Distributed Scheduling and Active Queue Management in Wireless Networks

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

***Abstract*—We propose a distributed scheduling and active queue management mechanism for wireless ad hoc networks. The approach is based on a random access scheduler where the transmission attempt probabilities depend on the local backlog. The resulting mechanism is simple and can be implemented in a distributed fashion. The performance of the resulting protocol can be modelled as a utility maximization problem to establish that it indeed leads to a high throughput and fair bandwidth allocation.**

I. Introduction

Currently, there is considerable interests the design of distributed scheduling algorithm for wireless networks which maximize network throughput subject to given fairness constraints. In this paper, we propose and analyze a scheduling mechanism based on a random access protocol with active queue management, where the probability that a node makes a transmission attempt depends on the local backlog. The resulting mechanism is simple and can easily been implemented in a distributed manner. The performance of the proposed mechanism can be modelled by a utility maximization problem to establish that it indeed leads to high throughput and a fair bandwidth allocation.

Related to our approach, in [1] the IEEE 802.11 MAC protocol was combined with an active queue management scheme called Neighborhood RED (NRED) to improve the fairness among TCP flows. NRED uses the channel utilization to estimate the total backlog in an interference region and to determine a packet-drop probabilities. As pointed out in [1], NRED is not guaranteed to accurately track the actual backlog in an interference region and no performance guarantees can be given. The approach presented here is able to overcome this problem by replacing IEEE 802.11 protocol with a CSMA mechanism with a backlog-dependent packet transmission probabilities. More recently, scheduling and bandwidth allocation mechanisms have been obtained by the means of solving a utility maximization problem (see for example [3], [4], [5]. 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, but has the drawback that it leads to solutions that tend to be too complex to be implemented in practice.

Due to space constraints, we state our results without proofs. A preliminary version of the paper has been presented in [2].

II. Properties of a Fair Scheduler

Consider a single-cell ad hoc network where a set of N nodes are within transmission range of each other and thus share a common communication channel. If a node transmits a packet, then this transmission is successful if it does not overlap with a transmission by another node in the network. If a packet transmission collides (overlaps) with transmission of another node, then the packet is lost and needs to be retransmitted.

Suppose that each node has a single buffer. Let λ_n be the packet arrival rate to the buffer at node n; let D_n be the expected delay of a packet at node n, and let P_n be the probability that a packet is dropped at node n due to a buffer-overflow. For

$$\lambda = \sum_{n=1}^{N} \lambda_n,$$

let $X(\lambda)$ be the network throughput under the network arrival rate λ. We are interested in schedulers with the following property.

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

(a) The expected delay D_n is identical at all nodes, i.e. we have $D_n = D$, $n = 1, ..., N$.
(b) The packet-drop probability P_n 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 scheduler 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 expected delay and the same probability of being dropped.

A. Centralized Scheduler

We first consider a centralized scheduler that satisfies Property 1. We assume that the scheduler has perfect information about the backlog at each node, but does not have any knowledge about the packet arrival rates.

Algorithm 1: Consider a single-cell wireless network with N nodes. If at least one buffer has a packet ready to be transmitted and there is currently no packet being transmitted,

then initiate a new transmission by scheduling node n with probability

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

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 local queue.

The above algorithm schedules nodes proportionally to their current backlog, hence nodes with a high arrival rate (and a higher backlog) tend to be scheduled more often resulting. We have the following result.

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 that 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 expected delay at a $M/M/1$ queue with arrival rate $\lambda = \sum_{n=1}^{N} \lambda_n$ and service rate μ.
Lemma 1 states that when the packet arrival process is Poisson and the service rates are exponentially distributed, then the above scheduler satisfies Property 1.

B. 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 scheduler 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.

We will refer to q as the *transmission attempt constant.*

The above algorithm implements a CSMA protocol [6] 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 expected number of transmission attempts after an idle slot is then given by

$$G = qB \tag{1}$$

where $B = \sum_{n=1}^{N} b_n$ is the total backlog over all nodes. We will also refer to G as the offered load.

Assuming that q is small, it is well-known that the instantaneous throughput (in packets per unit time) is a function of the expected number of transmission attempts G and is given by (see for example [6])

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

where L_p is the average duration of a packet transmission.

