Tuning the TCP Timeout Mechanism in Wireless Networks to Maximize Throughput via Stochastic Stopping Time Methods

George Papageorgiou and John S. Baras

Abstract— **We present an optimization problem that aims to maximize the throughput of a Transmission Control Protocol (TCP) connection between two nodes in a wireless ad-hoc network. More specifically, a persistent TCP connection is established between two nodes that are one hop away in a wireless unslotted Aloha network. The optimization is over the TCP timeout period, i.e. the problem is to find the optimal waiting period before the TCP sender declares a timeout event in the absence of a received acknowledgment for a transmitted packet. The problem is formulated as an optimal stopping problem. In the absence of a tractable analytical solution to the problem, a numerical method is proposed to achieve performance improvement of the system.**

I. Introduction

The performance of TCP over a wireless network is poor. This is because the TCP sender assumes that the reason for a packet loss is congestion in the network. Although this is a reasonable assumption for the Internet, it might not be the case for wireless networks. The successful transmission of a packet over a wireless channel depends on the channel quality. In wireless networks the channel quality exhibits high variability and it often causes unrecoverable errors for the packet at the receiver. In these cases, the TCP receiver cannot acknowledge the received packet and drops it. When packets are dropped due to bursts of errors introduced at the channel level, the TCP sender declares a timeout in the absence of received acknowledgments. When this happens, TCP enters the slow-start phase minimizing its window size, and thus its throughput. Clearly, this is not the best remedy to the problem, since the lost packets are dropped because of a temporary quality degradation at the channel and not because of congestion.

The same situation arises when the channel affects the transmission of the acknowledgments themselves. Typically, in the case of a bidirectional TCP connection, the acknowledgments for the one direction are piggybacked onto the data packets to the opposite direction. Hence, acknowledgments are subject to the effect of the channel as data packets do. Even in unidirectional TCP connections where the acknowledgments are sent to the sender on their own, the bursty nature of the channel errors increases the probability that more than three consecutive acknowledgments are lost and thus a data packet loss is falsely detected at the TCP sender and a timeout is declared.

It makes sense then to try to maximize the TCP throughput by appropriately tuning the timeout interval for each packet the TCP source sends to the network. This tuning should take into consideration the parameters that characterize the operation of the Medium Access Control (MAC) layer as well as the channel. In the setting considered here, these parameters are the mean backoff time λ_{ret} for unslotted Aloha and the probabilities the wireless channel is in the "good" or the "bad" state, π_g, π_b respectively.

After a packet is sent to the network the TCP sender starts a timer which is set to a value according to an estimate of the round-trip time (RTT). In general, in current TCP implementations this estimate does not take into account the fact that the communication takes place over a wireless network and thus it considers any delays to be associated with congestion. In a wireless environment however, especially in random access networks such as the IEEE 802.11 or Aloha (as is the case here), packets are delayed because of collisions and retransmissions at the MAC layer. If a packet is delayed because there is heavy background traffic (i.e. traffic from or to nodes in the neighborhood of the TCP sender or receiver), its acknowledgment might not reach the TCP sender before the expiration of the timeout timer. Then, the TCP sender will declare a timeout, it will minimize the window size and enter slow-start. This will happen even if there is no congestion between the TCP sender and receiver. In this case the TCP throughput is minimized without any congestion being present in the network.

Consider the situation from the point of view of the TCP sender. In the current TCP implementations the TCP sender at some point sends a packet and starts the timeout timer. If no acknowledgment is received for that packet when the timer expires, the TCP sender declares a timeout and enters slow-start. The TCP sender has no way to know the exact cause for the lack of a received acknowledgment and it always assumes there is congestion in the network. If we want to maximize the TCP performance we need to incorporate in the timeout mechanism some information regarding the wireless medium. Consider again the situation where the TCP sender sends a packet to the network and waits for an acknowledgment. There are two reasons for an acknowledgment to be delayed. Either there is congestion in the network or the packet is delayed because of collisions and retransmissions at the MAC layer. In the first case the TCP sender would have liked to declare immediately a timeout and enter slow-start and thus minimize the traffic that is sent

The material is based upon research supported by the National Aeronautics and Space Administration under the Cooperative Agreements No. NCC8-235 and NAG3-2844, and by Communications and Networks Consortium sponsored by the U.S. Army Research Laboratory under the Collaborative Technology Alliance Program Cooperative Agreement DAAD19-01-2-001.

The authors are with the Institute for Systems Research, University of Maryland, College Park, MD 20742 {gpapag, baras}@umd.edu

to the network in order to compensate for the congestion. In the second case however, a timeout event is not the best thing to do because the cause of the delay is the MAC layer and not congestion at the transport layer. In this case the TCP sender would have liked to wait and let the MAC layer resolve the collision. Then, the received acknowledgment would trigger the transmission of packets from the current state of the TCP sender and thus no performance decrease would be observed regarding the TCP throughput.

It is clear then, that the TCP sender has to choose between two different actions in the absence of a received acknowledgment. Either stop the waiting period and enter slow-start (by essentially declaring a timeout), or continue waiting for the acknowledgment, hoping that it is delayed at the MAC layer and not because of congestion. Thus, an optimal stopping problem can be defined. The solution to this problem provides the TCP sender with the optimal timeout period in order to increase its throughput.

