so that if $\hat{P}_I > P_I(cw_{ref}, \theta_{opt}/2)$ or $\hat{P}_I > P_I(cw_{ref}, \theta_{opt}/4)$ happens, the contention window should be decreased by half each time.

Based on the discussions above, the M-Level contention window tuning algorithm with a minimum tuning rate of γ is proposed in Algorithm 1. In this algorithm, the decisions on how to change contention window size and tuning rate are made based on a pair of vectors: $vector\_inc[M]$ and $vector\_dec[M]$. Each vector divides the contention channel into M different contention levels according to its M ordered elements using M thresholds of $P_I(cw_{ref}, \theta)$. As shown in Algorithm 1, the ordered thresholds of $P_I(cw_{ref}, \theta)$ are defined as $P_I(cw_{ref}, \theta_{opt} \cdot \gamma^i), i \in [0, M-1]$ and $P_I(cw_{ref}, \theta_{opt}/\gamma^i), i \in [0, M-1]$, which are pre-defined during the initialization period. A node running M-Level contention window tuning algorithm will compare the $\hat{P}_I$ with the elements of the vector. Once $\hat{P}_I$ is smaller than one of the elements in $vector\_inc[M]$, the contention window would be multiplied by γ. Likewise, if the $\hat{P}_I$ is larger than one of the elements in $vector\_dec[M]$, the contention window would be divided by γ. Based on the above, the update of M contention window tuning rate can be specified as $[\gamma, \gamma^2, \ldots, \gamma^M]$ according to different channel contention levels.

Algorithm 1 A M-Level contention window tuning algorithm with a minimum tuning rate of γ.

Initialization

unsigned $slot\_cnt = 0, idle\_slot\_cnt = 0, k$;
double $cw = CW\_MIN, vector\_inc[M], vector\_dec[M]$;
for $k = 0; k < M; k++$ **do**
  $vector\_inc[k] = P_I(32, \theta \cdot \gamma^k)$;
  $vector\_dec[k] = P_I(32, \theta/\gamma^k)$;
end for

Algorithm Body

while 1 **do**
  if detecting an idle slot **then**
    $slot\_cnt++$;
    if the slot is over with idle **then**
      $idle\_slot\_cnt++$;
    end if
  end if
end while
while 1 **do**
  if before generating a new backoff period **then**
    if $slot\_cnt - idle\_slot\_cnt \geq 5$ **then**
      for $k = 0; k < M; k++$ **do**
        if $idle\_slot\_cnt/slot\_cnt < vector\_inc[k]$ **then**
          $cw = cw * \gamma$;
        end if
        if $idle\_slot\_cnt/slot\_cnt > vector\_dec[k]$ **then**
          $cw = cw/\gamma$;
        end if
      end for
      $slot\_cnt = 0, idle\_slot\_cnt = 0$;
    end if
  end if
end while

3.2.2. The Selection of M

Through our proposed M-level contention window tuning algorithm, the contention window tuning accuracy can be improved by adopting a small γ, and a high tuning rate can also be achieved by applying large M. While the γ can be selected according to the need, the range of M is very limited. It is because the algorithm is based on the assumption that the thresholds including $vector\_inc[k]$, $k \in [0, M-1]$ and $vector\_dec[k], k \in [0, M-1]$ can be represented by $\hat{P}_I$. When the thresholds are very close to each other and the sample size which is used to calculate $\hat{P}_I$ is very small, the difference among thresholds is too small to matter. For example, when $\hat{P}_I$ is calculated based on 10 samples of the channel, the thresholds with the value of 0.71 and 0.74 could hardly be identified because the resolution of $\hat{P}_I$ is around 0.1 which is decided by the samples.

When selecting M, we should ensure the difference between neighboring thresholds is large enough to be represented by the $\hat{P}_I$. As shown in Algorithm 1, $\hat{P}_I$ is calculated based on the consecutive idle slots. We firstly start with a special case where $idle\_slot\_cnt = slot\_cnt$. In this case, $\hat{P}_I = 1$ which means the channel is always be busy which is obviously it impossible. Therefore, in order to make $\hat{P}_I$ valid, $\hat{P}_I$ should not be updated unless a busy slot is detected.

