Distributed Optimal Contention Window Control for Elastic Traffic in Single Cell Wireless LANs

Yaling Yang, *Member, IEEE,* Jun Wang, *Member, IEEE,* and Robin Kravets, *Member, IEEE*

***Abstract*— This paper presents a theoretical study on distributed contention window control algorithms for achieving arbitrary bandwidth allocation policies and efficient channel utilization. By modeling different bandwidth allocation policies as an optimal contention window assignment problem, we design a general and fully distributed contention window control algorithm, called GCA (General Contention window Adaptation), and prove that it converges to the solution of the contention window assignment problem. By examining the stability of GCA, we identify the optimal stable point that maximizes channel utilization and provide solutions to control the stable point near the optimal point. Due to the generality of GCA, our work provides a theoretical foundation to analyze existing and design new contention window control algorithms.**

I. Introduction

Due to the shared medium and the intrinsic scarcity of bandwidth in wireless LANs, nodes must contend for the channel and compete for bandwidth. While both contention resolution and bandwidth allocation can be achieved through centralized scheduling at a wireless LAN access point, such centralized control is not scalable to a large number of nodes, suggesting the use of distributed algorithms for both contention resolution and bandwidth allocation. Common distributed contention resolution protocols, including IEEE 802.11 [1], MACA [2] and MACAW [3], use contention windows to control the channel access of competing nodes. These window-based contention resolution protocols not only reduce network congestion, but also directly affect the bandwidth allocation between competing nodes. Therefore, it is natural to extend such algorithms to support bandwidth allocation.

Network applications generate either realtime traffic (e.g., video/audio streaming) or elastic traffic [4] (e.g., file transfer). While a realtime flow has a certain rate and requires QoS guarantees, an elastic flow always has backlogged packets and can adjust its rate to fill the available bandwidth. Hence, competing elastic flows are more concerned about the “fairness” and “efficiency” of bandwidth allocation. While efficiency is defined by bandwidth utilization, fairness must be defined by the goals of the particular network. For example, fairness may mean that every competing node obtains the same bandwidth, that the share of bandwidth to a node is proportional to its priority, or that the highest priority node should obtain all of the bandwidth. Due to these different goals, it is desirable to support any definition of fairness. Hence, the focus of our research is to use contention window control to allocate bandwidth to elastic traffic so that both an arbitrary definition of fairness and efficient channel utilization are achieved in single-cell wireless LANs. Using contention window control for service guarantees for realtime traffic is beyond the scope of this paper and can be found in our prior work [5], [6].

Y. Yang is with Virginia Polytechnic Institute and State University. Email: yyang8@vt.edu

J. Wang is with Microsoft Corporation. Email: jun.y.wang@gmail.com

R. Kravets is with University of Illinois at Urbana-Champaign. Email:rhk@cs.uiuc.edu

Despite extensive research on contention window control in wireless LANs, none of the current approaches can support both an arbitrary definition of fairness and efficient use of bandwidth. Most existing approaches, including AOB [7], MFS [8], Dynamic IEEE 802.11 [9], [10], IEEE 802.11e [11] and P-MAC [12], only support either uniform bandwidth allocation or weighted proportional fairness. Some approaches, including Dynamic IEEE 802.11, MFS and P-MAC, may even cause system instability under network disturbances. The only approach that tries to provide a more general definition of fairness is PFCR [13]. However, PFCR is only appropriate for a limited set of fairness definitions and may have poor channel utilization. (See Section II for details of related work.)

Due to the limitations of the existing approaches, we propose our distributed contention window adaptation algorithm, called GCA (**G**eneral **C**ontention window **A**daptation), which closely approximates optimal bandwidth allocation for competing wireless nodes in terms of efficient channel utilization and various fairness definitions. The goal of GCA is to provide a general solution for the design and analysis of dynamic contention window control algorithms in wireless LANs.

The challenges of designing fair and efficient distributed contention window control algorithms stem from the fact that a node’s bandwidth share depends on both its own contention window size and the contention window sizes of *all* competing nodes in the network. While an individual node can set its own contention window size, it has no control over the contention window sizes of other nodes. Additionally, by adjusting its own contention window size, the node directly affects other nodes’ share of the bandwidth. Without careful design, adjusting contention window sizes at different nodes may result in an unstable system. Therefore, it is non-trivial to design contention window control algorithms that are distributed, stable, efficient and fair based on any fairness definition.

Our research has three major contributions. First, we formulate, for the first time, the problem of efficient and fair bandwidth allocation based on an arbitrary definition of fairness as an optimization problem for contention window assignment. Second, to solve this optimization problem, we present the design of GCA, a fully distributed contention window control algorithm. Even though a node’s bandwidth share depends on the contention window sizes of *all* competing nodes, GCA *does not* require any global knowledge. We rigorously prove that in GCA, although a node only adjusts its own contention window size based on locally available information, the system automatically converges to a point that is very close to the solution of the optimization problem, meaning that GCA can achieve fairness for any given fairness definition while maintaining close to maximum channel utilization. Finally, we demonstrate that GCA provides a systematic scheme to generalize and evaluate related approaches by showing that we can use GCA to check the efficiency and fairness of many existing heuristic algorithms.

This paper is organized as follows. Section II discusses related work on bandwidth allocation in wired and wireless networks. Section III reveals the relationship between bandwidth allocation and contention window size. Section IV relates an arbitrary fairness definition and efficient bandwidth utilization to an optimal contention window assignment problem. Section V introduces our contention window control algorithm, GCA. Section VI shows that GCA converges to a fair bandwidth allocation point based on an arbitrary definition of fairness and Section VII shows how to control the stable point of GCA to be close to the maximum channel utilization point. Section VIII discusses guidelines for implementing GCA. In Section IX, we use GCA to analyze several existing approaches. Section X presents the evaluation of GCA using simulations. Finally, Section XI concludes our research. Notation for the entire paper can be found in Appendix A.

II. Related Work

The theoretical foundation for distributed bandwidth allocation in wired networks was established by F. Kelly, et al. [14], who describe a distributed rate control algorithm that solves the optimal bandwidth allocation problem. Since wired networks are assumed to be point-to-point and/or have high link bandwidth, the sending rate of a node is essentially its own TCP congestion window size over its own round trip time. Therefore, Kelly's rate control algorithm can be directly translated to congestion window control algorithms for TCP. Essentially, end hosts compete for bandwidth by modulating their TCP congestion window size to compete for buffer space in routers [15], [14], [16], [17], [18].

However, the shared nature and limited bandwidth of wireless links move the competition for bandwidth from router queues to channel access time so that the congestion window does not determine the transmission rate of a wireless node. Hence, research results in wired networks cannot be directly applied to wireless networks. As shown by Y. Xue et al. [19], to extend Kelly's rate control algorithm to wireless networks, neighboring nodes must frequently exchange information about each other's transmission rate, imposing heavy message overhead in the presence of frequent traffic variations.

To avoid such heavy message overhead, the wireless networking community is currently researching approaches that use distributed contention window allocation algorithms to support fair and efficient bandwidth allocation. The challenge to such approaches is that in wireless networks, the sending rate of a node is determined by the contention window sizes of *all* competing nodes. Since no node has complete control over its sending rate, Kelly's rate control algorithm cannot be directly translated to *contention* window control and new algorithms must be designed to achieve arbitrary fairness and efficient channel utilization.

Unfortunately, most of the existing approaches only focus on specific fairness definitions, either uniform bandwidth allocation or weighted proportional fairness. AOB [7], MFS [8] and the Dynamic IEEE 802.11 scheme [9], [10] only focus on uniform bandwidth allocation. IEEE 802.11e [11] and P-MAC [12] only provide weighted proportional fairness. Some of these existing algorithms, including the Dynamic IEEE 802.11 scheme, MFS and P-MAC, also create network instability when they try to achieve efficient channel utilization through dynamic contention window adaptation. This is because they require an individual node to run iterative algorithms to estimate both the contention windows used by the competing nodes and the number of competing nodes. Unfortunately, such estimation requires that all of the nodes, with or without packets for transmission, must start both the dynamic contention window adaptation and the iterative estimation algorithms simultaneously and run every step of these algorithms synchronously. Nodes with outdated knowledge of contention window sizes and the number of competing nodes, due to asynchronous algorithm executions or temporary failures, may cause these algorithms to fail.

The only existing approach that attempts to achieve a general definition of fairness through contention window adaptation is PFCR [13], which models fairness as an optimization problem of transmission rate allocation. Besides the lack of support for efficient bandwidth utilization, PFCR does not consider the fact that a node does not have complete control over its rate by simply adjusting its contention window size. Hence, the mapping between rate allocation and contention window adaptation in PFCR is only appropriate for a limited set of fairness definitions as demonstrated in Section IX-A.1.

Based on the limitations of existing approaches, we conclude that a distributed contention window control algorithm should satisfy three requirements. First, it should be flexible so that it can be configured to achieve arbitrary definitions of fairness. Second, it should utilize the channel efficiently. Third, it should asymptotically converge to an efficient and fair bandwidth allocation regardless of the initial contention window allocation so that it can maintain network stability. In the remainder of this paper, we present GCA (**G**eneral **C**ontention window **A**daptation) and show how it satisfies these requirements.

III. Bandwidth Allocation and Contention Windows

To realize fair bandwidth allocation by adapting contention window sizes, it is essential to understand the relationship between a node's bandwidth allocation and its contention window size. Since IEEE 802.11 [1] is the de facto industry standard for wireless LANs, we first briefly review IEEE 802.11 and then introduce the relationship between contention window size and bandwidth allocation in IEEE 802.11.

A. IEEE 802.11 DCF

IEEE 802.11's DCF mode, the dominating operation mode in practice, is based on Carrier Sense Multiple Access with Collision Avoidance (CSMA/CA) algorithm. Before a transmission, a node must determine whether the medium is busy or idle. If the medium remains idle for DIFS time units, the node can transmit. If the medium was initially busy or changed from idle to busy during the DIFS, the node must defer its transmission. The deferment period is determined by a random backoff timer, $BackoffTime = Random() \times aSlotTime$, where $Random()$ is a pseudo-random integer uniformly distributed over [0,W). The *contention window*, W, is an integer in the range [*minimum contention window* (W^{min}), *maximum contention window* (W^{max})]. For every idle $aSlotTime$, the backoff timer is decremented by $aSlotTime$. The timer is stopped when the medium is busy and restarted after the medium is idle. When the timer expires, the node can transmit. After a successful transmission, W is set to W^{min}. After an unsuccessful transmission, W is doubled, up to W^{max}.