II. Problem Formulation

We assume there exists a complete probability space $(\Omega, \mathcal{F}, P)$. We consider a wireless network where the MAC layer is unslotted (pure) Aloha [1], [2]. Each node in the network can hear the transmissions from any other node (single cell). We model the wireless channel as a two-state continuous time Markov chain (the Gilbert-Elliott [3], [4] model). One state corresponds to the case the channel is "good", i.e. the packets are not lost w.p. 1, and the other state corresponds to the case the channel is "bad" which means the transmitted packets are lost w.p. 1. The transition rate from the "bad" to the "good" state is λ_{bg} and the transition rate from the "good" to the "bad" state is λ_{gb}.

In the unslotted Aloha protocol a packet is immediately transmitted to the network. If this transmission overlaps with another packet transmission from another node, then there is a collision and none of the packets are received by the corresponding receivers, and have to be retransmitted. To avoid another collision, the nodes that participated in the collision have to delay their retransmissions for an exponentially distributed random interval. The mean of this exponential distribution is the same for all nodes and is denoted by $1/\lambda_{ret}$. Moreover, this randomly selected interval is independent of any possible previous delays at each node and across nodes.

We focus on a TCP connection between two nodes in the network that are one hop away. We assume this is a persistent TCP connection which implies that the sender has always packets to send. If W_t is the TCP window size at the sender at time t, we can define a stochastic process $W = (W_t)_{t\geq 0}$ that evolves according to the dynamics of the TCP protocol and is affected by the MAC layer and the wireless channel.

A stochastic differential equation for $W = (W_t)_{t\geq 0}$ is introduced in [5]. This stochastic differential equation describes the evolution of the window size as this evolution is driven by a point process that represents the arrival of acknowledgments from the TCP receiver to the TCP sender. More specifically, we have:

$$
\begin{aligned}
dW_t &= \begin{cases} dN_t, & W_t < H_t \\ \frac{1}{W_t} dN_t, & W_t \geq H_t \end{cases} \\
W_{S_n} &= W_0, \quad n = 1, 2, \dots \\
W_0 &\leq W_t \leq W_{max}, \quad t \geq 0
\end{aligned}
\tag{1}
$$

where $H = (H_t)_{t\geq 0}$ is the process that describes the slow-start threshold and its evolution is given in [5]. W_0 and W_{max} are the initial (minimum) and maximum values of the window size, and $\{S_n, n = 0, 1, \dots\}$ is the sequence of the time instances a timeout event is declared by the TCP sender. The stochastic process $N = (N_t)_{t\geq 0}$ is the counting process that corresponds to the point process $\{T_n^{MAC}, n = 0, 1, 2, \dots\}$ that represents the arrival of acknowledgments at the TCP sender. If $\mathcal{F}^N = (\mathcal{F}_t^N)_{t\geq 0}$ is the right continuous filtration that represents the history of this point process, it is shown in [5] that the $\mathcal{F}^N$-intensity λ_t of the process is

$$
\lambda_t = \frac{\lambda_{bg}}{\lambda_{bg} + \lambda_{gb}} \lambda_t^{MAC} \tag{2}
$$

where

$$
\lambda_t^{MAC} = \begin{cases} 0, & T_i^{MAC} \leq t < T_i^{MAC} + T_p \\ \lambda_{ret} p_{mac}, & T_i^{MAC} + T_p \leq t < T_{i+1}^{MAC} \end{cases} \tag{3}
$$

and $T_p = L/C$ is the transmission time of the packet of constant length L bits over the wireless channel of capacity C bps, and p_{mac} is the probability of a successful transmission of a packet for unslotted Aloha, which can be estimated by the TCP sender by listening to the channel and counting the successful transmissions and the collisions at the MAC layer.

As was mentioned in Section I we want to increase the TCP throughput by computing the optimal timeout period τ for the TCP sender to declare a timeout. This can be formulated as an optimal stopping time problem in the probability space $(\Omega, \mathcal{F}, P)$. If $\mathcal{F}^{TCP} = (\mathcal{F}_t^{TCP})_{t\geq 0}$ is the right continuous filtration that represents the history of events observed by the TCP sender we want to find the optimal $\mathcal{F}^{TCP}$-stopping time τ such that

$$
J(w, \tau; h) = \mathrm{E}_w \left[\int_0^\tau e^{-\beta t} k(W_t; h) dt + e^{-\beta \tau} g(W_\tau; h) \right] \tag{4}
$$

is maximized over τ, for $\beta > 0$, where $\mathrm{E}_w[\cdot]$ represents the expected value conditioned on the event that the initial value of the process W is w and the slow-start threshold is h. The value h of the slow-start threshold remains constant for the duration of the timeout period and it only affects the size of the window if an acknowledgment is received. The value function $V(\cdot; h)$ for the problem can then be defined as:

$$
V(w; h) = \sup_\tau J(w, \tau; h) \tag{5}
$$

Notice that the two filtrations, $\mathcal{F}^N$ and $\mathcal{F}^{TCP}$ are equal since the TCP sender observes the arrival of acknowledgments. Moreover, the arrival of acknowledgments dictates the evolution of the process $W = (W_t)_{t\geq 0}$.