We consider another case where $\hat{P}_I$ is used to update the contention window after each busy slot. In this case, the sample size right before calculating $\hat{P}_I$ can be interpreted as the estimation of the average number of consecutive idle slots between two busy slots. As the probability of an idle channel in a certain slot is a 0-1 distribution with probability of $P_I(cw_{ref}, \theta)$, the average number of consecutive idle slots between two busy slots can be expressed as $1/P_I(cw_{ref}, \theta)$. Therefore, we can calculate the sample size of $\hat{P}_I$ through Equation (39).

$$SS(\theta) = \left\lceil \frac{1}{P_I(cw_{ref}, \theta)} \right\rceil \tag{39}$$

The sample size can be obtained based on θ, and the sample sizes of $\hat{P}_I$ for all the thresholds in $vector\_inc$ and $vector\_dec$ can be calculated using $SS(vector\_inc[k]), k \in [0, M-1]$ and $SS(vector\_dec[k]), k \in [0, M-1]$. If Equation (40) holds, $P_I(cw_{ref}, vector\_inc[k]), k \in [0, M-1]$ and $P_I(cw_{ref}, vector\_dec[k]), k \in [0, M-1]$ can be totally differentiated because the numbers of consecutive idle slots between two busy ones for different thresholds are quite different, and these differences can be effectively indicated by $\hat{P}_I$ after each busy slot. Obviously a maximal M can be derived from Equations (39) and (40).

$$\begin{cases} |SS(vector\_inc[u]) - SS(vector\_inc[v])| \geq 1, u, v \in [0, M-1] \\ |SS(vector\_dec[u]) - SS(vector\_dec[v])| \geq 1, u, v \in [0, M-1] \end{cases} \tag{40}$$

In order to get larger M, we can limit the minimum number of busy slots before calculating $\hat{P}_I$ to 5 in this paper, and then we can have $SS(\theta) = [5/P_i(32, \theta)]$. According to the requirement in Equation (40), Table 2 lists the maximal M for different γ.

Table 2. The maximal M and corresponding equivalent maximal contention window tuning rate (γ_{max}) in M-level contention window tuning algorithm for different γ ranging from 1.2 to 2.0.

γ	1.2	1.3	1.4	1.5	1.6	1.7	1.8	1.9	2.0
M	10	9	9	7	7	6	6	5	5
γ_{max}	5.16	8.16	14.76	11.39	16.78	14.20	18.90	13.03	16.00

The contention window size will be updated if and only if the number of busy slots $\hat{P}_I$ is larger than 5 in our case. As the contention window tuning will only be triggered when a node is about to access the channel, the number of busy slots represented by $\hat{P}_I$ will be always larger than 5. This can sufficiently ensure the differentiation of thresholds in M-level contention window tuning algorithm. Now the M-level contention window tuning algorithm with parameters of M and γ as

inputs can be fully constructed. In next section, we will focus on the performance evaluation of our proposed algorithm.

4. Performance Evaluation of M-level Contention Window Tuning Algorithm

In this section, we study the performance of the proposed algorithm using OMNET++ simulator [29]. Some of the PHY and MAC layer parameters are listed in Table 3 based on the IEEE 802.11b standard [9]. The reason for studying the performance based on IEEE 802.11b network is that IEEE 802.11b supports longer distance than that of IEEE 802.15.4, so that the number of contending nodes in IEEE 802.11 networks can be much larger than that in IEEE 802.15.4 networks. Since the backoff processing is universal in all the IEEE 802.11x and IEEE 802.15.4 standards, the proposed algorithm and its performance study work for all the backoff based medium access schemes in IEEE 802.11x and 802.15.4 networks.

Table 3. PHY layer and MAC layer parameters used in simulation.

Parameters	Value	Parameters	Value
Channel Bit Rate	11 Mbps	Payload Length	1 KB
Slot Time(ST)	20 µs	MAC Header	224 bit
SIFS	10 µs	RTS	160 bit
DIFS	50 µs	CTS	112 bit
PHY Header	192 µs	ACK	112 bit

We apply two different types of networks in terms of node density: sparse networks where the number of contending nodes is from 4 to 20, and the super dense networks where the number of contending nodes can be up to 400. We firstly study the case when the number of contending nodes remains static during the whole simulation period, and then consider the dynamic case where the number of contending nodes changes as time goes.

