Performance Evaluation 68 (2011) 806–824

Contents lists available at ScienceDirect

Performance Evaluation

journal homepage: www.elsevier.com/locate/peva

TCP performance optimization in multi-cell WLANs[☆]

Ka-Lok Hung[*], Brahim Bensaou

Department of Computer Science and Engineering, The Hong Kong University of Science and Technology, Clear Water Bay, Kowloon, Hong Kong

A R T I C L E   I N F O

Article history:
Available online 5 May 2011

Keywords:
Analytical modeling
Multi-cell IEEE 802.11 WLANs
Hidden terminals
Cross-layer optimization
Markov chain

A B S T R A C T

Though significant attention has been given to understanding the performance of a single-cell WLAN, performance evaluation of a group of interfering basic service sets (BSSs) within an extended service set (ESS) is still an open area. In this paper, we first demonstrate that a severe throughput imbalance occurs between downlink TCP flows even in the simplest of multi-cell WLANs via simulation and real world experiments; then, to solve this unfairness problem, we derive an analytical model that describes the interaction between TCP flows at the MAC layer, and formulate a throughput allocation problem as a nonlinear optimization problem subject to certain fairness requirements. Our formulation considers real world complexity such as hidden terminals, packet transmission retry limit, and the unique characteristics of TCP traffic. Solving our optimization problem yields the optimal MAC layer contention window settings that can lead each TCP flow to its target end-to-end throughput without the need for any per-flow queuing nor modification of the TCP sender. Simulation results show that our approach can achieve a fair allocation on the end-to-end throughput and attest to the accuracy of our proposed method.

1. Introduction

Due to the small number of orthogonal channels and spatial constraints, it is very common that the coverage of different APs within the same extended service set (ESS) of an IEEE 802.11-based WLAN overlap. As a result, the transmission from different basic service sets (BSSs) within the ESS interfere with each other, and hidden terminals are bound to exist. In addition to such *inter-BSS* hidden terminals, in practice, in indoor environments, because of physical obstacles like walls or doors, wireless clients within the same BSS may be hidden from each other and the hidden terminal problem can exist even in a single BSS (dubbed here the *intra-BSS* hidden terminal problem). Hidden terminals in general result in a severe throughput unfairness. In this paper, we study and optimize the network performance in multi-cell WLANs in which both inter-BSS and intra-BSS hidden terminal problems exist.

In order to study the performance of such multi-cell WLANs, we consider a network consisting of two APs. The clients are partitioned into three disjoint sets L, R and M. Nodes within the same set are synchronized by the DCF [1]; as a consequence, collisions among clients from the same group only occur at the beginning of a transmission, i.e., only when the colliding nodes draw the same random backoff number. Nodes that belong to different sets are hidden from each other. Two APs, AP x and AP y, establish two BSSs within an ESS, and are also hidden from each other. Nodes in set L are associated with AP x and nodes in set R are associated with AP y. Nodes in set M can communicate with either AP x or AP y and are thus partitioned into two disjoint subsets M_x and M_y which are associated with AP x and AP y respectively. An instance of a network is shown in Fig. 1 in which the number of nodes in set L is 3, in set R is 2 and the number of nodes in set M is 3 with one node being in M_x and two being in M_y. Furthermore, we assume that all four groups of clients are receiving downlink TCP traffic from different file servers through independent TCP flows.

☆ This work is supported in part under grant GRF 610409.

* Corresponding author.
E-mail addresses: ether@cse.ust.hk (K.-L. Hung), brahim@cse.ust.hk (B. Bensaou).

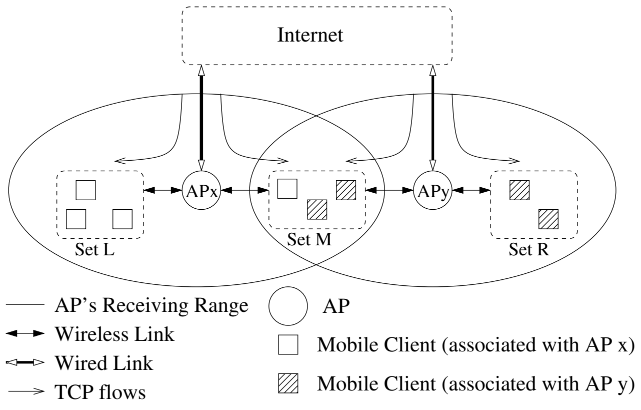


Fig. 1. The multi-cell WLANs scenario with inter-APs and intra-BSS hidden terminals.

Table 1
Simulation parameters.

Slot time σ	9 μs
SIFS time	10 μs
Preamble duration	20 μs
Signal extension time	6 μs
Default $CW_{\min}$, $CW_{\max}$	16, 1024
Data rate	54 Mbps
MTU	1500 bytes
Basic rate for ACK and MAC header	6 Mbps
Maximum queue length	100 Packets
Maximum transmission limit	7

Table 2
Average throughput (Mbps).

RTT_{as} (ms)	Basic access		RTS/CTS	
	Flow 1/4	Flow 2/3	Flow 1/4	Flow 2/3
1	20.12	0.008	17.06	0.051
10	19.75	0.026	16.85	0.053
20	19.26	0.038	16.15	0.087
30	18.39	0.050	14.73	0.128
40	17.45	0.087	13.33	0.23
50	16.70	0.103	11.97	0.32

In such a simple network, we observe that severe long term throughput imbalance occurs between the TCP flows when the wireless link is the end-to-end bottleneck. To demonstrate this, we simulate a basic scenario where $|L| = |M_x| = |M_y| = |R| = 1$ with TCP Reno flows using IEEE 802.11g standard in ns-2 [2] with the parameters shown in Table 1 for 30 random trials in which we seed the random number generator based on the current time at the beginning of each run. The length for each trial is 50 s. We label each TCP flow with a unique integer from 1 to 4. The downlink flow between the file server and the client in set L is flow 1; the downlink flow for the client in set M_x is flow 2; the downlink flow for the client in set M_y is flow 3; the downlink flow for the client in set R is flow 4. Since the number of nodes in sets L, R, M_x and M_y is the same, the throughput of flow 1 and that of flow 4 (respect. flow 2 and flow 3) are very close to each other in the simulation results. Here we only report the per-flow average throughput as shown in Table 2.

To avoid the side effects of hidden factors such as TCP RTT bias and congestion in the wired network, which may influence the simulation results, we set the RTTs between file servers and APs to the same value (RTT_{as}) for all 4 flows. We also set the wired link bandwidth to be larger than 20 Mbps which is the maximum throughput achieved by an isolated TCP Reno downlink flow from an AP to a client under the settings of Table 1 when the basic access scheme of the DCF is used. We repeated the simulation with different values of RTT_{as} ranging from 1 to 50 ms and the results, shown in Table 2, confirm the existence, and show the severity, of this starvation problem.

We observed that flows 2 and 3 are active only within the first few seconds of the simulation, and are blocked by the TCP transmission timeout and hidden node collisions afterward. As for flows 1 and 4, they occupy the channel throughout the whole experiment. We measured the standard deviation of the average throughput for these two (flows 1 and 4), and report the results in Table 3. We can see that the average throughput per second varies more dramatically when the RTT increases. The slow response to packet loss because of the long round trip delay induces such variances.

Table 3
Standard deviation (Mbps) of the throughput of flow 1/4.

