Active Queue Management and Scheduling for Wireless Networks: The Single-Cell Case

Peter Marbach and Yiyi Lu
Department of Computer Science
University of Toronto
Email: marbach@cs.toronto.edu

***Abstract*— It is well-known that TCP over IEEE 802.11 can lead to very poor network performance in terms of fairness. In this paper, we study active queue management and scheduling as an approach to overcome this problem. For the single-cell case, we show that the resulting mechanism indeed provides a fair bandwidth allocation among TCP connections.**

I. Introduction

It is well-known that the combination of TCP rate control with the IEEE 802.11 MAC protocol can lead to very poor performance in terms of fairness [1], [2]. By now, it is well-understood what causes this problem and it has been recognized that proper active queue management and scheduling mechanisms are needed to overcome it [1]. The goal of this paper is to provide a systematic approach to the design of these mechanisms.

The design of active queue management and scheduling for wireless networks needs to take into account the unique features of wireless networks (such as interference and the distributed nature of wireless networks). A systematic approach to this problem is still elusive as currently proposed approaches are either too complex, or do not provide clear design guidelines for obtaining good system performance (we will comment on this in more detail in Section VII).

For our analysis, we proceed as follows. We first characterize the properties of a fair MAC protocol for wireless networks. We then use these properties to derive a distributed active queue management and scheduling scheme. We analyze the resulting mechanism for the case where several TCP connections share a single-cell wireless network, and show that it indeed leads to a fair bandwidth allocation. Due to space constraints, we state our analytical results without proofs.

The rest of the paper is organized as follows. In Section II, we illustrate the performance problems of TCP over the IEEE 802.11 MAC protocol. In Section III, we derive the properties that a fair MAC protocol for wireless networks should have. In Section IV and Section V, we derive a distributed active queue management and scheduling scheme, and show that it satisfies the properties of Section III. In Section VI we study the performance of TCP under these schemes for the case of a single-cell wireless network. We discuss related work in Section VII. For our discussion, we assume that the reader is familiar with the basic mechanism of TCP and 802.11; a description of the two protocols can be found in [3].

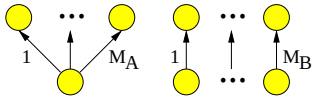

Fig. 1. Single-Cell Network

II. Interaction of TCP and IEEE 802.11

Consider the single-cell network (i.e. all nodes are within transmission range of each other) given by Fig. 1 which consists of two sets of connections, $\mathcal{M}_A = \{1, ..., M_A\}$ and $\mathcal{M}_B = \{1, ..., M_B\}$. We refer to the connections in the set $\mathcal{M}_A$ as subnet A and the connections in set $\mathcal{M}_B$ as subnet B. Note that the connections in subnet A share the same source node: this topology captures the situation where one node acts as a server for all other nodes in subnet A.

Fig. 1 shows the throughput[1] of the individual TCP connections for the case where $M_A = 3$ and $M_B = 6$. Note that even though all TCP connections share the same communication channel, the bandwidth is not evenly distributed among the TCP connections: the connections in subnet B tend to get a higher throughput compared with the connections in subnet A. This is due the fact that 802.11 provides per-node fairness (each node has an equal chance to access the communication channel) but not per-connection fairness (among competing TCP connections).

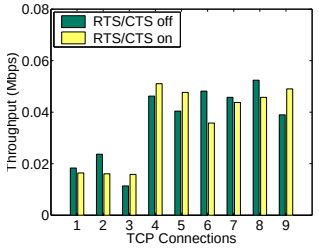

Fig. 2. Bandwidth Allocation among TCP Connections

[1]The results were obtained using ns2 to simulate TCP Reno and IEEE 802.11 with DCF. The channel rate was set 0.8Mbps, the DIFS (Distributed Inter-Frame Spacing) was set to $50\mu s$. All packets (data packets and ACKs) were of the same size with transmission delay $5000\mu s$. The maximal TCP window size was set to 30 packets, and the (MAC layer) buffer sizes at nodes was equal to 31 packets. For the simulation, we created FTP/TCP connections starting at 10s that last for 50s. Unless explicitly mentioned, we used the default settings given by ns2 for our simulations.

III. Properties of a Fair MAC Protocol

In this section, we address the following questions: What property should a MAC protocol have in order to provide fair bandwidth allocation among TCP connections?