4.1. Simulations in Sparse Networks

In this section, the throughput and fairness of the proposed algorithm are studied in sparse networks where the number of contending nodes varies from 4 to 20. 8 groups of simulations are conducted with different parameters for algorithms including BEB [6] and the idle sense algorithm [23], which is a classical and efficient estimation-based algorithm. We choose different γ and M for our proposed algorithm.

Figure 3 shows the results of the throughput while applying different backoff algorithms. In the figure, the curve named opt is the result of $S(cw, \theta_{opt})$ which is the optimum throughput that CSMA/CA based networks can achieve theoretically. From the figure we can see that BEB is efficient when the number of contending nodes is very small. However when number of contending nodes increases, the throughput decreases rapidly. This is because BEB algorithm can not estimate the channel contention accurately enough to fully utilize the channel when the channel contention becomes intense. The throughput of idle sense algorithm decreases when the number of contending nodes is small. This is because a smaller number of contending nodes has a shorter backoff time, so that a small number of samples could be obtained during the backoff period. The limit on the number of samples will lead to inaccurate estimates. The inaccurate estimates together with the unbalanced contention window tuning rate in idle state normally will introduce errors to the contention window size. This problem is effectively solved in our proposed algorithm. Two simulations of our proposed algorithm with parameters of $(\gamma = 1.2, M = 10)$ and $(\gamma = 1.8, M = 6)$ are presented in the figure. In order to evaluate their performance, we compare with the performance of the proposed algorithm with settings of $(\gamma = 1.2, M = 1)$ and $(\gamma = 1.8, M = 1)$. We also found that the throughput with

($\gamma = 1.2, M = 10$) and ($\gamma = 1.8, M = 6$) is much lower than that of other cases. This is mainly because inaccurate estimates result in even larger contention window deviation from the optimal size in M-level tuning scheme. Nevertheless, the overall throughput of our proposed algorithm is very close to the optimal value with a loss of no more than 1% for all the settings. We also studied the throughput of with ($\gamma = 1.2^{10} = 5.19, M = 1$) and ($\gamma = 1.8^{6} = 18.9, M = 1$). From the figure, we can see that larger contention window tuning rate decreases the throughput in sparse networks because tuning of contention window size happens very frequently even when the number of contending nodes changes slightly. Therefore, we can conclude that in order to achieve high throughput performance in sparse networks, small contention window tuning rate is necessary.

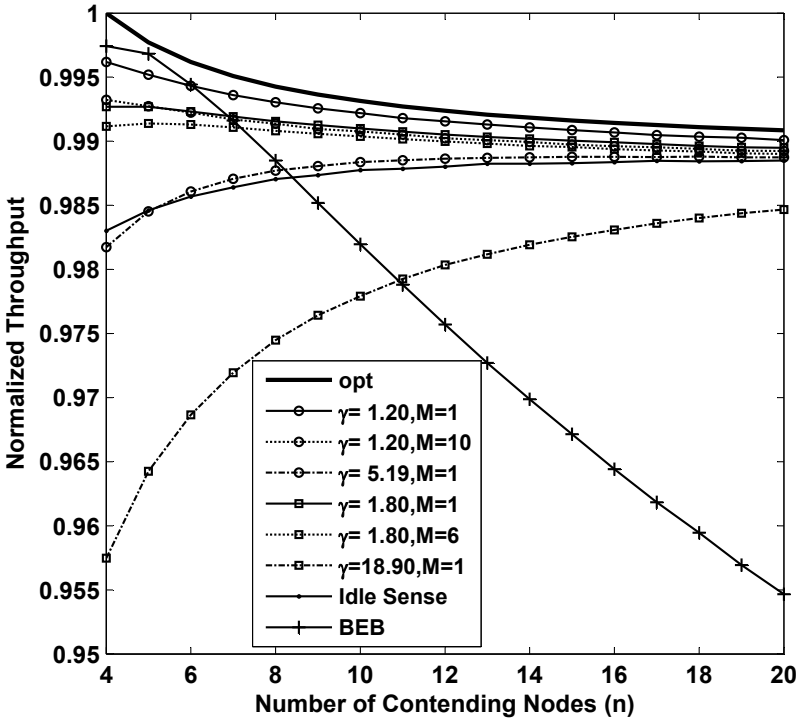

Figure 3. The normalized throughput performance of M-level contention window tuning algorithm with different M and γ, idle sense algorithm and BEB for sparse deployed networks.