III. Optimal Stopping

Let B denote the optimal stopping set, i.e. the process stops when the set B is reached for the first time. Then, the equation satisfied by the value function $V(\cdot;h)$ is known as the Hamilton-Jacobi-Bellman (HJB) equation and is [6], [7]:

$$\begin{cases} \mathcal{L}V(w;h) - \beta V(w;h) + k(w;h) = 0, & w \notin B \\ V(w;h) = g(w;h), & w \in B \end{cases} \tag{6}$$

where $\mathcal{L}$ is the infinitesimal generator of the process W defined on a function $f : \Re \rightarrow \Re$ as:

$$\mathcal{L}f(x) = \lim_{t \downarrow 0} \frac{\mathrm{E}_x\left[f(W_t)\right] - f(x)}{t} \tag{7}$$

As it can be seen from (3), the driving counting process $N = (N_t)_{t\geq 0}$ is not a Lévy process. This makes (6) analytically intractable. Because of this we need to solve the optimal stopping problem numerically. Following Kushner's Markov chain approximation method [7], we proceed by discretizing the time and thus moving from the original optimal stopping problem in continuous time to an equivalent problem in discrete time. In discrete time, the evolution of the system is described by a Markov chain, with transition probabilities that can be computed from the original continuous time dynamical system. We then formulate an optimal stopping time problem associated with the Markov chain and solve the corresponding dynamic programming equation using the value iteration method [8].

IV. Markov Chain Approximation

The discretization of the time is based on the transmission time T_p of a packet. In particular, we define the time increment δ to be

$$\delta = \frac{T_p}{K} \tag{8}$$

where K is a positive integer. For larger values of K, a smaller increment δ is defined and as K increases to infinity, the discretization becomes finer.

The stochastic process $W = (W_t)_{t\geq 0}$ that describes the evolution of the window size takes values in the interval $[W_0, W_{max}]$. When TCP operates in the slow-start regime, the window size takes values in the set of positive integers, and when TCP is in the congestion avoidance phase the window size is a positive real number. For the optimal stopping problem of the approximating Markov chain the window size does not need to be discretized. We only need to differentiate between the current value of the window size and the value that the window size will take after a new acknowledgment is received or a maximum waiting time is reached. Therefore, we do not discretize the interval $[W_0, W_{max}]$ where the process $W = (W_t)_{t\geq 0}$ takes its values.

Suppose at time $t = t_0$ there is a jump to $W_{t_0} = w$ for the original, continuous time system (1) and the slow-start threshold H_t is h. We define the Markov chain that describes the evolution of the system given that at time $t = t_0$ the size of the window is w. We want to solve the optimal stopping problem for the Markov chain for any such initial condition (w, t_0).

The approximating Markov chain $X^{\delta,\ h,\ t_0,\ w} = \{X_n^{\delta,\ t_0,\ w},\ n = 0, 1, \ldots\}$ that represents the evolution of the original, continuous time dynamical system (1) has a two dimensional state space:

$$\mathcal{X}^{\delta,\ h,\ t_0,\ w} = [W_0, W_{max}] \times \{t_0,\ t_0+\delta,\ t_0+2\delta,\ \ldots, \\ t_0 + (K+M+1)\cdot\delta\}, \tag{9}$$

where M is the number of time increments that we allow after the time $t_0+K\delta$ before we declare a timeout. Therefore, the parameter M defines an upper bound on the optimal stopping time of our problem and assures that the algorithm that computes this stopping time terminates.

In order to describe the transitions of the Markov chain $X^{\delta,\ h,\ t_0,\ w}$, suppose that the chain is in the state $(w, t_0 + i \cdot \delta)$. For $i = 0, 1, 2, \ldots, (K-1)$, the chain can only move in time leaving the first component of the state unchanged. This is true because in the original continuous time dynamical system (1) there is no new jump for a time duration equal to T_p (the transmission time of a packet) after a jump (which we assumed it happened at time t_0). Thus, for $i = 0, 1, 2, \ldots, (K-1)$ the state that follows $(w, t_0 + i\cdot\delta)$ can only be $(w, t_0 + (i+1)\cdot\delta)$ and this transition happens with probability 1.

After time $T_p = K\cdot\delta$ has elapsed from t_0 a new jump may occur. If the Markov chain $X^{\delta,\ h,\ t_0,\ w}$ is in state $(w, t_0 + (K+j)\cdot\delta)$ for $j = 0, 1, 2, \ldots, (K+M-1)$, there are two different events that may happen, and thus two possible transitions out of this state that represent these events.

The first event represents a new jump of the original system (1). Thus the second component of the state will increase by one to $t_0+(K+j+1)\cdot\delta$ and the first component of the state will be a new window size w'. The value of w' depends on whether TCP is in slow-start $(w < h)$ or congestion avoidance phase $(w \geq h)$. If TCP is in slow-start phase, then $w' = w + 1$, and if TCP is in congestion avoidance phase then $w' = w + \frac{1}{w}$. Such a transition indicates that a new acknowledgment has arrived at the TCP sender which implies that there is no need to find a timeout interval, and thus solve the optimal stopping problem. The transition from $(w, t_0 + (K+j)\cdot\delta)$ to $(w', t_0 + (K+j+1)\cdot\delta)$ happens with probability p_j that is computed from the p.d.f. for D^{MAC}. The random variable D^{MAC} represents the time between two successive packet transmissions at the MAC layer and its p.d.f. is computed in [5] to be

$$f_{D^{MAC}}(t) = p_{mac}\delta(t-T_p) + p_{mac}(1-p_{mac})\cdot \\ \lambda_{ret}e^{-\lambda_{ret}p_{mac}\cdot\left(t-T_p\right)}u_0(t-T_p) \tag{10}$$

for $t \geq 0$. In this new state the only transition that is allowed is to itself with probability 1.