In the following, we assume that $L_p = 1$ and the instantaneous throughput $X(G)$ is given by

$$X(G) = \frac{Ge^{-G}}{L_i + (1 - e^{-G})}, \qquad G \geq 0, \tag{2}$$

One can show that there exists an optimal offered load G^+ which maximizes the throughput $X(G)$ and that the throughput $X(G)$ becomes small as G becomes large. It is well-know that the optimal value G^+ is given by

$$G^+ = \sqrt{2L_i},$$

and that

$$\lim_{L_i \to 0} X(G^+) = 1,$$

i.e. if the duration of an idle slot L_i becomes small the throughput $X(G^+)$ is equal to the optimal throughput 1 (see for example [6]).

III. Active Queue Management

The performance (in terms of throughput) of Algorithm 2 depends on the offered load G. In order to achieve a high throughput, we use an active queue management mechanism that randomly drops incoming packets in order to keep the expected number of transmission attempts G at the desired level G^*. We let the probability that a new packet is dropped (called the packet-drop probability $p(u)$) depend on a congestion signal u.

A. The Basic Mechanisms

Consider the packet-drop probability $p(u)$ given by

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

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 the signal u is additively decreased by a constant $\alpha > 0$ and after each busy period 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 be decreased when the channel is idle.

One can show that the probability P_b that at least one node makes a transmission attempt after an idle slot is given by $P_b = 1 - e^{-G}$ (see for example [6]). The expected change Δu in the signal u between two idle periods of length L_i is then equal to

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

Let G^* be given by

$$G^* = \ln\big(\frac{\beta}{\beta-\alpha}\big). \tag{3}$$

Note that for $G = G^*$, the expected change in the congestion signal is equal to 0, i.e. we have

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

Furthermore, it can be shown 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). It follows that G^* is the unique operating point and the above active queue management mechanism will stabilize the offered load at G^*.

Eq. (3) provides a simple way for setting the system throughput. Suppose that we wish to set the rate of the virtual buffer equal to $X(G^*)$, $0 < G^* \leq G^+$, and the system backlog equal to B^*. This can be achieved by choosing $\beta > 0$ and set α equal to

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

In addition, using the relation $G = qB$ given by Eq. (1), we can also set the system backlog at a desired level B^* by setting q equal to

$$q = \frac{G^*}{B^*}. \tag{5}$$

We have the following result which states that the above distributed scheduling and active queue management algorithm satisfy Property 1.

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 $X(G^*)$.

IV. TCP Performance 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. For our analysis, we model the above network using the same approach that was used by Kelly in [7] to model TCP Reno in wireline networks. Suppose that a set $\mathcal{M} = \{1, ..., M\}$ connections share single-cell wireless network. For 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

$$x_m(t) = w_m(t)/D_m, \tag{6}$$

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

$$x_m(t)\frac{1-p(U(t)))}{w_m(t)} - x_m(t)p(U(t))\frac{1}{2}w_m(t), \tag{7}$$

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

As shown in [7], 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 offered load G^* as characterized in Section II-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.

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 D and the above optimization problem is given by (see also [7])

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

The optimal solution to this optimization problem is given by

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

indeed resulting in a fair bandwidth allocation among the TCP connections.

V. Scheduling and Active Queue Management in a Multihop Network

In this section, we extend the above mechanism to multihop networks. To do that, we have to extend the notion of a "distributed buffer" to the context of a multihop network, as well as adapt the active queue management mechanism of Section III to account for interference between nodes that are not within transmission range of each other.

A. Interference Region

For a multihop network, we associate a distributed buffer with the interference region of each node: the packet arrival rate to this buffer is equal to the sum of the packet arrival rates over all nodes in the interference region, and the queue size of the buffer is equal to the sum of the backlog over all nodes in the interference region. The interference region includes all 1-hop neighbors of the node, i.e. all nodes within transmission range of the given node. In addition, the interference region includes the 2-hop neighbors which indirectly interfere with the node: when a 2-hop neighbor is transmitting then it will prevent the node from making a transmission attempt as this will be detected as a collision at their common neighbor.