In estimate-based backoff algorithms, all nodes try to improve the channel utilization rate to approach the optimal rate. However, during the process, the contention window size on each contending node is different when network topology changes or estimate error occurs, and this may remain for quite a long time. Therefore, the fairness among nodes is another key factor of estimate-based backoff algorithms. In this paper, we apply the fairness index concept defined by Jain [30] to evaluate the fairness of each algorithm. The fairness index is defined as:

$$FI = \frac{(\sum_i TH_i)^2}{n \cdot \sum_i TH_i^2} \tag{41}$$

in which n is the number of contending nodes, and TH_i is the throughput of node i. Based on the Cauchy-Schwartz inequality, we obtain $FI \leq 1$, and the equality holds if and only if all TH_i are equal. Thus, the more FI approaches to 1, the better fairness the algorithm can achieve. Figure 4 shows the fairness of the 8 sets of simulations we analyzed above. Fortunately all of them achieved remarkable fairness which is above 0.995 where the number of contending nodes ranges from 4 to 20.

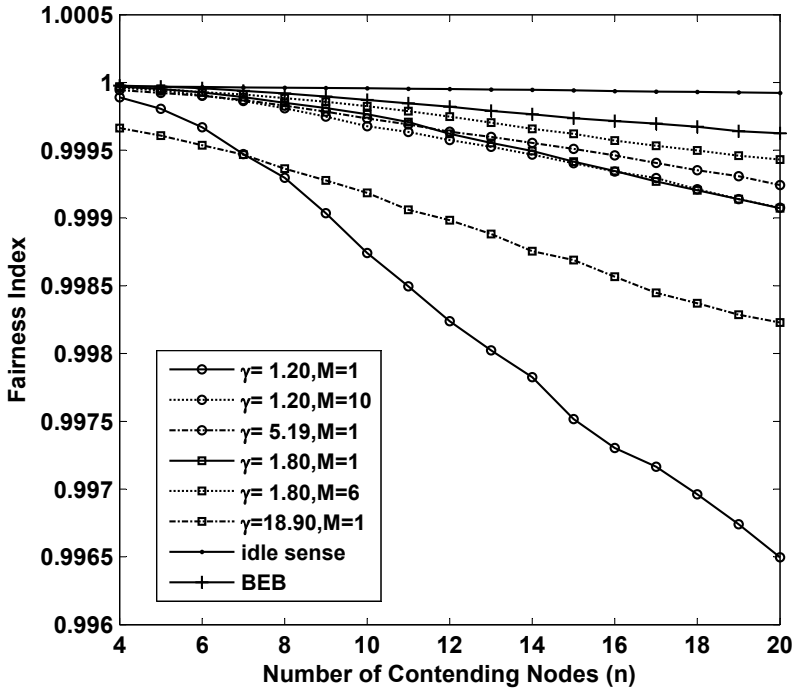

Figure 4. The fairness performance of M-level contention window tuning algorithm with different M and γ, idle sense algorithm and BEB for sparse deployed networks.

4.2. Performance in Super Dense Networks

In this section, the throughput and fairness are evaluated in super dense networks where the number of contending nodes varies from 10 to 400. Because smaller contention window tuning rate is required in sparse deployed networks and the throughput of BEB declines severely when the number of contending nodes reaches 20, in this section we focus on the performance of idle sense algorithm and our proposed algorithm with parameters of $(\gamma = 1.2, M = 10)$, $(\gamma = 1.8, M = 6)$, $(\gamma = 1.2, M = 1)$ and $(\gamma = 1.8, M = 1)$.

Figure 5 shows the simulation results of network throughput of different backoff algorithms. The curve named opt in the figure is the result of theoretical optimal throughput calculated by $S(cw, \theta_{opt})$. From the figure we can see that the throughput of simulation always oscillates around a certain level. It is reasonable because the contention window size of each node may be enumerated, and it is only optimal for some contending nodes. In other cases, the throughput slightly decreases because of the deviation of contention window size from their optimal size. The throughput with $(\gamma = 1.2, M = 1)$, $(\gamma = 1.8, M = 1)$, and that of the idle sense algorithm are all very close to the optimal value. The throughput of M-level contention window tuning is almost equivalent to that of 1-level tuning case and that of idle sense algorithm, and the overall differentiation is no more than 0.5%.