The second event represents the fact that no new arrival (and thus jump) has occurred. In this case, the first component of the next state remains the same and equal to w, and the second component indicates the increase in time by the increment δ. The transition from $(w, t_0 + (K+j)\cdot\delta)$ to $(w, t_0 + (K+j+1)\cdot\delta)$ happens with probability $1 - p_j$.

Finally, if the Markov chain is in the state $(w, t_0 + (K+M)\cdot\delta)$ it means that the maximum allowed waiting period has been reached, therefore a timeout has to be declared. This is indicated by a transition to $(1, t_0 + (K+M+1)\cdot\delta)$ with probability 1.

The two dimensional Markov chain $X^{\delta,\ h,\ t_0,\ w}$ can be represented schematically as in Fig. 1.

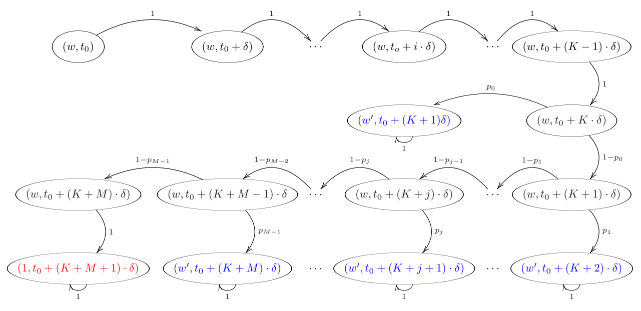
(w,t_0)
$(w,t_0+\delta)$
$(w,t_o+i\cdot\delta)$
$(w,t_0+(K-1)\cdot\delta)$
$(w',t_0+(K+1)\delta)$
$(w,t_0+K\cdot\delta)$
p_0
$1-p_0$
$(w,t_0+(K+M)\cdot\delta)$
$(w,t_0+(K+M-1)\cdot\delta)$
$(w,t_0+(K+j)\cdot\delta)$
$(w,t_0+(K+1)\cdot\delta)$
$1-p_{M-1}$
$1-p_{M-2}$
$1-p_j$
$1-p_{j-1}$
$1-p_1$
p_{M-1}
p_j
p_1
$(1,t_0+(K+M+1)\cdot\delta)$
$(w',t_0+(K+M)\cdot\delta)$
$(w',t_0+(K+j+1)\cdot\delta)$
$(w',t_0+(K+2)\cdot\delta)$
1

Fig. 1. The Markov chain approximation for the Optimal Stopping problem.

V. Transition Probabilities

To better represent the Markov chain $X^{\delta,\ h,\ t_0,\ w}$, we name each of the two dimensional states of the chain as follows:

$$
\begin{aligned}
S_i &= \Big(w, t_0 + i\cdot\delta\Big), \quad i = 0,1,2,\dots,(K+M)\\
F_j &= \Big(w', t_0 + (K+j)\cdot\delta\Big), \quad j = 1,2,\dots,M \qquad (11)\\
R &= \Big(1, t_0 + (K+M+1)\cdot\delta\Big)
\end{aligned}
$$

Using the representation of (11) the Markov chain $X^{\delta,\ h,\ t_0,\ w}$ is shown in Fig. 2. The transition probabilities

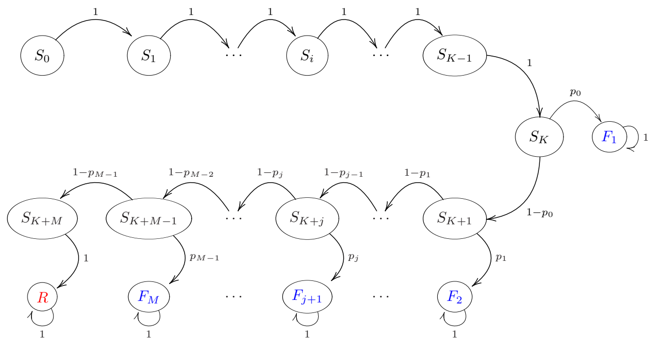

Fig. 2. The Markov chain approximation for the Optimal Stopping problem with different state representation.

can then be given as

- for $i = 0,1,2,\dots,(K-1)$,

$$
\begin{aligned}
Pr\{X^{\delta,\ h,\ t_0,\ w}_{n+1} = X \mid X^{\delta,\ h,\ t_0,\ w}_{n} = S_i\}\\
= \begin{cases} 1, & \text{if } X = S_{i+1} \\ 0, & \text{otherwise} \end{cases} \qquad (12)
\end{aligned}
$$

- for $i = K, (K+1), \dots, (K+M-1)$,

$$
\begin{aligned}
Pr\{X^{\delta,\ h,\ t_0,\ w}_{n+1} = X \mid X^{\delta,\ h,\ t_0,\ w}_{n} = S_i\}\\
= \begin{cases} 1 - p_{i-K}, & \text{if } X = S_{i+1} \\ p_{i-K}, & \text{if } X = F_{i-K+1} \\ 0, & \text{otherwise} \end{cases} \qquad (13)
\end{aligned}
$$

- for $i = K+M$,

$$
\begin{aligned}
Pr\{X^{\delta,\ h,\ t_0,\ w}_{n+1} = X \mid X^{\delta,\ h,\ t_0,\ w}_{n} = S_i\}\\
= \begin{cases} 1, & \text{if } X = R \\ 0, & \text{otherwise} \end{cases} \qquad (14)
\end{aligned}
$$