B. Bandwidth Allocation vs. Contention Window Size

In an IEEE 802.11 network where all nodes carry elastic traffic and always have backlogged packets, there is only a fraction of channel time, called the *effective capacity* (C), that is used for successful packet transmissions. The rest of the channel time is consumed by the backoff process or packet collisions. Hence, Node i's bandwidth allocation, s_i, is:

$$s_i = x_i C, \tag{1}$$

where x_i is the fraction of C allocated to Node i and $\sum_{i \in \mathcal{N}} x_i = 1$. $\mathcal{N}$ is the set of all transmitting nodes.

To obtain x_i, note that Li and Battiti [20] have shown that when $W_i \gg 1$ for $\forall i \in \mathcal{N}$, $s_i/s_j \approx (L_i W_j)/(L_j W_i)$, where L_i is the transmission rate at the physical layer multiplied by the duration of a successful transmission (including the DIFS and RTS/CTS/DATA/ACK handshake) at Node i. The approximation's condition, $W_i \gg 1$ for $\forall i \in \mathcal{N}$, is true when the network has fairly low collision probability, which will be enforced by GCA (See Section (VII) for details). Based on this approximation and Equation (1),

$$x_i \approx \frac{L_i/W_i}{\sum_{k \in \mathcal{N}} L_k/W_k}. \tag{2}$$

Equations (1) and (2) essentially reveal the relationship between contention window size and bandwidth allocation. Based on this relationship, GCA can directly adjust W_i to achieve fair and efficient bandwidth allocation. If the binary exponential increase of W_i after a collision is enabled, GCA can instead adjust the minimum contention window, W_i^{min}, to achieve the same fair bandwidth allocation, as shown in Section X. The fairness is still maintained because when W_i doubles after each collision, $s_i/s_j \approx (L_i W_j^{min})/(L_j W_i^{min})$ [20]. Therefore, $x_i \approx (L_i/W_i^{min})/(\sum_{k \in \mathcal{N}} L_k/W_k^{min})$, which is the same as Equation (2).

IV. Problem Formulation

As a first step in the design of our GCA algorithm, we first formulate the fairness and efficiency requirements as an optimal bandwidth allocation problem, and then translate it to an optimal contention window assignment problem.

A. Optimal bandwidth allocation (OPT_BW)

Following [14], [21], [17], fair and efficient bandwidth allocation in wireless LANs can be modeled as an optimal bandwidth allocation problem (OPT_BW). In this formulation, each Node i is assumed to have a utility of $U_i(s_i)$ when its bandwidth allocation is s_i. For elastic traffic, $U_i(s_i)$ is an increasing, strictly concave and continuously differentiable function of s_i over the range $s_i \geq 0$ [4]. Given the maximum effective capacity C_{max}, OPT_BW is:

$$\begin{aligned} &OPT\_BW(U,C) : \max \textstyle\sum_{i \in \mathcal{N}} U_i(s_i) \\ &\qquad\qquad \text{over: } \textstyle\sum_{i \in \mathcal{N}} s_i \leq C_{max} \text{ and } s_i \geq 0 \text{ for } i \in \mathcal{N}. \end{aligned}$$

According to the Karush-Kuhn-Tucker optimality condition [22], the *unique* solution to OPT_BW is given by [14]:

$$\begin{cases} U_i'(s_i) = \mu, & \text{for } i \in \mathcal{N} \\ \sum_{i \in \mathcal{N}} s_i = C_{max}, \end{cases} \tag{3}$$

where $\mu > 0$ is the Lagrange multiplier and $U_i'(s_i)$ is the derivative of $U_i(s_i)$ with respect to s_i. Different definitions of $U_i(s_i)$ result in different solutions to OPT_BW and so achieve different definitions of fairness [21], [17] (See examples in Section VIII-C).

As shown in Section II, Kelly's rate control algorithm [14] cannot be directly applied to solve OPT_BW since a node does not have complete control over its sending rate and its rate is determined by contention window assignments at all nodes. Therefore, we must translate OPT_BW to a problem of contention window assignment, called OPT_WIN, and find a contention window adaptation algorithm to solve OPT_WIN.

B. Optimal contention window assignment (OPT_WIN)

Based on Equations (1) and (2), we can translate OPT_BW to OPT_WIN as follows:

$$\boxed{\begin{aligned} &OPT\_WIN(U,C) : \max \textstyle\sum_{i \in \mathcal{N}} U_i\Big(\frac{\frac{L_i}{W_i}}{\sum_{k \in \mathcal{N}} \frac{L_k}{W_k}} C\Big) \\ &\quad \text{over } W_i > 0 \text{ for } i \in \mathcal{N} \text{ and } C \leq C_{max}, \end{aligned}}$$

where C is related to contention window assignments. Using Equations (1), (2) and (3), the solution to OPT_WIN is:

$$\boxed{\begin{aligned} &U_i'(x_i C_{max}) = \tilde{U}_i'\left(\frac{\frac{L_i}{W_i}}{\sum_{k \in \mathcal{N}} \frac{L_k}{W_k}}\right) = \mu > 0, \forall i \in \mathcal{N} \quad (4) \\ &\qquad C = C_{max} \quad (5) \end{aligned}}$$

where $U_i(s_i) = \tilde{U}_i(\frac{s_i}{C_{max}})$ and $\tilde{U}_i'(x_i)$ is the derivative of $\tilde{U}_i(x_i)$ with respect to x_i. Similar to $U_i(s_i)$, $\tilde{U}_i(x_i)$ is an increasing, strictly concave and continuously differentiable function of x_i over $x_i \geq 0$. Essentially, Equation (4) captures various definitions of fair bandwidth allocation and Equation (5) represents the requirement for efficient channel utilization.

The solution to Equation (4) is not unique since it only defines the ratio among contention window sizes. Given and positive constant a and a contention window assignment $\mathbf{W} =$

$\{W_i : i \in \mathcal{N}\}$ that solves OPT_WIN, $a\mathbf{W} = \{aW_i : i \in \mathcal{N}\}$ is also a solution to Equation (4). Among the possible solutions to Equation (4), the effective capacity (C) can be quite different due to different contention window assignments. Therefore, it is important to identify the solution to Equation (4) that also satisfies Equation (5), essentially maximizing network channel utilization. In Section VI, we prove that GCA converges to a solution of Equation (4). In Section VII, we discuss how we control GCA's converged point to also approximately satisfy Equation (5).

V. General Contention Window Adaptation Algorithm (GCA)

Although OPT_WIN looks very similar to OPT_BW, a simple copy of Kelly's rate control algorithm to a contention window control algorithm may not work for OPT_WIN (see an example in Section IX-A.1). This is because the convergence of Kelly's rate control algorithm relies on the concavity of OPT_BW's object function, $\sum_{i\in\mathcal{N}} U_i(s_i)$, with respect to the parameters that need to be optimized (s_i). This property, however, does not hold for OPT_WIN since its object function, $\sum_{i\in\mathcal{N}} U_i(\frac{L_i/W_i}{\sum_{k\in\mathcal{N}} L_k/W_k}C)$, is not concave with respect to W_i, making it more difficult to design distributed algorithms that solve OPT_WIN. In this section, we present GCA, our distributed contention window control algorithm that achieves both fair bandwidth allocation for arbitrary definitions of fairness and high bandwidth utilization. In GCA, a Node i adapts its W_i as follows:

$$\boxed{\dot{W}_i(t) = -\alpha W_i(t)[\tilde{U}_i'(x_i) - f(\Omega)],} \tag{6}$$

where α is a positive constant factor, $\dot{W}_i(t)$ is the time derivative of W_i, $f(\cdot)$ is a function of a locally observable channel state Ω and Ω can be known by any node through monitoring the channel locally. The intuition behind the design of GCA is that when the bandwidth allocated to Node i is smaller than its fair share, the derivative of its utility function is larger than the cost function $f(\Omega)$. In such cases, the contention window size at Node i will be decreased to increase the bandwidth share of Node i. On the other hand, if Node i gets more bandwidth than its fair share, the derivative of its utility function is smaller than the cost function. This causes the contention window size of Node i to increase and hence decreases the bandwidth allocated to Node i.

Although there are various choices of Ω and $f(\cdot)$, GCA does make three assumptions about them. First, Ω must be observable by all nodes sharing the channel and all nodes must be pre-configured with the same $f(\cdot)$. Since in the networks targeted by GCA, every node can hear each other and hence see the same channel state, the first assumption is not very restrictive. Second, to guarantee that the system stabilizes at a unique point (see details in Section VI), the value of Ω must depend on the window sizes and packet lengths of all nodes. In other words, if $\mathbf{W} = \{W_i : i \in \mathcal{N}\}$ and $\mathbf{L} = \{L_i : i \in \mathcal{N}\}$, GCA requires $\Omega = \Omega(\mathbf{W}, \mathbf{L})$. Such a Ω is not hard to find since many channel states depend on $\mathbf{W}$ and $\mathbf{L}$ (e.g., packet transmission delay, average length of an idle period or collision probability). Third, $f(\Omega)$ must be continuous and be strictly increasing with respect to $\sum_{i\in\mathcal{N}} \frac{L_i}{W_i}$ inside a certain set of system states. Given the relationship between Ω and $(\mathbf{W}, \mathbf{L})$, by choosing the right form of $f(\cdot)$, $f(\Omega)$ can easily meet the third assumption. As long as the three assumptions are satisfied, GCA is not limited to any specific Ω or $f(\cdot)$. We demonstrate in Section IX that these assumptions are easy to meet and GCA can be used to model different dynamic contention window control algorithms.