To do this, consider a single-cell ad hoc network where a set of N nodes are within transmission range and compete for a common communication channel. Let λ_n be the packet arrival rate to node n. If the buffer at node n is full, then a newly arriving packet is dropped; otherwise the packet is buffered at node n waiting to be transmitted. Let P_n be the probability that a packet is dropped at node n, and let D_n be the expected delay at node n for packets that are not dropped. Furthermore, let $X(\lambda)$ be total network throughput (in packets per unit time) under the arrival rate $\lambda = \sum_{n=1}^{N} \lambda_n$. We then use the following property to characterize a MAC protocol that provides a fair bandwidth allocation. ‘

Property 1: For a single-cell wireless network consisting of nodes $n = 1, ..., N$, we say that a MAC protocol *implements a distributed buffer with service rate* μ if the following is true.

(a) All packets experience the same expected delay, i.e. we have $D_n = D$, $n = 1, ..., N$.
(b) The packet-drop probability is identical at all nodes, i.e. we have $P_n = P$, $n = 1, ..., N$.
(c) The throughput $X(\lambda)$ is an non-decreasing function in λ with $\lim_{\lambda \to \infty} X(\lambda) = \mu$.

The above property states that a fair MAC protocol should serve packets as if the network traffic shares a common buffer that is served at rate μ, i.e. all packets entering the network should experience the same average delay and drop probability.

Applied to the case where the packet arrival rates are controlled by TCP, the above property states that the performance obtained by individual TCP connections in a single-cell wireless network should be the same as in the case where all connections share a single buffer that is served with rate μ. Note that the IEEE 802.11 protocol does not satisfy Property 1, as it does not always divides equally the network throughput among TCP connections that share a single communication channel (see Fig. 1 given in the previous section).

IV. Design of a Fair MAC Protocol

In order to design a MAC protocol that satisfies Property 1, we first propose a centralized approach which we later extend to a fully distributed mechanism.

A. A Centralized Scheduler

Consider the problem of finding a centralized scheduler that satisfies Property 1 where we assume that the scheduler has perfect information about the backlog at individual nodes, but does not have any knowledge about the packet arrival rates. Using backlog information, the scheduler decides which node is transmits the next packet. We consider the following algorithm.

Algorithm 1: Consider a single-cell wireless network with N nodes. If at least one node has a packet ready to be transmitted and there is currently no packet being transmitted, then schedule a packet transmission. When scheduling the next transmission, choose a node at random and independent of past scheduling decisions as follows. The probability that node n is scheduled is given by

$$q_n = \frac{b_n}{B}, \qquad n = 1, ..., N,$$

where b_n, $n = 1, ..., N$, is the current backlog at node n and $B = \sum_{n=1}^{N} b_n$ is the total backlog over all nodes. If node n is scheduled, then it will transmit the packet at the head of its queue.

The intuition behind the above algorithm is that nodes are scheduled proportionally to their backlog. As a result, nodes that have a higher arrival rate (and a higher backlog) are more likely to be scheduled resulting in a fair allocation of the total throughput. To verify this intuition, we characterize the performance of the above algorithm under the assumption that nodes have infinite buffers, packets arrive according to a Poisson process, and service times are exponentially distributed with mean $\frac{1}{\mu}$. The lemma below states that in this case the above centralized algorithm indeed meets Property 1 as it implements a $M/M/1$ queue with rate μ.

Lemma 1: Consider a single-cell wireless network where each node has an infinite buffer, and suppose that packets arrive to node n according to an independent Poisson process with rate λ_n and packet service times are independently and exponentially distributed with mean $\frac{1}{\mu}$. Then Algorithm 1 implements a distributed buffer with rate μ, i.e. the expected delay D is equal to the delay of a $M/M/1$ queue with arrival rate $\lambda = \sum_{n=1}^{N} \lambda_n$ and service rate μ.

B. A Distributed Scheduler

The above centralized algorithm suggests that the probability that a node is scheduled should depend on the current backlog at this node. Using this insight, we consider a distributed algorithm which implements a MAC protocol with backlog-dependent transmission probabilities.

Algorithm 2: Let q, $0 < q < 1$, be a constant which is assumed to be small. Each node n, $n = 1, ..., N$ uses the following algorithm to schedule its packet transmissions.

