A Fair MAC Protocol for IEEE 802.11-Based Ad Hoc Networks: Design and Implementation

Brahim Bensaou, *Senior Member, IEEE*, and Zuyuan Fang *Member, IEEE*

***Abstract*— In this paper, we model the problem of bandwidth sharing in wireless multi-hop networks as a general utility maximization problem with link bandwidth constraints. Lagrangean relaxation and duality are invoked to derive a gradient-based iterative algorithm to solve the problem. We then investigate the practical aspects of the problem and discuss how such theoretical framework can be used to design practical fair media access control frameworks that can be implemented in real systems based on the IEEE 802.11 distributed coordination function.**

***Index Terms*— IEEE 802.11, ad hoc networks, medium access control, fairness, backoff, convex programming, implementation.**

I. Introduction

AMONG the fundamental problems in multi-hop wireless networks is the problem of bandwidth sharing between competing links in the network. The IEEE 802.11 distributed coordination function (DCF), has gained such a wide popularity in ad hoc networks that it has become nowadays the de facto MAC layer standard for such networks. Yet, this protocol is known to suffer from a dramatic unfairness problem in the ad hoc operation mode. The unfairness in the DCF is caused by the existence of hidden terminals in ad hoc networks, and is exacerbated by the so-called binary exponential backoff (BEB) algorithm, adopted by the IEEE 802.11 standard to resolve collisions. This problem of unfairness is known to be especially severe when the network topology is non-homogeneous and the sources generate a high traffic load. A direct consequence of such a problem is the strong dependence of the feasible capacity of the network on the topology of the network making links or nodes that have a large number of competitors suffer a dramatically low link bandwidth.

Many schemes [1]–[5] have been proposed in the literature to overcome this fairness problem by modifying the IEEE 802.11 MAC protocol. Most, if not all, work well only under some scenarios. This is because none of these protocols (except [2]) consider the systematic study of the effects of bandwidth sharing principles on the fair share achieved by each link in an ad hoc network. Nowadays, this approach is becoming more and more popular and has been the subject of many recent papers. For example, [6] studies the problem of relaying (node capacity sharing) in ad hoc networks through and price-based approach; [7] studies the problem of fair link bandwidth sharing in ad hoc networks through a game theoretic approach and derives both cooperative and non-cooperative games to solve the problem; in an independent effort, [8] follows a similar approach as [7] and focuses on achieving max-min fair rates in ad hoc networks.

More recently, the systematic study of bandwidth sharing principles in ad hoc networks has been considered in holistic or cross-layer approach. The control algorithms of different protocol layers are considered as optimization problems acting on variables from different protocol layers. For example, to cite just a few, congestion control with MAC design is considered in [9], [10]. Though most such works contribute a great deal to understanding the interactions between different control algorithms from different layers, the proposed solutions remain inherently complex. As a consequence, most, if not all, existing cross-layer optimization-based algorithms are often sanctioned as theoretical only because they seldom lead to realistic simulations or practical implementations.

While advocating a systematic study of the problem, we are still driven by the practicality of the proposed solutions. Therefore in this paper we will omit the theoretical details such as the study of convergence, and so on, as these have already been addressed in our previous paper [7] and the plethora of other published papers on network utility maximization (e.g., [11]–[13]).

To address the problem, we first model the contention relations between link flows in the network by a link contention graph [3], [7], then, based on the decomposition of this graph into cliques, the fair link bandwidth allocation problem is modeled as a concave utility maximization problem in Section II. Using Lagrange relaxation and duality, we propose a distributed algorithm to solve the dual and the primal in Section III. The details, discussions and proofs for this part can be found in our earlier paper [7]. After this, the rest of the paper is spent on discussing and showing how such a distributed algorithm can be implemented in a real system. Finally, simulation and experimental results are provided to assess the performance of our proposed algorithm.

II. Model and problem formulation

An ad hoc network topology can be represented by an undirected graph whose vertices represent the network nodes, and where an edge between any pair of vertices represents radio proximity between two nodes. Due to such proximity, simultaneous transmissions often result in collisions. To model such interactions between links, from the topology graph, we can construct a link contention graph, where a vertex represents an active link, and an edge between two vertices denotes wireless proximity between the two links: that is,

Manuscript received December 16, 2005; revised October 21, 2006; accepted November 20, 2006. The associate editor coordinating the review of this paper and approving it for publication was E. Hossain. This work was supported in part under grant RGC-CERG HKUST 610506.

The authors are with the Department of Computer Science and Engineering, The Hong Kong University of Science and Technology (email: {brahim, zyfang}@cse.ust.hk).

Digital Object Identifier 10.1109/TWC.2007.051014.

either the sender or the receiver of one link is within radio proximity of the sender or the receiver of the other.

Note that this contention graph only represents the contentions that occur within the carrier sensing range and neglects the additive nature of interference. A more accurate representation of contentions should take into account interference generated by all nodes in the network, however, since interference is additive, it can only be represented accurately by knowing the whole topology of the network, which is impractical in ad hoc networks.

The cliques of the link contention graph represent accurately the time-domain bandwidth sharing constraints. That is, no two links in the same clique can be active simultaneously. Thus, each clique in the contention graph stands for a contention context, and is an accurate abstraction of a "channel resource" in the real network. Links in the same clique share the capacity of this channel resource according to the target fairness principle. So, from the bandwidth sharing viewpoint, a particular link succeeds transmission if and only if all other links in all the cliques containing this particular link do not transmit. Examining the problem from another viewpoint (the scheduling point of view, [14]), links that form an independent set in the link contention graph can be scheduled to transmit simultaneously. While the search for cliques in a graph and the search for all independent sets of a graph are both NP hard, the latter is less appealing for ad hoc networks because independent sets are spatially distributed and thus require the knowledge of the global contention graph, and thus the whole network topology, whereas cliques are by nature local subgraphs of this global graph.