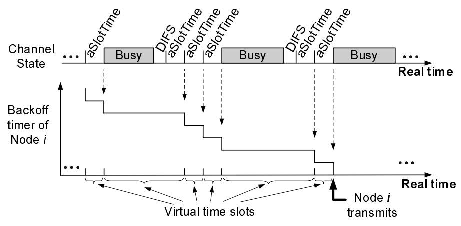


Fig. 1. Virtual slots

To implement GCA in a real system, the update algorithm in Equation (6) must be translated to its discrete counterpart. To find the appropriate time interval between updates, note that under the condition that all nodes can hear each other (typical wireless LAN environment), the state transition of a wireless LAN is a discrete-time Markov process as shown in the Bianchi model [23] of IEEE 802.11 (See Figure 1) . The time unit for this discrete-time process, called a *virtual slot*, is the time period for a backoff node to decrement its backoff timer by one. At most one packet can be transmitted in a virtual slot and a collision happens when there are multiple transmission attempts in the same virtual slot. The example shows that a virtual slot can either be exactly an $aSlotTime$ period during the idle time of the channel (e.g., the first virtual slot) or include a busy period, a DIFS and an $aSlotTime$ if the channel is busy (e.g., the second virtual slot).

Since the network state only changes in the steps of virtual slots, the effects of any contention window update cannot be seen in any smaller time unit than a virtual slot. Therefore, the update interval of GCA should not be smaller than a virtual slot. If a node updates its contention window size at the end of every backoff slot, essentially at every virtual slot, the discrete version of GCA in Equation (6) becomes:

$$W_i^{k+1} = W_i^k - \alpha W_i^k[\tilde{U}_i'(x_i^k) - f(\Omega^k)], \tag{7}$$

where Ω^k and x_i^k are the measurements of Ω and x_i at the k_{th} iteration and W_i^k is the contention window size at the k_{th} iteration. If updates are performed for each packet transmission so that the average number of virtual slots between each update is $\frac{W_i}{2}$, the discrete version of GCA becomes:

$$W_i^{k+1} = W_i^k - 0.5\alpha(W_i^k)^2[\tilde{U}_i'(x_i^k) - f(\Omega^k)]. \tag{8}$$

GCA is simple and only requires locally observable information about network state. Despite this simplicity, GCA converges to the solution of OPT_WIN (see Section VI) and can also achieve efficient channel utilization (see Section VII).

VI. Convergence and Fairness of GCA

In this section, we prove that GCA, as expressed in Equation (6), asymptotically converges to a unique point that is a solution to Equation (4) given the three assumptions about $f(\Omega)$. Our proof of convergence is different from the one for Kelly's rate control algorithm since the object function in OPT_WIN is not concave with respect to W_i. Due space limitations, part of the proof is presented in our technical report [24].

Theorem 1: Denote the solution set of Equation (4) as Γ:

$$\Gamma = \Big\{\mathbf{W} : \tilde{U}_i'(\frac{L_i/W_i}{\sum_{k\in\mathcal{N}}\frac{L_k}{W_k}}) = \tilde{U}_j'(\frac{L_j/W_j}{\sum_{k\in\mathcal{N}}\frac{L_k}{W_k}}), \forall i,j \in \mathcal{N}\Big\}. \tag{9}$$

If $\Omega = \Omega(\mathbf{W},\mathbf{L})$ and $f(\Omega)$ is a continuous function that is strictly increasing with respect to $\sum_{i\in\mathcal{N}}\frac{L_i}{W_i}$ inside Γ, then starting from any initial state of $\mathbf{W}$, GCA converges to a unique point $\widehat{\mathbf{W}} \in \Gamma$.

Proof: The proof consists of three steps. At step one, for notation simplicity, we translate GCA in Equation (6) to an equivalent algorithm of $Z_i = 1/W_i$, called GCA-Z. We also translate Γ to its corresponding set R of $\mathbf{Z}$. At step two, based on Lemmas 1 to 3 in [24], we prove that GCA-Z asymptotically converges to a unique equilibrium point $\widehat{\mathbf{Z}} \in R$. In step three, we conclude that GCA asymptotically converges to a unique equilibrium point $\widehat{\mathbf{W}} \in \Gamma$.

Step 1: From $Z_i = 1/W_i$, we have $\dot{Z}_i = -\dot{W}_i/(W_i)^2 = -Z_i^2\dot{W}_i$. By replacing x_i in Equation (6) with W_i based on Equation (2) and then replacing W_i and $\dot{W}_i$ with Z_i and $\dot{Z}_i$, GCA is translated to GCA-Z as follows:

$$\dot{Z}_i = \alpha Z_i\Big[\tilde{U}_i'(\frac{Z_iL_i}{\sum_{k\in\mathcal{N}}Z_kL_k}) - f(\Omega)\Big]. \tag{10}$$

The assumptions about $f(\Omega)$ in this theorem are equivalent to $\Omega = \Omega(\mathbf{Z},\mathbf{L})$ and $f(\Omega)$ is continuous and strictly increasing with respect to $\sum_{i\in\mathcal{N}}Z_iL_i$ in R. Γ also translates to R:

$$R = \Big\{\mathbf{Z} : \tilde{U}_i'(\frac{Z_iL_i}{\sum_{k\in\mathcal{N}}Z_kL_k}) = \tilde{U}_j'(\frac{Z_jL_j}{\sum_{k\in\mathcal{N}}Z_kL_k}), \forall i,j \in \mathcal{N}\Big\}. \tag{11}$$

Step 2: According to Lemma 1 in [24], starting from any initial state, GCA-Z asymptotically converges to R and R is an invariant set. According to Lemma 3 in [24], GCA-Z has a unique equilibrium point $\widehat{\mathbf{Z}}$ in R and starting from any point in R, GCA-Z converges to $\widehat{\mathbf{Z}}$. According to Lemma 4 in [24], this convergence set R can actually be extended to a larger set $N(R)\bigcup\hat{\mathbf{Z}}$, where $N(R)$ is a neighborhood of set $(R-\hat{\mathbf{Z}})$. Note that Lemma 1 in [24] shows that starting from any $\mathbf{Z}$, GCA-Z asymptotically converges to R. Hence, any trajectory that converges to $(R-\hat{\mathbf{Z}})$ must first converge to $N(R)$ in finite time and then when the trajectory is inside $N(R)$, it converges to $\hat{\mathbf{Z}}$ as $t \to \infty$. Therefore, starting from any initial $\mathbf{Z}$, GCA-Z converges to the unique fix point $\widehat{\mathbf{Z}}$ as $t \to \infty$.

Step 3: Based on the equivalence between GCA and GCA-Z, starting from any initial state of $\mathbf{W}$, GCA converges to a unique point $\widehat{\mathbf{W}} \in \Gamma$ that solves Equation (4). ■

Theorem 1 shows that GCA converges to a unique point that solves Equation (4). The next section shows that GCA satisfies Equation (5) by achieving high channel utilization.

VII. Channel Utilization of GCA

Maximizing effective capacity (C) is essentially a problem of choosing the right $f(\cdot)$ and Ω. This is because, as shown by Equation (4), the choice of utility functions only defines fairness since it only determines the ratios of W_i. Multiple assignments of $\mathbf{W}$ may satisfy the same ratio condition and solve Equation (4) with very different effective capacity. If $\mathbf{W}$ is too large, channel bandwidth is not fully utilized since idle periods are too long. If $\mathbf{W}$ is too small, collisions increase, which also results in inefficient use of bandwidth. While the conditions about $f(\cdot)$ and Ω in Theorem 1 ensure that GCA converges to a unique fixed point that is a solution to Equation (4), $f(\cdot)$ and Ω must also be carefully chosen so that GCA's stable point also approximates maximum effective capacity, essentially satisfying Equation (5) as well.

A. Optimal Stable Point

To choose $f(\Omega)$ to stabilize the system at a point that maximizes effective capacity, we need to identify the optimal stable point of GCA that satisfies both Equations (4) and (5). In this section, we analyze the property of the optimal stable point of GCA and show that at this point, the sum of the reciprocals of all W_i's, denoted ω, is quasi-constant regardless of the number of competing nodes and therefore can be pre-calculated. Using this property, we can design $f(\Omega)$ to ensure that GCA converges around the optimal stable point.

To identify the property of ω at the optimal stable point, note that the ratios between contention window sizes are determined by utility functions (see Equation (4)). Hence, assuming there are n competing nodes belonging to m classes $c_1, c_2, \ldots c_m$ and nodes in the same class have the same utility function,

$$\varphi_i = \varphi_j = \varphi_{c_k}, \forall i,j \in c_k, \tag{12}$$

$$\text{where } \varphi_i = \frac{1/W_i}{\sum_{j=1}^{n}1/W_j} = \frac{1/W_i}{\omega}, \forall 1 \le i \le n, \tag{13}$$

and φ_{c_k} is the value of φ shared by all the nodes in class c_k. Denoting the fraction of nodes in each class is $\beta_1, \beta_2, \ldots \beta_m$, $\sum_{i=1}^{n}\varphi_i = \sum_{k=1}^{m}n\beta_k\varphi_{c_k} = 1$.

To maximize effective capacity, the average time between each successful transmission, denoted F, must be minimized. F includes two types of virtual slots, idle slots and slots with collisions. Given the probability that a slot is an idle slot, P_I, and the probability that a slot includes a collision, P_c, the average number of virtual slots in F is $\frac{1}{1-P_c-P_I} - 1$. On average, a $\frac{P_I}{P_I+P_c}$ fraction of these slots is idle and a $\frac{P_c}{P_I+P_c}$ fraction includes collisions. Therefore,

$$F = (\textit{aSlotTime} \cdot P_I + T_cP_c)/(1-P_c-P_I), \tag{14}$$

where $aSlotTime$ is the duration of an idle virtual slot and T_c is the duration of a virtual slot with a collision.

