Distributed Optimal Contention Window Control for Elastic Traffic in Wireless LANs

Yaling Yang, Jun Wang and Robin Kravets
University of Illinois at Urbana-Champaign
{ yyang8, junwang3, rhk@cs.uiuc.edu}

***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 of GCA 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 nature of wireless channels 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 [20], MACA [10] and MACAW [3], use contention windows to control the channel access of nodes. Contention windows not only reduce network congestion, but also directly affect the share of bandwidth that a node achieves during competition for the channel. Therefore, it is natural to extend such algorithms to support bandwidth allocation.

Applications requiring bandwidth allocation can either be realtime traffic (e.g., video/audio streaming) or elastic traffic [19] (e.g., file transfer). While realtime traffic requires service guarantees to ensure optimal bandwidth allocation, elastic traffic always has backlogged packets and adjusts its rate to fill the available bandwidth. Hence, competing flows with elastic traffic are more concerned about fairness and efficiency of bandwidth allocation. While efficiency is defined by bandwidth utilization, fairness must be defined by the goals of the particular network, which may mean uniform bandwidth allocation or weighted proportional bandwidth allocation or the highest priority node obtaining all bandwidth. The focus of this paper 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. Using contention window control for service guarantees for realtime traffic is beyond the scope of this paper and can be found in our technical report [22].

There have been extensive studies on contention window control in wireless LANs. However, none of these approaches can support both an arbitrary definition of fairness and efficient use of bandwidth. The first type of algorithm, including IEEE 802.11e [15] and [1], assigns different minimum contention window sizes to different types of nodes to achieve weighted fairness. However, since minimum contention window sizes are pre-configured and do not adapt to congestion, such approaches do not utilize the channel efficiently. The second type of algorithm, including AOB [5], MFS [13], [6] and [7], only focuses on efficient channel utilization in the context of uniform bandwidth allocation and the support for other definitions of fairness is limited. The third type of algorithm, including PFCR [17], tries to provide a more general definition of fairness by modeling fairness as an optimization problem of transmission rate allocation. However, the mapping between rate allocation and contention window adaptation in PFCR is only appropriate for a limited set of fairness definitions (See Section VIII-A). The final type of algorithm, P-MAC [18], tries to achieve both proportional fairness and efficient utilization by estimating the contention windows used by the competing nodes. Such estimation requires that every node, with or without packets for transmission, must start P-MAC simultaneously and calculate the contention window sizes for all other nodes synchronously. Nodes with outdated contention window sizes of the other nodes due to asynchronous starting time or temporary failure may cause the algorithm to fail.

Due to the limitations of the existing approaches, we propose our distributed contention window control algorithm, called GCA (**G**eneral **C**ontention window **A**daptation), which can be used to achieve optimal bandwidth allocation for competing wireless nodes in terms of efficient channel utilization and various definitions of

fairness. The goal of GCA is to provide a general solution for design and analysis of dynamic contention window control algorithms in wireless LANs.

There are four major contributions of this paper. First, we identify and model, for the first time, an arbitrary fairness definition as an optimization problem for contention window assignment. Second, even though a node's bandwidth share depends on the contention window sizes of *all* competing nodes, GCA *does not* require any global knowledge. In GCA, a node only adjusts its own contention window size based on locally available information and the system automatically converges to any given fairness definition. Third, by studying the properties of the stable point of GCA, we show that efficient channel utilization can also be achieved by controlling the stable point. Finally, we demonstrate that GCA provides a systematic scheme to generalize and evaluate related approaches.

This paper is organized as follows. Section II reviews the IEEE 802.11 contention resolution and the relationship between bandwidth allocation and contention window size. Section III relates an arbitrary fairness definition to an optimal contention window assignment problem. Section IV introduces our contention window control algorithm, GCA. Section V shows that GCA converges to the solution of the optimal contention window assignment problem and Section VI shows how to control the stable point of GCA to achieve high channel utilization. Section VII discusses guidelines for implementing GCA. In Section VIII, we use GCA to analyze several existing approaches. Section IX presents the evaluation of GCA using simulations. Finally, Section X concludes and discusses future research.

II. Bandwidth Allocation and Contention Window Size

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. In this section, we first briefly review the contention resolution algorithm in IEEE 802.11 and then introduce the relationship between the contention window size and bandwidth allocation.

A. IEEE 802.11 DCF

In IEEE 802.11 DCF [20], 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 first part of the deferment period is determined by the success of the last transmission. If the last frame was successful, the node waits DIFS time units. If the last frame was not successful, the node waits EIFS time units. The second part of the deferment period is determined as *Backoff Time = 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 for a DIFS. When the timer expires, the node can transmit. The transmission includes either a four-way RTS-CTS-DATA-ACK handshake or just a two-way DATA-ACK handshake. 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

To understand the relationship between bandwidth allocation and contention window size, we use the following result presented in [14]:

$$\frac{s_i}{s_j} = \begin{cases} \frac{L_i W_j^{min}}{L_j W_i^{min}}, & \text{exponential increase of } W_i \\ & \text{after a collision,} \\ \frac{L_i W_j}{L_j W_i}, & W_i \text{ does not change after a collision,} \end{cases} \tag{1}$$

where s_i is the absolute bandwidth allocated to Node i in bits per second. L_i is the channel transmission rate, S, multiplied by the duration of a successful transmission at Node i, including the DIFS and RTS/CTS/DATA/ACK handshake. Since all nodes carry elastic traffic and always have backlogged packets, the combined transmissions of all nodes consume the entire network capacity, C. Hence,

$$\sum_{i \in \mathcal{N}} s_i = C, \tag{2}$$

where $\mathcal{N}$ is the set of all transmitting nodes. Combining Equations (1) and (2), the fraction of channel bandwidth allocated to Node i, x_i, is

$$x_i = \frac{s_i}{C} = \begin{cases} \frac{L_i/W_i^{min}}{\sum_{k \in \mathcal{N}} L_k/W_k^{min}}, & \text{exponential increase} \\ & \text{of } W_i \text{ after a collision,} \\ \frac{L_i/W_i}{\sum_{k \in \mathcal{N}} L_k/W_k}, & \text{otherwise} \end{cases} \tag{3}$$

Equation (3) shows that the relationship between x_i and W_i is approximately the same as the relationship between x_i and W_i^{min}. Therefore, in the rest of this paper, we design GCA assuming no exponential increase of W_i after a collision. Our algorithm can also be used to dynamically adjust W_i^{min} when exponential increase is used, which we validate through simulation (see Section IX). Notation for the entire paper can be found in Appendix D.

III. Fairness Formulation

In this section, to design a contention window control algorithm that supports various fairness definitions, we first

formulate the general fairness requirement as an optimal bandwidth allocation problem, and then translate it to an optimal contention window assignment problem.

A. Optimal bandwidth allocation (OPT_BW)

Following [11], [16], fair bandwidth allocation in wireless LAN can be modeled as an optimization problem. 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 $s_i \geq 0$ [19]. Assuming that the channel capacity is C, the optimal bandwidth allocation can be formulated as follows:

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

According to the Karush-Kuhn-Tucker optimality condition [2], the *unique* solution to OPT_BW is given by [11]:

$$\begin{cases} U_i'(s_i) = \mu, & \text{for } i \in \mathcal{N} \\ \mu(C - \sum_{i\in\mathcal{N}} s_i) = 0, & \\ \mu \geq 0, & \end{cases} \tag{4}$$

where μ is the Lagrange multiplier. Different definitions of utility functions result in different solutions to $OPT\_BW(U,C)$ and so achieve different definitions of fairness [16] (See examples in Section VII-B).