The fairness of different algorithms in super dense deployed networks is shown in Figure 6. From the figure we can see that as the number of contending nodes increases, there is a significant decline of the fairness in all the simulations. In sparse networks, smaller contention tuning rate improves the throughput, however it has negative impact on the fairness. This is because differential contention window sizes with different contending nodes are much easier to be increased during a substantial change of contention window size through small amplitude adjustments. Therefore, the fairness of idle sense algorithm decreases rapidly after the number of contending nodes is larger than 50. However, the superiority of M-level contention window tuning algorithm is fully demonstrated in the simulations. The fairness indexes for $(\gamma = 1.2, M = 10)$, $(\gamma = 1.8, M = 6)$ are larger than 0.97 even when the number of contending nodes increases to 400.

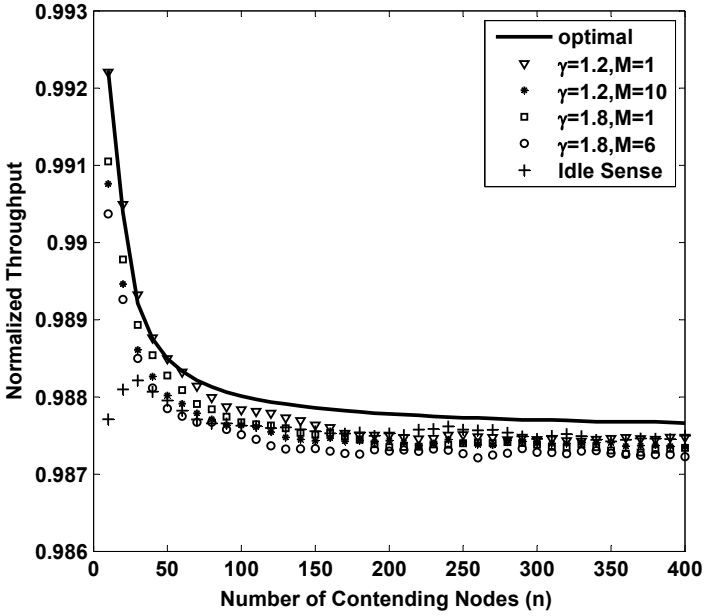

Figure 5. The normalized throughput performance of M-level contention window tuning algorithm with different M and γ and idle sense algorithm for super dense deployed networks.

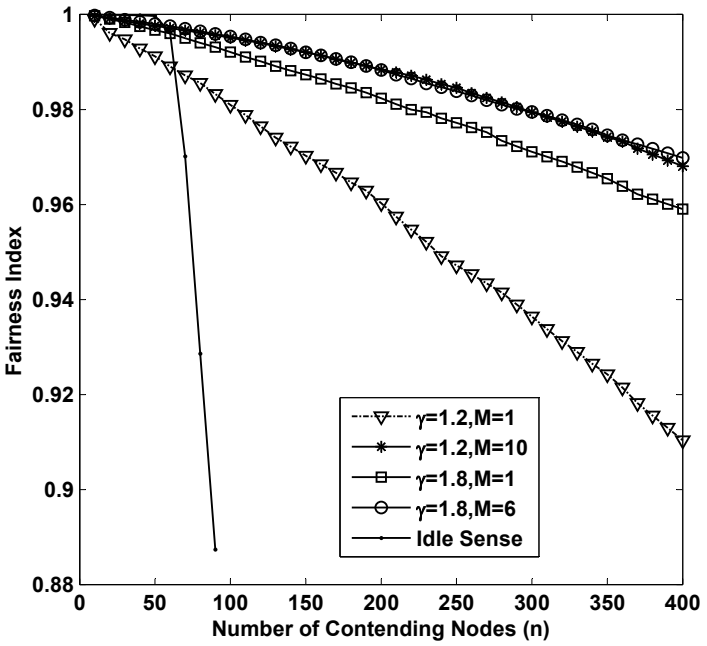

Figure 6. The fairness performance of M-level contention window tuning algorithm with different M and γ and idle sense algorithm for super dense deployed networks.