Based on the Bianchi model [23], the probability that Node i transmits in a virtual slot is $1/(W_i/2+1)$ and P_I equals the probability that no node transmits in a slot. Therefore, using Equations (12) and (13),

$$P_I = \prod_{i=1}^{n}(1-\frac{1}{W_i/2+1}) = \frac{1}{\prod_{k=1}^{m}(1+2\omega\varphi_{c_k})^{n\beta_k}}. \tag{15}$$

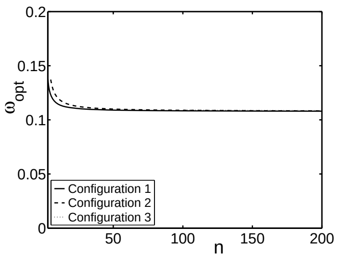


Fig. 2. ω_{opt} in three network configurations. Configuration 1 has 1 class, Configuration 2 has 2 classes with $\varphi_{c_1} : \varphi_{c_2} = 1 : 5$ and $\beta_1 : \beta_2 = 0.5 : 0.5$ and Configuration 3 has 4 classes with $\varphi_{c_1} : \varphi_{c_2} : \varphi_{c_3} : \varphi_{c_4} = 1 : 5 : 10 : 20$ and $\beta_1 : \beta_2 : \beta_3 : \beta_4 = 0.5 : 0.3 : 0.15 : 0.05$

Since a collision happens when more than one node transmits in a slot,

$$\begin{aligned} P_c &= 1 - P_I - \textstyle\sum_{i=1}^{n} \frac{1}{W_i/2+1} \prod_{j=1, j\neq i}^{n} (1 - \frac{1}{W_j/2+1}) \\ &= 1 - (1 + 2\omega) P_I. \end{aligned} \tag{16}$$

Using Equations (15) and (16), Equation (14) becomes

$$F(\omega) = \frac{1}{2\omega}\Big[T_c \prod_{k=1}^{m} (1+2\omega\varphi_{c_k})^{n\beta_k} - T_c(1 + 2\omega) + \textit{aSlotTime}\Big].$$

Since the ω that minimizes F, denoted ω_{opt}, satisfies $F'(\omega_{opt}) = 0$, by setting $F'(\omega) = 0$, ω_{opt} can be obtained by solving:

$$\Big[1 - \sum_{k=1}^{m} (\frac{2\omega_{opt} n\beta_k\varphi_{c_k}}{1 + 2\omega_{opt}\varphi_{c_k}})\Big] \prod_{k=1}^{m} (1+2\omega_{opt}\varphi_{c_k})^{n\beta_k} = 1 - \frac{\textit{aSlotTime}}{T_c}. \tag{17}$$

Since $\sum_{k=1}^{m} n\beta_k\varphi_{c_k} = 1$, when $n \to \infty$, Equation (17) becomes:

$$(1 - 2\omega_{opt}) e^{2\omega_{opt}} = 1 - (\textit{ aSlotTime}/T_c). \tag{18}$$

Solving this equation gives the lower bound of ω_{opt}. Figure 2 depicts how ω_{opt} changes as n increases. It shows that ω_{opt} is a quasi-constant for a large n. The differences between different configurations of classes are hard to distinguish. Therefore, we can pre-calculate this quasi-constant and pre-configure GCA to converge around this value by proper design of $f(\Omega)$.

B. Choice of $f(\Omega)$

Since ω_{opt} is a quasi-constant, we can control the system to stabilize near ω_{opt}, which will result in an effective capacity that is close to the maximum value, essentially ensuring that $C \approx C_{max}$. This means that GCA can stabilize at a contention window allocation that is very close to the solution of OPT_WIN, essentially solving OPT_WIN practically.

To achieve this, $f(\Omega)$ should be a large negative value when the ω of the system is much larger than ω_{opt}. This large negative value of $f(\Omega)$ forces the system to increase its $\mathbf{W}$ (see Equation (6)), driving its ω back to ω_{opt}. Similarly, when the ω of the system is much smaller than ω_{opt}, $f(\Omega)$ should be a large positive value to drag the system back to ω_{opt}. $f(\Omega)$ that satisfies these conditions as well as the conditions in Theorem 1 is very easy to design. Examples of $f(\Omega)$ are presented in Section IX. Our simulation results in Section X verify the effectiveness of this approach.

VIII. Implementation Considerations

In previous sections, we introduced GCA and analyzed its fairness and efficiency. In this section, we address three implementation issues of GCA: estimation of Ω and x_i in Equation (6) and choice of utility functions.

A. Estimation of Ω

In our discussion of GCA, we assumed that $f(\Omega)$ was determined by the instantaneous contention window allocation $\mathbf{Z}$, i.e., $f(\Omega) = f(\mathbf{Z}, \mathbf{L})$. However, the effect of contention window size on the channel state Ω is stochastic. Therefore, measurements of Ω need to be averaged over time to eliminate randomness. Such estimation, however, introduces extra delay in the feedback for GCA control and may potentially affect the stability of the system. In this section, we design an estimation algorithm for Ω, which can maintain the stability of GCA.

First, we choose an Ω that is related to $\omega = \sum_{i\in\mathcal{N}} 1/W_i$ through a known function $\Omega = h(\omega)$. This is not an unrealistic assumption since we need to design $f(\Omega)$ to ensure that GGA's convergent point is not far from ω_{opt}, which requires knowledge of the relationship between ω and Ω (See Section IX-C for an example of function $h(\cdot)$). Given this assumption, we estimate the time-average of Ω, Ω_{av}, as follows:

$$\omega^k = h^{-1}(\Omega^k), \tag{19}$$

$$\omega_{av}^k = (1 - \frac{1}{\epsilon})\omega_{av}^{k-1} + \frac{1}{\epsilon}\omega^k, \tag{20}$$

$$\Omega_{av}^k = h(\omega_{av}^k), \tag{21}$$

where k represents the k_{th} iteration of the averaging algorithm. First, Equation (19) calculates the value of ω at the k_{th} iteration through the observed value of Ω (Ω^k) at this iteration. Then, Equation (20) calculates ω's moving average, ω_{av}. Next, Equation (21) produces the time-average state Ω_{av} from ω_{av}. Finally, GCA uses Ω_{av} to adapt the contention window size:

$$\dot{W}_i(t) = -\alpha W_i(t)[\tilde{U}_i'(x_i) - f(\Omega_{av})]. \tag{22}$$

To prove the convergence of this algorithm, note that Theorem 1 is affected by the estimation algorithm since even though $f(\Omega)$ may be strictly increasing with respect to $\theta = \sum_{i\in\mathcal{N}} L_i/W_i$ in Γ, $f(\Omega_{av})$ may not be strictly increasing with respect to θ. However, by approximating the discrete estimation algorithms in Equations (19), (20) and (21) to their continuous time counterpart, Lemma 5 in [24] shows that GCA-Z still converges to the unique equilibrium point $\widehat{\mathbf{Z}}$. Therefore, due to the equivalence of GCA and GCA-Z, we can conclude that under this estimation algorithm of Ω, GCA still asymptotically converges to a unique equilibrium point that solves Equation (4).

B. Estimation of x_i

Since $x_i = s_i/C$, a node can calculate its x_i by estimating s_i and C. While s_i can be easily obtained from Node i's own transmission rate, C needs to be observed from the channel.

Since a collision is seen as a busy period with an undecodable message, a node can obtain the channel time that is used for effective data transmission, T_e, by subtracting the channel collision time and channel idle time. Denoting the raw capacity of the physical channel as S, $C = T_e S$.

A drawback of the above approach is that a node must try to decode every transmission on the channel, which imposes a heavy computation overhead. Hence, we use a second approach, which calculates x_i based on an approximated relationship between x_i and the frequencies and durations of busy virtual slots.

First, given the probability that Node i successfully transmits in a virtual slot is P_i, the average network throughput in this virtual slot is $ST_s \sum_{i\in\mathcal{N}} P_i$, where $\sum_{i\in\mathcal{N}} P_i$ is the probability that this virtual slot has a successful transmission and T_s is the average length of a virtual slot with a successful transmission. In addition, since $P_i L_i$ is Node i's average throughput in a virtual slot,

$$x_i = (P_i L_i)/(ST_s \sum_{i\in\mathcal{N}} P_i). \tag{23}$$

To derive $T_s \sum_{i\in\mathcal{N}} P_i$ in Equation (23), note that a virtual slot is busy due to either a successful transmission or a collision. Hence, the duration of a busy virtual slot, T_b, is:

$$T_b = \big(T_s \sum_{i\in\mathcal{N}} P_i\big)/P_b + (T_c P_c/P_b) \approx \big(T_s \sum_{i\in\mathcal{N}} P_i\big)/P_b, \tag{24}$$

where P_b is the probability that a virtual slot is a busy slot and T_c is the duration of a virtual slot with a collision. The approximation in Equation (24) holds because when RTS/CTS exchanges are used, collisions usually happen between RTS packets. Therefore, T_c is much smaller than T_s. In addition, since GCA controls the system's stable point to avoid congestion, P_c is also smaller than $\sum_{i\in\mathcal{N}} P_i$. Combining Equations (24) and (23),

$$x_i \approx (P_i L_i)/(ST_b P_b). \tag{25}$$

To derive P_b in Equation (25), note that

$$P_b = 1 - P_I = 1/(I+1), \tag{26}$$

where I is the average number of idle virtual slots between two consecutive busy virtual slots. To derive P_i in Equation (25), note that P_i equals the probability that Node i is the only node that transmits in a virtual slot. Hence,

$$P_i = \frac{1}{W_i/2+1} \prod_{j\in\mathcal{N}, j\neq i} (1 - \frac{1}{W_j/2+1}). \tag{27}$$

Combining Equations (15), (26) and (27), P_i becomes

$$P_i = \frac{2}{W_i}(1 - \frac{1}{I+1}). \tag{28}$$

Replacing P_i and P_b in Equation (25) using Equations (26) and (28), we finally get the estimation algorithm for x_i based on the observations of T_b and I:

$$\boxed{x_i \approx (2L_i I)/(W_i T_b S).} \tag{29}$$

Both T_b and I are currently monitored by the IEEE 802.11 MAC layer. Hence, there is no overhead associated with calculating x_i using Equation (29).

C. Choice of Utility Functions

Depending on the system goal, GCA supports a large range of utility functions that define a variety of fairness definitions. These utility functions can be either pre-configured into nodes or selected by nodes at run time according to application requirements. In this section, we briefly review several common utility functions and their corresponding fairness definitions. How to enforce a node to use a certain utility function is beyond the scope of this paper.

1) Strict Priority: For a system that needs to achieve strict priority (i.e., the highest-priority nodes get all of the bandwidth), we can use a weighted linear utility function $\tilde{U}_i(x) = \rho_i x_i$, where ρ_i is the priority-based weight. The corresponding update algorithm is $\dot{W}_i = -\alpha W_i[\rho_i - f(\Omega)]$.

Note that this utility function does not satisfy the stability conditions since $\tilde{U}(\cdot)$ is not strictly concave. Therefore, our update algorithm will never converge to a certain $\mathbf{W}$. However, since the nodes with the highest weight essentially drive $f(\Omega)$ to be equal to $\max\{\rho_i, i \in \mathcal{N}\}$, the other competing nodes infinitely increase their W_i's. Therefore, the nodes with the highest weight quickly obtain all of the bandwidth and our update algorithm achieves this strict priority between nodes.