In wired networks, $OPT\_BW(U,C)$ is solved using a distributed rate adaptation algorithm [11]. Because 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, the rate adaptation algorithm is easy to implement in wired networks through TCP *congestion* window control. However, in wireless networks, the sending rate of a node depends on the contention window sizes of *all* competing nodes, and so no node has direct control over its sending rate. Therefore, the same rate control algorithm can not be directly applied to *contention* window control (see an example in Section VIII-A). To solve $OPT\_BW(U,C)$, we must translate $OPT\_BW(U,C)$ to a problem of contention window assignment, called OPT_WIN.

B. Optimal contention window assignment (OPT_WIN)

To translate the $OPT\_BW(U,C)$ problem to the OPT_WIN problem, we first map $U_i(s_i)$ in $OPT\_BW(U,C)$ to a function of x_i by substituting $U_i(s_i)$ with $\tilde{U}_i(x_i)$, where $U_i(s_i) = U_i(x_iC) = \tilde{U}_i(x_i)$. Similar to $U_i(s_i)$, $\tilde{U}_i(x_i)$ is an increasing, strictly concave and continuously differentiable function of x_i over the range $x_i \geq 0$. Then, based on Equation (3) and using the fact that $\sum_{i\in\mathcal{N}} x_i \equiv 1$, we replace x_i with W_i in the $OPT\_BW(U,C)$ and finally get the OPT_WIN problem as follows:

$$\begin{aligned} &OPT\_WIN(\tilde{U},C): \\ &\qquad \max \textstyle\sum_{i\in\mathcal{N}} \tilde{U}_i\left(\frac{\frac{L_i}{W_i}}{\sum_{k\in\mathcal{N}} \frac{L_k}{W_k}}\right) \\ &\text{over} \\ &\qquad W_i > 0 \text{ for } i \in \mathcal{N}. \end{aligned}$$

By replacing s_i in Equation (4) with W_i based on Equation (3) and using the fact that $(1 - \sum_{i\in\mathcal{N}} \frac{\frac{L_i}{W_i}}{\sum_{k\in\mathcal{N}} \frac{L_k}{W_k}}) \equiv 0$, the solution to $OPT\_WIN(\tilde{U},C)$ is given by:

$$\begin{cases} \tilde{U}_i'\left(\frac{L_i/W_i}{\sum_{k\in\mathcal{N}} L_k/W_k}\right) = \mu, & \text{for } \forall i \in \mathcal{N} \\ \mu \geq 0. & \end{cases} \tag{5}$$

It is very important to note that, although the solution to $OPT\_WIN(\tilde{U},C)$ is unique in terms of $x_i = \frac{L_i/W_i}{\sum_{k\in\mathcal{N}} L_k/W_k}$, it is not unique in terms of W_i. Consider contention window sizes $\mathbf{W} = \{W_i : i \in \mathcal{N}\}$ that solve $OPT\_WIN(\tilde{U},C)$. When $\mathbf{W}$ is multiplied by a constant factor a, the resulting contention window assignment $a\mathbf{W} = \{aW_i : i \in \mathcal{N}\}$ is also a solution to $OPT\_WIN(\tilde{U},C)$. Among the possible solutions to $OPT\_WIN(\tilde{U},C)$ that satisfy the fairness requirement, channel utilization can be quite different. Therefore, the identification of the solution of $OPT\_WIN(\tilde{U},C)$ that maximizes channel utilization is important (see Section VI-A).

IV. General Contention Window Adaptation Algorithm (GCA)

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. Since GCA is fully distributed, each node needs only collect local information and adjust its contention window size accordingly. In addition, unlike previous work on dynamic contention window control [5], [6], [7], [17], [18], GCA can be used in a network where nodes have different frame sizes.

In GCA, a Node i adapts its W_i according to the following differential equation:

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

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.

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

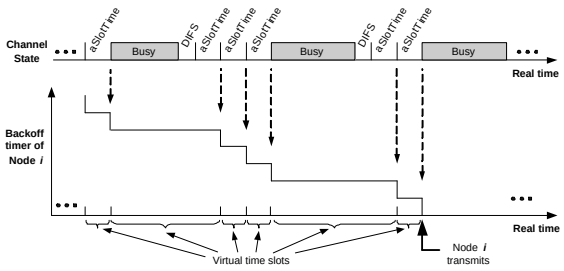


Fig. 1. Virtual slots

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 V), the value of Ω must depend on the window sizes and packet lengths of all nodes. In other words, denoting $\mathbf{W} = \{W_i : i \in \mathcal{N}\}$ and $\mathbf{L} = \{L_i : i \in \mathcal{N}\}$, GCA requires $\Omega = \Omega(\mathbf{W}, \mathbf{L})$. Such 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 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 VIII that these assumptions are easy to meet and GCA can be used to model different dynamic contention window control algorithms.

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 the state transition of a wireless LAN is a discrete-time Markov process as shown in the Bianchi model [4] of IEEE 802.11. 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. Figure 1 illustrates an example of this model. 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 can not 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 becomes:

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

If a node only performs the window size update for each packet transmission, which means that the average number of virtual slots between each update is $\frac{W_i}{2}$, the discrete version of GCA becomes:

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

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

V. 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 OPT_WIN given the three assumptions about $f(\Omega)$. Our proof includes two theorems. Theorem 1 states that under the first assumption of $f(\Omega)$, GCA converges to an *invariant set* [12] where each element of the set is a solution to OPT_WIN. Based on the second and third assumptions for $f(\Omega)$, the second theorem shows that GCA converges to a unique point that solves OPT_WIN.

Theorem 1: Starting from any initial state of $\mathbf{W}$, GCA converges to an invariant set $\Gamma = \{\mathbf{W} : \frac{\dot{W}_i}{W_i} = \frac{\dot{W}_j}{W_j}, \text{ for } \forall i, j \in \mathcal{N}\}$ and every element in Γ is a solution to OPT_WIN. In addition, inside Γ, the $\tilde{U}_i'(x_i)$ in Equation (6) remains a constant and $\tilde{U}_i'(\frac{L_i/W_i}{\sum_{k\in\mathcal{N}} L_k/W_k}) = \tilde{U}_j'(\frac{L_j/W_j}{\sum_{k\in\mathcal{N}} L_k/W_k}), \text{ for } \forall i, j \in \mathcal{N}$.

Proof: The proof consists of three steps. At step one, for notation simplicity, we translate GCA in Equation (6) to an equivalent algorithm of $Z_i = \frac{1}{W_i}$, called GCA-Z. At step two, we prove that GCA-Z converges to an invariant set R. At step three, using the equivalence between GCA and GCA-Z, we find the invariant set Γ of GCA and show that every point in Γ is a solution to OPT_WIN.

Step 1: From $Z_i = 1/W_i$, we have $\dot{Z}_i = -\frac{1}{(W_i)^2}\dot{W}_i = -Z_i^2\dot{W}_i$. By replacing x_i in Equation (6) with W_i based on Equation (3) and then replacing W_i and $\dot{W}_i$ with Z_i and $\dot{Z}_i$, GCA is translated to GCA-Z as follows:

$$\dot{Z}_i = \alpha Z_i \left[\tilde{U}_i'(\frac{Z_i L_i}{\sum_{k\in\mathcal{N}} Z_k L_k}) - f(\Omega)\right]. \quad (9)$$

Step 2: Define a scalar function $V(\mathbf{Z})$ as:

$$V(\mathbf{Z}) = \sum_{i\in\mathcal{N}} \tilde{U}_i(\frac{Z_i L_i}{\sum_{k\in\mathcal{N}} Z_k L_k}), \quad (10)$$

where $\mathbf{Z} = \{Z_i : i \in \mathcal{N}\}$. According to Lemma 1 in Appendix A, V is a Lyapunov function for GCA-Z with $\dot{V} \geq 0$ [1] and the zero values of $\dot{V}$ are obtained in the set:

$$R = \left\{\mathbf{Z} : \frac{\dot{Z}_i}{Z_i} = \frac{\dot{Z}_j}{Z_j}, \forall i, j \in \mathcal{N}\right\}. \quad (11)$$

[1]Note that for a maximization problem, the convergence condition is $\dot{V} \geq 0$ [11], [21].