A. Contention Graph and Clique Capacity

Let G be the link contention graph of an ad hoc network, and denote by $\mathcal{I}$ the family of all the independent sets of G. A conflict free schedule S is defined as an infinite sequence of independent sets, $I_1, I_2, \cdots, I_k, \cdots$ of $\mathcal{I}$. The frequency of link i in schedule S is then defined as [15]: $f_i = \lim_{t\to\infty} \sum_{k=1}^{t} S(i,k)/t$, where, $S(i,k) = 1$ if $i \in I_k$, and $S(i,k) = 0$ otherwise.

For a link contention graph with N vertices, a given vector of frequencies $\hat{f} = (f_1, \cdots, f_N)$ is said to be feasible if there exists a schedule S such that the frequency of the ith vertex in schedule S is at least f_i [14]. The frequency of vertex i can be treated as the normalized bandwidth allocated to the corresponding link in schedule S. Furthermore, a schedule is said to follow a given fairness principle if there exists a feasible vector of frequencies that verifies the fairness condition under consideration. For instance a vector of frequencies $\hat{f}^* = (f_1^*, \cdots, f_N^*)$ is said to be proportionally fair if and only if, for any other feasible vector of frequencies $\hat{f} = (f_1, \cdots, f_N)$, $\sum(f_i - f_i^*)/f_i^* \leq 0$ holds [16].

Thus the feasible capacity of a clique K in a given link contention graph and under a given fairness principle is $\sum_{i\in K} f_i^*$. However, verifying the feasibility of a vector of frequencies over a general graph is an NP-hard problem, since this equates with finding a schedule that achieves the given vector of frequencies. So characterizing the capacity of the cliques in general graphs is not easy.

In the particular case of perfect graphs, verifying the feasibility of a frequency vector can be achieved in polynomial time through the so-called *Clique feasibility* where we have $\sum_{i\in K} f_i \leq 1, \forall K$. Put simply, in perfect graphs clique capacities can be normalized to 1. Unfortunately this is not the case in general graphs including non-perfect graphs (for further discussion on this issue refer to [7]).

According to the *Strong perfect graph Conjecture* [17], a graph is perfect if and only if it has no induced subgraph that is isomorphic to an odd hole of size at least 5 or a complement of such an odd hole. In graph terminology, a hole is a cycle without chords. The ring C_5 is the smallest odd-hole (its complement, the pentagram is also an odd hole). Therefore, if a link contention graph contains one or more odd holes, clique feasibility does not hold and capacities cannot be normalized to 1. The capacity of any clique that includes edges of an odd hole (or the components that can be reduced to an odd hole by edge elimination) must be reduced by a small fraction. In practice, in distributed ad hoc networks, it is not easy to determine whether a clique contains an edge that belongs to an odd hole, because the odd whole may span all the edges in the network and thus global information would be required. So as a rule of thumb, to ensure feasibility in a general graph, each clique's capacity is systematically reduced to a fraction β. Using the results in [18] we can show that if the capacity is reduced to $\beta = 2/3$ then clique feasibility is sufficient to ensure feasibility in general graphs without having to check the existence of odd holes. This yields a lower bound on the allocated frequency vector. In practice, in carrier sensing multiple access (CSMA) based networks it is not necessary to reduce the clique capacity as such class of MAC protocols self regulates excess fair shares assigned to the nodes through collisions and backoffs.

B. Problem formulation

The link bandwidth allocation problem in ad hoc networks can be formulated as a general utility maximization problem subject to the constraints imposed by link contentions. Assume the number of links in the graph to be N. Denote the set of links as $\mathcal{N}= \{1, \cdots, N\}$ and defined as x_i to be the rate of link i, $i = 1, \cdots, N$. Denote the set of maximal cliques in G by $\mathcal{M}= \{1, \cdots, M\}$, and let the capacity of clique j be $c_j, j \in \mathcal{M}$. A link may belong to several maximal cliques: this can be described by a matrix $A = (a_{i,j})$ where:

$$a_{i,j} = \begin{cases} 1, & \text{if link } j \text{ belongs to clique } i\ ,\ j \in \mathcal{N} \\ 0, & \text{if link } j \text{ does not belong to clique } i\ ,\ i \in \mathcal{M} \end{cases}$$

Each clique's capacity is shared among the links in the clique, thus, the capacity constraints on the link rates can be written in matrix form as:

$$\boldsymbol{A}\boldsymbol{x} \leq \boldsymbol{C} \tag{1}$$

where $\boldsymbol{x} = (x_1, \cdots, x_N)^t$ is the column vector of link rates and $\boldsymbol{C} = (c_1, \cdots, c_M)^t$ is the clique capacity vector. In addition, rates must not take negative values, so:

$$x_i \geq 0, \qquad i = 1, \cdots, N. \tag{2}$$

The set of link rate vectors Ω that satisfy conditions (1) and (2) is called the set of feasible link rates. The objective of fair bandwidth sharing is to find a rate vector in Ω that satisfies some fairness property. According to [13], [19], maximizing a concave utility function of the rates can guarantee fairness in general. This is due to the so called diminishing return property enforced by concave non decreasing utility functions.

Let $f_i(x_i)$ be a strictly concave non decreasing utility function for link i. The objective of fair bandwidth sharing in ad hoc networks can be described as:

$$\text{maximize} \qquad \sum_{i=1}^{N} w_i f_i(x_i), \tag{3}$$

where $w_i(>0)$ is a positive constant. In general, each link can use a different utility function. However, if the links choose the same utility function, then w_i can be treated as an additional parameter for differentiating the links.