2) Weighted Proportional Fairness: Some systems aim to achieve weighted proportional fairness [14] (i.e., bandwidth allocations satisfy $x_i/\rho_i = x_i/\rho_j, \forall i, j \in \mathcal{N}$, where ρ_i is the weight of Node i). The utility function for such a system is a weighted log function $\tilde{U}_i(x_i) = \rho_i \log x_i$. Our update algorithm for this system is: $\dot{W}_i = -\alpha W_i[(\rho_i/x_i) - f(\Omega)]$.

3) Minimum Potential Delay: If the policy of the system is to minimize the total delay of file transfers, the utility function can be expressed as $\tilde{U}_i(x) = -\rho_i/x_i$, where ρ_i is the size of the file. Our update algorithm for this system is $\dot{W}_i = -\alpha W_i[\frac{\rho_i}{x_i^2} - f(\Omega)]$.

4) Mixed Utility: It is also possible that different nodes have different goals and hence different utilities. In such situations, each node simply updates its contention window according to its own utility function. The system automatically converges to a stable point where the aggregated utility of all competing nodes is maximized. In general, the variety of choices of the utility functions gives GCA the flexibility to be used in systems that have different fairness policies.

IX. Case Study

In the previous sections, we analyzed the optimality, stability and optimal stable point of GCA. Since GCA is a general algorithm for contention window control, these analyses can be used as a powerful tool to examine existing approaches and to design new algorithms. In this section, we present a brief analysis of four examples. The first two examples show how to use GCA to check the fairness of existing algorithms. The third example shows how to use GCA to analyze the stability and efficiency of an existing algorithm. The final example shows how to use GCA to design a new algorithm.

A. Fairness Analysis

To examine an algorithm's fairness property, it is necessary to find out the contention window allocation at the stable point of this algorithm and use this contention window allocation to check the type of fairness that this algorithm achieves.

1) Case 1: PFCR: In [13], it is proposed to directly translate a variant of Kelly's rate adaptation algorithm $\dot{s}_i = \alpha - (\beta P_c)/\tilde{U}_i'(s_i)$ to a contention window control algorithm:

$$\dot{Z}_i = \alpha - [\beta P_c/\tilde{U}_i'(Z_i)] \tag{30}$$

to solve OPT_BW (α and β are positive constants). PFCR is a special case of the algorithm with a weighted log utility function. Assuming uniform packet size, it can be shown that this algorithm cannot achieve an arbitrary fairness definition.

At the equilibrium point of the algorithm, $\dot{Z}_i = 0$, which results in: $\tilde{U}_i'(Z_i) = \tilde{U}_j'(Z_j) = \beta P_c/\alpha, \forall i, j \in \mathcal{N}$. By replacing Z_i with $1/W_i$,

$$\tilde{U}_i'(1/W_i) = \tilde{U}_j'(1/W_j) = \beta P_c/\alpha, \forall i, j \in \mathcal{N}, \tag{31}$$

which does not satisfy the optimality condition for OPT_WIN in Equation (4). Although, for log utility functions (e.g., PFCR), when Equation (31) is satisfied, the fairness condition in Equation (4) is also satisfied so that proportional fairness can be achieved. However, such a property does not hold for many utility functions (e.g., $\tilde{U}_i(x_i) = \rho_i x_i + log x_i$). Hence, this algorithm cannot achieve arbitrary fairness.

Another problem of this algorithm is the inaccuracy of the measurement method for P_c. Essentially, P_c should be the collision probability of all transmission attempts on the channel. However, P_c is approximated by the measurement of the collision probability of an individual node's transmission attempts in [13]. Based on Equations (15) and (16):

$$\begin{aligned} P_c &= 1 - \textstyle\prod_{i=1}^{n}(1 - \frac{1}{W_i/2+1}) \\ &\quad - \textstyle\sum_{i=1}^{n} \frac{1}{W_i/2+1} \prod_{j=1, j\neq i}^{n}(1 - \frac{1}{W_j/2+1}) \end{aligned} \tag{32}$$

$$\begin{aligned} &= 1 - \frac{1}{1-\frac{1}{W_i/2+1}} \textstyle\prod_{j=1}^{n}(1 - \frac{1}{W_j/2+1}) \\ &\quad - \textstyle\sum_{j=1, j\neq i}^{n} \frac{\frac{1}{W_j/2+1}}{1-\frac{1}{W_j/2+1}} \prod_{j=1}^{n}(1 - \frac{1}{W_j/2+1}). \end{aligned} \tag{33}$$

Node i's collision probability, ϕ_i, is:

$$\begin{aligned} \phi_i &= 1 - \textstyle\prod_{j=1, j\neq i}(1 - \frac{1}{W_j/2+1}) \\ &= 1 - \frac{1}{1-\frac{1}{W_i/2+1}} \textstyle\prod_{j=1}^{n}(1 - \frac{1}{W_j/2+1}) \end{aligned} \tag{34}$$

Equations (33) and (34) show that ϕ_i can approximate P_c when $\sum_{j=1, j\neq i}^{n} \frac{\frac{1}{W_j/2+1}}{1-\frac{1}{W_j/2+1}} \prod_{j=1}^{n}(1 - \frac{1}{W_j/2+1})$ in Equation (33) is very small. In other cases, ϕ_i is different from P_c and varies from node to node. These cases violate the assumption that the cost function should be the same for all nodes. Hence, the fairness achieved by PFCR can be severely degraded.

2) Case 2: AOB with service differentiation: In [7], AOB (Asymptotically Optimal Backoff Algorithm) is proposed to dynamically adjust contention windows to achieve maximum bandwidth utilization. In AOB, at every packet transmission, Node i sets its contention window size as:

$$Z_i^{k+1} = 0.5\Big[1 - \min\big(1, \frac{1}{(I^k+1)2\omega_{opt}}\big)^{m_k}\Big], \tag{35}$$

where I is the average number of idle virtual slots between two busy virtual slots, ω_{opt} is pre-computed and m_k is the number of transmission attempts for the current packet. Although AOB is mainly proposed to provide efficient channel utilization, an extension to AOB, named AOB-DIF [7], supports service differentiation using the following adaptation algorithm:

$$Z_i^{k+1} = 0.5\Big[1 - \min\big(1, \frac{1}{(I^k+1)2\omega_{opt}}\big)^{m_k \rho_i}\Big], \tag{36}$$

where the weight ρ_i is introduced to provide different services to different nodes. In this section, we examine the type of fairness achieved by AOB-DIF.

At the stable point of AOB-DIF, $Z_i^{k+1} - Z_i^k = 0$. Combining this with Equation (36),

$$Z_i = 0.5\Big[1 - \big(\frac{1}{(I+1)2\omega_{opt}}\big)^{m_i \rho_i}\Big]. \tag{37}$$

Given Node i's collision probability, ϕ_i, m_i is:

$$m_i = 1/(1-\phi_i), \tag{38}$$

Based on Equation (15),

$$\phi_i = 1 - \prod_{j=1, j\neq i}^{n}(1 - \frac{1}{W_j/2+1}) = 1 - (1 + 2/W_i)P_I. \tag{39}$$

Since I is the average number of idle slots between busy slots,

$$1/(I+1) = 1 - P_I. \tag{40}$$

Therefore, combining Equations (38), (39) and (40),

$$m_i = \frac{1}{(1+2/W_i)P_I} = \frac{I+1}{(1+2/W_i)I}. \tag{41}$$

Combining Equations (37) and (41), we get:

$$Z_i = 0.5\Big[1 - \big(\frac{1}{(I+1)2\omega_{opt}}\big)^{\frac{I+1}{(1+2/W_i)I}\rho_i}\Big] = 0.5[1 - \beta^{\frac{\rho_i}{1+2/W_i}}], \tag{42}$$

where $\beta = (\frac{1}{(I+1)2\omega_{opt}})^{\frac{I+1}{I}}$. Hence the bandwidth allocation achieved by AOB-DIF can be expressed a

$$s_i/s_j = Z_i/Z_j = (1 - \beta^{\frac{\rho_i}{1+2/W_i}})\big/(1 - \beta^{\frac{\rho_j}{1+2/W_j}}), \forall i, j \in \mathcal{N}. \tag{43}$$

Since AOB controls the congestion level of the network by adapting the contention window sizes, as the number of nodes in the network increases, the W_i of each node increases as well. Therefore, for a large number of nodes, $2/W_i \ll 1$. Hence, the fairness achieved by AOB in large networks is:

$$s_i/s_j \approx (1 - \beta^{\rho_i})\big/(1 - \beta^{\rho_j}), \forall i, j \in \mathcal{N}. \tag{44}$$

Obviously, the fairness achieved by AOB is not arbitrary.

B. Stability and Efficiency Analysis

In this section, we examine AOB's stability and efficiency in a network with a uniform priority and packet size. The following analysis shows that AOB is a special form of GCA and achieves high channel utilization.

Using the utility function $\tilde{U}(x) = x - 0.5x^2$, which is strictly increasing and concave in the range $[0, 1]$, Equation

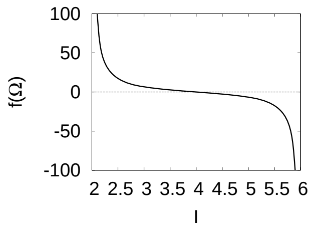


Fig. 3. $f(\Omega) = f(I) = 10/(I-2) + 10/(I-6)$

(35) becomes:

$$\begin{aligned} Z_i^{k+1} - Z_i^k &= \tfrac{1}{2(I+1)}[\tilde{U}'(2Z_i^k(I_k+1)) - \\ &\min(1, \tfrac{1}{(I_k+1)^{m_k-1}2\omega_{opt}^{m_k}} - I)]. \end{aligned} \tag{45}$$

By approximating $\sum_{k\in\mathcal{N}} Z_k \approx \frac{1}{2(I+1)}$, the discrete form of GCA-Z in Equation (10) becomes:

$$Z_i^{k+1} - Z_i^k = 0.5\alpha[\tilde{U}_i'(2Z_i^k(I+1)) - f(\Omega)]. \tag{46}$$

Equations (45) and (46) show that AOB is a special case of GCA with:

$$f(\Omega) = \min\Big(1, \frac{1}{(I+1)^{m-1}2\omega_{opt}^m} - I\Big). \tag{47}$$

For $f(\Omega)$ to satisfy the convergence condition of GCA, $f(\Omega)$ must be strictly increasing with respect to $\theta = \sum_{i\in\mathcal{N}} L_i/W_i$ in the invariant set Γ. In addition, Lemma 1 in [24] shows that inside Γ, $\tilde{U}_i'(\frac{Z_iL_i}{\sum_{k\in\mathcal{N}} Z_kL_k})$ remains a constant. Denoting the constant as $\hat{\gamma}$, $Z_i = 1/W_i = [\tilde{U}_i'^{-1}(\hat{\gamma})\theta]/L_i$. Combining this with Equations (15), (40) and (47) , $f(\Omega)$ can be transformed to a function of θ, whose derivative can be shown to be larger than 0. Therefore, $f(\Omega)$ in AOB satisfies GCA's convergence condition and hence AOB is a stable algorithm that converges to a unique point.