According to Lemma 2 in Appendix B, R is an invariant set for GCA-Z and every element inside R satisfies

$$\tilde{U}_i'(\frac{Z_i L_i}{\sum_{k\in\mathcal{N}} Z_k L_k}) = \tilde{U}_j'(\frac{Z_j L_j}{\sum_{k\in\mathcal{N}} Z_k L_k}) = \text{constant}, \forall i,j \in \mathcal{N}. \tag{12}$$

Therefore, whenever the system state evolves into R, it remains in R. Using the La Salle Invariant Set Principle [12], we conclude that GCA-Z converges to R.

Step 3: Due to the equivalence between GCA and GCA-Z, we conclude that GCA converges to an invariant set Γ. Replacing W_i and $\dot{W}_i$ with Z_i and $\dot{Z}_i$ in Equations (11) and (12), Γ can be expressed as $\Gamma = \left\{ \mathbf{W} : \frac{\dot{W}_i}{W_i} = \frac{\dot{W}_j}{W_j}, \forall i,j \in \mathcal{N} \right\}$ and every element inside Γ satisfies $\tilde{U}_i'(\frac{L_i/W_i}{\sum_{k\in\mathcal{N}} L_k/W_k}) = \tilde{U}_j'(\frac{L_j/W_j}{\sum_{k\in\mathcal{N}} L_k/W_k}) = \text{constant}$, $\forall i,j \in \mathcal{N}$. Clearly, Γ matches the optimality condition for OPT_WIN in Equation (5) and so any element in Γ solves OPT_WIN. ■

Theorem 2: If $\Omega = \Omega(\mathbf{W}, \mathbf{L})$ and $f(\Omega)$ is strictly increasing with respect t $\sum_{i\in\mathcal{N}} \frac{L_i}{W_i}$ inside Γ, starting from any initial state of $\mathbf{W}$, GCA converges to a unique point $\widehat{\mathbf{W}} \in \Gamma$ that solves OPT_WIN.

Proof: Since $Z_i = \frac{1}{W_i}$, the assumptions in this theorem are equivalent to that $\Omega = \Omega(\mathbf{Z}, \mathbf{L})$ and $f(\Omega)$ is strictly increasing with respect to $\sum_{i\in\mathcal{N}} Z_i L_i$. According to Lemma 3 in Appendix C, with these assumptions, GCA-Z has a unique equilibrium point $\widehat{\mathbf{Z}}$ in R and starting from any point in R, GCA-Z converges to $\widehat{\mathbf{Z}}$. Based on the equivalence between GCA and GCA-Z, we conclude that starting from any point in Γ, the algorithm of Equation (6) converges to a unique equilibrium point $\widehat{\mathbf{W}}$ in Γ. Note that Theorem 1 already shows that starting from any initial state of $\mathbf{W}$, the algorithm of Equation (6) converges to Γ and every element of Γ is a solution to OPT_WIN. Therefore, we conclude that in the context of the assumptions, starting from any initial state of $\mathbf{W}$, GCA converges to a unique point $\widehat{\mathbf{W}} \in \Gamma$ that solves OPT_WIN. ■

Theorems 1 and 2 demonstrate that the system is stable under the control of GCA and that GCA converges to a unique point that solves OPT_WIN. Therefore, GCA achieves arbitrary fairness definitions. Next, we present how GCA can be used to achieve high channel utilization.

VI. Channel Utilization of GCA

Theorem 1 shows that the choice of utility functions defines the ratios of W_i's at the stable point of GCA, and, therefore, the fairness between nodes. However, multiple assignments of $\mathbf{W}$ may satisfy the same ratio condition and their channel utilization may be quite different.

Theorem 2 shows that the choice of $f(\Omega)$ ensures that the system only has one stable point and hence determines channel utilization. If $\mathbf{W}$ at the stable point is too large, channel bandwidth is not fully utilized since idle periods are too long. If $\mathbf{W}$ at the stable point is too small, collisions increase, which also results in inefficient use of bandwidth. Therefore, the problem of maximizing channel utilization is essentially the problem of choosing the right $f(\cdot)$ and Ω. Together with the choice of utility functions, this should enable the system to stabilize at a point that achieves both the fairness definition and high channel utilization.

A. Optimal Stable Point

To choose $f(\Omega)$ to stabilize the system at a point that maximizes channel utilization, we need to identify the optimal stable point. In this section, we analyze the property of the optimal stable point of GCA and show that at the optimal stable 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, assume 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. The fraction of nodes in each class is $\beta_1, \beta_2, \ldots \beta_m$, where $\sum_{i=1}^{m} \beta_i = 1$. To capture the fact that the ratios between contention window sizes are determined by the choice of utility functions and the fact that the nodes in the same class share the same utility function, we define a new variable φ_i as follows:

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

$$\varphi_i = \varphi_j = \varphi_{c_k}, \forall i,j \in c_k, \tag{14}$$

where $\sum_{i=1}^{n} \varphi_i = \sum_{k=1}^{m} n\beta_k \varphi_{c_k} = 1$.

To maximize channel utilization, the average time between each successful transmission, denoted as 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 $\frac{1}{1-P_c-P_I} - 1$. In 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

$$F = \frac{1}{1-P_c-P_I}(aSlotTime \cdot P_I + T_c P_c), \tag{15}$$

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 [4], 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 (13) and (14) ,

$$P_I = \prod_{i=1}^{n}(1 - \frac{1}{W_i/2+1}) = \frac{1}{\prod_{k=1}^{m}(1+2\omega\varphi_{c_k})^{n\beta_k}}. \tag{16}$$

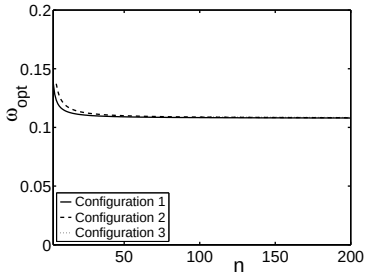


Fig. 2. ω_{opt} in three network configurations. Configuration 1 has 1 class, Configuration 2 has 2 classes with $\varphi_{c_1} : \varphi_{c_2} = 1 : 5$ and $\beta_1 : \beta_2 = 0.5 : 0.5$ and Configuration 3 has 4 classes with $\varphi_{c_1} : \varphi_{c_2} : \varphi_{c_3} : \varphi_{c_4} = 1 : 5 :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

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

$$\begin{aligned} P_c &= 1 - P_I - \textstyle\sum_{i=1}^{n} \frac{1}{W_i/2+1} \prod_{j=1, j\neq i}^{n} (1 - \frac{1}{W_j/2+1}) \\ &= 1 - (1+2\omega)P_I. \end{aligned} \tag{17}$$

Using Equations (16) and (17), Equation (15) becomes

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

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

$$\left[1 - \sum_{k=1}^{m} (\frac{2\omega_{opt} n\beta_k\varphi_{c_k}}{1+2\omega_{opt}\varphi_{c_k}})\right] \prod_{k=1}^{m} (1+2\omega_{opt}\varphi_{c_k})^{n\beta_k} = 1 - \frac{aSlotTime}{T_c}. \tag{18}$$

Since $\sum_{k=1}^{m} n\beta_k\varphi_{c_k} = 1$, when $n \to \infty$, Equation (18) becomes:

$$(1-2\omega_{opt})e^{2\omega_{opt}} = 1 - \frac{aSlotTime}{T_c}. \tag{19}$$

Solving this equation gives the lower bound of ω_{opt}. Figure 2 depicts how ω_{opt} changes as n increases. As we can see, for a large n, ω_{opt} is a quasi-constant and 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 a proper design of $f(\Omega)$.