Definition 1: The interference region H_n of node n consists of the node itself plus all its 2-hop neighbors.

In the following, we assume that each node can sense whether a node in its interference region is currently transmitting. This could be achieved through the use of a *busy tone*. Whenever a node senses a transmission by a node in its 1-hop neighborhood (transmission range) then it starts sending a busy tone signal in a frequency band that is separate from the packet transmission's. When a node senses the channel to be busy (by a transmission in its 1-hop neighborhood) or detects a busy tone (triggered by a transmission in its 2-hop neighborhood), then it will not make a transmission attempt, thus avoiding a potential collision.

B. Scheduling

Using the busy tone, a node will sense an idle channel only if all nodes in its interference region are idle. If the channel is idle, the following algorithm is used to schedule a packet transmission.

Algorithm 3: 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 transmission.

1) *Channel Sensing:* Before each transmission attempt, node n senses whether the channel is idle (no other node makes a transmission). If the channel has been 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.

C. Distributed Active Queue Management

To set the packet-drop probabilities, each node follows the following algorithm.

Algorithm 4: Each node n, $n = 1, ..., N$, keeps a list of the congestion signals of the nodes in its interference region H_n.

- *Computation of Congestion Signals*: After each idle period of length L_i, nodes n decreases its signal u_n by a factor $\alpha > 0$. After each busy period of length L_p, node n increases its congestion signal u_n by a factor $\beta > 0$.
- *Exchange of Congestion Signals*: Whenever a node transmits a packet, it piggybacks its congestion signal u_n, as well as the congestion signals of all its 1-hop neighbors on the packet transmission.
- *Collection of Congestion Signals*: Whenever a node overhears a successful transmission, it uses the obtained congestion signal to update the congestion signal of the nodes in its interference region.
- *Packet-Drop Probability*: Each node forms an aggregated congestion signal
$$U_n = \sum_{n' \in H_n} u_{n'}$$
and drops incoming packets with probability $p(U_n)$, where the function $p(U_n)$ for the packet-drop probability is as given in Section III.

Let G_n be the offered load in the interference region of node n (expected number of transmission attempts after an idle slot in interference region of node n) and let
$$G^* = \ln\left(\frac{\beta}{\beta - \alpha}\right).$$
The same analysis as given in Section III shows that when $G_n < G^*$, then node n will decrease its congestion signal u_n, and vice versa. Thus, each node tries to stabilize the expected number of transmission attempts G_n in its interference region at G^*. In order to achieve this, all nodes in the interference region of node n should react to the congestion signal u_n, i.e. the packet-drop probability of a node n' in the interference region of node n needs to include u_n. This is the reason why nodes need to know all congestion signals in their interference neighborhood.

D. Asymptotic Throughput Analysis

When node n stabilizes the offered load G_n in its interference region at G^*, the fraction of time T_n that exactly one node $n' \in H_n$ transmits a packet is give by (see also Section II-B)
$$T_n = \frac{G^* e^{-G^*} L_p}{L_i + (1 - e^{-G^*}) L_p}.$$
Note however that T_n is not equal to the fraction of time that a node $n' \in H_n$ makes successfully transmits a packet as (a) a transmission by a node $n' \in H_n$ can collide with a packet

transmission by a node outside the interference region of node n and (b) it is possible for two (or more) nodes in the 2-hop neighborhood of node n to simultaneously transmit packets without causing a collision if the transmission do not result in a collision at the destination nodes. We have the following result for the throughput $X_n(G^*)$ at node n under the offered load G^*.

Lemma 3: Let N_n be the number of nodes in the interference region of node n. Then we have

$$X(G^*)e^{-G^*} \leq X_n(G^*) \leq X(G^*) + N_n \frac{1 - G^* e^{-G^*}}{L_i \cdots}.$$

The lowerbound in the above lemma accounts for the fact that a transmission by a node $n' \in H_n$ can potentially collide with a transmission with a node outside the interference region of node n (and e^{-G^*} is the probability that this is not the case). The upperbound assumes that all simultaneous transmissions in N_n result in a successful transmissions. Note that the upperbound is not tight as it is based on a very optimistic assumption.

In general, the throughput $X_n(G^*)$ depends on the actual network topology; however, we have the following asymptotic result.