4.3. Simulations in Very Dynamic Super Dense Networks

According to the definition of θ in Section 2.1, a sudden change of the number of contending nodes will make θ deviates from its optimal value. As shown in Figure 2, throughput declines with respect to such deviations. Backoff algorithms detect such changes and try to make θ back to optimal by tuning the contention window. Thus, the contention window tuning efficiency can be represented by the time spent on.

In this section we study the performance of our proposed algorithm in settings where the channel contention changes frequently as the number of nodes contending the channel changes dramatically. In the simulations, the number of contending nodes changes following steps of (4, 8, 4, 15, 4, 40, 4, 100, 4, 200, 4, 300, 4, 400 and 4) and each step lasts for 5 seconds. Figure 7 shows the dynamics of throughput of different algorithms running the simulation above. From the figure we can see that the throughput becomes extremely low when the number of contending nodes changes suddenly, and the throughput gradually increases to the optimal value after a certain period of time with the help of the proposed algorithm. When the number of contending nodes changes in small scale, all the algorithms have similar performance. However when the number of contending nodes changes tremendously, the throughput of BEB decrease very much. The idle sense algorithm has the slowest contention window tuning speed because of its small contention window tuning rate as shown by the Figure. Similar situation appears in our proposed algorithm with $(\gamma = 1.2, M = 1)$. In the situation where the number of contending nodes changes from 4 to 400, their adaptation time is more than 3 s. By applying the larger $\gamma = 1.8$, the adaptation time is effectively reduced to around 2s. Our proposed M-level tuning algorithm can reduce the adaptation time dramatically since a much larger contention window tuning rate can be applied when the channel contention becomes much more intensive according to our algorithm. From the figure we can see that for the situations where $n \leq 400$, our proposed algorithm with parameters of $(\gamma = 1.2, M = 10)$ and $(\gamma = 1.8, M = 6)$ has a very small adaptation time smaller than 0.5 s.

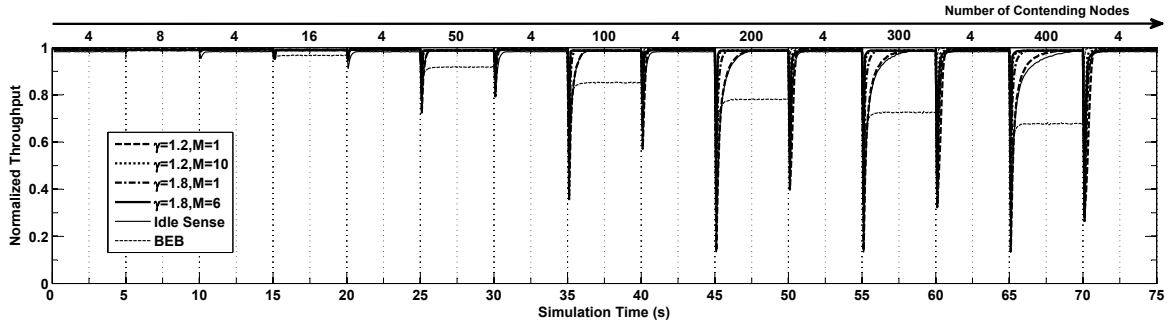

Figure 7. The normalized throughput performance of M-level contention window tuning algorithm with different M and γ, idle sense algorithm and BEB when the number of contention nodes changes.

5. Conclusions

In this paper, an adaptive and efficient channel access method for super dense wireless sensor networks was proposed. By constructing and verifying a normalized channel contention model, we mapped the diverse channel parameters onto reference cases. Our proposed model significantly reduces the complexity of channel contention analysis. Based on this normalized model, a multi-level contention window tuning algorithm was proposed. By estimating the channel contention level based on the deviation degree of the channel parameters from its optimal value, the contention window tuning rate can be set dynamically and adaptively according to the number of contending nodes to accelerate the contention window adaptation process.

To evaluate the performance of the proposed algorithm, we performed comprehensive simulations in sparse networks, super dense networks and dynamic networks. Simulation results show that our proposed algorithm can provide stable throughput and fairness performance close to the theoretical throughput bound in networks with different node densities and dynamics. Our algorithm can also achieve high contention window tuning speed in dynamic networks where the number of active nodes changes rapidly.