Taken together, (1), (2) and (3) form the system model of the fair bandwidth sharing problem in ad hoc networks.

Let $Q(i)$ be the set of cliques that include a link i, and let $S(j)$ be the set of links that form clique j. The fair bandwidth sharing problem in ad hoc networks can be formulated as a constrained maximization problem, (called system problem or primal problem $\boldsymbol{P}$) defined as follows:

$$\begin{aligned} \boldsymbol{P}: \quad \max_{x_i} \quad & \sum_i w_i f_i(x_i) \\ \text{subject to:} \quad & \sum_{i\in S(j)} x_i \le c_j, \qquad j=1,\cdots,M \\ & x_i \ge 0, \qquad i=1,\cdots,N. \end{aligned} \tag{4}$$

First consider the feasible set Ω. Since the constraints are linear inequalities, and link rates are non-negative and upper bounded by the capacity of the largest maximal clique to which the links belong, we can easily show that Ω is a nonempty, convex and compact set. Moreover, the objective function $\sum_i f_i(x_i)$ is strictly concave, therefore, the problem is a convex optimization problem which admits a unique maximizer [20].

In this formulation, the utility function plays an important role in determining the target fairness objective. As shown in [13], by adopting a function of the form:

$$f_\alpha(x) = \begin{cases} logx, & \text{if } \alpha = 1 \\ (1-\alpha)^{-1}x^{1-\alpha}, & \text{otherwise}, \end{cases} \tag{5}$$

we can achieve system optimal fairness for $\alpha = 0$; proportional fairness for $\alpha = 1$; harmonic mean fairness for $\alpha = 2$; and max-min fairness for $\alpha \to \infty$.

Solving problem $\boldsymbol{P}$ requires coordination of possibly all the links in the network, which is not practical in ad hoc networks. In order to derive a practical distributed algorithm, we invoke Lagrangean relaxation and duality.

III. Distributed Algorithm

Problem $\boldsymbol{P}$ is convex, thus, we can invoke Lagrangean relaxation and duality to solve the dual problem and recover the solution of the primal with no duality gap.

Consider the Lagrangean $L(\boldsymbol{x}, \boldsymbol{\lambda})$ of system problem P with respect to the inequality constraints. We have:

$$L(\boldsymbol{x}, \boldsymbol{\lambda}) = \sum_i w_i f_i(x_i) + \boldsymbol{\lambda}^t(\boldsymbol{C} - \boldsymbol{A}\boldsymbol{x}), \tag{6}$$

where $\boldsymbol{\lambda}$ is a vector of Lagrange multipliers. Let $\mathfrak{L}(\boldsymbol{\lambda}) = \max_{\boldsymbol{x}} L(\boldsymbol{x}, \boldsymbol{\lambda})$. The dual problem can be defined as:

$$\boldsymbol{D}: \qquad \min_{\boldsymbol{\lambda}\succeq \boldsymbol{0}} \mathfrak{L}(\boldsymbol{\lambda}).$$

The dual function is separable in $\boldsymbol{x}$, thus we can write

$$\mathfrak{L}(\boldsymbol{\lambda}) = \sum_i \max_{x_i}(w_i f_i(x_i) - x_i \sum_{j\in Q(i)} \lambda_j) + \sum_j \lambda_j c_j. \tag{7}$$

In (7), Lagrange multiplier λ_j can be considered as the price to pay per unit of bandwidth in clique $j, j \in \mathcal{M}$, when the total offered link rate to clique j is $\sum_{i\in S(j)} x_i$. Alternatively, assuming a fixed price λ_j, the optimal link rate vector can be obtained by solving the following problem for each link i:

$$\max_{x_i}(w_i f_i(x_i) - x_i \sum_{j\in Q(i)} \lambda_j). \tag{8}$$

Since function $f_i(x_i)$ is strictly concave non decreasing and is differentiable, the unique maximizer for problem (8) exists, and is given by:

$$x_i^* = f_i'^{-1}\left(\frac{\sum_{j\in Q(i)} \lambda_j}{w_i}\right), \quad i=1,\cdots,N. \tag{9}$$

So, for given values of λ_j, $j \in Q(i)$ each link i is able to calculate its own flow rate distributively. Solving the dual problem is done iteratively to obtain the optimal value of $\boldsymbol{\lambda}$ for problem $\boldsymbol{D}$. Since the objective function is continuous and twice differentiable, we can invoke the gradient projection method to solve the dual problem. In this method, in step $k+1$, a new price λ_j is calculated for clique j as:

$$\lambda_j(k+1) = max\left(0, \lambda_j(k) - \gamma(c_j - \sum_{i\in S(j)} x_i^*)\right) \tag{10}$$

where γ is a given step size, c_j is the capacity of clique j, $\sum_{i\in S(j)} x_i^*$ is the total rate of all links in clique j in the previous round. Intuitively, (10) implies the basic requirement of supply and demand: if the total offered link rate to a clique is less (respectively more) than the capacity of the clique, the price decreases (respectively increases).

In practice, with scheduled MAC protocols it is necessary to know the exact value of c_j in order to calculate the exact flow rates; with contention based protocols such as CSMA-based MAC protocols, link rates are achieved only statistically therefore, it is not necessary to know the exact capacity c_j, and normalizing it results in the same achieved throughput. In addition, in wired networks the router computes iteratively the counterpart of (10) and the sources calculate the counterpart of (9), in ad hoc networks, a clique is only an abstraction of the physical bandwidth resource and has no real centralized control entity. Therefore the clique algorithm in (10) must also be executed distributively by the links that belong to the clique.