RTT_{as} (ms)	Basic access Flow 1/4	RTS/CTS Flow 1/4
1	1.69	2.13
10	2.53	2.54
20	3.45	3.42
30	4.29	4.20
40	4.88	4.49
50	5.06	4.61

The starvation of flows 2 and 3 is due to the uneven degree of contention between TCP flows at the MAC layer. TCP data packets of flows 1 and 4 are immune to hidden node collisions while TCP data packets of flows 2 and 3 suffer from hidden node collisions as AP x and AP y are unable to detect each other's transmissions. TCP ACK packets of flow 1 (or 4) can collide with TCP ACK packets of flow 2 (or 3) because of the hidden terminal problem. However, due to the cumulative acknowledgment property of the TCP ACK, the TCP sender is less sensitive to the ACK packet collisions than to the TCP data packet drops. Flows 1 and 4 can maintain a sustainable throughput right from the onset of the competition with flows 2 and 3. The TCP congestion windows of flows 2 and 3 are limited by the packet drop due to hidden nodes' transmissions. Since data packets of flows 1 and 4 are free from hidden node collisions, flows 1 and 4 eventually acquire more bandwidth than flows 2 and 3. In turn, this increased TCP window (and TCP data rate) of flows 1 and 4 exacerbates further the hidden node collision problem for flows 2 and 3, which succeed even less in their transmissions. When data packets of flows 2 and 3 are dropped frequently, the corresponding TCP sources spend most of their time waiting for the timeout events because of the TCP exponential backoff procedure [3]. In the TCP exponential backoff procedure, the time interval between successive retransmissions is doubled after each unsuccessful retransmission (detected by a timeout). For example, if a timeout event occurs at t seconds after the first transmission of a given data packet, this packet will be retransmitted and the timeout value for this retransmission will be doubled to $2 \times t$. The upper limit of the TCP timeout value is 64 s. While flows 2 and 3 are idling and waiting for a timeout event, the TCP congestion windows of flows 1 and 4 continue to grow until the two flows fully utilize the wireless channel and occupy the buffers at the APs. This finally starves flows 2 and 3.

In addition to simulation studies, we show in the body of the paper that such starvation occurs in real-world WLANs. We demonstrate that such a problem exists due to the downlink TCP data collision at the clients through extensive test-bed experiments. Although the impact of the hidden terminal problem can be cushioned by channel assignment algorithms which avoid having hidden nodes from partially overlapping cells operating on the same channel, such an approach only resolves the problem to some extent because of the limited number of orthogonal channels. Recent studies [4] reveal that the hidden terminal problem cannot be completely eliminated by channel assignments in well managed production WLANs even with RF site surveys. In indoor environments like offices and shopping malls, numerous APs are deployed, and hidden node collisions make some clients inaccessible to their associated APs.

To resolve this starvation problem, the maximum transmission rate of flows 1 and 4 must be controlled such that TCP data and ACK packets of flows 2 and 3 can face less contention and their successful transmission probability can be increased. Furthermore, when the rate of flows 1 and 4 is controlled, the share of buffer occupancy at the APs by flows 2 and 3 is improved and the buffer lockout problem is avoided. Rate control can be achieved in various ways. For example, it can be achieved in end systems by limiting the TCP congestion window size at the TCP source or it can be achieved at the network level by employing AQM at the APs' queues. The two approaches have, however, drawbacks that make them less appealing for WLANs. The first approach may not be feasible as the operator of the ESS may not be able to control the remote servers. For example a Wi-Fi hotspot operator has no control over the data servers from which the customers download their files therefore it is not possible to modify the TCP sender (at the server) to limit the congestion window size. For the second approach (AQM), per-flow queuing and careful tuning of many parameters of the AQM system are necessary. Furthermore, while AQM schemes proved to be good means to limiting congestion and resolving issues like synchronization, they are still not conclusive as to achieving fairness because they still rely on the responsiveness of the end systems.

In this paper, we propose a simple yet effective technique to address the problem at the MAC layer. Our approach achieves a fair balance between TCP flows in the wireless realm by tuning the MAC layer contention windows. To achieve this, we first propose an analytical model of the MAC layer that consists of a set of nonlinear equations, which describes the interaction between all wireless nodes at the MAC layer. Then, we study the interaction between the TCP layer and the MAC layer by constructing a set of constraints that relate to the transport layer and embed them into the MAC layer model. Based on this cross-layer model, we study two problems: the rate allocation problem in which we can obtain the optimal contention window setting for each node in the network based on some specified throughput requirements; and the more general rate optimization problem by formulating the throughput maximization problem as a nonlinear optimization problem subject to max–min or weighted max–min fairness constraints. Both problems are generalized to the case of multiple clients in each set (L, R, and M). We validate our models and method via extensive simulation, and the results show that the network bandwidth can be shared fairly and the target throughput of each TCP flow can be reached. The approach is appealing because it requires only the incremental modification of the firmware of the APs which are in practice under the control of a single entity (e.g. a hotspot operator, a University, and so on).

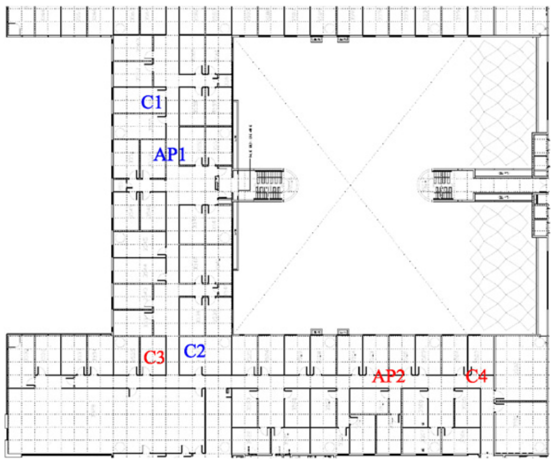


Fig. 2. A multi-cell WLAN at our department office.

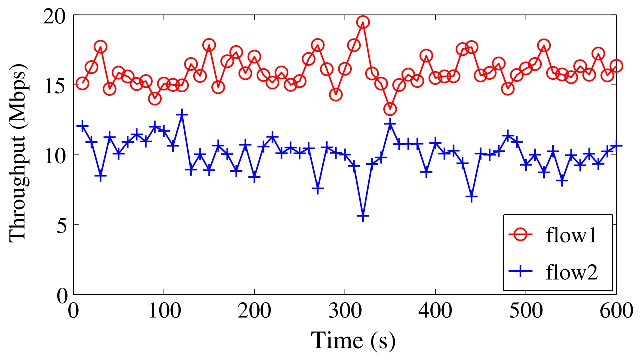
The rest of the paper is organized as follows. In Section 2, we demonstrate the existence of the throughput starvation problem in a real world test-bed. We present our MAC layer analytical model in Section 3. In Section 4, we model the interactions between the TCP and the MAC layers, and propose a throughput allocation problem and a throughput optimization problem. In Section 5, we validate our model and bandwidth allocation scheme via simulation. In Section 6, we discuss some related work, and finally conclude the paper in Section 7.