- for $i = 1,2,\dots,M$,

$$
\begin{aligned}
Pr\{X^{\delta,\ h,\ t_0,\ w}_{n+1} = X \mid X^{\delta,\ h,\ t_0,\ w}_{n} = F_i\}\\
= \begin{cases} 1, & \text{if } X = F_i \\ 0, & \text{otherwise} \end{cases} \qquad (15)
\end{aligned}
$$

- and,

$$
\begin{aligned}
Pr\{X^{\delta,\ h,\ t_0,\ w}_{n+1} = X \mid X^{\delta,\ h,\ t_0,\ w}_{n} = R\}\\
= \begin{cases} 1, & \text{if } X = R \\ 0, & \text{otherwise} \end{cases} \qquad (16)
\end{aligned}
$$

The cardinality of the state space for the Markov chain $X^{\delta,\ h,\ t_0,\ w}$ is $K + 2M + 2$. If we order the states in the following manner:

$$
\Big(S_0,\ S_1,\ \dots,\ S_{K+M},\ F_1,\ F_2,\ \dots,\ F_M,\ R\Big) \qquad (17)
$$

then using the transition probabilities defined above we can write the $(K+2M+2)\times(K+2M+2)$ transition matrix $Q^{(\delta)}$ for the Markov chain $X^{\delta,\ h,\ t_0,\ w}$ in block form:

$$
Q^{(\delta)} = \begin{pmatrix}
\mathbb{O}_{K\times 1} & \mathbb{I}_K & \mathbb{O}_{K\times M} & \mathbb{O}_{K\times M} & \mathbb{O}_{K\times 1}\\
\mathbb{O}_{M\times 1} & \mathbb{O}_{M\times K} & \mathbb{I}_M - \mathbb{P}_M & \mathbb{P}_M & \mathbb{O}_{M\times 1}\\
0 & \mathbb{O}_{1\times K} & \mathbb{O}_{1\times M} & \mathbb{O}_{1\times M} & 1\\
\mathbb{O}_{M\times 1} & \mathbb{O}_{M\times K} & \mathbb{O}_{M\times M} & \mathbb{I}_M & \mathbb{O}_{M\times 1}\\
0 & \mathbb{O}_{1\times K} & \mathbb{O}_{1\times M} & \mathbb{O}_{1\times M} & 1
\end{pmatrix} \qquad (18)
$$

where $\mathbb{O}_{\cdot\times\cdot}$ is the zero matrix, $\mathbb{I}_{\cdot}$ is the identity matrix and $\mathbb{P}_M$ is a diagonal matrix with the element at the i^{th} diagonal position equal to p_i:

$$
\mathbb{P}_M = \begin{pmatrix}
p_1 & 0 & \cdots & 0 & \cdots & 0 & 0\\
0 & p_2 & \cdots & 0 & \cdots & 0 & 0\\
\vdots & \vdots & \ddots & \vdots & \cdots & \vdots & \vdots\\
0 & 0 & \cdots & p_i & \cdots & 0 & 0\\
\vdots & \vdots & \vdots & \vdots & \ddots & \vdots & \vdots\\
0 & 0 & \cdots & 0 & \cdots & p_{M-1} & 0\\
0 & 0 & \cdots & 0 & \cdots & 0 & p_M
\end{pmatrix} \qquad (19)
$$

VI. Computation of the Transition Probabilities

To compute the transition probabilities p_i, $i = 0, 1, \dots, (M-1)$ we use the p.d.f. for D^{MAC} given by (10).

We first compute the probability:

$$Pr\{D^{MAC} > s\} = \int_s^\infty f_{D^{MAC}}(t)\ dt = \begin{cases} 1, & 0 \le s \le T \\ (1-p_{mac})e^{-\lambda_{ret}p_{mac}\cdot(s-T_p)}, & s > T \end{cases} \tag{20}$$

For $i = 0, 1, 2, \dots,$ we compute the probability $q_i^{T_p,\ \delta} = Pr\{T_p + (i+1)\delta > D^{MAC} > T_p + i\delta\}$

$$q_i^{T_p,\ \delta} = \int_{T_p+i\delta}^{T_p+(i+1)\delta} f_{D^{MAC}}(t)\ dt = \begin{cases} 1-(1-p_{mac})e^{-\lambda_{ret}p_{mac}\delta}, & i=0 \\ (1-p_{mac})e^{-\lambda_{ret}p_{mac}i\delta} \\ \quad \cdot \left(1-e^{-\lambda_{ret}p_{mac}\delta}\right), & i=1,2,\dots \end{cases} \tag{21}$$

Using (20) and (21) we compute the probability:

$$Pr\Big\{T_p + (i+1)\delta > D^{MAC} > T_p + i\delta \mid D^{MAC} > T_p + i\delta\Big\}$$

$$= \frac{Pr\Big\{T_p + (i+1)\delta > D^{MAC} > T_p + i\delta\Big\}}{Pr\Big\{D^{MAC} > T_p + i\delta\Big\}}$$

$$= \begin{cases} 1-(1-p_{mac})e^{-\lambda_{ret}p_{mac}\delta}, & i=0 \\ 1-e^{-\lambda_{ret}p_{mac}\delta}, & i=1,2,\dots \end{cases} \tag{22}$$