The resulting cooperative distributed algorithm is shown in Algorithm 1.

Algorithm 1 Cooperative Gradient-based Algorithm

1: Initially, link i chooses a feasible flow rate $x_i(0)$.
2: Link i collects local link rate information (from its neighbors), constructs its local flow contention graph, and decomposes it into a set of cliques $Q(i)$.
3: Initial price $\lambda_j(0)$ is set for each clique in $Q(i)$. The initial price is a global parameter of the system.
4: In round k, link i calculates its new link rate according to (9).
5: Link i broadcast one hop in the contention graph the new link rate to all its competing links.
6: In round $k+1$, node i calculate clique j's new price according to Equation (10).
7: If the flow contention graph has changed (e.g., due to mobility), go back to 1, otherwise go back to step 1.

We can easily prove that under the appropriate value of the step size γ in (10), for any initial feasible link rate vector $\boldsymbol{x}_0$ and price $\boldsymbol{\lambda}^0$, any accumulation point $(\boldsymbol{x}^*, \boldsymbol{\lambda}^*)$ generated by the algorithm is primal-dual optimal. Therefore, the system is globally stable.

IV. Design and Implementation Issues

In this section we address the issues related to the implementation of such an algorithm in a real system based on the IEEE 802.11 protocol.

To be able to implement the algorithm in a real system each sender node must be able to substitute itself to all its links to i) collect and distribute local link contention information and rates; ii) adapt its rate according to clique prices; and iii) synchronize its operation with other nodes in the neighborhood. Due to such cooperation between neighbors, we call this algorithm the cooperative gradient-based algorithm (CGA).

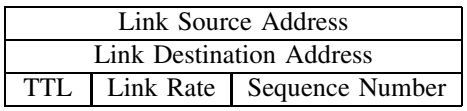
The first step towards implementing the CGA in a real network is therefore to substitute the sender node of each link to the link and to the clique in the execution of the rate algorithm (9) and the clique price algorithm (10). In doing so, many issues that did not exist in the theoretical algorithm arise. We discuss hereafter these issues and propose methods to address them.

A. Local link contention graph issues

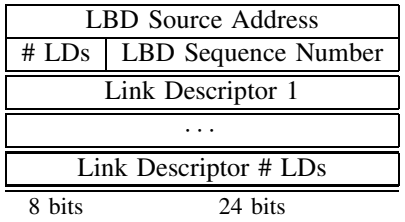
In wireless networks the interference range of a link is roughly determined by: i) the distance between the sender and the receiver; ii) the distance of the interfering nodes from the receiver; and iii) different power thresholds used to validate received data as a frame or dismissing it as noise. Define the transmission range R_{tx} of a tagged node as the radius of the circle centered at the node's location and in which any node can receive a frame transmitted by the tagged node in the absence of interference. More precisely, given the fixed transmit power level P_{tx}, the transmission range R_{tx} is defined as the area in which the received signal power level is larger than a packet receiving threshold κ if there is no interference. Using the two-ray ground propagation model (and ignoring the impact of antennae gains and heights), $R_{tx} \leq (P_{tx}/\kappa)^{1/4}$. Likewise, define the interference range R_I of a tagged receiver as the radius of the circle centered at this receiver and in which any transmitter interferes with an ongoing reception at the tagged receiver. If the link between a transmitter and a receiver spans a distance d, and assuming, for simplicity, that interference is generated by a single node as in [21] (for a more accurate interference model see [22]), then we can easily show that $R_I/d \geq \sqrt[4]{\xi}$, where ξ is the signal to interference and noise ratio (SINR) threshold. In simpler words, if the distance d between transmitter and receiver is such that $d \leq (\xi)^{-1/4} \times R_{tx}$, then $R_I \leq R_{tx}$ and only the nodes that are within the circle of radius R_{tx} centered at the receiver can interfere with the receiver. Such nodes can sense the receiver's CTS frame and thus are silenced by it. Otherwise, if $d > (\xi)^{-1/4} \times R_{tx}$ then $R_I > R_{tx}$ and thus nodes that are outside the circle of radius R_{tx} centered at the receiver can interfere with the receiver. Unfortunately, such nodes are not able to decode the receiver's frames, and a dramatic consequence of this on our algorithm is that nodes in the same clique perceive a different clique size which leads to undesirable behavior of the CGA. This illustrates the need for deploying explicit means to enable nodes to collect information on competing links.

For this purpose we propose to use a protocol for information dissemination in the network. In this protocol, the sender and receiver of each link must declare their links by periodically sending beacons. Such beacons must propagate to a given extent to cover the desired interference range. This can be done by retransmissions of the beacons (implemented in the real system), or by increasing the transmit power of such beacons (not to be confused with control frames). Alternatively, to avoid increasing the interference range with the power increase, data messages and control messages can be separated in two channels; the data channel, used for RTS, CTS, data and ACK frames; and a control channel, used to carry broadcast data frames that contain the beacons or control messages by using the basic CSMA/CA access method. The transmission range R_{tx}^C of the control channel must be set to equal the interference range R_I^D of the data channel. To create these two channels, two methods are possible:

- Physical channel separation: In practice, IEEE 802.11 based networks have at least 3 orthogonal channels. Only one is used at any given time by the current standard. Two orthogonal channels among those available can be used to implement the control and data channels. This approach however requires modifications to the IEEE 802.11 transceivers.
- Logical channel separation: a single physical channel can be used to support two logical channels. The separation can be implemented by a non preemptive priority queueing scheme in each node. This queueing scheme gives higher priority to control messages over data messages intra-node. To enable control messages with a higher inter-nodes priority, a smaller inter-frame space and smaller backoff window are assigned to the control channel. Control messages are broadcast in data frames using the basic access method of IEEE 802.11 DCF



(a) Link Descriptor



(b) Link Batch Descriptor

Fig. 1. Link Information Protocol Messages

(CSMA/CA without RTS/CTS). Aside from being a good tradeoff between complexity and accuracy, this approach can in fact be implemented without modifying existing commodity wireless LAN hardware. Off-the-shelf WLAN chipsets already implement multiple queues at the MAC layer with per-queue, inter-frame spaces, backoff windows, and so on. These parameters are controllable by the device driver on a per frame basis.