1) *Channel Sensing:* Before each transmission attempt, node n senses whether the channel is idle (no other node is currently transmitting). If the channel has been sensed to be idle for a duration L_i time units, then the node makes a transmission attempt with probability $q_n = \min\{1 - \epsilon, qb_n\}$, where b_n is the current backlog at node n and $\epsilon > 0$ is a small constant to ensure that the attempt probability is strictly smaller than 1.
2) *Transmission:* After finishing its transmission, node n waits for an ACK from the receiver. If no such ACK is obtained within a fixed period of time, then the node assumes that a collision happened and the packet needs to be retransmitted. If an ACK is obtained, the packet has been transmitted successfully and is removed from the buffer.

Note that the above algorithm implements the well-known CSMA protocol [4] with backlog-dependent transmission

probabilities: the larger the backlog at a node, the more aggressive a node will be in making a transmission attempt. Below we characterize the throughput under this algorithm.

Suppose that the current backlog at node n is equal to b_n such that node n makes transmission attempts with probability qb_n. The probability that at least one node make a transmission attempt after an idle period (and the channel is busy) is given by

$$P_b = 1 - \Pi_{n=1}^{N}(1 - qb_n)$$

and the probability that no node makes a transmission attempt is given by

$$P_i = 1 - P_b = \Pi_{n=1}^{N}(1 - qb_n). \quad (1)$$

Furthermore, the probability that exactly one node makes a transmission attempt (and hence the packet will be successfully transmitted) is given by

$$P_s = \sum_{n=1} \frac{qb_n}{1 - qb_n} \Pi_{m=1}^{N}(1 - qb_m).$$

Assuming that q is small, we can approximate the above probabilities by

$$P_b = 1 - e^{-qB} \quad (2)$$

and

$$P_s = qBe^{-qB},$$

where $B = \sum_{n=1}^{N} b_n$ is the total backlog over all nodes. Note that the probability of having an idle (busy) channel depends on the average numbers of transmission attempts

$$G = qB.$$

We will refer to G as the offered load.

Using the above expression, the instantaneous throughput (in packets per unit time) as a function of the offered load G is given by (see for example [4], [11])

$$X(G) = \frac{Ge^{-G}}{L_i + (1 - e^{-G})L_p}, \qquad G \geq 0, \quad (3)$$

where L_p is the average duration of a packet transmission.

It is well-known (see for example [4]) that the throughput $X(G)$ becomes small as G becomes large, i.e. we have

$$\lim_{G \to \infty} X(G) = 0,$$

and the unique offered load G^+ which maximizes the throughput $X(G)$ given by

$$G^+ = \sqrt{\frac{2L_i}{L_p}}.$$

The above analysis shows that the performance (in terms of throughput) of Algorithm 2 depends on the offered load $G = qB$. In the next section, we propose an active queue management mechanism to stabilize the queue length at a desired operating point B^*.

V. Active Queue Management

In this section we consider an active queue management mechanism that randomly drops incoming packets in order to keep the total backlog B at the desired level B^*. We let the probability that a new packet is dropped (called the packet-drop probability $p(u)$) depend on a congestion signal u. Note that Eq. (2) implies that the larger the backlog, the higher the probability that the channel is busy. Using this intuition, we increase the congestion signal u (and increase the packet-drop probability) when the channel is busy, and decreasing it otherwise. We first consider an approach where the control signal is updated in a centralized fashion which we then extend to a fully distributed implementation mechanism.

A. The Basic Mechanisms

The packet-drop probability $p(u)$ is given by a linear function of the congestion control signal u, i.e. we have

$$p(u) = \begin{cases} \kappa u, & 0 \leq u \leq 1/\kappa, \\ 1, & u > 1/\kappa, \end{cases}$$

where $\kappa > 0$ characterizes the slope of the of the function $p(u)$.

The congestion signal u is computed as follows. After each idle period of length L_i the signal u is additively decreased by a constant $\alpha > 0$, and after each busy period of length L_p the signal u is additively increased by a constant $\beta > 0$. Note that this rule follows the intuition that the congestion signal u should be increased when the channel is busy, and should be decreased if the channel is idle.