Lemma 4: For $G^* = G^+ = \sqrt{2L_i}$ we have

$$\lim_{L_i \to 0} X_n(G^*) = 1.$$

The above lemma states that in the limit, when the duration L_i of an idle slot is negligible small compared with the duration of a packet transmission, then throughput $X_n(G^+)$ in the interference region of node n is equal to 1. This result is quite remarkable as it implies that the above distributed scheduling and active queue management mechanism can (asymptotically as L_i becomes small) achieves the theoretical optimal throughput of 1.

VI. TCP Performance in a Multihop Network

Consider a multihop networks consisting of the set $\mathcal{N} = \{1, ..., N\}$ of nodes and the set $\mathcal{M} = \{1, ..., M\}$ of TCP connections. Let $A(n)$ be the set of TCP connections $m \in \mathcal{M}$ that pass through node $n \in \mathcal{N}$.

Using Lemma 4, the model of Section IV can be extended to the multihop case, i.e. in the limit as L_i becomes very small, the throughput of individual TCP connections under the above active queue management and scheduling mechanism is modelled by the following optimization problem,

$$\max \sum_{m=1}^{M} \frac{\sqrt{2}}{D_m} \arctan\left(\frac{x_m D_m^2}{2}\right) \qquad (8)$$

$$s.t. \sum_{m \in A(n)} x_m \leq 1/2, \qquad n = 1, ..., N, \qquad (9)$$

$$x_m \geq 0, m = 1, ..., M. \qquad (10)$$

Note that the capacity constraint accounts for the fact that each TCP connection consists of two flows: the flow of data packets from source to destination and the flow of ACKs from the destination to the source. If we assume that ACKs are piggybacked on data packets, then the performance is given by the following optimization problem.

$$\max \sum_{m=1}^{M} \frac{\sqrt{2}}{D_m} \arctan\left(\frac{x_m D_m^2}{2}\right)$$
$$s.t. \sum_{m \in A(n)} x_m \leq 1, \qquad n = 1, ..., N,$$
$$x_m \geq 0, m = 1, ..., M.$$

The is result states that the above scheduling and active queue management mechanism can (asymptotically) be modelled as a utility maximization problem. Moreover, the mechanism is asymptotically optimal under the interference model given by Definition 1 in the sense that the capacity constraint for each interference region is equal to theoretical optimal value throughput 1.

VII. Conclusions

We presented a distributed scheduling and active queue management scheme and provided analytical and experimental results to show that it leads to an efficient and fair bandwidth allocation. Compared with the IEEE 802.11 protocol, the proposed scheduling mechanism only requires a redefinition of the transmission probabilities at individual nodes. This could be done by redefining the contention window size (CW) of the current 802.11 protocol, which only requires changes in the software but not hardware.

For our analysis, the interference region (and the capacity constraint in the optimization problem given by Eq. (8) is given by the top-hop neighborhood of a node. This definition allows a simple implementation as collisions can be detected using a busy tone. However, the definition is not efficient from a performance point of view as it suffers from the exposed-terminal problem. Future work is to investigate approaches to avoid this problem by improving the channel feedback.

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 ACM MOBICOM, San Diego, September 2003.

[2] P. Marbach and Y. Lu, “Active Queue Management and Scheduling for Wireless Networks: The Single-Cell Case”, in Proceedings of Conference on Information Sciences and System (CISS), Princeton, March 2006.

[3] 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, Barcelon, April 2006.

[4] Y. Xue, B. Li, and K, Nahrstedt, “Optimal Resource Allocation in Wireless Ad Hoc Networks: A Price-based Approach,” IEEE Transactions on Mobile Computing, vol. 6, no. 2, pp. 961-970, December 2005.

[5] X. Lin and N. B. Shroff, ”The Impact of Imperfect Scheduling on Cross-Layer Congestion Control in Wireless Networks,” IEEE/ACM Transactions on Networking, vol. 14, no. 2, pp. 302-315, April 2006.
G. Holland and N. Vaidya, “Analysis of TCP Performance over Mobile Ad Hoc Networks,” In Proceedings of ACM MOBICOM, 1999.

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

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