To understand the channel utilization of AOB, note that due to the uniform priority and packet size, each node should have the same contention window size at AOB's stable point, indicating $Z_i = \omega/n$. Since at the stable point, $Z_i^{k+1} - Z_i^k = 0$, based on Equations (35) and (40),

$$\frac{2\omega}{n} = 1 - (\frac{1}{2(I+1)\omega_{opt}})^m, \text{where } \frac{1}{I+1} = 1 - \prod_{i=1}^{n} \frac{1}{1+\frac{2\omega}{n}}. \tag{48}$$

By setting $n = 1$ and $n \to \infty$, we obtain the bounds of ω as $[\omega_1, \omega_2]$, where:

$$\begin{cases} 1 - 2\omega_1 &= (\frac{\omega_1}{(1+2\omega_1)\omega_{opt}})^m, \\ (\frac{1-e^{-2\omega_2}}{2\omega_{opt}})^m &= 1. \end{cases} \tag{49}$$

Essentially, AOB bounds the ω of the system inside a range that includes ω_{opt}, which explains why AOB can almost achieve maximum channel utilization.

C. New Algorithm Design

To show how to design a special case of GCA, we assume any utility function that is strictly increasing and concave and that the observed channel state Ω is I. I can be obtained by a node through monitoring the channel state and the time average of I can be calculated according to the discussion in Section VIII-A. Based on Equations (15) and (40), for a large n, $I = h(\omega) \approx \frac{1}{e^{2\omega}-1}$. Hence, $f(\Omega)$ is defined as:

$$f(\Omega) = f(I) = \lambda/(I - I_{min}) + \lambda/(I - I_{max}),$$

where $I_{min} < \frac{1}{e^{2\omega_{opt}}-1} < I_{max}$ and the range of $[I_{min}, I_{max}]$ is small. Figure 3 shows the shape of $f(\Omega)$ with $I_{min} = 2$ and $I_{max} = 6$. According to Section VII-B, this function $f(\Omega)$ ensures that at the converged point, ω is near ω_{opt}, so that the system utilization is close to the maximum. The performance of this algorithm is evaluated in Section X.

X. Evaluation

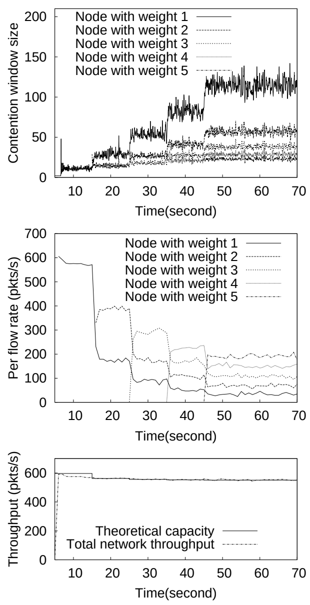
Our evaluation focuses on four aspects of GCA: (1) support for different definitions of fairness, (2) maintaining fairness, (3) maintaining efficiency and (4) convergence speed under drastic load changes. Although GCA supports various fairness definitions, we only present the performance of GCA for strict priority and proportional fairness in this paper. These two types of fairness represent opposite extremes, where strict priority requires that all bandwidth is allocated to the node with the highest priority while proportional fairness requires that every node gets a fraction of bandwidth proportional to its priority. To demonstrate the correctness of GCA, we use the newly designed special case of GCA discussed in Section IX-C. Except in Section X-D, we set $\alpha = 6e^{-4}$ (Equation (8)) and $1/\epsilon = 1e^{-2}$ (Equation (20)). Two variants of GCA are examined. GCA-EXP adjusts W^{min}, where W is exponentially increased after a collision. GCA-DIRECT directly adjusts W, without exponential increase of W after a collision. We compare both GCA-DIRECT and GCA-EXP's performances with IEEE 802.11e [11], AOB [7] and PFCR and PFCR's extension to other utility functions [13]. The theoretical maximum network capacity is calculated using MATLAB as a baseline for comparison. NS2 simulator [25] is used for simulation and the channel bandwidth is 11Mbps.

A. System Evolution

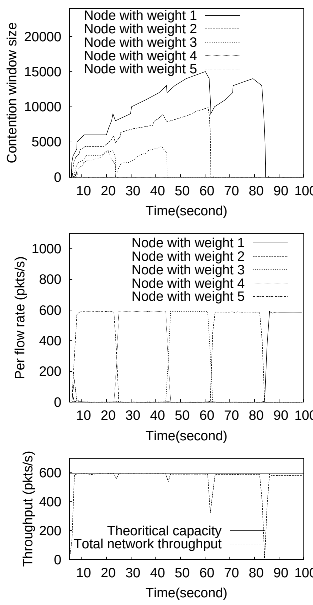
In this section, we illustrate how GCA adapts contention window size to support fair and efficient bandwidth allocation. Only GCA-DIRECT is shown here. The behavior of GCA-EXP is the same and is omitted due to space limitations.

To examine the behavior of GCA for proportional fairness, in a 70-second simulation, five competing nodes start transmitting 512B packets at time 5s, 15s, 25s, 35s and 45s respectively and use weighted log utility functions with weights 1 to 5. Figure 4(a) shows that as the number of competing nodes increases, GCA quickly increases the contention window sizes of all competing nodes to prevent congestion. Therefore, GCA operates near its optimal point and maintains a steady throughput regardless of the number of competing nodes. In addition, GCA keeps the ratio between contention window sizes to provide each node its weighted fair share of bandwidth.

To examine the behavior of GCA for strict priority, in a 100s simulation with five competing nodes, each node starts transmitting a file at 5s and each is equipped with a linear utility function with weight ranging from 1 to 5. Figure 4(b)

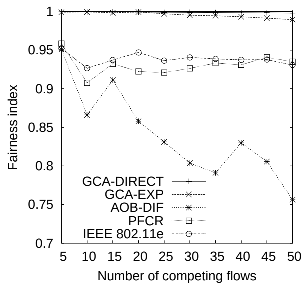


(a) Weighted proportional fair

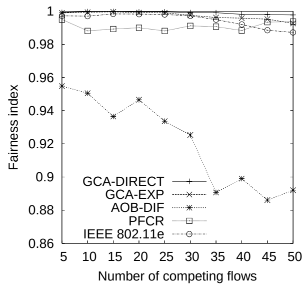


(b) Strict Priority

Fig. 4. Evolution of GCA.

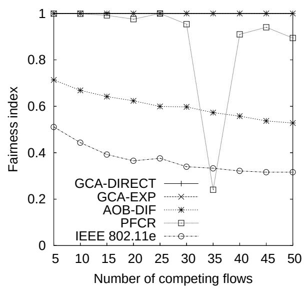


(a) Different packet size

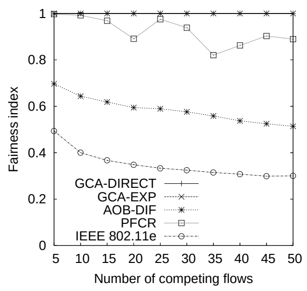


(b) Same packet size

Fig. 5. Fairness index for weighted proportional fairness



(a) Different packet size



(b) Same packet size

Fig. 6. Fairness index of strict priority

shows that at the beginning of the simulation, the node with weight 5 has a very small contention window size while the other nodes with lower weights keep on increasing their contention window sizes. Therefore, the node with weight 5 soon obtains all channel bandwidth. After the node with weight 5 finishes its transmission, the contention window size of the node with weight 4 drops down and grabs the channel. Then after the node with weight 4 finishes, the node with weight 3 gets the channel. The process goes on until only the node with the lowest priority is left, demonstrating GCA's ability in achieving strict priority between competing nodes using weighted linear utility functions.

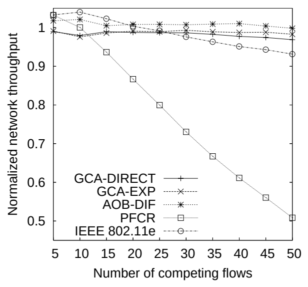
B. Fairness

Next, we compare GCA, AOB, PFCR and IEEE 802.11e's ability to achieve weighted proportional fairness and strict priority. The purpose of this comparison is to show the generality of GCA since it can support both types of fairness while other approaches lack GCA's flexibility. For weighted proportional fairness, we use Jain's fairness index [26], a common measure of proportional fairness. Given n competing nodes, Jain's fairness index is $\Psi = \left(\sum_{i=1}^{n} s_i/r_i\right)^2 \Big/ \left[n \sum_{i=1}^{n} (s_i/r_i)^2\right]$, where r_i is the share of bandwidth proportional to Node i's weight and s_i is Node i's achieved bandwidth. The fairness index is a real value between 0 and 1 with values closer to 1 indicating better proportional fairness. For strict priority, since all of the bandwidth should be allocated to the flow with the highest priority, we define the fairness index for strict priority as $\Psi = s_k/(\sum_{i=1}^{n} s_i)$, where flow k has the highest priority. For perfect strict priority, $\Psi = 1$.

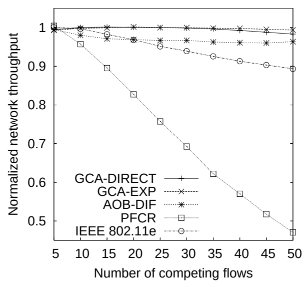
1) Weighted proportional fairness: In this set of simulations, 5 to 50 competing nodes with weights from 1 to 5 start in the first 10s. Both GCA and PFCR use weighted log utility functions. AOB-DIF is used to provide differentiated services for nodes with different weights (Section IX-A.2). For IEEE 802.11e, the minimum contention window sizes of nodes with weight 5, 4, 3, 2, and 1 are 30, 37, 50, 75 and 150, respectively. These contention window sizes are selected to ensure similar weighted bandwidth allocation to GCA. We examine protocol performances under both heterogeneous packet sizes ranging from 400B to 1000B and homogeneous packet sizes of 512B.