2. Test-bed experiments

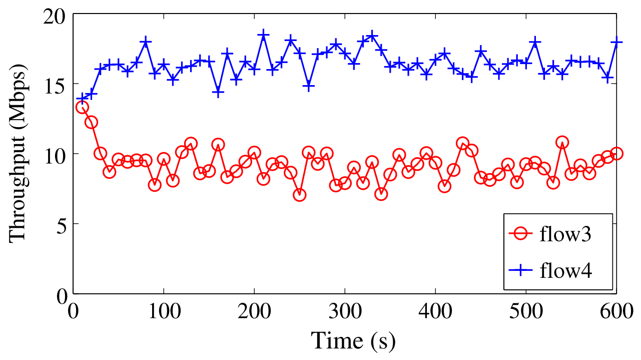
We have shown in the previous section via simulation studies that a severe throughput starvation problem exists if APs are hidden from each other. To determine how such an inter-cell hidden terminal problem affects the network performance in a practical situation, we deploy a multi-cell WLAN in our department office. There are two APs and four clients in the example network as shown in Fig. 2. Clients 1 and 2 are associated with AP1 while clients 3 and 4 are associated with AP2. The two APs are ASUS RT-N16 and clients are four identical Lenovo S10-2 netbooks. All the network interfaces operate under the IEEE 802.11g standard. The APs are configured with the same default factory settings and transmit in the same channel. Due to the "L-shaped" corridor and concrete walls, AP1 and AP2 are hidden from each other. Thus, clients 2 and 3 suffer from hidden node collisions. Since the focus of this experimental study is about inter-cell hidden terminal problems, we do not consider the intra-cell hidden terminal problem. Slightly different from the example network in Fig. 1, Clients 1(3) and 2(4) in Fig. 2 are not hidden from each other. PCs are connected to the APs via Ethernet to serve as TCP sources.

To study the basic characteristics of this network, we first establish two downlink TCP flows between AP1 and clients 1 and 2 under the basic access scheme and measure the throughput with the utility iperf [5]. The TCP downlink throughput is averaged and reported every 10 s and results are shown in Fig. 3(a). We can see that flow 1 (the downlink traffic between AP1 and client 1) takes more bandwidth than flow 2 (the downlink traffic between AP1 and client 2). Such a difference can be the result of multi-path fading and capture effect as the transmission between client 2 and AP1 has to pass through the hallway. Similar measurement results can be obtained for flows 3 and 4 as shown in Fig. 3(b). We also study the DCF synchronization between AP1 and AP2. To do so, we activate flow 1 and flow 4 at the same time and their throughput is reported in Fig. 3(c). We can see that both flows 1 and 4 achieve more than 20 Mbps most of the time. It indicates that AP1 and AP2 are out of each other's sensing range and flows 1 and 4 belong to different WLANs. The average throughput of flow 1 is 20.1 Mbps and flow 4 is 23.5 Mbps.

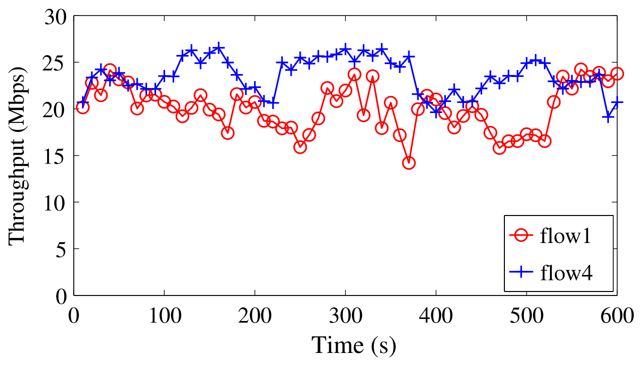
After understanding the basic characteristics of the network in a real operational environment, we study the influence of the inter-cell hidden terminal problem. We establish four downlink TCP flows between the clients and the APs. The throughput measurement under the basic access scheme is shown in Fig. 4(a). We can clearly see that a throughput starvation problem occurs. The average throughput of flows 1 to 4 is 10.3 Mbps, 1.3 Mbps, 0.3 Mbps and 8.1 Mbps respectively. Such a problem occurs as TCP flows 2 and 3 cannot sustain the growth of their congestion windows for long periods of time due to the hidden node collisions and TCP retransmission timeout. We repeated our experiment with an RTS/CTS exchange required for each TCP data packet. The throughput measurement is shown in Fig. 4(b). We can see that flow 3, as in the previous experiment, achieves nearly zero bandwidth allocation all the time. Flow 2 performs sightly better than the previous experiment but most of the bandwidth is still allocated to flows 1 and 4. The average throughput of flows 1 to 4 are 9.1 Mbps, 1.93 Mbps, 0.3 Mbps and 7 Mbps. This experiment shows that the RTS/CTS mechanism cannot remove the root of the problem completely. To resolve the starvation problem, we have to control the rate of each flow and reduce the hidden node collisions between the two APs.



(a) TCP throughput for flows 1 and 2.



(b) TCP throughput for flows 3 and 4.



(c) TCP throughput for flows 1 and 4.

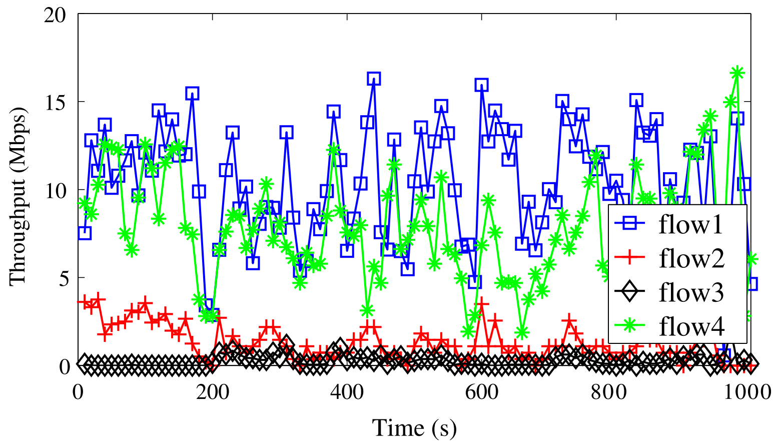
Fig. 3. Throughput measurement without the hidden terminal problem.

3. Analytical model

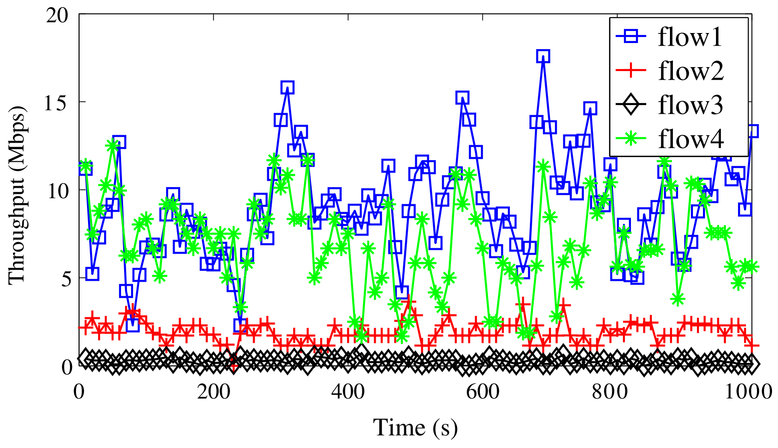
In this section, we develop an analytical model of the channel activities of multi-cell WLANs at the MAC layer as shown in Fig. 1. This model serves as the basic building block for TCP performance optimization in Section 4. In this model, we first assume that $|L| = |R| = |M_x| = |M_y| = 1$ then we relax this assumption and extend our model to support multiple clients in Section 4.2.3.

In this model, we assume the APs and the clients to be saturated with TCP data packets and TCP ACK packets. We further assume the TCP sources are capable of fully utilizing all available bandwidth on the wireless bottleneck link. Under such assumptions, we derive how the link throughput is related to the average backoff duration. In Section 4, by examining the unique characteristics of TCP traffic, we construct the necessary constraints which allow TCP to effectively work according to these assumptions. Then by embedding such constraints into the analytical model, we can formulate the throughput allocation problem and throughput maximization problem in which the optimal average backoff time of each node can be obtained by solving these problems in Sections 4.2 and 4.3.