B. Local link information dissemination

Link Information dissemination is accomplished by the link information protocol (LIP) which defines and carries two types of messages between neighboring nodes. Each link is represented by a link descriptor (LD) shown in Fig. 1(a). The link descriptor identifies the link by its transmitter and receiver addresses which can be MAC addresses, or simply IP addresses. The LD contains a time-to-live (TTL) field set to 2 to limit the maximum number of hops over which the LD is rebroadcast, an LD sequence number, and a link rate. To reduce communication overhead, LDs are transmitted or retransmitted in batches, similar to the distance vectors in the implementation of RIP in the Internet protocol stack. For this purpose, we define a link batch descriptor (LBD) as shown in Fig. 1(b). The LBD is a message that contains a batch of LDs. The LBD is identified by its source node address, it comprises, a sequence number, a field that indicates the number of LDs followed by the list of LDs.

The pseudo-code of the LIP protocol is shown by Algorithm 2 as a set of event handlers.

C. Link Sending Rate Adjustment

Initially, the nodes may use a preselected sending rate x_0 as defined in Algorithm 1 to access the channel until the algorithm converges. However, in practice, choosing an appropriate initial value is not trivial as it depends on the load of the network. Another more practically interesting alternative is to let the nodes use the legacy IEEE802.11 DCF protocol to exchange data as long as the CGA algorithm did not converge,

Algorithm 2 Link Information Dissemination Protocol (LIP)

Events:

LBD_in Received:
- **Foreach** *LD* in *LBD_in*{
  - Update Link Contention Graph
  - **If** graph changed {
    - Restart *CGA_Timer*;
    - *CGA_mode = False*;}
  - Decrement *LD.TTL*
  - **If** *LD.TTL* > 0 Insert *LD* in *LBD_out*
- }

CGA Timer Elapsed:
- Decompose Link Contention graph into cliques;
- CGA_mode = True

New Link Detected:
- Create *LD*;
- Insert *LD* in *LBD_init*;
- Update Link Contention Graph
- **If** (*Broadcast_timer* not running)
  - *Broadcast_timer.start*;
- *CGA_Timer.start*;
- *CGA_mode = False*;

Link Breakdown Detected:
- Update Link Contention Graph
- Remove *LD* from *LBD_init*;
- **If** isEmpty(*LBD_init*){
  - *Broadcast_timer.stop*
  - *CGA_Timer.stop*;
  - *CGA_mode = False*;}
- else{
  - *Broadcast_timer.start*;
  - *CGA_Timer.start*;
  - *CGA_mode = False*;}

Broadcast Timer Elapsed:
- Broadcast *LBD_out*;
- *LBD_out* $\leftarrow$ *LBD_init*;
- **Foreach** *LD* in *LBD_init*{
  - Update *LBD_init.LD.Sequence_number*;
  - **If** *CGA_mode*{
    - Calculate a new price λ using (10);
    - Calculate a new link rate x^* using (9);
    - Set *LBD_init.LD.Rate* $\leftarrow x^*$;}
  - else Set *LBD_init.LD.Rate* $\leftarrow 0$;
- }

as indicated by the CGA_mode variable in the LIP protocol. In this non-CGA, mode the nodes declare a rate of 0 in their LDs. When the topology steadies, the CGA timer eventually expires, and the algorithm enters the CGA mode. In this mode, the link rate is calculated by the algorithm as a normalized fraction of bandwidth.

How to effectively achieve this bandwidth in a real system is another tedious problem in ad hoc networks. A simple approximation consists in modifying the IEEE802.11 DCF backoff algorithm to assign each station i a success probability that is proportional to its optimal rate x_i^*. A simple approximation to this is to set the sender's contention window CW_i for link i to:

$$CW_i = min(\frac{\delta}{x_i^*} CW_{min}, CW_{max}), \qquad (11)$$

where CW_{min} is the minimal contention window, CW_{max} is the maximal contention window, and $\delta(\geq 1)$ is a scaling factor, which depends on the density of the network neighborhood. This approach has been used in the simulations of the protocol in NS-2 (shown in [7]). In practice, on commodity hardware, it is not possible to modify the backoff algorithm as in most of today's hardware, the MAC protocol is hardwired. Therefore we devised another simple approach to achieving the optimal rates by invoking the so-called credit-based fair queueing algorithm [23]. From many previous simulation studies of the IEEE 802.11 DCF in ad hoc mode (e.g., [1]), we can observe that unfairness only occurs when the total traffic load is too high. When this is the case, collisions result in some stations failing consecutively because of the BEB. To prevent this, yet continue to use the BEB we tune down the traffic load seen by the network: the optimal rates x_i^* are converted into fractions of a predefined super-frame size, and a round-robin scheduler is implemented in each node to schedule the node's local links according to their associated rates. Each link is associated with a credit counter, periodically incremented by x_i^* in each super-frame duration. Access to the channel is attempted only for links whose counters exceed the unit value. The super-frame size is predefined to be the transmission time of the largest possible data frame including RTS, CTS, DATA, ACK and all the overheads incurred by this transmission, including those incurred at the physical layer convergence protocol (PLCP). We can show that if the LIP protocol converges, then this scheduler can achieve throughput values that are in proportion to the x_i^* in the log run. This approach has been implemented in commodity hardware in the network card device driver in our test-bed. Packets that come down from the IP layer to the network device driver are diverted before entering the MAC buffer to a battery of per-link queues that we implemented in the device driver. Each of the queues is associated with one of the links and a credit counter. Once the credit counter of the queue reaches 1, one packet is pushed down from the link queue to the MAC transmit buffer. Note that a more refined version is implemented where the counters count bytes to take into account variable packet sizes. Furthermore, in order to avoid delaying any control frames, LIP control packets (LBDs), and other control packets from other protocols, only IP packets that contain application data (TCP segments and/or UDP datagrams with port numbers other than those known to be used for control messages) are diverted from the normal datapath toward the per-link queues. All control packet (e.g. routing packets, DNS requests) bypass our system of queues and enter directly into the MAC transmit buffer, which gives them a higher priority over application generated data.