Acknowledgments: This work was supported in part by the National Natural Science Foundation of China (No. 41-174-158) and SinoProbe-09-04 (No. 2010-1108-1-4).

Author Contributions: All the authors have contributed to ensure the quality of this work. Chunyang Lei designed the experiments, conceived the new proposed algorithms and wrote the paper. Xuekun Zhang performed the experiments. Gengfa Fang analyzed the data and contributed to write the paper. Finally, Hongxia Bie coordinated and supervised the work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. Lopez-Iturri, P.; Aguirre, E.; Azpilicueta, L.; Fernandez-Valdivielso, C.; Matias, I.; Falcone, F. Analysis of efficient dense wireless sensor network deployment in Smart City environments. In Proceedings of the 2014 IEEE SENSORS, Valencia, Spain, 2–5 November 2014; pp. 8–10.
2. Das, K.; Havinga, P. Evaluation of DECT-ULE for robust communication in dense wireless sensor networks. In Proceedings of the 2012 IEEE International Conference on the Internet of Things, Wuxi, China, 24–26 October 2012; pp. 183–190.
3. Matsuo, K.; Goto, K.; Kanzaki, A.; Hara, T.; Nishio, S. A study on efficient event monitoring in dense mobile wireless sensor networks. In Proceedings of the 2014 IEEE 33rd International Symposium on Reliable Distributed Systems (SRDS), Nara, Japan, 6–9 October 2014; pp. 341–342.
4. Al-Ahmadi, S.; Al-Dhelaan, A. EDGS: Efficient data gathering scheme for dense wireless sensor networks. In Proceedings of the 2013 IEEE Globecom Workshops (GC Wkshps), Atlanta, GA, USA, 9–13 December 2013; pp. 848–854.
5. Cali, F.; Conti, M. Dynamic tuning of the IEEE 802.11 protocol to achieve a theoretical throughput limit. *IEEE/ACM Trans. Netw.* **2000**, *8*, 785–799.
6. Al-Ammal, H.; Goldberg, L.A.; MacKenzie, P. Binary exponential backoff is stable for high arrival rates. In Proceedings of the 17th Annual Symposium on Theoretical Aspects of Computer Science, Lille, France, February 2000; pp. 169–180.
7. Kwak, B.J.; Song, N.O.; Miller, M.E. Performance analysis of exponential backoff. *IEEE/ACM Trans. Netw.* **2005**, *13*, 343–355.
8. LAN/MAN Standards Committee of the IEEE Computer Society. IEEE Standard for Local and Metropolitan Area Networks ł Part 15.4: Low-Rate Wireless Personal Area Networks (LR-WPANs). In *IEEE Std 802.15.4-2006*; IEEE: New York, NY, USA, 2006.
9. LAN/MAN Standards Committee of the IEEE Computer Society. IEEE Standard for Local and Metropolitan Area Networks ł Part 11b: Higher Speed Physical Layer Extension in the 2.4GHz Band. In *IEEE Std 802.11b-1999*; IEEE: New York, NY, USA, 1999.
10. LAN/MAN Standards Committee of the IEEE Computer Society. IEEE Standard for Local and Metropolitan Area Networks ł Part 11n: Enhancement for higher throughput. In *IEEE Std 802.11n-2009*; IEEE: New York, NY, USA, 2009.
11. Felemban, E.; Ekici, E. Single hop IEEE 802.11 DCF analysis revisited: Accurate modeling of channel access delay and throughput for saturated and unsaturated traffic cases. *IEEE Trans. Wirel. Commun.* **2011**, *10*, 3256–3266.
12. Xu, D.; Sakurai, T.; Vu, H.L. An analysis of different backoff functions for an IEEE 802.11 WLAN. In Proceedings of the 68th IEEE Vehicular Technology Conference, Calgary, AL, Canada, 21–24 September 2008; pp. 1–5.
13. Shi, S.; Zhi, Y.; Deng, M.; Peng, Y. Logarithm increase-linear decrease backoff algorithm based on self-adaptive optimal contention window. In Proceedings of the IEEE International Conference on Computer Application and System Modeling (ICCASM), Taiyuan, China, 22–24 October, 2010; Volume 10, pp. V10–V657.
14. Singh, D.; Pandey, B.; Tomar, G.S.; Sarkar, B. Performance evaluation of backoff method–effect of backoff factor on exponential backoff algorithm. In Proceedings of the IEEE 5th International Conference on Computational Intelligence and Communication Networks (CICN), Mathura, India, 27–29 September 2013; pp. 82–86.
15. Tang, J.; Cheng, Y.; Zhuang, W. Real-time misbehavior detection in IEEE 802.11-based wireless networks: An analytical approach. *IEEE Trans. Mob. Comput.* **2014**, *13*, 146–158.
16. Haas, Z.; Deng, J. On optimizing the backoff interval for random access schemes. *IEEE Trans. Commun.* **2003**, *51*, 2081–2090.