We summarize the key notations used in this paper in Table 4, and explain each of them subsequently.



(a) Starvation problem (basic access scheme).



(b) Starvation problem (RTS/CTS).

Fig. 4. Throughput measurement under the influence of hidden terminals.

Table 4
Key notations.

	Model input
$T_F(n)$	Average channel occupation time of a packet from node n
$\|L\|, \|M_x\|, \|M_y\|, \|R\|$	Number of nodes in sets L, M_x, M_y, R
	Model output
$th(l_{ii'})$	Link throughput from node i to node i'
$P(l_{ii'})$	Conditional probability on link $l_{ii'}$
	Model variables
$BO(n)$	Average backoff time of node n
$f_i(i')$	Fraction of time that node i transmits to node i'
	Markov chain
Ω	States space
$\Theta(N)$	The set of states where all the nodes in set N are transmitting
$G(i)$	The set of states where node i is in backoff stage
$C(i, i')$	a subset of $G(i)$ in which nodes that interfere with link (i, i') are transmitting
$\pi(S)$	Steady state probability of Markov chain state S

3.1. Channel activities model

To describe the interaction between the nodes activities, we use a continuous time Markov chain model in which each state is based on the activities of all the nodes. To map the time scale between the Markov chain and the DCF, the time unit of the Markov chain is chosen as the physical time slot σ.

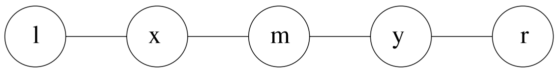


Fig. 5. The node synchronization graph.

In our example network, coordinated collisions occur sharply at the beginning of a transmission from synchronized senders. The occurrence of such collisions is due to the discrete nature of the backoff counters, reducing the network throughput significantly in single-cell WLANs when the number of stations in the network is large. However, as explained in the introduction, the starvation problem in our multi-cell WLANs is mainly the result of collisions between the two APs which are hidden from each other. Also, as shown in [6], hidden node collisions are the major source of collisions whenever hidden nodes exist. Therefore, coordinated collisions are of little significance. Moreover, the clients are only required to handle TCP ACK traffic whose volume is much smaller than the one of the downlink data traffic. Therefore, in our cross-layer optimization control which will be explained in Section 4, the contention windows of stations are much larger than the APs in optimal throughput allocation. With a large contention window, coordinated collisions rarely happen. Thus, omitting such collisions does not result in a significant degradation of model accuracy. This is a reasonable tradeoff between simplicity of the modeling approach and realism yet the resulting solution may yield a practically appealing approach. Therefore, in our model, we focus on modeling the hidden node collision, and consider how throughput is diminished by such collisions. In Section 5, simulation results show that predictions on network throughput and the successful packet delivery probability are not affected dramatically by such a simplification and indicate that indeed hidden node collisions are the dominant factor in packet loss, imposing the throughput limitations observed in such scenarios.

Based on the above discussion, ignoring the coordinated collisions, the backoff time and the transmission time can be modeled as continuous random variables. We further assume that they are exponentially distributed. We will relax this assumption on the distribution in Section 3.1.2.

3.1.1. State space

Based on our assumption that $|L| = |R| = |M_x| = |M_y| = 1$, we label the node in set L and set R as node l and node r. Since the two nodes in set M are synchronized by the DCF with a common channel view, their overall behavior is instantiated by that of a single virtual node denoted m. For example, if node m is active, it means that one of the two nodes in set M is transmitting while the other is frozen because of the ongoing transmission. There can only be one active node in the set at any time as we excluded the possibility of coordinated node collisions. If node m is inactive, it means that both nodes in set M are frozen due to the transmission of AP x or AP y, or both nodes are in countdown state.

The activity state of a node n is denoted S_n whose value can be either 1 or 0 for $n \in \{x, y, l, m, r\}$ (AP x, AP y, node l, node m and node r). Since we assume all nodes are saturated with traffic, S_n is equal to 0 only if node n is frozen or in backoff state. S_n is equal to 1 if node n is transmitting a MAC data frame or is waiting for a MAC ACK frame after a MAC data frame transmission. That is, S_n is 1 when node n has captured the channel. The overall system state is defined as $S = S_lS_xS_mS_yS_r$.

The node synchronization graph of our network is shown in Fig. 5. Each node of the network is represented by a vertex. There is an edge between two vertices if they can sense each other's transmission, and they are synchronized by the DCF. By treating the sequence $S_lS_xS_mS_yS_r$ as a morphism of the node synchronization graph, and excluding coordinated nodes collisions, the states of node n's neighbors have to be 0, if $S_n = 1$. The state space Ω contains 13 possible states: {00000, 10000, 01000, 00100, 00010, 00001, 10100, 10010, 10001, 01010, 01001, 00101, 10101}.

3.1.2. State transitions and connectivity

In general, the system state transits from one state to another state when a node starts or finishes its transmission. For example, the transition from state 01000 to 01001 occurs if node r starts its transmission while AP x is transmitting. The reverse transition from 01001 to 01000 occurs if node r finishes its transmission while AP x is still transmitting. Fig. 6 shows the state transition diagram of the continuous time Markov chain and we label each state with a unique integer from 1 to 13 shown in the parentheses. The undirected edges between two states represent transitions in both directions.

As the coordinated collisions are neglected (and the backoff is considered continuous), we can assume the probability that two or more nodes start or finish their transmission at the same time is zero. Also, with probability zero a node starting its transmission coincides with another node finishing its transmission. For example, transitions such as 00000 to 10100 and 10000 to 00100 do not take place. There is a direct transition between two states if and only if the number of changes in transmitting nodes is equal to 1. For example, the number of changes in transmitting nodes from state 10000 to 00100 is 2, so no direct transition exists between them.

As shown in [7], the steady state probability of the Markov chain shown in Fig. 6 is insensitive to the actual distributions of the backoff timers and transmission time. It only depends on the mean values of the actual distributions. Our modeling approach is still valid even if the transmission time of a packet is a constant in practice.

3.1.3. Steady state probabilities

We define the average backoff time of node n as $BO(n)$ for $n \in \{l, x, m, y, r\}$. We emphasize that $BO(m)$ is not the average backoff time of an individual node in set M but indeed $BO(m)$ describes the overall behavior of set M. For example, if $BO(m)$

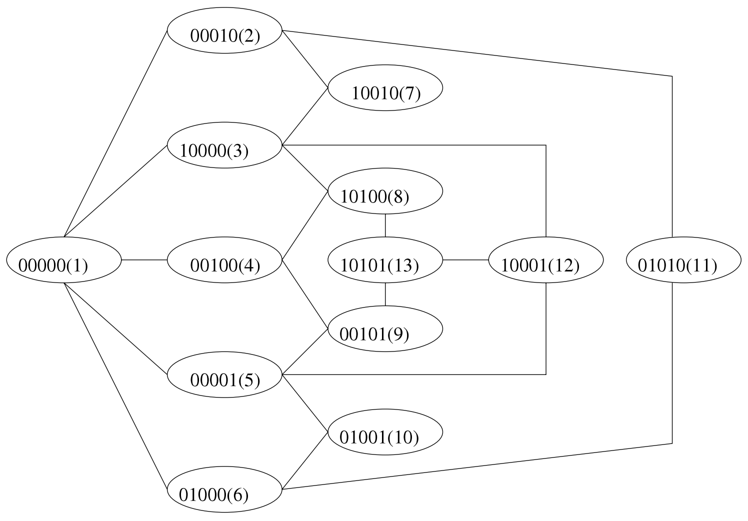


Fig. 6. The continuous time Markov chain model for channel activities.

is 100, it means that between two consecutive transmissions that are generated by any node in set M, 100 slots are spent in backoff procedure on average. If we only focus on a particular node m' in set M, the average backoff time between two consecutive transmissions that are both generated by m' is larger than $BO(m)$. We discuss how $BO(m)$ is related to the individual backoff time of a node in Section 4.2.1. In this section, we express the steady state probability in terms of the average backoff time $BO(n)$. We will show later how these average backoff times can be obtained via the throughput allocation/optimization problem.