D. Operation of the CGA Protocol

The CGA protocol consists of a control plan (C-plan) and a user plan (U-plan). The C-plan implements the control logic and operates the control channel according the LIP protocol described by Algorithm 2. In the C-plan, actions taken by the sender are triggered by one of five events: a new link is created by this sender, an old link has broken down, an LBD is received from a neighbor node, or one of the two timers has expired. To avoid overloading the control channel with a large number of small LDs, updates are triggered by a timer. Exchanges of LBDs between neighbors are carried out periodically and asynchronously, each time the Broadcast_timer expires. Each LBD groups all the LDs of the node, as well as all the LDs with a non zero TTL received from the neighbors of this node.

When an LBD is received, the node extracts the LDs, updates its local link contention graph, decrements the TTL of each LD, and appends all those with non-zero TTL to its outgoing LBD. From the received LDs, the node can determine up to date contention relations by examining the sender address of the LBD, and for each LD, the sender address, the receiver address, the LD TTL, and the LD sequence number.

The link contention graph is also updated whenever a new link is created or an old link has broken down. Routing protocols such as AODV or OLSR rely on a HELLO protocol for neighbor discovery and on timers for stale link invalidation. We are currently integrating our protocol with AODV, and once this is accomplished, this information can be used to detect new links and stale links and to update the local topology information.

When the CGA timer expires, the system enters the CGA mode in which the CGA algorithm is invoked to calculate the rate. To avoid premature transition to the CGA mode, the CGA timer is normally set to be at least one order of magnitude larger than the broadcast timer and it is restarted each time a change in the local topology is detected.

The D-plan logic is simplified as much as possible to avoid stalling the datapath unnecessarily. Whenever a frame is ready for transmission, the node checks the *CGA_mode* state variable (maintained by the C-Plan), to determine how to enqueue the frame: if the *CGA_mode* variable is true, any of the methods described above can be used, for example, the frame is simply appended to one of our per-link queues. Otherwise, if the CGA_mode is false, the frame bypasses our per-link queue in the device driver and immediately enters the MAC transmit buffer.

E. Implementation of the CGA Protocol

The CGA algorithm has been implemented in two parts: the LIP protocol is implemented in the user space as an application layer protocol. Since we have yet to modify a routing protocol such as AODV or OLSR to interact with our LIP protocol, the latter is also in charge of neighbor discovery and link activity monitoring. Communication between peers is done through IP datagrams using raw sockets. The scheduling part of the algorithm is implemented in the kernel space, in the Multiband Atheros Driver for WIFI [24] (MADWIFI) device driver. When the CGA timer elapses in a node, the LIP protocol launches the CGA algorithm, starts to calculate the fair share of each link in which this node is a transmitter, and passes these fair shares to the device driver through the so-called Linux proc file system, which is somehow a window into the current state of the running kernel, and allows user space processes to communicate with, and modify some configuration parameters of, kernel space processes. The fair shares are used by the device driver to schedule access to the channel among the node's local links using the credit based approach.

TABLE I
FAIR SHARE CONVERGENCE VS. TIME: CHAIN TOPOLOGY

Time	5s	10s	15s	20s
Node 1	0.365	0.365	0.341	0.344
Node 2	0.189	0.183	0.170	0.172
Node 3	0.189	0.183	0.171	0.172
Node 4	0.377	0.365	0.339	0.344
Experiment I				
Node 1	0.336	0.336	0.334	0.334
Node 2	0.168	0.168	0.167	0.167
Node 3	0.168	0.168	0.167	0.167
Node 4	0.336	0.336	0.334	0.334
Experiment II				

TABLE II
FAIR SHARE CONVERGENCE VS. TIME: RING (C5) TOPOLOGY

Time	10s	20s	25	30	35	40
Node 1	0.110	0.122	0.132	0.133	0.133	0.133
Node 2	0.110	0.122	0.132	0.133	0.133	0.133
Node 3	0.110	0.122	0.132	0.133	0.133	0.133
Node 4	0.110	0.122	0.132	0.133	0.133	0.133
Node 5	0.110	0.122	0.132	0.133	0.133	0.133
Experiment I						
Node 1	0.129	0.132	0.132	0.133	0.133	0.133
Node 2	0.129	0.132	0.133	0.133	0.133	0.133
Node 3	0.129	0.132	0.133	0.133	0.133	0.133
Node 4	0.129	0.132	0.133	0.133	0.133	0.133
Node 5	0.129	0.132	0.133	0.133	0.133	0.133
Experiment II						

V. PERFORMANCE EVALUATION

I addition to the mathematical algorithms obtained in the previous sections, we have implemented the CGA protocol in the ns-2 [25] simulator as well as in retail wireless LAN network interface cards within the MADWIFI (Multiband Atheros Driver for WIFI) [24]. Since our numerical results and simulations are already available in [7], we focus here only on some measurement results obtained from the implementation to illustrate the convergence time of the the LIP protocol.