Using Eq. (1) and (2), the expected change in the signal u between two idle periods of length L_i is then given by

$$\begin{aligned} \Delta u &= -\alpha P_i + (-\alpha + \beta)P_b \\ &= -\alpha + \beta P_b = -\alpha + \beta(1 - e^{-G}), \end{aligned}$$

where $G(k) = qB$ is the offered load.

Note that for $G^* = \ln\left(\frac{\beta}{\beta - \alpha}\right)$, the expected change in the congestion signal is equal to 0, i.e. we have

$$-\alpha + \beta(1 - e^{-G^*}) = 0.$$

Furthermore for $G < G^*$ we have that

$$-\alpha + \beta(1 - e^{-G}) > 0$$

and for $G > G^*$ we have that

$$-\alpha + \beta(1 - e^{-G}) < 0.$$

This implies that if the offered load G is smaller than G^* then the congestion signal u tends to decrease (and hence the packet-drop probability tends to decrease), whereas for $G > G^*$ the congestion signal u tends to increase (and hence the packet-drop probability tends to increase). As a result, one can show that the above active queue management mechanism stabilizes the offered load at G^*, and the system backlog at $B^* = G^*/q$.

The above equations provides a simple algorithm for choosing the parameters α and β to set system throughput and backlog. Suppose that we wish to achieve a network throughput

equal to $X(G^*)$, $0 < G^* \leq G^+$, and the system backlog equal to B^*. Then we need to do the following. Choosing $\beta > 0$, set α equal to

$$\alpha = \beta(1 - e^{-G^*}) \tag{4}$$

and q equal to $q = \frac{G^*}{B^*}$. We have the following result.

Lemma 2: Consider a single-cell wireless network and suppose that packets arrive to node n according to an independent Poisson process with rate λ_n. Then the above active queue management mechanism, together with the scheduling Algorithm 2, implements a distributed buffer with rate $X(G^*)$. Proving requires to model and analyze the dynamics of the congestion signal u as well as the backlog at each node n. The proof follows a similar argument as given in [5].

B. A Distributed Implementation

The above implementation assumes a single congestion signal u that is common to all nodes. Here, we consider an approach to compute u in a fully distributed way. Each node n computes a congestion signal u_n in the same way as described above, i.e. after each idle period of length L_i the signal u_n is decreased by a factor $\alpha > 0$ and after each busy period of length L_p the signal u_n is increased by a factor $\beta > 0$. Furthermore, odes periodically exchange their congestion signals. The exchange could be done by piggybacking the congestion signal on data packets that the node transmits. Using the broadcast nature of the wireless channel, all other nodes will then receive this congestion signal (given that no collision occurred and the transmission was successful). Each node collects the congestion signal of all other nodes, and uses the aggregated congestion signal $U_n = \sum_{n=1}^{N} u_n$ to set the packet-drop probability for packets arriving at node n.

Ignoring delay in the exchange of congestion information, all nodes will use the same aggregated congestion signal $U_n = U = \sum_{n=1}^{N} u_n$ and hence use the same probability $p(U)$ to drop incoming packets. In practice, there will be a delay in the exchange of the congestion signal between nodes - we will investigate this issue in next section.

VI. TCP in a Single-Cell Network

In this section, we study the interaction of the above distributed active queue management and scheduling mechanism with TCP Reno rate control in single-cell wireless network.

A. Numerical Results

We start out by illustrating the above distributed active queue management and scheduling mechanism for a single-cell wireless network where we assume that all packets (data packets and ACKs) are of the same size. Setting $L_i = 1$ and $L_p = 100$, the parameters G^*, α, β, and q, are given as follows. The offered load G^* is chosen to be

$$G^* = G^+ = \sqrt{\frac{2L_i}{L_p}} = 0.1414$$

in order to maximize system throughput. Using Eq. (4), G^* is obtained by setting $\alpha = 0.1319$ and $\beta = 1$. The constant q is chosen to be equal to $q = 0.05/16$. The packet-drop probability $p(u)$ was set equal to

$$p(u) = \in \{0.002u, 1\}.$$

The constant ϵ in the scheduling Algorithm 2 is set equal to $\epsilon = 0.01$. For TCP Reno, we set maximal TCP window size was set to 30 packets, and the (MAC layer) buffer sizes at nodes was equal to 31 packets. Using this setup, We simulated the system for 200,000 time-steps of length $L_i = 1$.