We define the channel occupation time per-MAC data frame from node n as $T_F(n)$ slots which is defined as

$$T_F(n) = T_{PHY} + T_{MAC} + T_{DATA}(n) + T_{SIFS} + T_{ACK} + T_{DIFS}, \tag{1}$$

where T_{DIFS} and T_{SIFS} represent the duration of a DIFS and a SIFS; T_{PHY} represents the transmission time for the physical layer preamble and header of a data frame; T_{MAC} represents the transmission time for the MAC header of a data frame; T_{ACK} represents the transmission time of a MAC ACK frame including physical layer and MAC layer overhead; $T_{DATA}(n)$ is equal to $T_{tcp_{data}}$ when $n \in \{x, y\}$ and is equal to $T_{tcp_{ack}}$ when $n \in \{l, m, r\}$ with $T_{tcp_{data}}$ and $T_{tcp_{ack}}$ being the average transmission duration (measured in the unit of a physical time slot σ) of a TCP data packet and a TCP ACK packet respectively. The average on period of node n is $T_F(n)$ slots.

It can easily be verified that the Markov chain in Fig. 6 is time-reversible and ergodic [8]. The steady state probability of state S is denoted $\pi(S)$. When the system is in state S where node n and all its conflicting nodes are inactive, the Markov chain transits from state S to state $S + \{n\}$ with the transition rate $1/BO(n)$. In state $S + \{n\}$, node n is transmitting while the status of other nodes remains the same as in state S. For example, we have the following transition from state 1 to state 6,

$$\pi(01000) = \frac{1}{BO(x)} \times \pi(00000). \tag{2}$$

Similarly, the Markov chain transits from state $S + \{n\}$ back to S with the transition rate $1/T_F(n)$. For example, we have the following transition from state 6 to state 1,

$$\pi(00000) = \frac{1}{T_F(x)} \times \pi(01000). \tag{3}$$

It is not difficult to see that any state which has at least one transmitting node can be expressed in terms of π_{00000} $(\pi(1))$ as

$$\pi(S) = \left(\prod_{i \in A(S)} \frac{T_F(i)}{BO(i)}\right) \pi(00000), \tag{4}$$

where $A(S)$ represents all the active nodes that are transmitting. For example, $A(10010) = \{l, y\}$. By one of the probability axioms, we also have

$$\sum_{S \in \Omega} \pi(S) = 1, \tag{5}$$

where Ω is the set of all possible states. Solving Eqs. (4) and (5), we can obtain the steady state probability vector $\boldsymbol{\pi}$.

3.1.4. Comment

A similar Markovian modeling technique is used in [9,10] to analyze general CSMA networks without considering the details of a particular random access protocol like IEEE 802.11. Such a classic technique is also used to describe interaction among a set of single-hop UDP flows in a IEEE 802.11 ad hoc network in [6]. However, the objective of the analysis in [6] and ours is completely different. In [6], the goal of the analysis is to evaluate the throughput of each link given the flow pattern. Whereas, the goal of our analysis is to optimize the TCP performance and resolve the starvation problem in multi-cell WLANs.

Due to this optimization driven purpose and the unique characteristics of TCP traffic, like TCP ACK packets and the link transmission reliability requirement, the methodology we developed for handling interactions between TCP and the MAC layer, packet collision probability at the MAC layer and throughput calculation is unique to our paper and is not addressed in [9,6,10].

3.2. Conditional collision probabilities

In this section, we derive the conditional collision probability of a link given that this link is active. We relate this conditional collision probability of each link to the steady state probability π. We denote a directional link from the node i to i' as $\ell_{ii'}$ and its conditional collision probability as $P(\ell_{ii'})$. In general, there are two types of collisions:

- Collisions at the receiver due to the overlap of two MAC data frames generated by a pair of hidden nodes, a *Data-to-data* collision;
- Collisions at the receiver due to the overlap of a MAC data frame and a MAC ACK frame generated by a pair of hidden nodes, a *Data-to-ACK* collision.

We represent the occurrence probability of a *Data-to-data* collision as P_{dd} and the occurrence probability of a *Data-to-ACK* collision as P_{da}. The conditional collision probability of the link $\ell_{ii'}$ is equal to

$$P(\ell_{ii'}) = 1 - (1 - P_{dd}(\ell_{ii'}))(1 - P_{da}(\ell_{ii'})). \tag{6}$$

3.2.1. Data-to-data collision

Data-to-data collisions occur when the transmission of two MAC data frames overlap at a common receiver. Both data frames are lost. It is due to the lack of synchronization between a pair of hidden nodes. For example, a MAC data frame (which contains a TCP ACK packet) coming from node l collides with another MAC data frame (which contains a TCP ACK packet) coming from node m at AP x as nodes l and m are hidden from each other.

Since links ℓ_{xl} and ℓ_{yr} are free from hidden node interference, we have $P_{dd}(\ell_{xl}) = P_{dd}(\ell_{yr}) = 0$. For other links, a data frame can be successfully transmitted to the receiver if and only if (1) the sender i starts the transmission to the receiver i' when the nodes hidden from receiver i' are not transmitting; (2) these hidden nodes remain silent for at least the next $T_{DATA}(i)$ slots. We denote the occurrence probability of the above two events as $p_1(\ell_{ii'})$ and $p_2(\ell_{ii'})$ respectively. We have $P_{dd}(\ell_{ii'}) = 1 - p_1(\ell_{ii'})p_2(\ell_{ii'})$.

To evaluate $p_1(\ell_{ii'})$, we need to first consider the system states where sender i can start its transmission. That is, all its neighbors in the synchronization graph are not transmitting. We denote the set of such states as $G(i)$. Within the set $G(i)$, we define the subset of states where the interfering hidden node is transmitting as $C(i, i')$. By definition, we have $C(i, i') \subset G(i)$. The probability $p_1(\ell_{ii'})$ is defined as

$$p_1(\ell_{ii'}) = 1 - \sum_{u \in C(i,i')} \pi(u) \times \left\{ \sum_{v \in G(i)} \pi(v) \right\}^{-1}. \tag{7}$$

Let us take link ℓ_{ry} as an example, we have $C(r, y) = \{S(4), S(8)\}$ and $G(r) = \{S(1), S(3), S(4), S(6), S(8)\}$.