B. Choice of $f(\Omega)$

Since ω_{opt} is a quasi-constant, by controlling the system to stabilize near ω_{opt}, the channel utilization is close to the maximum value. 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}. Examples of $f(\Omega)$ are presented in Section VIII. Our simulation results in Section IX verify the effectiveness of this approach.

VII. Implementation Considerations

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

A. Estimation of x_i

Given the network capacity C, a node can simply observe its own sending rate, s_i, to obtain x_i, since $x_i = s_i/C$. However, the capacity of a wireless channel may vary due to outside interference, such as a microwave. Therefore, this method is not practical for use in real networks. However, a node can directly estimate its x_i by observing two states of the channel, the average number of idle virtual slots between two busy virtual slots, I, and the average length of a busy virtual slot, T_b. Since in IEEE 802.11 networks a node monitors the channel continuously, I and T_b can be obtained at the MAC layer easily. In the rest of this section, IEEE 802.11 DCF RTS/CTS mode is used as an example to show how this can be done.

Let P_b be the probability that a virtual slot is a busy slot. Since I is the average number of idle slots between two consecutive busy slots,

$$P_b = 1 - P_I = \frac{1}{I+1}. \tag{20}$$

Since a busy virtual slot is caused either by a successful transmission or by a collision, the average length of a busy slot, T_b, can be expressed as:

$$T_b = T_s \frac{\sum_{i\in\mathcal{N}} P_i}{P_b} + T_c \frac{P_c}{P_b}, \tag{21}$$

where P_i is the probability that Node i successfully transmits in a virtual slot, T_s is the average length of a virtual slot with a successful transmission and T_c is the duration of a virtual slot with a collision. T_s can be expressed as $T_s = RTS + CTS + 3 \times SIFS + DATA + ACK + DIFS + aSlotTime$. Since RTS/CTS exchange is used, collisions usually happen between RTS packets. Hence, $T_c = RTS + EIFS + aSlotTime$. Based on IEEE 802.11 configurations, T_c is much smaller than T_s. Therefore, as long as P_c is not much larger than $\sum_{i\in\mathcal{N}} P_i$, using Equation (20), Equation (21) can be simplified to:

$$T_b \approx T_s \frac{\sum_{i\in\mathcal{N}} P_i}{P_b} = T_s(I+1)\sum_{i\in\mathcal{N}} P_i. \tag{22}$$

Note that T_s also satisfies $T_s = \sum_{i\in\mathcal{N}} \frac{L_i P_i}{S\sum_{j\in\mathcal{N}} P_j}$. Therefore, from Equation (22), we can get $\sum_{i\in\mathcal{N}} P_i L_i = T_b S/(I+1)$. Since $\sum_{i\in\mathcal{N}} P_i L_i$ is the average network throughput per virtual slot and $P_i L_i$ is Node i's average throughput per virtual slot,

$$x_i = \frac{P_i L_i}{\sum_{j\in\mathcal{N}} P_j L_j} \approx \frac{P_i L_i (I+1)}{T_b S}. \tag{23}$$

To calculate P_i from I, note that Node i transmits in a slot successfully if and only if it is the only node that transmits in that slot. Therefore, $P_i = \frac{1}{W_i/2+1}\prod_{j\in\mathcal{N},j\neq i}(1-\frac{1}{W_j/2+1})$. Combining the fact that $P_b = 1 - P_I$ and using Equations (16) and (20), P_i becomes $P_i = \frac{2}{W_i}(1-\frac{1}{I+1})$. Integrating this with Equation (23), we finally obtain the estimation of x_i based on I and T_b:

$$\boxed{x_i \approx \frac{2L_iI}{W_iT_bS}.} \tag{24}$$

B. Choice of Utility Functions

Depending on the system goal, GCA supports a large range of utility functions that define a variety of fairness. These utility functions can be either pre-configured in 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 the paper.

1) Strict Priority: For a system that needs to achieve strict priority (i.e., the highest-priority nodes get all 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 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 the bandwidth of the channel and our update algorithm achieves this strict priority between nodes.

2) Weighted Proportional Fairness: Some systems aim to achieve weighted proportional fairness [11] (i.e., bandwidth allocations satisfy $\frac{x_i}{\rho_i} = \frac{x_j}{\rho_j}, \forall i,j \in \mathcal{N}$, where ρ_i is the weight of Node i). The utility function for such a system is a weighted log function $U_i(x_i) = \rho_i \log x_i$. Our update algorithm for this system is: $\dot{W}_i = -\alpha W_i[\frac{\rho_i}{x_i} - f(\Omega)]$.

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 $U_i(x) = -\frac{\rho_i}{x_i}$, where ρ_i is the size of the file that Node i is transmitting. 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 in a system, 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 give GCA the flexibility to be used in systems that have different fairness policies.

VIII. Case Study

In the previous sections, we have 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 design new algorithms. Due to space limitations, we can only present a brief analysis of three examples. (Additional examples can be found in our technical report [23].) The first example shows how to use GCA to check the fairness of an existing algorithm. The second case shows how to use GCA to analyze the stability and efficiency of an existing algorithm. The final case shows how to use GCA to design a new contention window control algorithm.

A. Case 1: Fairness Analysis

In [17], it is proposed to directly translate the rate adaptation algorithm $\dot{s}_i = \alpha - \beta\frac{P_c}{\tilde{U}_i'(s_i)}$ to a contention window control algorithm,

$$\dot{Z}_i = \alpha - \beta\frac{P_c}{\tilde{U}_i'(Z_i)}, \tag{25}$$

to solve $OPT\_BW(U,C)$ (α and β are positive constants). A special case of the algorithm with a weighted log utility function is named PFCR. Assuming uniform packet size, it can be shown that this algorithm can not achieves an arbitrary fairness definition.

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

$$\tilde{U}_i'(\frac{1}{W_i}) = \tilde{U}_j'(\frac{1}{W_j}) = \frac{\beta P_c}{\alpha}, \forall i,j \in \mathcal{N}, \tag{26}$$

which does not satisfy the optimality condition for OPT_WIN in Equation (5), and hence can not achieve an arbitrary fairness. Although, for log utility functions (e.g. PFCR), when Equation (26) is satisfied, the fairness condition in Equation (5) is also satisfied. However, such a property does not hold for many utility functions (e.g $\tilde{U}_i(x_i) = \rho_i x_i + log x_i$).

B. Case 2: Stability and Efficiency Analysis

In [5], an algorithm named AOB (Asymptotically Optimal Backoff Algorithm) is proposed to dynamically adjust contention window to achieve maximum bandwidth utilization. In this section, we examine AOB's stability and efficiency in a network with a uniform priority and packet size. It is also easy to show that AOB only achieves a specific fairness definition in a multi-priority network. Due to space limitation, this analysis is not presented in this paper and can be found in our technical report [23].