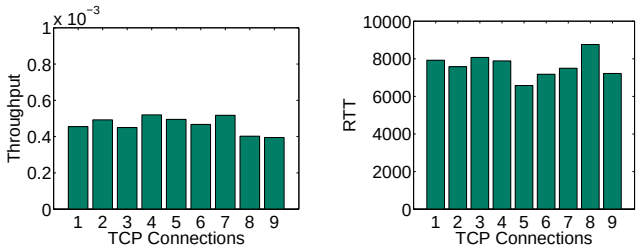

Fig. 3. Time-Average Throughput and Round-Trip Time (RTT) for $M_A = 3$ and $M_B = 6$

Note that each TCP connection consists of two flows: the flow of data packets from the source to the destination and the flow of ACK's from the destination to the source. As a result, the network throughput is twice the total TCP throughput (sum over all connections), i.e. the predicted total TCP throughput under G^+ is equal to $X(G^+)/2 = 0.0043$.

We again consider the network topology given by Fig. 1 with $M_A = 3$ and $M_B = 6$. Fig. 3 shows the throughput, and the round-trip time (RTT), of the individual TCP sessions. Compared with the results in Section II, the bandwidth allocation is indeed more balanced. The time-average throughput was equal to 0.0042 which close to the predicted throughput of $X(G^+)/2 = 0.0043$. The time-average packet-drop probability at all nodes was equal to 0.92. These results suggest that the above active queue management and scheduling mechanism indeed implement a fair MAC protocol as defined by Property 1, i.e. all TCP connections obtain (roughly) the same performance in terms of throughput, delay, and packet-drop probabilities.

Through additional simulations, we observed that dimensioning the buffer size B^* is important in order to obtain good performance in terms of throughput, and that the buffer size should be of the order $O(N)$ where N is the number of interfering nodes. Fig. 4 shows the result of this rule applied to the case where the number of TCP connections in subnet A is kept constant ($M_A = 3$) while the number of TCP connections in subnet B increases (see Fig 1), i.e. the transmission attempt constant q is set equal to

$$q = \frac{0.05}{4 + 2M_B}.$$

Note that the total throughput stays constant as M_B increases and that the queue length at the virtual buffer (i.e. the total backlog) increases roughly linear in M_B as predicted by the above analysis. There is not significant decrease in the total

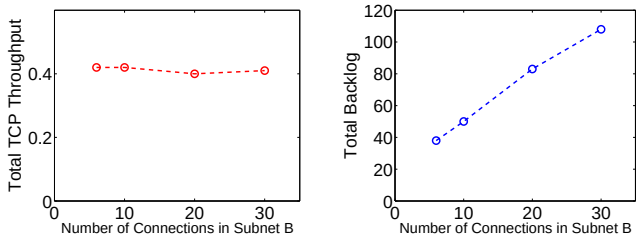

Fig. 4. System Throughput (top) and Backlog (bottom) as a Function of M_B.

throughput as M_B increases, even though an increase in the number of nodes in subnet B increases the delay in the price-updates as mentioned in the previous section.

B. Performance Modeling

The above experimental results suggest that the distributed active queue management and scheduling mechanism given by Algorithm 2 indeed leads to a fair bandwidth allocation among the TCP sessions. A natural question to ask is whether this results can be verified analytically and be shown to be true in general for single-cell wireless networks. This is indeed possible following a similar approach as used to describe the performance of TCP Reno in wireline networks. A detailed derivation of these results is beyond the scope of this paper and we only provide an outline of the necessary analysis.

We can model the above network using the same approach that was used by Kelly in [6] to model TCP Reno in wireline networks. For a connection $m \in \mathcal{M}$, let $w_m(t)$ be the window size (in terms of packets) of connection m during time slot t, $t \geq 0$, and let D_m be the equilibrium round trip delay. Furthermore, let

$$x_m(t) = w_m(t)/D_m, \tag{5}$$

be the transmission rate (in terms of packets per time slot) of connection m. TCP Reno uses packet loss as a congestion indicator, where window size w_m is increased by $\frac{1}{w_m}$ for each acknowledged packet and halved for each packet that is not acknowledged. Ignoring delay in the exchange of congestion signals between nodes, the expected change in the window size w_m is then given by

$$x_m(t)\frac{1-p(U(t))}{w_m(t)} - x_m(t)p(U(t))\frac{1}{2}w_m(t), \tag{6}$$

where $U = \sum_{n=1}^{N} u_n(t)$ is the aggregated congestion signal as defined in Section V.