Figure 5(a) shows that when packet sizes are different, both GCA-EXP and GCA-DIRECT achieve a much larger fairness index than IEEE 802.11e, AOB-DIF and PFCR and the fairness index is very close to 1 regardless of the number of competing nodes. Since in IEEE 802.11e, AOB-DIF and PFCR, the contention window size is independent of the packet size, nodes that send larger packets obtain more bandwidth than their fair share, resulting in severe unfairness. With uniform packet size, the fairness of IEEE 802.11e and PFCR approach GCA. AOB-DIF's performance, however, is still worse than GCA (see Figure 5(b)) since it is not able to provide proportional fairness. The fairness index of GCA-EXP is also slightly smaller than GCA-DIRECT because the exponential increase of the contention window after a collision changes the ratio between contention window sizes and hence degrades the fairness of bandwidth allocation. However, since GCA-EXP is able to adjust the minimum contention window to avoid excessive collisions, it essentially limits the effects of collisions on fairness, resulting in better fairness performance than IEEE 802.11e and PFCR.

2) Strict priority: To demonstrate the ability of GCA to achieve strict priority, we use similar set up as in Section X-B.1 except that the utility function is a weighted linear function. Figures 6(a) and 6(b) show that for both uniform and heterogeneous packet sizes, both GCA-EXP and GCA-

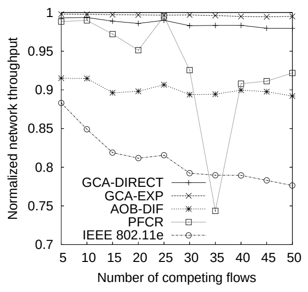


(a) Different packet size

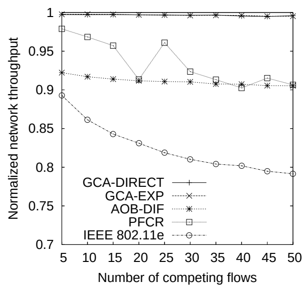


(b) Same packet size

Fig. 7. Normalized total network throughput for weighted proportional fairness



(a) Different packet size



(b) Same packet size

Fig. 8. Normalized total network throughput of strict priority

DIRECT achieve a fairness index that is very close to the ideal allocation of a strict priority policy while none of AOB, PFCR and IEEE 802.11e can realize strict priority.

C. Channel utilization

We evaluate GCA's ability to achieve high channel utilization by comparing it with IEEE 802.11e, AOB, PFCR and the theoretical maximum capacity of the network under both weighted proportional fairness and strict priority.

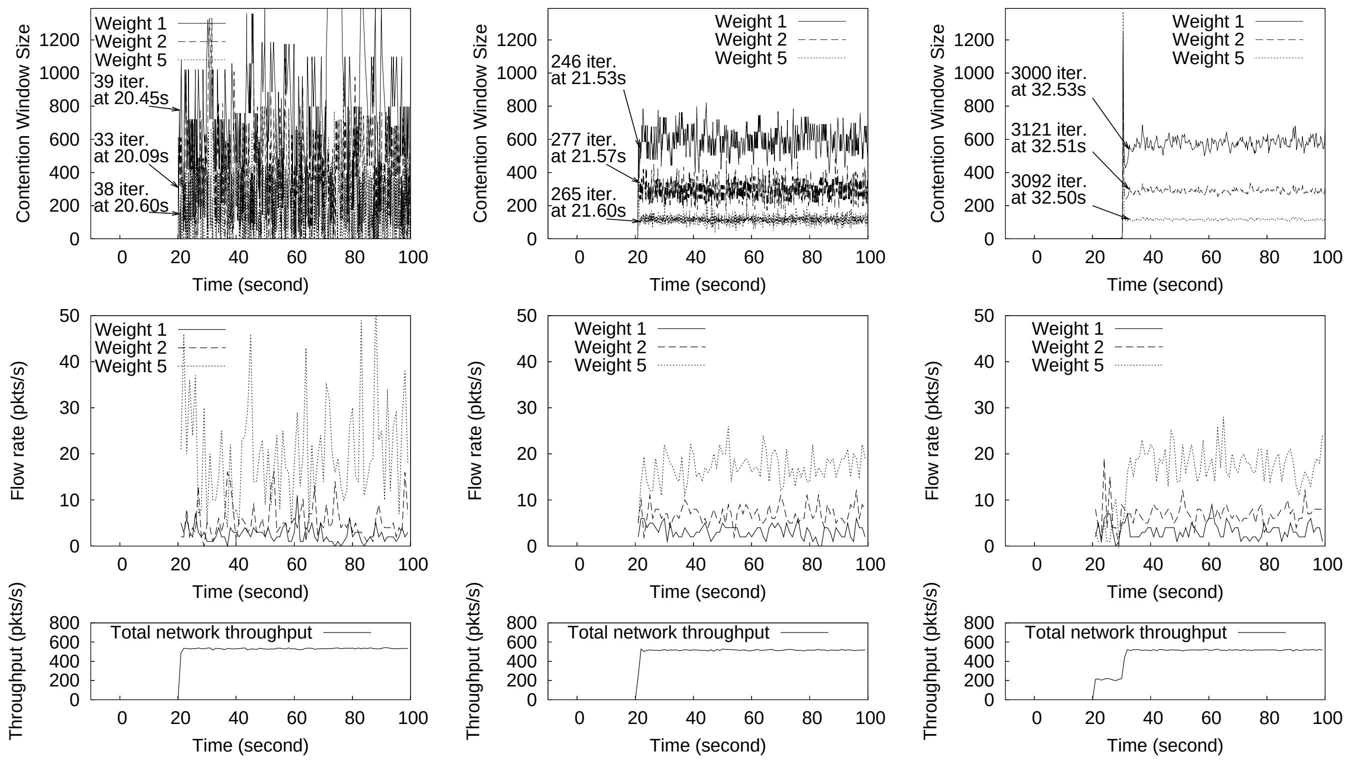
1) Weighted proportional fairness: The setting of the simulations is the same as Section X-B.1. Figures 7(a) and 7(b) show that the throughput of GCA, normalized to the theoretical maximum effective capacity of an IEEE 802.11 network, is very close to the theoretical limit, indicating efficient channel usage. Since AOB is also designed for maximizing channel utilization, its throughput is also very close to the theoretical maximum. Since IEEE 802.11e can not dynamically adjust its minimum contention window size according to the congestion level, its channel utilization degrades as the number of competing nodes increases. PFCR's throughput also drops as the network load increases since its design does not consider channel utilization. It is worth noting that when packet sizes are different, PFCR, IEEE 802.11e and AOB may achieve a slightly higher channel utilization than both the theoretical maximum effective capacity and GCA (see left part of Figure 7(a)). This is because the theoretical maximum effective capacity is calculated based on perfect weighted proportional fairness. PFCR, IEEE 802.11e and AOB unfairly allocate more bandwidth to flows with large packets, which results in higher channel utilization due to smaller amortized communication overhead of large packets. This favoring of flows with large packets in PFCR, IEEE 802.11e and AOB results in their degraded fairness as demonstrated in Figure 5(a).

2) Strict priority: To demonstrate the ability of GCA to maximize network utilization under strict priority, we compare the channel utilization of GCA, AOB-DIF, PFCR and IEEE 802.11e to the theoretical maximum throughput of a single flow network (essentially a single flow with contention window size of 1), under both fixed and heterogeneous packet sizes. The setting of the simulations is the same as Section X-B.2. Figures 8(a) and 8(b) show that the throughput of GCA (with multiple competing nodes) normalized to the maximum single-flow throughput is very close to the theoretical limit of the network. On the other hand, the channel utilization of the other approaches is much lower than GCA since they are unable to allocate all of the bandwidth to a single flow with the highest priority, which essentially hurts the network throughput since collisions and backoffs, which are unavoidable when multiple flows compete, waste network bandwidth.

D. Convergence under different step sizes

Although our theoretical analysis has focused on continuous-time models to design GCA, continuous adaptation of contention window size is not possible in practice since individual nodes can only periodically adjust their contention window sizes at a finite granularity. The discretization of GCA may introduce "imperfections" into GCA's control since the step size of the update algorithm may affect the network convergence speed and nodes may update their contention window sizes asynchronously. Such imperfections may cause a loss of network performance or instability in the contention window sizes and flow rates. Therefore, in this section, we examine the convergence speed of GCA under different step sizes. Note that in all simulations in Section X, since every node updates its contention window size before it transmits a packet and different nodes have different transmission rates, the contention window updates at nodes are not synchronized and are performed at different rates.

During the first 20 seconds of the simulations, there is no traffic in the network. Then, 50 flows using weighted log utility functions with weights ranging from 1 to 5 start transmitting at exactly the 20th second. Figure 9 shows the changes in contention window sizes, flow rates and total network throughput under different step sizes (i.e., α in Equation (8) and $1/\epsilon$ in Equation (20)). Figure 9(a) shows that under a very large step size, GCA responds to changes in network load very quickly (e.g., the contention window size of the flow with weight 1 jumps from 31 to 777 in only 39 iterations and 0.45 seconds). Therefore, there is very small loss in total network throughput. However, very large step sizes also create larger variations in contention window sizes, resulting in undesirable large variations of flow rates. Figure 9(c) shows that under a very small step size, although the contention window allocation is much more stable, GCA responds to network load changes much slower (e.g., the contention size of the flow with weight 1 changes from 31 to 544 in 3000 iterations and 12.53 seconds). Hence, some network capacity is lost during the convergence process. However, by setting the



(a) $\alpha = 2e^{-3}$ and $1/\epsilon = 1e^{-1}$ (b) $\alpha = 6e^{-4}$ and $1/\epsilon = 1e^{-2}$ (c) $\alpha = 2e^{-5}$ and $1/\epsilon = 1e^{-3}$

Fig. 9. Convergence under different step sizes. Three flows are randomly picked from the 50 flows that start from 20_{th} second.

right step size, which can be determined through experiments, both reasonably fast convergence speed and stable contention window allocation can be achieved (e.g., Figure 9(b)).