Using (22) and taking into account the operation of the wireless channel, we can compute the transition probability p_i:

$$p_i = \pi_g \cdot Pr\Big\{T_p + (i+1)\delta > D^{MAC} > T_p + i\delta \mid D^{MAC} > T_p + i\delta\Big\}$$

$$= \begin{cases} \pi_g\Big(1-(1-p_{mac})e^{-\lambda_{ret}p_{mac}\delta}\Big), & i=0 \\ \pi_g\Big(1-e^{-\lambda_{ret}p_{mac}\delta}\Big), & i=1,\dots,M-1 \end{cases} \tag{23}$$

for $i = 0, 1, 2, \dots, (M-1)$, where π_g is given by [5]

$$\pi_g = \frac{\lambda_{bg}}{\lambda_{bg} + \lambda_{gb}} \tag{24}$$

VII. Running and Final Rewards

The running reward $k(\cdot; h)$ is defined in such a way as to represent our unwillingness to declare a timeout and thus minimize the window size. At the same time though this unwillingness should be decreasing with time, since as time increases and no event has occurred (no arrival of an acknowledgment) is an indication of bad channel quality. This means that the chances of finally receiving an acknowledgment become smaller. On the other hand, if we have already built a large window size we might be reluctant to declare a timeout since declaring a timeout brings the window size to its minimum value. Thus, the running reward $k(\cdot; h)$ is an increasing function of the current window size and a decreasing function of the waiting time.

In case an acknowledgment has arrived and the Markov chain $X^{\delta,\ h,\ t_0,\ w}$ has moved to an F_i, $i = 1, 2, \dots, M$, state, there is no need to declare a timeout and thus solve the optimal stopping problem. Because of this, the running cost is 0 for these states.

If no acknowledgment has arrived and the maximum waiting time has been reached, the Markov chain $X^{\delta,\ h,\ t_0,\ w}$ will move to state R. When this transition happens a timeout is declared anyway, and no further action is needed. Hence, the running reward for the R state is 0.

More precisely, the running reward is given by (25)

$$\begin{aligned} &k(S_i; h) = \frac{w}{W_{max}}\Big(K + M + 1 - i\Big)\alpha\delta,\ i = 0, \dots, K+M \\ &k(F_i; h) = 0, \quad i = 1, 2, \dots, M \\ &k(R; h) = 0 \end{aligned} \tag{25}$$

where α is a parameter that depends on whether the TCP sender is in the slow-start $(w < h)$ or the congestion avoidance phase $(w \ge h)$ and can be tuned based on the performance we want to achieve through the optimization problem.

The final reward $g(\cdot; h)$ also depends on both components of the state of the Markov chain $X^{\delta,\ h,\ t_0,\ w}$. For the states S_i, $i = 0, 1, 2, \dots, (K-1)$, the final reward should be 0, since these states represent waiting time during transmission of a packet and thus no event will occur with probability 1. For the rest of the S_i, $i = K, (K+1), \dots, (K+M)$ states, the final reward is defined to be an increasing function on both the window size and the waiting time.

If an acknowledgment is received, and thus the Markov chain $X^{\delta,\ h,\ t_0,\ w}$ moves to an F_i, $i = 1, 2, \dots, M$, state, the final reward depends on the new window size which is different and depends on whether the TCP sender is in the slow-start $(w < h)$ or the congestion avoidance phase $(w \ge h)$.

Finally, the final reward for the state R depends on the maximum waiting time that we allow before we declare a timeout.

More specifically, the final reward $g(\cdot; h)$ is given by (26)

$$\begin{aligned} &g(S_i; h) = 0, \quad i = 0, 1, \dots, (K-1) \\ &g(S_i; h) = \frac{w}{W_{max}}\ i\alpha, \quad i = K, (K+1), \dots, (K+M) \\ &g(F_i; h) = \frac{w'}{W_{max}}(i+K)\alpha, \quad i = 1, 2, \dots, M \\ &g(R; h) = (K+M+1)\alpha \end{aligned} \tag{26}$$

where α is as in the case of the running reward $k(\cdot)$.

VIII. Optimal Stopping and Dynamic Programming

The optimal stopping problem presented in Section II can now be posed on the Markov chain $X^{\delta,\ h,\ t_0,\ w}$. If N_δ is a stopping time for the approximating chain $X^{\delta,\ h,\ t_0,\ w}$, we define the discounted reward according to (4) to be:

$$J^{\delta,\ h}(x, N_\delta) = \mathrm{E}_x \left\{ \sum_{n=0}^{N_\delta - 1} e^{-\beta\ t_n^\delta} \cdot k\big(X_n^{\delta,\ h,\ t_0,\ w}; h\big) \cdot \delta + e^{-\beta\ N_\delta} \cdot g\big(X_{N_\delta}^{\delta,\ h,\ t_0,\ w}; h\big) \right\} \tag{27}$$

where $t_n^\delta = n\delta$ and $\beta > 0$ the discount factor. If

$$V^{\delta,\ h}(x) = \sup_{N_\delta} J^{\delta,\ h}(x, N_\delta) \tag{28}$$

is the corresponding value function for the problem, it satisfies the dynamic programming equation:

$$V^{\delta,\ h}(x) = \max\Big\{ \sum_y e^{-\beta\ \delta} Q^{(\delta)}(x,y) \cdot V^{\delta,\ h}(y) + k(x) \cdot \delta, g(x) \Big\} \tag{29}$$

For numerical purposes, we can approximate $e^{-\beta\delta}$ in (29) with $\frac{1}{1+\beta\cdot\delta}$ [7].