To evaluate $p_2(\ell_{ii'})$, we have to derive the probability that the hidden node on receiver i' is going to start its transmission while sender i is sending. Given $\ell_{ii'}$, we define $h_{ii'}$ as the hidden node that conflicts with sender i at receiver i', for example, $h_{lx} = m$. We define $\Theta(N)$ as the set of system states in which all nodes in set N are transmitting. For example, we have $\Theta(l) = \{S(3), S(7), S(8), S(12), S(13)\}$ and $\Theta(\{l, m\}) = \{S(8), S(13)\}$. A hidden node cannot start a new transmission if any of its neighbors is active. While node i is transmitting, the set of system states where the hidden node $h_{ii'}$ can start its transmission is $G(h_{ii'}) \cap \Theta(i)$ and the set of system states where the hidden node $h_{ii'}$ is not transmitting is $\Theta(i) \setminus \Theta(\{i, h_{ii'}\})$. Therefore, while sender i is transmitting to node i', the hidden node $h_{ii'}$ starts a new transmission at any time slot with the probability

$$p_s(h_{ii'}) = \frac{\sum\limits_{u \in G(h_{ii'}) \cap \Theta(i)} \pi(u)}{\sum\limits_{v \in \Theta(i) \setminus \Theta(\{i, h_{ii'}\})} \pi(v)} \times \frac{1}{BO(h_{ii'})}. \tag{8}$$

Altogether, $p_2(\ell_{ii'})$ can be expressed as

$$p_2(\ell_{ii'}) = (1 - p_s(h_{ii'}))^{T_{DATA}(i)}. \tag{9}$$

We remind readers that $T_{DATA}(i)$ is measured in the unit of a physical time slot σ.

3.2.2. Data-to-ACK collision

We start the derivation of the Data-to-ACK collision probability with a simple example. A *data-to-ACK* collision occurs at link ℓ_{xm} if node r successfully transmitted a MAC data frame to AP y and AP y returns a MAC ACK frame to node r. At the same time, AP x is transmitting a data frame to node m. The data frame from AP x and the ACK frame from AP y collide at node m. Unlike the data-to-data collision, such packet loss is asymmetric. Node r and AP y are not affected and only the data frame from AP x to node m is lost. Similar collisions can occur at link ℓ_{lx}, ℓ_{ry}, and ℓ_{ym}. For other links $\ell \in \{\ell_{mx}, \ell_{xl}, \ell_{my}, \ell_{yr}\}$, we have $P_{da}(\ell) = 0$.

For a link $\ell_{ii'}$, we define $\ell_e(ii')$ as the link that causes a Data-to-ACK collision at link $\ell_{ii'}$ and we have: $\ell_e(lx) = \ell_{ym}$, $\ell_e(xm) = \ell_{ry}$, $\ell_e(ry) = \ell_{xm}$, and $\ell_e(ym) = \ell_{lx}$. We define $s(\ell_e(ii'))$ as the sender of link $\ell_e(ii')$ and $r(\ell_e(ii'))$ as the receiver. To cause a data-to-ACK collision on link $\ell_{ii'}$, node $s(\ell_e(ii'))$ has to be three hops away from sender i in the synchronization graph. The first necessary condition for such collisions to occur is that sender i starts its transmission when $s(\ell_e(ii'))$ is transmitting; we denote the occurrence probability of such an event as $p_e(\ell_{ii'})$. Since nodes x, m and y transmit data to multiple destinations, we define $f_i(i')$ as the fraction of time node i transmits to node i' for $i \in \{x, m, y\}$ and i' as the possible destinations of node i's transmission. We have

$$\begin{aligned} p_e(\ell_{lx}) &= \frac{f_y(m)\pi(2)}{\sum_{v\in G(l)} \pi(v)}, \qquad p_e(\ell_{xm}) = \frac{\pi(5)}{\sum_{v\in G(x)} \pi(v)}, \\ p_e(\ell_{ry}) &= \frac{f_x(m)\pi(6)}{\sum_{v\in G(r)} \pi(v)}, \qquad p_e(\ell_{ym}) = \frac{\pi(3)}{\sum_{v\in G(y)} \pi(v)}. \end{aligned} \tag{10}$$

For example, $p_e(\ell_{lx}) = f_y(m)\pi(2)/(\pi(1) + \pi(2) + \pi(4) + \pi(5) + \pi(9))$.

The second necessary condition that causes a data-to-ACK collision is a successful data frame transmission from $s(\ell_e(ii'))$ to $r(\ell_e(ii'))$ such that $r(\ell_e(ii'))$ returns a MAC ACK frame to $s(\ell_e(ii'))$. We approximate the probability of such an event by $(1 - P_{dd}(\ell_e(\ell_{ii'})))$. Putting it together, $P_{da}(\ell_{xm})$ and $P_{da}(\ell_{ym})$ are given by

$$\begin{aligned} P_{da}(\ell_{xm}) &= p_e(\ell_{xm}) \times (1 - P_{dd}(\ell_{ry})), \\ P_{da}(\ell_{ym}) &= p_e(\ell_{ym}) \times (1 - P_{dd}(\ell_{lx})). \end{aligned} \tag{11}$$

Since the transmission time for a TCP data packet is longer than that of a TCP ACK packet, when we derive the occurrence probability of the data-to-ACK collision on links ℓ_{lx} and ℓ_{ry}, we have to consider the effect of such an asymmetric packet size. For example, if node l starts its transmission too early, the MAC ACK frame from node m, due to a successful TCP data packet transmission from AP y, may not overlap the TCP ACK packet transmission from node l at AP x. Assuming that node l starts its transmission at any time slot randomly when AP y is transmitting, the probability that the MAC ACK frame collides with the MAC data frame at AP x is $T_F(l)/T_F(y)$. Therefore, $P_{da}(\ell_{lx})$ is defined as

$$P_{da}(\ell_{lx}) = p_e(\ell_{lx}) \times (1 - P_{dd}(\ell_{ym})) \times T_F(l)/T_F(y). \tag{12}$$

Similarly, we have

$$P_{da}(\ell_{ry}) = p_e(\ell_{ry}) \times (1 - P_{dd}(\ell_{xm})) \times T_F(r)/T_F(x). \tag{13}$$

3.3. Throughput calculation

The throughput of link ℓ is obtained by multiplying its normalized activation time $n(\ell)$ by its conditional successful transmission probability and the actual useful data rate. The normalized activation time of each link is defined as follows

$$\begin{aligned} &n(\ell_{lx}) = \Theta(l), \qquad n(\ell_{ry}) = \Theta(r), \\ &n(\ell_{xl}) = \Theta(x) \times f_x(l), \qquad n(\ell_{xm}) = \Theta(x) \times f_x(m), \\ &n(\ell_{yr}) = \Theta(y) \times f_y(r), \qquad n(\ell_{ym}) = \Theta(y) \times f_y(m), \\ &n(\ell_{mx}) = \Theta(m) \times f_m(x), \qquad n(\ell_{my}) = \Theta(m) \times f_m(y), \end{aligned} \tag{14}$$

where $f_x(l), f_y(r), f_m(x)$ are between 0 and 1 and $f_x(m) = 1 - f_x(l), f_y(m) = 1 - f_y(r), f_m(y) = 1 - f_m(x)$. The throughput of link $\ell_{ii'}$ can be obtained by

$$th(\ell_{ii'}) = n(\ell_{ii'}) \times (1 - P(\ell_{ii'})) \times \frac{T_{DATA}(i)}{T_F(i)} \times R, \tag{15}$$

where R is the physical layer data rate, for example, $R = 54$ Mbps in 802.11 g. We show how $f_x(l), f_y(r)$ and $f_m(x)$ can be determined in following section.

4. TCP performance optimization

In this section, we use our model to optimize TCP performance and resolve the starvation problem. In Section 4.1, we first construct the necessary constraints for TCP to work effectively in multi-cell WLANs. Then, given a throughput requirement for each TCP flow, for instance, 1 Mbps for flow 1, 2 Mbps for flow 2 and so on, we formulate a throughput allocation problem as a system of nonlinear equations which can be solved via numerical methods in Section 4.2. The solution yields the optimal MAC contention window setting for each node in the network. Nevertheless, in practice, one may be interested in how to operate the network at its full capacity. Therefore, in Section 4.3 we consider a more general optimization problem where we maximize the TCP throughput under certain fairness constraints without fixed throughput requirements. As the problem is non-convex, we propose an iterative algorithm to search for the global optimum. The stopping criteria are based on solutions of a series of throughput allocation problems. The throughput allocation problem, is therefore part of the routine for solving the more general problem, the throughput optimization problem.