In AOB, at every packet transmission, a node sets its contention window size to:

$$Z_i^{k+1} = 0.5\left[1 - \min(1, \frac{1}{(I^k+1)2\omega_{opt}})^{m_k}\right], \quad (27)$$

where ω_{opt} is pre-computed and m_k is the number of transmission attempts for the current packet. The following analysis shows that AOB is a special form of GCA.

Using utility function $\tilde{U}(x) = x - 0.5x^2$, which is a strictly increasing concave function in the range of $[0, 1]$, Equation (27) becomes:

$$\begin{aligned} Z_i^{k+1} - Z_i^k &= \frac{1}{2(I+1)}[\tilde{U}'(2Z_i^k(I_k+1)) - \\ &\quad \min(1, \frac{1}{(I_k+1)^{m_k-1}2\omega_{opt}^{m_k}} - I)]. \end{aligned} \quad (28)$$

By approximating $\sum_{k\in\mathcal{N}} Z_k \approx \frac{1}{2(I+1)}$, the discrete form of GCA-Z in Equation (9) becomes:

$$Z_i^{k+1} - Z_i^k = 0.5\alpha[\tilde{U}_i'(2Z_i^k(I+1)) - f(\Omega)], \quad (29)$$

where each iterative step is a packet transmission. Comparing Equations (28) and (29) shows that the AOB algorithm is a special case of GCA with:

$$f(\Omega) = min\left(1, \frac{1}{(I+1)^{m-1}2\omega_{opt}^m} - I\right). \quad (30)$$

For $f(\Omega)$ to satisfy the convergence condition of GCA, $f(\Omega)$ must be strictly increasing with respect to $\theta = \sum_{i\in\mathcal{N}} \frac{L_i}{W_i}$ in the invariant set Γ. Combining Equations (20) and (16), the relationship between I and W_i is:

$$\frac{1}{I+1} = (1 - \prod_{j=1}^{n}(\frac{W_j/2}{W_j/2+1})). \quad (31)$$

In addition, Theorem 1 shows that inside Γ, $\tilde{U}_i'(\frac{1/W_i}{\sum_{k\in\mathcal{N}} 1/W_k})$ remains a constant. Denoting the constant as $\hat{\gamma}$, we get $W_i = 1/[\tilde{U}_i'^{-1}(\hat{\gamma})\theta]$. Combining this with Equations (30) and (31), $f(\Omega)$ 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 $Z_i = \frac{\omega}{n}$. Since at the stable point, $Z_i^{k+1} - Z_i^k = 0$, based on Equations (27) and (31),

$$\frac{2\omega}{n} = 1 - (\frac{\frac{1}{I+1}}{2\omega_{opt}})^m, \quad \text{where} \frac{1}{I+1} = 1 - \prod_{i=1}^{n}\frac{1}{1+\frac{2\omega}{n}}. \quad (32)$$

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

$$\begin{cases} 1 - 2\omega_1 & = (\frac{\omega_1}{(1+2\omega_1)\omega_{opt}})^m, \\ (\frac{1-e^{-2\omega_2}}{2\omega_{opt}})^m & = 1. \end{cases} \quad (33)$$

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.

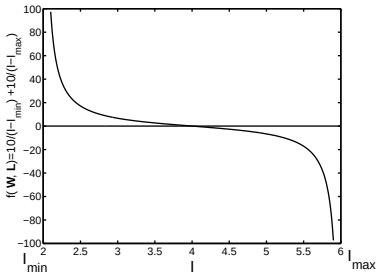


Fig. 3. $f(\Omega)$ for Case 3

C. Case 3: New Algorithm Design

In this section, we present an example of the process of designing a special case of GCA. In this example, we assume any utility function that is strictly increasing and concave and that the observed channel state Ω is I. According to Equation (31), for a large n, $I \approx \frac{1}{e^{2\omega}-1}$. Therefore, $f(\Omega)$ can be defined as:

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

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 VI-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 IX.

IX. Evaluation

In this section, we evaluate the performance of two variants of GCA using simulations in ns2 [8]. 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. The evaluation focuses on three aspects: (1) support for different definitions of fairness, (2) maintaining fairness and (3) maintaining efficiency. Although GCA supports various fairness definition, 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 VIII-C. The performance of AOB, which is an existing variant of GCA, can be found in [5]. Finally, 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, using uniform 512B packet size.

To examine the behavior of GCA for proportional fairness, in a 70-second simulation, five competing nodes start transmitting 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 keeps the system operating near its optimal point and maintains a steady throughput regardless of the number of competing nodes (see Figure 4(c)). At the same time, GCA maintains the ratio between contention window sizes to provide each node its weighted fair share of bandwidth (see Figure 4(b)).

To examine the behavior of GCA for strict priority, in a 100s simulation, five competing nodes start to transmit bulk data at 5s and each is equipped with a linear utility functions with weights ranging from 1 to 5. Figure 5(a) 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 (see Figure 5(b)). 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 evaluate GCA's accuracy in achieving fairness by Jain's fairness index [9], a common measure of fairness for bandwidth allocation. 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. When perfect proportional fairness is achieved, the fairness index equals 1. If, on the other hand, only one node out of n is allocated bandwidth, the fairness index is $1/n$. Since bandwidth allocation based on strict priority aims to give all the bandwidth to the node with the highest priority, the fairness index should be $1/n$ for a perfect strict priority based bandwidth allocation.

1) Weighted proportional fairness: To examine GCA's ability for achieving weighted proportional fairness, in this simulation, 5 to 50 competing nodes start in the first 10s and use weighted log utility functions with weights from 1 to 5. GCA's performance under both heterogeneous packet sizes ranging from 400B to 1000B and homogeneous packet sizes of 512B are examined.

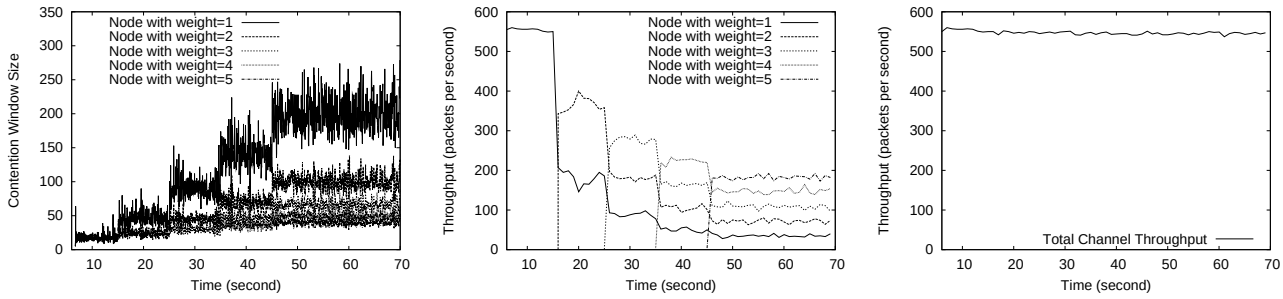
Figure 6(a) shows that when packet sizes of nodes are different, both GCA-EXP and GCA-DIRECT achieve a much larger fairness index than IEEE 802.11e and the fairness index is very close to 1 regardless of the number of competing nodes. Since in IEEE 802.11e, the contention window size is independent of the packet size, nodes that send larger packets obtain more bandwidth than their fair share, resulting in IEEE 802.11e's severe unfairness. With uniformed packet size, the fairness for IEEE 802.11e is greatly improved (see Figure 6(b)), although its performance is still worse than GCA. The fairness index of GCA-EXP is also slightly smaller than GCA-DIRECT because the exponential increment 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.