As shown in [6], the expected rate of change in the transmission rate λ_m at time t is given by

$$x_m(t+1) = \left[x_m(t) + \frac{1-p(U(t))}{D_m^2} - \frac{1}{2}p(U(t))x_m^2(t)\right]^+.$$

To characterize the performance, we consider the operating point of the above system, i.e. the values of $x_m^*, m = 1, ..., M$, and U^* such that

$$\frac{1-p(U^*)}{D_m^2} - \frac{1}{2}p(U^*)(x_m^*)^2 = 0,$$

and

$$\sum_{m=1}^{M} x_m^* = X(G^*)/2$$

where $X(G^*)$ is the throughput under the overload G^* as characterized in Section IV-B. The factor 2 in the above constraint on the total transmission rate accounts for the fact that each TCP connection consists of two flows: the flow of data packets from the source to destination and the flow of ACK's from the destination to the source. As a result, the total bandwidth used by the TCP connections is twice the sum of the transmission rates.

Using the above relation, the performance can be characterized as follows. The TCP throughput $(x_1^*, ..., x_M^*)$ at the operating point is equal to the unique optimal solution to the following optimization problem (see also [6]),

$$\begin{aligned} &\max \sum_{m=1}^{M} \frac{\sqrt{2}}{D_m} \arctan\left(\frac{x_m D_m^2}{2}\right) \\ &s.t. \sum_{m=1}^{M} x_m \leq X(G^*)/2, \\ &x_m \geq 0, m = 1, ..., M. \end{aligned}$$

The above model states that the the bandwidth allocated to connection m depends on the round-trip time D_m. For a single-cell network, one can show that under the above active queue management and scheduling scheme all TCP sessions have the same round-trip time and the optimal solution to this optimization problem is given by

$$x_m^* = \frac{X(G^*)}{2M}, \qquad m = 1, ..., M,$$

resulting in a fair allocation of the bandwidth, as illustrated by the numerical results of the previous subsection.

VII. Related Work

Below we highlight the literature that is most relevant to the work presented in this paper.

It has long been recognized that active queue management plays a crucial role in improving TCP throughput in wireline networks. There are two basic approaches to active queue management in wireline networks: Random Early Detection (RED) [7] and BLUE [8]. RED uses queue length information to determine the packet-drop probabilities: as the queue size at a node increases, the drop-probability increases, hence giving sources an early congestion indication. BLUE, on the other hand, uses packet loss and link idle time (utilization) to set the packet-drop probabilities. It has been shown in [8] that BLUE tends to perform better than RED. It is interesting to observe that the active queue management mechanism used here is similar to BLUE, as it uses observation on channel utilization to set the packet-drop probabilities.

The approach that we pursue in this paper is in spirit very similar to the work presented in [1], which was one of the first papers to recognize that distributed active queue management plays a key role in providing TCP fairness in

wireless networks. In [1], an active queue management scheme called Neighborhood RED (NRED) was proposed, which was implemented on the top of the IEEE 802.11 MAC protocol. In NRED, nodes use the channel utilization to estimate the queue size over all nodes in a given neighborhood (interference region) and to determine a packet-drop probabilities that should be used in this neighborhood (interference region). Through numerical results, it was shown that NRED works well even though it does not alway predict accurately the actual queue size, i.e. after the backlogged packets in a neighbor reach a certain threshold, then the queue size estimate does not increase further. Using a similar analysis as presented in Section IV and V, this behavior can be explained: as the channel access probabilities under 802.11 does not depend on the backlog at a node, it can be shown that NRED does not estimate the queue-size of the virtual buffer, but instead estimates the number of nodes in the interference region that are trying to transmit a packet. As a result, it is not possible for NRED to set the protocol parameters in a systematic way in order to obtain a desired queue-length of the virtual buffer. The approach presented here is able to overcome this issue.