Because we only have 5 nodes available (notebooks and desktops enabled with wireless LAN network cards), we have so far tested the implementation with a chain topology of 5 nodes with traffic flowing from node 1 to node 5 (making nodes 1, 2 , 3 and 4 transmitters at the MAC layer), and with a ring topology of 5 nodes (C_5) where all the nodes are transmitters. Each node is a computer running Linux and comprising a retail wireless LAN card (PCI or PCMCIA) that uses Atheros Chipset (the only hardware requirement is that the network card uses this chipset to be supported by the MADWIFI driver). A list of compatible cards can be found in [24]. Note that, to confine the testing to a small area of a few tens of meters, it is necessary sometimes to reduce the transmit power of the wireless LAN cards, making it easy to control the wireless proximity between nodes and form the topology under test accurately.

Table I and Table II illustrate the convergence of the fair share for the string topology and the ring topology, respectively, as a function of time. We set the broadcast timer to 1s, and the CGA timer to 20s, and recorded the time and values of the fair share at 5s, 10s, 15s, and so on, after the CGA timer is triggered. In these scenarios, the utility function used is logarithmic, therefore the fairness principle enforced is the proportional fairness. Under proportional fairness. We can see from Table I for example that the fair shares obtained are very accurate. Indeed, under the proportional fairness, in the chain topology for example, nodes 1 and 4 control links that reside in one clique of degree 3 each, whereas nodes 2 and 3 control links that reside in two cliques of degree 3 each. As such nodes 1 and 4 obtain twice as large a fair share as nodes 2 and 3 do. With the C_5 ring topology the same remarks hold.

TABLE III
CONVERGENCE TIME VS. STEP SIZE γ

Topology	Chain			Ring		
Step size γ	1	2	3	1	2	3
Node 1	11s	6s	6s	40s	31s	25s
Node 2	11s	7s	5s	39s	30s	24s
Node 3	12s	7s	6s	39s	30s	24s
Node 4	11s	6s	5s	40s	31s	24s
Node 5	–	–	–	40s	31s	25s

To speed up convergence, one can increase the value of the step size γ in (10) as discuss previously. Table III shows the convergence time against the value of the step size γ for the two topologies. With larger values of γ, the fair share calculation converges faster.

We have tested the same topologies many times with varying values for the CGA timer, step size and broadcast timer. Although these are only preliminary results, overall we can conclude that if the CGA timer is set to an order of magnitude larger than the broadcast timer, convergence is guaranteed. In our tests for a CGA timer of 5s and a broadcast timer of 1s, the system converged in about 70% to 80% of the experiments. When we increased the CGA timer to 10s and above, the system converged in 100% of the tests. This can be explained by the fact that topology information must propagate 3 hops from one node to another, therefore, taking into account both directions and the possible asynchrony between the nodes in the network, at least 6 rounds of broadcasts are needed to synchronize the topology information. Also, when the traffic load is high it is worthwhile to increase the CGA timer to account for possible collisions and avoid triggering the timer prematurely. Finally, we have tested the system where the clique capacity is systematically reduced to 2/3 versus a system where cliques have normalized capacities. In both cases the ratios of the fair shares of different links were of

the same proportion. For example the fair shares in Table II are obtained with clique capacities reduced to 2/3. When the clique capacities are normalized, the fair shares obtained for the same scenario converged to 0.2 for all nodes in the ring topology. In conjunction with the credit based contention MAC protocol we proposed, this had no major impact on the achieved throughput because nodes are regulated by the credits and also by their collisions.

VI. Conclusion

In summary, we have studied the bandwidth sharing problem as a constrained concave utility maximization problem to guarantee fairness. Lagrangean relaxation and duality are invoked to solve the problem and an iterative gradient-descent algorithm is proposed. This algorithm requires some local cooperation between nodes, which translates into the exchanges of link information between neighboring nodes. We considered implementation issues and illustrated how this simple theoretical result and algorithm can be practically implemented as a protocol in complex real system, and carried out the implementation in the device driver of wireless LAN cards based on the so-called Atheros Chipsets in the Linux environment. Preliminary measurements from the implementation are given to illustrate the effectiveness of the proposed protocol in meeting its targets. Of course such results cannot be taken as conclusive because our tests have been only carried out in small networks and further experiments are needed to ensure the robustness of the protocol.

References

[1] B. Bensaou, Y. Wang, and C. Ko, "Fair media access in 802.11 based wireless ad-hoc networks," in *Proc. ACM MobiHOC 2000*, pp. 99–106.

[2] T. Nandagopal, T. E. Kim, X. Gao, and V. Bharghavan, "Achieving MAC layer fairness in wireless packet networks," in *Proc. ACM MOBICOM 2000*, pp. 87–98.

[3] X. L. Huang and B. Bensaou, "On max-min fairness and scheduling in wireless ad-hoc networks: analytical framework and implementation," in *Proc. ACM MobiHOC 2001*, pp. 221–231.

[4] H. Luo and S. Lu, "A topology independent fair queueing model in ad hoc wireless networks," in *Proc. IEEE ICNP'00*, (Osaka, Japan), Aug. 2000.

[5] L. Bao and J. J. Garcia-Luna-Aceces, "A new approach to channel access scheduling for ad hoc networks," in *SIGMOBILE: ACM Special Interest Group on Mobility of Systems, Users, Data and Computing*, pp. 210–221, 2001.