2) Strict priority: To demonstrate the ability of GCA to achieve strict priority, we use similar set up as in Section IX-B.1 except that the utility function is a weighted linear function. Figure 6(c) shows that regardless of uniform or different packet sizes, both GCA-EXP and GCA-DIRECT achieve a fairness index that is very close to the ideal allocation of strict priority policy.

C. Channel utilization

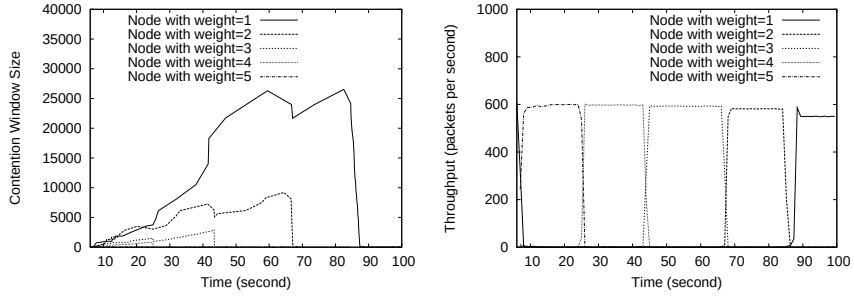
Finally, we evaluate GCA's ability to achieve high channel utilization by comparing it with IEEE 802.11, IEEE 802.11e and the theoretical network capacity.

1) Weighted proportional fairness: In this set of simulations, the channel utilization of GCA is compared with IEEE 802.11e and the theoretical maximum network capacity. For GCA, the weights of the log utility functions range from 1 to 5. For IEEE 802.11e, there are five classes of traffic, with the minimum contention window sizes of the classes being 30, 37, 50, 75 and 150, respectively. These contention window sizes are selected to ensure similar weighted bandwidth allocation to GCA. Each simulation runs for 100s with 5 to 50 competing nodes, each starting at 10s and sending 512B packets. Figure 7(a) shows that the throughput of GCA, normalized to the theoretical maximum capacity of an IEEE 802.11 network, is very close to the theoretical limit, indicating efficient channel usage. 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.



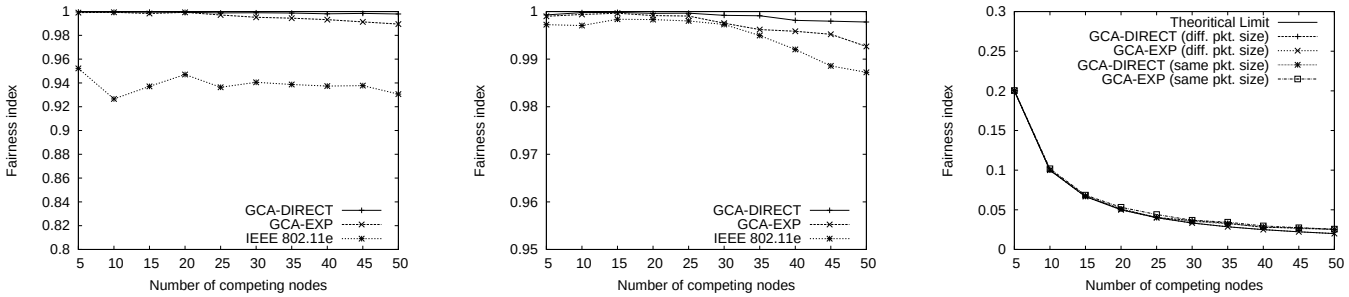
(a) Contention window (b) Per node throughput (c) Total network throughput

Fig. 4. Evolution of GCA: proportional fairness.



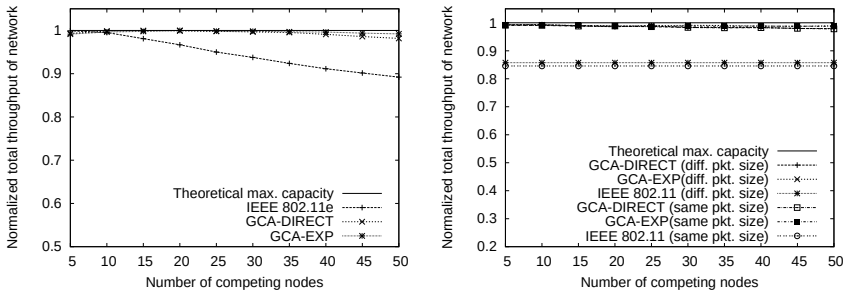
(a) Contention window (b) Per node throughput

Fig. 5. Evolution of GCA: strict priority



(a) Proportional fairness (diff. pkt. size) (b) Proportional fairness (same pkt. size) (c) Strict priority

Fig. 6. Fairness index



(a) Proportional fairness (b) Strict priority

Fig. 7. Channel Utilization

2) Strict priority: Since strict priority requires that only the node with the highest priority wins the bandwidth, we compare the channel utilization of GCA to a single-node IEEE 802.11 network and the theoretical maximum network capacity, under both fixed and heterogeneous packet sizes. Each simulation runs for 100s. All nodes start in the first 10s. Figure 7(b) shows that the throughput of GCA (with multiple competing nodes) normalized to the maximum network capacity is very close to the theoretical limit of the network, while the channel utilization of the

single-node IEEE 802.11 network is much lower than GCA due to its inability to adapt to the light load in the network.

X. 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 requirement 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.

For future work, we plan on comparing the performances of different choices of $f(\Omega)$ and extending GCA into multihop wireless networks to support fair bandwidth allocation along with efficient channel utilization.

References

[1] Imad Aad and Claude Castelluccia. Differentiation Mechanisms for IEEE 802.11. In *Proceedings of INFOCOM*, 2001.

[2] Dimitri Bertsekas. *Nonlinear Programming: Second Edition*. Athena Scientific, 1999.

[3] Vaduvur Bharghavan, Alan J. Demers, Scott Shenker, and Lixia Zhang. MACAW: A media access protocol for wireless LAN's. In *SIGCOMM*, pages 212–225, 1994.

[4] Giuseppe Bianchi. Performance analysis of the IEEE 802.11 distributed coordination function. *IEEE Journal on Selected Areas in Communications*, 18(3), 2000.

[5] Luciano Bononi, Marco Conti, and Enrico Gregori. Run-time optimization of IEEE 802.11 wireless LANs performance. *IEEE Transactions on Parallel and Distributed Systems (IEEE TPDS)*, 15(1), January 2004.

[6] Frederico Cali, Marco Conti, and Enrico Gregori. Dynamic tuning of the IEEE 802.11 protocol to achieve a theoretical throughput limit. *IEEE Transactions on Networking*, 8(6), December 2000.

[7] Frederico Cali, Marco Conti, and Enrico Gregori. IEEE 802.11 protocol: Design and performance evaluation of an adaptive backoff mechanism. *IEEE Journal on Selected Areas in Communications*, 18(9), September 2000.

[8] Kevin Fall and Kannan Varadhan. NS notes and documentation. In *The VINT Project, UC Berkely, LBL, USC/ISI, and Xerox PARC*, 1997.

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

[10] Phil Karn. A New Channel Access Method for Packet Radio. In *Proceedings of the 9th ARRL Computer Networking Conference*, 1996.

[11] 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*, volume 49, 1998.

[12] Hassan Khalil. *Nonlinear systems*. Prentice Hall, 1996.

[13] Hwangnam Kim and Jennifer Hou. Improving Protocol Capacity with Model-based Frame Scheduling in IEEE 802.11-operated WLANs. In *ACM Mobicom*, 2003.