Another approach to the design of active queue management and scheduling in wireless networks is through a utility maximization problem with constraints on the transmission rates (rate control) and channel access probabilities (scheduling) (see for example [9], [10]. Active queue management arises in this context naturally in the form of Lagrange multipliers that enforce the rate and scheduling constraints. This approach is very elegant and lucid, as it makes the interaction between rate control (transport layer) and scheduling (MAC layer) transparent. One drawback of this approach is that it abstracts out many of the practical aspects/constraints of protocol design for wireless networks and lead to solutions that tend to be too complex to be implemented in practice.

The approach presented here also shares some common ideas with the MAC protocol Idle Sense [11], where observations of the channel utilization are used to set contention window of 802.11. While Idle Sense addresses and overcomes several issues of 802.11 at the MAC layer, it does not improve 802.11 with respect to TCP fairness, i.e Idle Sense as a MAC protocol does not satisfy Property 1 given in Section III. The reason for this is that in Idle Sense (as it is the case for 802.11) (a) the channel access probabilities do not depend on the current backlog at a node and (b) the channel access probabilities are decreased as channel contention increases.

VIII. Conclusions and Future Work

We presented a systematic approach to the design of active queue management and scheduling schemes in wireless networks which leads to simple, distributed protocols. Through analysis and experimental results, we showed that the resulting distributed algorithms lead to fair bandwidth allocation among TCP connections. Another important aspect of the approach is that it can be easily used to set the queue length at the (virtual) buffers, which is important to ensure a high TCP throughput.

In this paper, we only considered the case of a single-cell wireless network, however the above approach can be extended to multihop networks. This can be done by associating with each node a "distributed buffer" and adapt the active queue management mechanism of Section V to account for interference between nodes that are not within transmission range of each other. A discussion of this case will be presented in a forthcoming paper.

Another aspect that has not been fully explored in this paper is how to dimension the queue length at the virtual buffer. Experimentally, we observed that the queue length should be proportional to the number of nodes in an interference region (see Section VI). However, in the light of recent advances in the understanding of buffer dimensioning for wireline networks [12], it might be possible to derive rigorous guidelines for choosing an optimal queue length.

References

[1] K. Xu, M.Gerla, L. Qi and Y. Shu, "Enhancing TCP Fairness in Ad Hoc Wireless Networks Using Neighborhood RED," In Proceedings of MOBICOM 2003

[2] C. Chaudet, D. Dhoutaut, and'I. Lassous, "Experiments of some Performance Issues with IEEE 802.11b in Ad Hoc Networks," in Proceedings of WONS 2005.

[3] J. F. Kurose ancd K. W. Ross, "Computer Networking: A Top-Down Approach Featuring the Internet," Addison Wesley, 2005.

[4] D. Bertsekas and R. Gallager, *Data Networks*, 2nd ed. Prentice-Hall, Inc., 1992.

[5] C. Yuen and P. Marbach, "Price-Based Rate Control in Random Access Networks," IEEE/ACM Transactions on Networking, 13 (5), Oct. 2005.

[6] F. Kelly, "Mathematical modelling of the Internet," Fourth International Congress on Industrial and Applied Mathematics, pp. 105–116, July 1999.

[7] S. Floyd and V. Jacobson, "Random Early Detection Gateways for Congestion Avoidance," IEEE/ACM Transactions on Networking, 1(4), Aug. 1993.

[8] W. Feng, D. Kandlur, D. Saha, and K. Shin, "The Blue Queue Management Algorithms", IEEE/ACM Transactions on Networking, 10(4), August 2002.

[9] Y. Xue, B. Li, and K, Nahrstedt, "Optimal Resource Allocation in Wireless Ad Hoc Networks: A Price-based Approach," IEEE Transactions on Mobile Computing, 2005.

[10] L. Chen, S. Low, M. Chiang, and J. Doyle, "Jointly Optimal Congestion Control, Routing, and Scheduling for Wireless Ad Hoc Networks, In Proceedings of IEEE INFOCOM 2006.

[11] M. Heusse, F. Rousseau, R. Guillier, and A. Duda, "Idle Sense: an Optimal Access Method for High Throughput and Fairness in Rate Diverse Wireless LANs, Proceedings of ACM SIGCOMM, 2005.

[12] M. Enachescu, Y. Ganjali, A. Goel, N. McKeown, and T. Roughgarden, "Routers with Very Small Buffers," Proceedings of the IEEE INFOCOM, Barcelona, 2006.