Because of the discounting, the metric $J^{\delta,\ h}(x, N_\delta)$ in (27) is finite and well defined. The dynamic programming equation (29) is also well defined since the transformation

$$\sum_y e^{-\beta\ \delta} Q^{(\delta)}(x,y) \cdot V^{\delta,\ h}(y) + k(x) \cdot \delta \tag{30}$$

defines a contraction mapping.

IX. Simulation Results

To solve the optimal stopping problem for the approximating Markov chain $X^{\delta,\ h,\ t_0,\ w}$ we use the dynamic programming equation (29). Because of the contraction mapping property of (30) we use a combination of iteration methods in both the value and the policy space to get the solution to the stopping problem. The value iteration method solves for the value function $V^{\delta,\ h}(\cdot)$ and the policy iteration provides the optimal stopping set B of (6) for the problem. The stopping set B defines practically our optimal policy, in the sense that, whenever the Markov chain $X^{\delta,\ h,\ t_0,\ w}$ moves into a state $x \in B$ then we stop.

The parameters that define the experiments are related to the wireless channel, the MAC and the TCP. They are summarized in Table I.

TABLE I
Simulation Parameters.

Wireless Channel	MAC	TCP	Approximation
λ_{bg}	λ_{ret}	T_p	δ, K
λ_{gb}	p_{mac}	β	M
			α

We run different experiments for different values of the parameters in Table I and compare against the standard timeout mechanism of TCP [9]. The channel capacity in all experiments is 2Mbps.

In Fig. 3 and Fig. 4 we compare the instantaneous rate when using the timeouts that are solutions to the optimal stopping time against the standard timeout mechanism of TCP. In the case of Fig. 3, the mean waiting time before a retransmission at the MAC layer is 0.1sec ($\lambda_{ret} = 10$), and in Fig. 4 the corresponding mean waiting time is 0.01sec ($\lambda_{ret} = 100$). In both cases, there are no losses at the wireless channel ($\pi_g = 1$). The probability of a successful transmission at the MAC layer is $p_{mac} = 0.3$ and the discount parameter β is 0.9. Also, in both cases the parameter α in the running and final rewards is 1 when TCP is in slow-start and 10 when in congestion avoidance. As it can be seen in both Fig. 3 and Fig. 4 the timeout mechanism that is produced from the numerical approximation to the stopping problem has better performance than the standard implementation of the timeout mechanism.

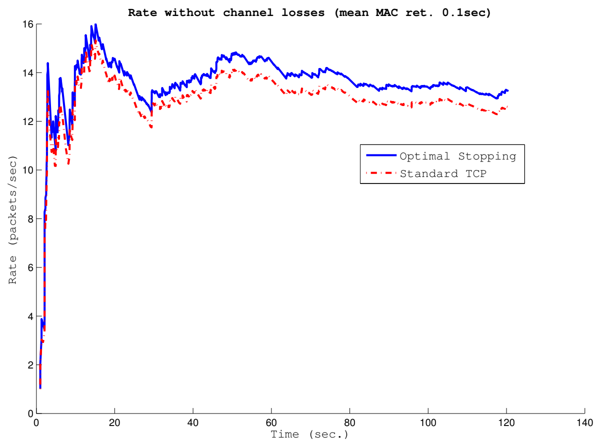

Fig. 3. Comparison of the instantaneous rate between the stopping problem and the TCP mechanism in the absence of losses at the channel and mean retransmission waiting time 0.1sec at the MAC layer.

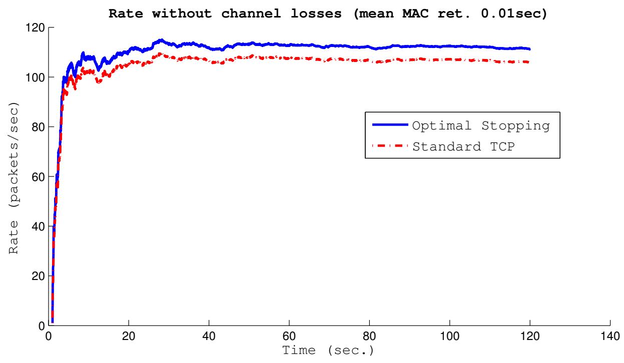

Fig. 4. Comparison of the instantaneous rate between the stopping problem and the TCP mechanism in the absence of losses at the channel and mean retransmission waiting time 0.01sec at the MAC layer.

The case of channel losses is shown in Fig. 5 and Fig. 6. In both cases the channel loss probability is $\pi_b = 0.5$. Fig. 5

shows the case where the mean waiting time after a collision is 0.1sec ($\lambda_{ret} = 10$), and Fig. 6 corresponds to the case where the mean waiting time after a collision at the MAC layer is 0.01sec ($\lambda_{ret} = 100$). As before, the probability p_{mac} of not having a collision at the MAC layer is 0.3 for both cases. The discount parameter β is 0.9, and the parameter α in the running and final rewards is 1 for slow-start and 10 for congestion avoidance.