4.1. TCP traffic constraints

As mentioned in the introduction, the starvation problem arises in our multi-cell WLAN example due to (1) the inhomogeneous contention experienced by different TCP flows at the MAC layer; and (2) the TCP retransmission mechanism. To resolve such a problem without modifying the behavior of TCP, we try to construct a reliability constraint on data transmission for each TCP flow. As reported in [11], the random packet loss probability has to be less than or equal to 10^{-3} at the link layer in order to allow TCP Reno to fully utilize the link bandwidth. Therefore, when we apply our model to allocate wireless link bandwidth for the TCP downlink traffic, we have to add the following constraints

$$
\begin{aligned}
&0.999 \le (1 - P(\ell_{xm})^k) \times (1 - P(\ell_{mx})^k),\\
&0.999 \le (1 - P(\ell_{ym})^k) \times (1 - P(\ell_{my})^k),\\
&0.999 \le (1 - P(\ell_{lx})^k) \times (1 - P(\ell_{xl})^k) = (1 - P(\ell_{lx})^k),\\
&0.999 \le (1 - P(\ell_{ry})^k) \times (1 - P(\ell_{yr})^k) = (1 - P(\ell_{ry})^k),
\end{aligned}
\tag{16}
$$

where k is the maximum transmission limit per data frame at the MAC layer. The first constraint in (16) is the reliability requirement for the downlink traffic from AP x to node m. $1 - P(\ell_{xm})^k$ is the probability that a TCP data packet transmission from AP x to node m succeeds and $1 - P(\ell_{mx})^k$ is the probability that node m returns a TCP ACK packet successfully to AP x. The overall successful transmission probability is the product of the two. The rest of the constraints apply similarly to flows 3, 1 and 4 respectively.

Similar to the TCP analysis in a single-cell WLAN [12–15], to cope with the problem complexity, we adopt the following assumptions:

(A1) The last hop wireless link is the bottleneck link of a TCP downlink flow.
(A2) The wireless channel is error-free.
(A3) The RTT between all APs and remote file servers is the same. That is, TCP RTT bias does not occur among the traffic sources and APs. This assumption is not vital to our scheme; it simply states that our scheme only targets the resolution of the unfairness that is due to the hidden terminals (not the bias that is due to RTT imbalance).
(A4) The delayed ACK mechanism is disabled.

Based on assumptions A1 to A3 and the reliability constraints, each TCP flow should be able to operate just like in a reliable wired network and the end-to-end throughput is constrained by the last hop wireless link. Therefore, if we consider the long term behavior of TCP, we have the following relation between links' throughput and end-to-end TCP flows' throughput

$$
\begin{aligned}
&th(\ell_{xl}) = t_1, \qquad th(\ell_{xm}) = t_2,\\
&th(\ell_{ym}) = t_3, \qquad th(\ell_{yr}) = t_4
\end{aligned}
\tag{17}
$$

where the target throughput requirement on TCP flow i is t_i for $i \in \{1, 2, 3, 4\}$.

Without a delayed ACK mechanism, the throughput requirement for the reverse TCP ACK packets is $\beta \times t_i$ where $\beta = T_{tcp_{ack}}/T_{tcp_{data}}$. We construct the following constraints for the ACK traffic,

$$
\begin{aligned}
&th(\ell_{lx}) = \beta \times th(\ell_{xl}), \qquad th(\ell_{mx}) = \beta \times th(\ell_{xm}),\\
&th(\ell_{ry}) = \beta \times th(\ell_{yr}), \qquad th(\ell_{my}) = \beta \times th(\ell_{ym}).
\end{aligned}
\tag{18}
$$

4.2. Throughput allocation problem

Given the throughput requirements t_i on TCP flow i, (4), (5) and (16)–(18) form a set of nonlinear constraints that describe the interaction between the TCP and MAC layers and how the throughput constraints should be enforced. The unknowns in these equations are $f_x(l), f_y(r), f_m(x)$ and $BO(n)$ for $n \in \{l, x, m, y, r\}$. Finding the values of these unknowns to fulfill the given values t_i is a feasibility problem which can be solved numerically [16] by formulating it as an optimization problem in which the objective function is equal to zero, that is

$$\begin{aligned} &\max \quad 0, \\ &s.t. \quad (4), (5), (16), (17), (18). \end{aligned} \tag{19}$$

Solving this problem with numerical methods, the value of the objective function will be equal to zero and a feasible point will be returned if the feasible set is nonempty. Although a detailed modeling of congestion window dynamic is omitted in our problem formulation, the simulation results in Section 5 show that our scheme nonetheless provides an accurate solution that leads to the target throughput.

4.2.1. Optimal window settings

After solving the bandwidth allocation problem, we can obtain the optimal average backoff time BO'. According to the IEEE 802.11 standard [1], the duration of each backoff period is uniformly distributed between 0 and CW where CW is the current contention window. The size of the contention window is governed by the binary exponential backoff procedure. To achieve the optimum average backoff time at nodes l, r and APs x, y, we can simply set $CW(n)_{\max}$ and $CW(n)_{\min}$ to $2 \times BO'(n)$ for $n \in \{l, r, x, y\}$. Approaches to achieve and maintain the optimal average backoff other than this fixed optimum contention window approach, e.g., statistical techniques, are also possible but are outside the scope of the present paper.

Node m is in fact a virtual node that instantiates the behavior of two real nodes which are associated with different APs. Denoting the node that communicates with AP x as m_x and the node that communicates with AP y as m_y, we have to design $BO'(m_x)$ and $BO'(m_y)$ based on $BO'(m)$ and $f'_m(x)$ where $f'_m(x)$ is the optimal value of $f_m(x)$. The virtual node m should allocate $f'_m(x)$ of its transmission time for sending data to AP x and $1 - f'_m(x)$ of its transmission time should be used to communicate with AP y at optimum.

Since the average backoff time between two consecutive transmissions is $BO'(m)$ for the virtual node m, we have the following relationship

$$\left(1 - \left(1 - \frac{1}{BO'(m_x)}\right)\left(1 - \frac{1}{BO'(m_y)}\right)\right)^{-1} = BO'(m). \tag{20}$$

As we assume the coordinated node collisions are negligible, given a node in set M is going to start a new transmission, the probability that this node is node m_x should be $f'_m(x)$ at optimum. That is,

$$\frac{\left(\frac{1}{BO'(m_x)}\right)\left(1 - \frac{1}{BO'(m_y)}\right)}{1 - \left(1 - \frac{1}{BO'(m_x)}\right)\left(1 - \frac{1}{BO'(m_y)}\right)} = f'_m(x). \tag{21}$$

Similarly, we have the following relationship for node m_y,

$$\frac{\left(1 - \frac{1}{BO'(m_x)}\right)\left(\frac{1}{BO'(m_y)}\right)}{1 - \left(1 - \frac{1}{BO'(m_x)}\right)\left(1 - \frac{1}{BO'(m_y)}\right)} = 1 - f'_m(x). \tag{22}$$