[14] Bo Li and Roberto Battiti. Performance Analysis of An Enhanced IEEE 802.11 Distributed Coordination Function Supporting Service Differentiation. In *International Workshop on Quality of Future Internet Service*, 2003.

[15] Stefan Mangold, Sunghyun Choi, Peter May, Ole Klein, Guido Hiertz, and Lothar Stibor. IEEE 802.11e Wireless LAN for Quality of Service. In *Proceedings of European Wireless*, 2002.

[16] Jeonghoon Mo and Jean Walrand. Fair end-to-end window-based congestion control. *IEEE/ACM Transaction of Networking*, 8(5):556–567, 2000.

[17] Thyagarajan Nandagopal, Tae-Eun Kim, Xia Gao, and Vaduvur Bharghavan. Achieving MAC Layer Fairness in Wireless Packet Networks. In *ACM Mobicom*, 2000.

[18] Daji Qiao and Kang G. Shin. Achieving efficient channel utilization and weighted fairness for data communications in IEEE 802.11 WLAN under the dcf. In *IWQoS*, 2002.

[19] Scott Shenker. Fundamental design issues for the future internet. *IEEE Journal on Selected Areas in Communication*, 13(7), September 1995.

[20] IEEE Computer Society. 802.11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications.

[21] Rayadurgam Srikant. *The mathematics of Internet Congestion Control*. Birkhauser, 2004.

[22] Yaling Yang and Robin Kravets. Distributed QoS Guarantees for Realtime Traffic in Ad Hoc Networks. Technical Report UIUCDCS-R-2004-2446, June 2004.

[23] Yaling Yang, Jun Wang, and Robin Kravets. Distributed Optimal Contention Window Control for Elastic Traffic in Wireless LANs. Technical Report UIUCDCS-R-2004-2427, April 2004.

Appendix

A. Proof that V is a Lyapunov function

Lemma 1: Scalar function $V(\mathbf{Z})$ in Equation (10) is a Lyapunov function for GCA-Z in Equation (9) with $\dot{V} \geq 0$. The zero values of $\dot{V}$ are obtained in the set $R = \{\mathbf{Z} : \frac{\dot{Z}_i}{Z_i} = \frac{\dot{Z}_j}{Z_j}, \forall i, j \in \mathcal{N}\}$.

Proof: Note that the derivative of $V(\mathbf{Z})$ to Z_i is:

$$\begin{aligned}
\frac{\partial V}{\partial Z_i} &= \tilde{U}_i'\left(\frac{Z_i L_i}{\sum_{k\in\mathcal{N}} Z_k L_k}\right)\frac{L_i \sum_{k\in\mathcal{N}} Z_k L_k - Z_i L_i^2}{(\sum_{k\in\mathcal{N}} Z_k L_k)^2} \\
&\quad - \sum_{j\in\mathcal{N}, j\neq i} \tilde{U}_j'\left(\frac{Z_j L_j}{\sum_{k\in\mathcal{N}} Z_k L_k}\right)\frac{Z_j L_j L_i}{(\sum_{k\in\mathcal{N}} Z_k L_k)^2} \\
&= \frac{L_i}{(\sum_{k\in\mathcal{N}} Z_k L_k)^2}\sum_{j\in\mathcal{N}, j\neq i}[\tilde{U}_i'\left(\frac{Z_i L_i}{\sum_{k\in\mathcal{N}} Z_k L_k}\right) \\
&\quad - \tilde{U}_j'\left(\frac{Z_j L_j}{\sum_{k\in\mathcal{N}} Z_k L_k}\right)]Z_j L_j \\
&\quad \text{(From Equation (9))} \\
&= \frac{L_i}{(\sum_{k\in\mathcal{N}} Z_k L_k)^2}\sum_{j\in\mathcal{N}, j\neq i}\{[\frac{\dot{Z}_i}{\alpha Z_i} + f(\Omega)] \\
&\quad - [\frac{\dot{Z}_j}{\alpha Z_j} + f(\Omega)]\}Z_j L_j \\
&= \frac{L_i}{\alpha(\sum_{k\in\mathcal{N}} Z_k L_k)^2}\sum_{j\in\mathcal{N}, j\neq i}[\frac{\dot{Z}_i}{Z_i} - \frac{\dot{Z}_j}{Z_j}]Z_j L_j \\
&\quad \text{(Denote } \eta = \frac{1}{\alpha(\sum_{k\in\mathcal{N}} Z_k L_k)^2},\ \eta > 0\text{)} \\
&= \eta[(\sum_{j\in\mathcal{N}, j\neq i} Z_j L_j)\frac{L_i}{Z_i}\dot{Z}_i - (\sum_{j\in\mathcal{N}} \dot{Z}_j L_j)L_i].
\end{aligned}$$

Hence, the time derivative of $V(\mathbf{Z})$ is:

$$\begin{aligned}
\dot{V} &= \sum_{i\in\mathcal{N}} \frac{\partial V}{\partial Z_i}\dot{Z}_i \\
&= \eta\sum_{i,j\in\mathcal{N}, j\neq i}[\frac{Z_j L_j L_i}{Z_i}\dot{Z}_i^2 + \frac{Z_i L_i L_j}{Z_j}\dot{Z}_j^2 - 2\dot{Z}_i\dot{Z}_j L_i L_j] \\
&= \eta\sum_{i,j\in\mathcal{N}, j\neq i}[(\sqrt{\frac{Z_j L_j L_i}{Z_i}} \mid \dot{Z}_i \mid - \sqrt{\frac{Z_i L_i L_j}{Z_j}} \mid \dot{Z}_j \mid)^2 \\
&\quad + 2 \mid \dot{Z}_i \mid\mid \dot{Z}_j \mid L_i L_j - 2\dot{Z}_i\dot{Z}_j L_i L_j] \\
&\geq \eta\sum_{i,j\in\mathcal{N}, j\neq i}(\sqrt{\frac{Z_j L_j L_i}{Z_i}} \mid \dot{Z}_i \mid - \sqrt{\frac{Z_i L_i L_j}{Z_j}} \mid \dot{Z}_j \mid)^2 \\
&\geq 0.
\end{aligned}$$

The equality holds if and only if $\frac{\dot{Z}_i}{Z_i} = \frac{\dot{Z}_j}{Z_j}$, for $\forall i, j \in \mathcal{N}$. ■

B. Proof that R is an invariant set

Lemma 2: For GCA-Z, $R = \left\{ \mathbf{Z} : \frac{\dot{Z}_i}{Z_i} = \frac{\dot{Z}_j}{Z_j}, \text{ for } \forall i,j \in \mathcal{N} \right\}$ is an invariant set and inside this invariant set, $\tilde{U}_i'(\frac{Z_i L_i}{\sum_{k\in\mathcal{N}} Z_k L_k}) = \tilde{U}_j'(\frac{Z_j L_j}{\sum_{k\in\mathcal{N}} Z_k L_k}) = \text{constant}, \forall i,j \in \mathcal{N}$