17. Bianchi, G.; Tinnirello, I. Kalman filter estimation of the number of contending terminals in an IEEE 802.11 networ. In Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003), San Francisco CA, USA, 1–3 April 2003; pp. 844–852.
18. Kang, S.W.; Cha, J.R.; Kim, J.H. A novel estimation-based backoff algorithm in the IEEE 802.11 based wireless network. In Proceedings of the 7th Annual IEEE Consumer Communications & Networking Conference (CCNC 2010), Las Vegas, NV, USA, 9–12 January 2010; pp. 1–5.
19. Liang, H.M.; Zeadally, S.; Chilamkurti, N.K.; Shieh, C.K. A novel pause count backoff algorithm for channel access in IEEE 802.11 based wireless LANs. In Proceedings of the IEEE International Symposium on Computer Science and Its Applications (CSA'08), Hobart, Australia, 13–15 October 2008; pp. 163–168.
20. Jamali, A.; Hemami, S.; Berenjkoub, M.; Saidi, H. An adaptive MAC protocol for wireless LANs. *J. Commun. Netw.* **2014**, *16*, 311–321.
21. Kwon, Y.; Fang, Y.; Latchman, H. A novel MAC protocol with fast collision resolution for wireless LANs. In Proceedings of the 22nd Annual Joint Conference of the IEEE Computer and Communications Societies (INFOCOM 2003), San Francisco CA, USA, 1–3 April 2003; Volume 2, pp. 853–862.
22. Kwon, Y.; Fang, Y.; Latchman, H. Design of MAC protocols with fast collision resolution for wireless local area networks. *IEEE Trans. Wirel. Commun.* **2004**, *3*, 793–807.
23. Heusse, M.; Rousseau, F.; Guillier, R.; Duda, A. Idle sense: An optimal access method for high throughput and fairness in rate diverse wireless LANs. *ACM SIGCOMM Comput. Commun. Rev.* **2005**, *35*, 121–132.
24. Abdeddaim, N.; Theoleyre, F.; Heusse, M.; Duda, A. Adaptive IEEE 802.15.4 MAC for throughput and energy optimization. In Proceedings of the 2013 IEEE International Conference on Distributed Computing in Sensor Systems (DCOSS 2013), Cambridge, MA, USA, 21–23 May 2013; pp. 223–230.
25. Alizadeh-Shabdiz, F.; Subramaniam, S. Analytical models for single-hop and multi-hop ad hoc networks. *Mob. Netw. Appl.* **2006**, *11*, 75–90.
26. Toledo, A.L.; Vercauteren, T.; Wang, X. Adaptive optimization of IEEE 802.11 DCF based on bayesian estimation of the number of competing terminals. *IEEE Trans. Mob. Comput.* **2006**, *5*, 1283–1296.
27. Gottapu, S.; Tatineni, M.; Kumar, M. Performance analysis of collision alleviating distributed coordination function protocol in congested wireless networks - a markov chain analysis. *IET Netw.* **2013**, *2*, 204–213.
28. Gao, Y.; Sun, X.; Dai, L. Throughput optimization of heterogeneous IEEE 802.11 DCF networks. *IEEE Trans. Wirel. Commun.* **2013**, *12*, 398–411.
29. OMNET++. OMNeT++ Discrete Event Simulator. Available online: https://omnetpp.org/ (accessed on 30 September 2015).
30. Jain, R.; Chiu, D.M.; Hawe, W.R. *A Quantitative Measure of Fairness and Discrimination for Resource Allocation in Shared Computer System*; Digital Equipment Corporation: Hudson, MA, USA, 1984.