[6] Y. Qiu and P. Marbach, "Bandwidth allocation in ad hoc networks: a price based approach," in *Proc. IEEE INFOCOM2003*.

[7] Z. Fang and B. Bensaou, "Fair bandwidth sharing algorithms based on game theory frameworks," in *Proc. INFOCOM 2004*.

[8] Y. Xue, B. Li, and K. Nahrstedt, "Optimal resource allocation in wireless ad hoc networks: a price-based approach," *IEEE Trans. Mobile Comput,*, vol. 5, pp. 347–364, April 2006.

[9] L. Chen, S. H. Low, and J. C. Doyle, "Joint congestion control and media access control design for ad hoc wireless networks," in *Proc. IEEE Infocom 2005*.

[10] M. Chiang, "To layer or not to layer: balancing transport and physical layers in wireless multihop networks," in *Proc. INFOCOM 2004*.

[11] F. P. Kelly, A. Maulloo, and D. Tan, "Rate control for communication networks: shadow prices, proportional fairness and stability," *J. Operation Research Society*, vol. 49, pp. 237–252, Mar. 1998.

[12] S. H. Low and D. E. Lapsley, "Optimization flow control–I: basic algorithm and convergence," *IEEE/ACM Trans. Networking*, vol. 7, pp. 861–874, Dec. 1999.

[13] J. Mo and J. Walrand, "Fair end-to-end window-based congestion control," *IEEE/ACM Trans. Networking*, vol. 8, pp. 556–567, Oct. 2000.

[14] A. Bar-noy, A. Mayer, B. Schieber, and M. Sudan, "Guaranteeing fair service to persistent dependent tasks," *SIAM J. Comput.*, vol. 27, pp. 1168–1189, Aug. 1998.

[15] B. Hajeck and G. Sasaki, "Link scheduling in polynomial time," *IEEE Trans. Inf. Theory*, vol. 34, pp. 910–917, Sept. 1988.

[16] L. Massoulié and J. Roberts, "Bandwidth sharing: Objectives and algorithms," *IEEE/ACM Trans. Networking*, vol. 10, pp. 320–328, June 2002.

[17] G. Cornuejols, "The strong perfect graph conjecture," in *Proc. International Congress of Mathematicians*, 2002, Beijing, China.

[18] C. E. Shannon, "A theorem on coloring the lines of a network," *J. Math. Phys.*, vol. 28, pp. 148–151, 1949.

[19] S. Shenker, "Fundamental design issues for the future internet," *IEEE J. Sel. Areas Commun.*, vol. 13, pp. 1176–1188, 1995.

[20] S. G. Nash and A. Sofer, *Linear and Nonlinear Programming*. McGraw-Hill International Editions, 1996.

[21] K. Xu, M. Gerla, and S. Bae, "How effective is the IEEE 802.11 RTS/CTS handshake in ad hoc networks?" in *Proc. IEEE Globecom 2002*.

[22] J. Zhang and B. Bensaou, "Core-PC: a class of correlative power control algorithms for single channel mobile ad hoc networks," *IEEE Trans. Wireless Commun.*, to appear.

[23] B. Bensaou, D. H. K. Tsang, and K. T. Chan, "Credit-based fair queueing (CBFQ): A simple service-scheduling algorithm for packet-switched networks," *IEEE/ACM Trans. Networking*, vol. 9, no. 5, pp. 591–604, 2001.

[24] madwifi, "Multiband atheros driver for wifi (madwifi)," http://sourceforge.net/projects/madwifi.

[25] "The network simulator -ns-2," http://www.isi.edu/nsnam/ns/,2.1b9a.

Brahim Bensaou received an Engineer Degree in Computer Science from the University of Science and Technology Houari Boumediene of Algiers Algeria in 1982, and a DEA degree from University Paris XI in Computer Science in 1988. He earned his Doctorate in Computer Science from the University Paris VI in 1993. From 1990 to 1994, he was a research assistant at France Telecom Research labs, France, where he was involved in the early designs and study of ATM technology. From 1995 to 1997 he was a Research Associate at the Hong Kong University of Science and technology (HKUST). In 1997 he joined the Centre for Wireless Communications in Singapore, a national R&D center (known now as the Institute of Infocomm Research, I2R A-Star) as a Member of Technical Staff (MTS), where he worked as a System Architect on QoS-enabled MAC protocols and scheduling algorithms for wireless ATM networks; then, as Senior MTS, he led a small R&D group in the area of wireless networking at the CWC. Since fall 2000 he is a faculty member in the Department of Computer Science and Engineering at the HKUST. His general areas of research are in QoS-enabled wired/wireless networks, including ad hoc networks sensor networks and wireless LAN, where he has published more than 70 research papers in prominent conferences and journals, received numerous research grants, graduated nearly 20 postgraduate students and invented 3 US patents of which one was licensed. He is an Associate Editor of *IEEE Communications Letters* and is a senior member of the IEEE, and a member of the ACM.

Zuyuan Fang received a B.S. degree in mechanical engineering from Northwestern Polytechnic University, Xian, China, in 1987, and a M.S degree in computer science and engineering from Huazhong University of Science and Technology (HUST), Wuhan, China in 1994. He received his PhD degree in Computer Science from The Hong Kong University of Science and Technology (HKUST), Hong Kong in 2004. Since graduation, he has been a R&D engineer in the Hong Kong Applied Science and Technology Research Institute (ASTRI). Prior to HKUST, He was a lecturer and project leader for the State Key Lab. of Storage Systems at the HUST. His research interests focus on the algorithm design and protocol development for WPAN, WiMAX, and wireless ad hoc networks at MAC layer and above. He is a member of both IEEE and ACM.