Proof: Combining Equation (9) with the definition of R, we get that inside R, $\tilde{U}_i'(\frac{Z_i L_i}{\sum_{k\in\mathcal{N}} Z_k L_k}) = \tilde{U}_j'(\frac{Z_j L_j}{\sum_{k\in\mathcal{N}} Z_k L_k})$ holds. To prove that R is an invariant set, note that:

$$\begin{aligned}\tfrac{d}{dt}\tilde{U}_i'(\tfrac{Z_i L_i}{\sum_{k\in\mathcal{N}} Z_k L_k}) &= \textstyle\sum_{j\in\mathcal{N}} \left[\frac{\partial}{\partial Z_j}\tilde{U}_i'(\frac{Z_i L_i}{\sum_{k\in\mathcal{N}} Z_k L_k})\right] \dot{Z}_j \\ &= \tilde{U}_i''(\tfrac{Z_i L_i}{\sum_{k\in\mathcal{N}} Z_k L_k}) \tfrac{L_i}{(\sum_{k\in\mathcal{N}} Z_k L_k)^2} \times \\ &\quad \left[\textstyle\sum_{j\in\mathcal{N}, j\neq i} Z_j \dot{Z}_i L_j - \sum_{j\in\mathcal{N}, j\neq i} Z_i \dot{Z}_j L_j\right] \\ &= 0. \text{(Inside } R, \dot{Z}_i Z_j = \dot{Z}_j Z_i.\text{)}\end{aligned}$$

Therefore, if at any time GCA-Z gets inside R, $\tilde{U}_i'(\frac{Z_i L_i}{\sum_{k\in\mathcal{N}} Z_k L_k})$ for $\forall i \in \mathcal{N}$ remains the same constant for all future time. Combined with Equation (9), $\frac{\dot{Z}_i}{Z_i} = \frac{\dot{Z}_j}{Z_j}$, for $\forall i,j$ holds for all future time. Hence, R is an invariant set for GCA-Z. ■

C. Proof for convergence inside the invariant set

Lemma 3: If $\Omega = \Omega(\mathbf{Z}, \mathbf{L})$ and $f(\Omega)$ is strictly increasing with respect to $\sum_{i\in\mathcal{N}} Z_i L_i$ inside R, then starting from any point in R, GCA-Z converges to a unique equilibrium point $\widehat{\mathbf{Z}} \in R$.

Proof: For ease of notation, we define $\theta = \sum_{i\in\mathcal{N}} Z_i L_i$ and use $f(\mathbf{Z}, \mathbf{L})$ to represent $f(\Omega(\mathbf{Z}, \mathbf{L}))$. The rest of the proof consists of three steps. First, we translate $f(\mathbf{Z}, \mathbf{L})$ to a function of θ. Then, using the property that $f(\mathbf{Z}, \mathbf{L})$ is strictly increasing with respect to θ, we identify the unique equilibrium point $\widehat{\mathbf{Z}}$. Finally, we prove that $\widehat{\mathbf{Z}}$ is the unique stable point and so the update algorithm in Equation (9) converges to $\widehat{\mathbf{Z}}$.

Step 1: Lemma 2 shows that when GCA-Z converges to R, $\tilde{U}_i'(\frac{Z_i L_i}{\sum_{k\in\mathcal{N}} Z_k L_k})$ for $\forall i \in \mathcal{N}$ remains a constant. Defining this constant as $\hat{\gamma}$,

$$\tilde{U}_i'(\frac{Z_i L_i}{\sum_{k\in\mathcal{N}} Z_k L_k}) = \tilde{U}_i'(\frac{Z_i L_i}{\theta}) = \hat{\gamma}, \text{ for } \forall i \in \mathcal{N}. \tag{34}$$

Next, by defining $\mathbf{g}(\cdot) = \{\tilde{U}_i'^{-1}(\cdot) : i \in \mathcal{N}\}$ and $\mathbf{L}^{-1} = \{\frac{1}{L_i} : i \in \mathcal{N}\}$, $\mathbf{Z}$ can be expressed as ($\{\cdot\}$ means set):

$$\mathbf{Z} = \{Z_i\} = \left\{\frac{\tilde{U}_i'^{-1}(\hat{\gamma})\theta}{L_i}\right\} = \theta[\mathbf{g}(\hat{\gamma})]^T \mathbf{L}^{-1}. \tag{35}$$

Therefore, $f(\mathbf{Z}, \mathbf{L}) = f(\theta[\mathbf{g}(\hat{\gamma})]^T \mathbf{L}^{-1}, \mathbf{L})$.

Step 2: From Equations (9) and (34), when $\dot{Z}_i = 0$, the value of θ, denoted $\hat{\theta}$, satisfies:

$$f(\mathbf{Z}, \mathbf{L}) = f(\hat{\theta}[\mathbf{g}(\hat{\gamma})]^T \mathbf{L}^{-1}, \mathbf{L}) = \hat{\gamma}. \tag{36}$$

Since $f(\mathbf{Z}, \mathbf{L})$ is strictly increasing with respect to θ, $\hat{\theta}$ is unique. Therefore, the unique equilibrium point in R is $\widehat{\mathbf{Z}} = \hat{\theta}[\mathbf{g}(\hat{\gamma})]^T \mathbf{L}^{-1} = \{\hat{Z}_i : i \in \mathcal{N}\}$ and:

$$\hat{\theta} = \sum_{k\in\mathcal{N}} \hat{Z}_k L_k. \tag{37}$$

Step 3: To show that $\widehat{\mathbf{Z}}$ is the unique stable point of the system, we define a scalar function $V_2(\mathbf{Z})$ as follows:

$$V_2(\mathbf{Z}) = \sum_{i\in\mathcal{N}} \int_{\hat{Z}_i}^{Z_i} \frac{1}{\sigma}(\hat{Z}_i - \sigma) L_i d\sigma.$$

The following proof shows that $V_2(\mathbf{Z})$ is a Lyapunov function, and therefore GCA converges to $\widehat{\mathbf{Z}}$.

$$\dot{V}_2 = \sum_{i\in\mathcal{N}} (\frac{\partial V_2}{\partial Z_i}) \dot{Z}_i = \sum_{i\in\mathcal{N}} \frac{1}{Z_i}(\hat{Z}_i - Z_i) L_i \dot{Z}_i. \tag{38}$$

Note that inside R, according to Equations (9) and (34),

$$\dot{Z}_i = \alpha Z_i (\tilde{U}_i'(\frac{Z_i L_i}{\theta}) - f(\mathbf{Z}, \mathbf{L})) = \alpha Z_i (\hat{\gamma} - f(\mathbf{Z}, \mathbf{L})). \tag{39}$$

Combining Equations (38) and (39),

$$\begin{aligned}\dot{V}_2 &= \textstyle\sum_{i\in\mathcal{N}} \alpha(\hat{Z}_i - Z_i) L_i (\hat{\gamma} - f(\mathbf{Z}, \mathbf{L})) \\ &= \alpha(\textstyle\sum_{i\in\mathcal{N}} \hat{Z}_i L_i - \sum_{i\in\mathcal{N}} Z_i L_i)(\hat{\gamma} - f(\mathbf{Z}, \mathbf{L})) \\ &\quad \text{(Using Equations (36) and (37))} \\ &= \alpha(\hat{\theta} - \theta)\left[f(\hat{\theta}[\mathbf{g}(\hat{\gamma})]^T \mathbf{L}^{-1}, \mathbf{L}) - f(\theta[\mathbf{g}(\hat{\gamma})]^T \mathbf{L}^{-1}, \mathbf{L})\right].\end{aligned}$$

Since $f(\theta[\mathbf{g}(\hat{\gamma})]^T \mathbf{L}^{-1}, \mathbf{L})$ is strictly increasing with respect to θ in R, $\dot{V}_2 \geq 0$. This equality holds if and only if $\theta = \hat{\theta}$. Therefore, we have shown that the algorithm in Equation (9) converges to a unique point $\widehat{\mathbf{Z}} \in R$. ■

D. Notation

- $\mathcal{N}$: the set of transmitting nodes $1, 2 \cdots n$
- C: the network 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 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