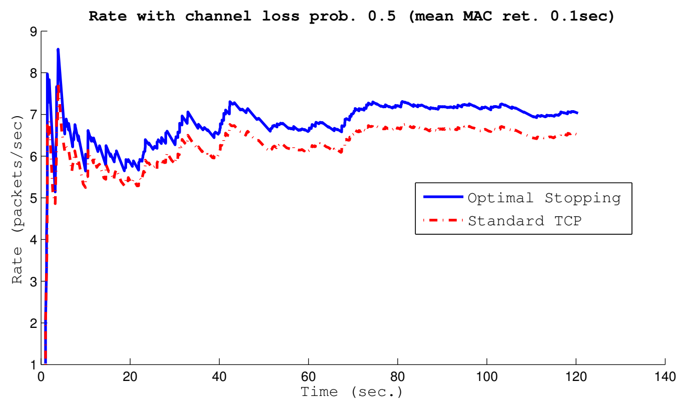

Fig. 5. Comparison of the instantaneous rate between the stopping problem and the TCP mechanism with a loss prob. 0.5 at the channel and mean retransmission waiting time 0.1sec at the MAC layer.

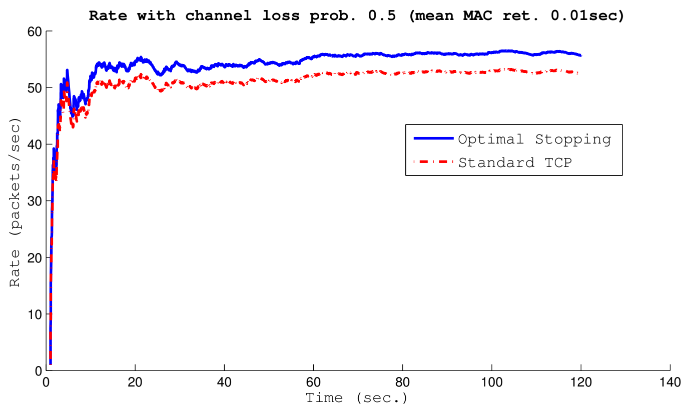

Fig. 6. Comparison of the instantaneous rate between the stopping problem and the TCP mechanism with a loss prob. 0.5 at the channel and mean retransmission waiting time 0.01sec at the MAC layer.

Finally, In Fig. 7 the results of the comparison are shown where the discount factor β is 0.001. The loss probability $(1 - \pi_g)$ at the wireless channel is 0.3, and the mean waiting time before retransmissions at the MAC layer is 0.01sec ($\lambda_{ret} = 100$).

X. Conclusions

The focus of the current work is to fine tune the timeout mechanism of TCP to achieve better throughput in wireless networks. We pose the problem as an optimal stopping problem using the stochastic differential equation [5] as a constraint. The fact that the driving point process is not a Poisson process makes the analysis intractable. Motivated by the approximation method of Kushner for stochastic control problems in continuous time, we develop a numerical approximation to the original problem. Using dynamic programming we solve the discrete time version of the original problem and retrieve stopping policies that define the new timeout mechanism. We verify the performance increase by comparing our solution to the standard TCP timeout mechanism using simulations for different values of the involved parameters.

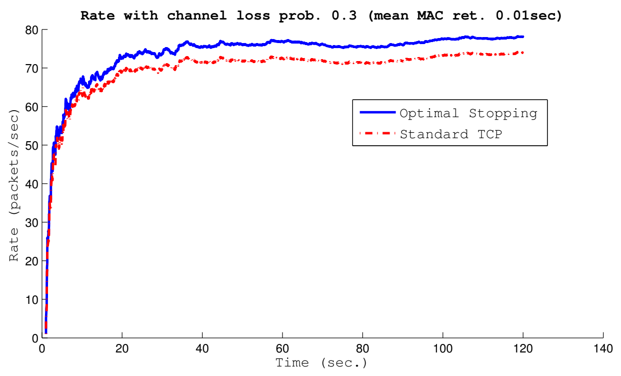

Fig. 7. Comparison of the instantaneous rate between the stopping problem and the TCP mechanism with a loss prob. 0.3 at the channel, mean retransmission waiting time 0.01sec at the MAC layer and discount factor $\beta = 0.001$.

References

[1] N. Abramson, "The Aloha system-another alternative for computer communications," in *AFIPS Conference Proceedings*, vol. 37, Nov. 1970, pp. 281–285.
[2] L. G. Roberts, "Aloha packet system with and without slots and capture," *ACM SIGCOMM Computer Communications Review*, vol. 5, no. 2, pp. 28–42, Apr. 1975.
[3] E. N. Gilbert, "Capacity of a burst-noise channel," *Bell System Technical Journal*, vol. 39, no. 3, pp. 1253–1265, Sep. 1960.
[4] E. O. Elliott, "Estimates of error rates for codes on burst-noise channels," *Bell System Technical Journal*, vol. 42, no. 3, pp. 1977–1997, Sep. 1963.
[5] G. Papageorgiou and J. S. Baras, "A simple model for the window size evolution of TCP coupled with MAC and PHY layers," in *Proceedings of the 43rd Annual Conference on Information Sciences and Systems (CISS 2009)*, Baltimore, MD, Mar. 2009.
[6] G. Peskir and A. Shiryaev, *Optimal Stopping and Free-Boundary Problems*. Birkhauser, 2006.
[7] H. J. Kushner and P. Dupuis, *Numerical Methods for Stochastic Control Problems in Continuous Time*, 2nd ed. Springer, 2001.
[8] M. L. Puterman, *Markov Decision Processes: Discrete Stochastic Dynamic Programming*. Wiley-Interscience, 2005.
[9] V. Paxson and M. Allman, *Computing TCP's Retransmission Timer*. RFC2988, Nov. 2000.