XI. Conclusion and Future Work

In response to the limitations of current algorithms, in this paper, we provide a systematic method for designing stable contention window control algorithms that can be used to achieve fair and efficient bandwidth allocation. We decompose the requirements for both fairness and efficiency to the problem of choosing proper utility functions and functions of observable channel states. Due to the inclusion of a wide diversity of both of these types of functions, we essentially broaden the scope of designing dynamic contention window control algorithms. The general dynamic contention window control algorithm (GCA) proposed by us can be used to achieve both arbitrary fairness and efficient channel utilization. However, this paper is just the first step toward a comprehensive solution for contention window control in wireless LANs and many open issues are yet to be addressed in the future.

The first open problem is the gap between modeling and real systems. In this paper, we mainly use continuous-time deterministic model to design and analyze GCA. However, contention window updates in real wireless LANs are discrete-time and stochastic due to the probabilistic nature of contention window based channel access and the asynchronous updates among nodes. To close the gap between reality and our current model, more accurate network models (e.g., stochastic models) need to be used. Some related work in the area of congestion control in wired networks [27] has shown that for proportional fairness, when the number of users in a single-link network is large, the system can be accurately described by deterministic equations. The connections between deterministic models and stochastic models under general utility functions in wired networks are still unknown. Therefore, one of the objectives of our future research is to build a stochastic model for GCA and study whether this stochastic model reaches the deterministic Equation (6) when the number of users in the system is large.

Another open problem is that GCA can only be used in single-cell wireless LANs. For more general wireless networks, such as ad hoc networks, GCA may not work since every node may observe different $f(\Omega)$. Therefore, one direction of our future work is to investigate how to achieve fair and efficient bandwidth allocation in multihop networks.

References

[1] IEEE 802.11 WG, “Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications,” in *IEEE*, 2003.

[2] P. Karn, “A New Channel Access Method for Packet Radio,” in *the 9th ARRL Computer Networking Conference*, 1996.

[3] V. Bharghavan, A. J. Demers, S. Shenker, and L. Zhang, “MACAW: A media access protocol for wireless LAN’s,” in *ACM SIGCOMM*, 1994, pp. 212–225.

[4] S. Shenker, “Fundamental design issues for the future internet,” *IEEE Journal on Selected Areas in Communication*, vol. 13, no. 7, Sept. 1995.

[5] Y. Yang and R. Kravets, “Distributed QoS Guarantees for Realtime Traffic in Ad Hoc Networks,” in *IEEE International Conference on Sensor and Ad Hoc Communications and Networks (SECON)*, 2004.

[6] ——, “Achieving Delay Guarantees in Ad Hoc Networks using Distributed Contention Window Adaptation,” in *IEEE INFOCOM*, 2006.

[7] L. Bononi, M. Conti, and E. Gregori, "Run-time optimization of IEEE 802.11 wireless LANs performance," *IEEE Transactions on Parallel and Distributed Systems (IEEE TPDS)*, vol. 15, no. 1, January 2004.

[8] H. Kim and J. Hou, "Improving Protocol Capacity with Model-based Frame Scheduling in IEEE 802.11-operated WLANs," in *ACM Mobicom*, 2003.

[9] F. Cali, M. Conti, and E. Gregori, "Dynamic Tuning of the IEEE 802.11 Protocol to Achieve a Theoretical Throughput Limit," *IEEE/ACM Transactions On Networking*, vol. 8, no. 6, pp. 785–799, Dec. 2000.

[10] ——, "IEEE 802.11 Protocol: Design and Performance Evaluation of an Adaptive Backoff Mechanism," *IEEE Journal On Selected Areas in Communications*, vol. 18, no. 9, pp. 1774–1786, September 2000.

[11] IEEE P802.11, TASK GROUP E, "Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) specifications Amendment 8: Medium Access Control (MAC) Quality of Service Enhancements," in *IEEE*, 2005.

[12] D. Qiao and K. G. Shin, "Achieving efficient channel utilization and weighted fairness for data communications in IEEE 802.11 WLAN under the dcf," in *IWQoS*, 2002.

[13] T. Nandagopal, T.-E. Kim, X. Gao, and V. Bharghavan, "Achieving MAC Layer Fairness in Wireless Packet Networks," in *ACM Mobicom*, 2000.

[14] F. Kelly, A. Maulloo, and D. Tan, "Rate control in communication networks: shadow prices, proportional fairness and stability," in *Journal of the Operational Research Society*, vol. 49, 1998.

[15] S. Golestani and S. Bhattacharyya, "A class of end-to-end congestion control algorithms for the internet," in *IEEE ICNP*, 1998.

[16] S. Kunniyur and R. Srikant, "End-to-end congestion control schemes: Utility functions, random losses and ECN marks," in *IEEE INFOCOM*, 2000, pp. 1323–1332.

[17] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Transactions on Networking*, vol. 8, no. 5, pp. 556–567, 2000.

[18] R. Srikant, *The mathematics of Internet Congestion Control*. Birkhauser, 2004.

[19] Y. Xue, B. Li, and K. Nahrstedt, "Price-Based Resource Allocation in Wireless Ad Hoc Networks," in *IWQoS*, 2004.

[20] B. Li and R. Battiti, "Performance Analysis of An Enhanced IEEE 802.11 Distributed Coordination Function Supporting Service Differentiation," in *Intl. Workshop on Quality of Future Internet Service*, 2003.

[21] L. Massoulié and J. Robert, "Bandwidth sharing: Objective and algorithms," *IEEE/ACM Transactions on Networking*, vol. 10, no. 3, June 2002.

[22] D. Bertsekas, *Nonlinear Programming: Second Edition*. Athena Scientific, 1999.

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

[24] Y. Yang, J. Wang, and R. Kravets, "Distributed Optimal Contention Window Control for Elastic Traffic in single Cell Wireless LANs - Complete Version," Tech. Rep. [Online]. Available: http://www.ece.vt.edu/yyang8/papers/ton07-complete.pdf

[25] K. Fall and K. Varadhan, "NS notes and documentation," in *The VINT Project, UC Berkely, LBL, USC/ISI, and Xerox PARC*, 1997.

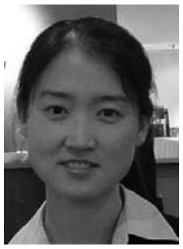
[26] R. Jain, *The Art of Computer Systems Performance Analysis: Techniques for ExperimentalDesign, Measurement, Simulation and Modeling*. John Wiley and Sons, Inc, 1991.

[27] S. Shakkottai and R. Srikant, "How Good are Deterministic Fluid Models of Internet Congestion Control," in *IEEE INFOCOM*, 2002.

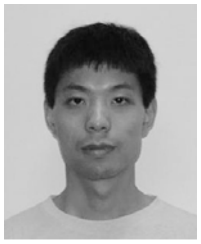
APPENDIX

A. Notation

- $\mathcal{N}$: the set of transmitting nodes $1, 2 \cdots n$
- C: the network effective capacity
- C_{max}: the maximum effective capacity
- P_i: the probability that Node i successfully transmits in a virtual slot
- s_i: the bandwidth allocated to Node i in bits per second
- S: the channel sending rate of IEEE 802.11
- L_i: $S\times$ the average duration of a successful transmission at Node i, including the DIFS and the RTS/CTS/DATA/ACK handshake
- x_i: the fraction of channel bandwidth of Node i
- W_i: the contention window size of Node i
- W_i^{min}: the minimum contention window size of Node i
- $\mathbf{W}$: $\{W_i : i \in \mathcal{N}\}$
- $\mathbf{L}$: $\{L_i : i \in \mathcal{N}\}$
- $\mathbf{P}$: $\{P_i : i \in \mathcal{N}\}$
- Ω: locally observable channel state
- Z_i: $\frac{1}{W_i}$
- θ: $\sum_{i\in\mathcal{N}} \frac{L_i}{W_i} = \sum_{k\in\mathcal{N}} Z_k L_k$
- Γ: the invariant set of GCA
- R: the invariant set of GCA-Z
- F: the average time between successful packet transmissions
- I: the average number of idle virtual slots between two busy virtual slots
- ω: $\sum_{i\in\mathcal{N}} \frac{1}{W_i}$
- φ_i: $\frac{1/W_i}{\omega}$
- T_b: the average length of a busy virtual slot
- T_c: the average duration of a virtual slot including a collision
- P_I: the probability that a virtual slot is an idle slot
- P_c: the probability that a collision happens in a virtual slot
- ϕ_i: the collision probability of node i's transmission
- m_k: the number of transmission attempts for a packet



Yaling Yang (M'03/ACM'03) is currently an assistant professor at the Electrical and Computer Engineering Department at Virginia Polytechnic Institute and State University (Virginia Tech). She received her Ph.D. degree of Computer Science from the University of Illinois at Urbana-Champaign. Her research interests include resource management and QoS in wireless networks, network routing, cognitive networks and congestion control. For a list of publications and more detailed information, please visit: http://www.ece.vt.edu/yyang8.



Jun Wang (M'01/ACM'01) received the Ph.D. degree in computer science from the University of Illinois at Urbana-Champaign (UIUC) in 2003. From 2003 to 2006, he was a senior security engineer and research staff member in the National Center for Supercomputing Applications (NCSA). He is a Software Design Engineer in the Enterprise Networking Group at Microsoft. His research interests include computer networks and data communications, network survivability and security, network QoS, multimedia systems, and distributed systems. For a list of publications and more detailed information, please visit: http://www.ncsa.uiuc.edu/~wangj/.

Robin Kravets is currently an associate professor at the Computer Science Department at the University of Illinois, Urbana-Champaign. Dr. Kravets received her Ph.D. from the College of Computing, Georgia Institute of Technology in 1999. She is the head of the Mobius group at UIUC, which researches communication issues in mobile and ad hoc networking, including power management, connectivity management, transport protocols, admission control, location management, routing and security. Her research has been funded by various sources, including the National Science Foundation and HP Labs. She is currently a member of the Editorial Board for IEEE Transactions on Mobile Computing and Elsevier Ad Hoc Networks Journal and was an Associate Editor of MC2R: Mobile Computing and Communications Review. For a list of publications and more detailed information, please visit: http://www-sal.cs.uiuc.edu/~rhk/.