There is no feasible solution for Eqs. (20)–(22) except when $f'_m(x) = 0.5$. To approximate a solution, we first consider Eqs. (20) and (21). Solving these two equations, we obtain the unique solution

$$\begin{aligned} BO'(m_x) &= (BO'(m) - 1 + f'_m(x))/f'_m(x) + 1, \\ BO'(m_y) &= BO'(m)/(1 - f'_m(x)). \end{aligned} \tag{23}$$

Solving Eqs. (20) and (22) yields the unique solution:

$$\begin{aligned} BO'(m_x) &= BO'(m)/f'_m(x), \\ BO'(m_y) &= (BO'(m) - f'_m(x))/(1 - f'_m(x)) + 1. \end{aligned} \tag{24}$$

To balance between these two solutions fairly without having a bias against a particular set when $f'_m(x) \neq 0.5$, we approximate $BO'(m_x)$ and $BO'(m_y)$ as

$$\begin{aligned} BO'(m_x) &= BO'(m)/f'_m(x), \\ BO'(m_y) &= BO'(m)/(1 - f'_m(x)). \end{aligned} \tag{25}$$

We again set $CW_{\max}$ and $CW_{\min}$ of nodes m_x and m_y to $2 \times BO'(m_x)$ and $2 \times BO'(m_y)$.

4.2.2. *Transmission opportunity sharing at the APs*

The solution of the bandwidth allocation problem contains $f'_x(l)$ and $f'_y(r)$ which is the optimal fraction of transmission time APs x and y should spend on transmitting data to nodes l and r. We emphasize that our methodology does not require any modification or scheduling at the transport layer to enforce such division of transmission time. Such transmission opportunity sharing is enforced by controlling the throughput allocated on TCP ACK flows in our approach. In steady state, due to the TCP self-clocking behavior, by appropriately setting the transmission rate of the ACK packets, we can control the TCP source rate and prevent certain TCP flows from monopolizing all the transmission time at the AP.

4.2.3. *Extension to multiple clients*

Our model and discussion so far are limited to the case where $|L| = |R| = |M_x| = |M_y| = 1$. In this section, we discuss how to extend our model to multiple clients. To apply our model, we can set t_1, t_2, t_3 and t_4 as the aggregate downlink throughput requirements for all nodes in sets L, M_x, M_y, R respectively. Then, we obtain the values BO' and $f'_m(x)$ by solving the throughput allocation problem. Denoting the optimal average backoff time of a single node in set L as $BO'_i(l)$, we have the following relationship between $BO'_i(l)$ and $BO'(l)$,

$$\left(1-\left(1-\frac{1}{BO'_i(l)}\right)^{|L|}\right)^{-1} = BO'(l) \tag{26}$$

where $(1-\frac{1}{BO'_i(l)})^{|L|}$ is the probability that all nodes in set L are in backoff state at a time slot. Similarly, the optimal average backoff time of a single node in sets M_x, M_y and R, which are represented by $BO'_i(m_x)$, $BO'_i(m_y)$ and $BO'_i(r)$, are related to $BO'(m_x)$, $BO'(m_y)$, $BO'(r)$ via

$$\begin{aligned}
&\left(1-\left(1-\frac{1}{BO'_i(m_x)}\right)^{|M_x|}\right)^{-1} = BO'(m_x),\\
&\left(1-\left(1-\frac{1}{BO'_i(m_y)}\right)^{|M_y|}\right)^{-1} = BO'(m_y),\\
&\left(1-\left(1-\frac{1}{BO'_i(r)}\right)^{|R|}\right)^{-1} = BO'(r)
\end{aligned} \tag{27}$$

where $BO'(m_x)$ and $BO'(m_y)$ can be obtained by (25) given $f'_m(x)$. Solving Eqs. (26) and (27) yields the optimal average backoff time for each client in the network. We can again set the values of $CW_{\min}(c)$ and $CW_{\max}(c)$ to $2 \times BO'_i(c)$ for each client node c.

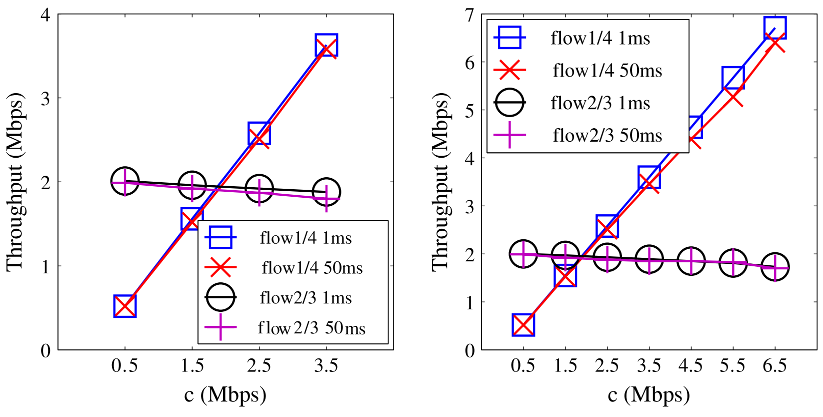
4.3. *Throughput optimization problem*

In this section, we optimize the network performance by maximizing each TCP flow's throughput subject to max–min fair or weighted max–min fair requirement by formulating a nonlinear optimization problem. Given the number of nodes in each set, we can formulate the optimization problem as

$$\begin{array}{rrcl}
 & \max & t, & \\
s.t. & \text{equations} & & (4), (5), (16), (18) \\
 & th(\ell_{xl}) & = & |L| \times w(L) \times t, \\
 & th(\ell_{xm}) & = & |M_x| \times w(M_x) \times t, \\
 & th(\ell_{ym}) & = & |M_y| \times w(M_y) \times t, \\
 & th(\ell_{yr}) & = & |R| \times w(R) \times t,
\end{array} \tag{28}$$

where $w(V)$ is the weight for set V and $V \in \{L, R, M_x, M_y\}$. The weight is used to provide weighted max–min fairness or service differentiation. If the weight is 1 for all sets, max–min fairness is the target.

Since our problem is a general non-convex problem, in theory, starting with different starting points may lead to different local optima. However, in our experience, even starting with multiple starting points, the same global optimum is reached. To ensure the quality of the solution, we propose a simple checking procedure. After we obtain the optimum solution t' by solving (28), we increase it by 5%, that is $t' = 1.05 \times t'$, then we recalculate $|L| \times w(L) \times t'$, $|M_x| \times w(M_x) \times t'$, $|M_y| \times w(M_y) \times t'$, $|R| \times w(R) \times t'$ and set them as the throughput requirements for the feasibility test in (19). As we specify all the throughput requirements, the decision of the feasible test is always the same even if we start with different starting points. If we find that the new t' is feasible, we increase it again and repeat the test. The whole test will be terminated when the latest t' is infeasible. The final solution will be the last feasible t' and the corresponding BO', $f'_m(x)$, $f'_x(l)$ and $f'_y(r)$.



(a) The maximum transmission limit $k = 7$. (b) The maximum transmission limit $k = 14$.

Fig. 7. Simulation and analysis results of the throughput allocation problem.

4.3.1. Extension to general scenarios

In this paper, though we only model a two cell scenario, the modeling techniques, like the calculation of successful transmission probability and construction of the Markov chain, are topology independent. The only requirement for applying all these techniques is the network synchronization graph, which can be obtained within millisecond-level time scales in practice, according to recent studies in [17,18]. Ahmed et al. show that interference like hidden node collisions can be identified in multi-cell WLANs whose APs are interconnected through a wired infrastructure network. Our optimization problem can be formulated based on the synchronization graph constructed by the micro-probing schemes proposed in [17,18].

5. Simulation and model validation

In this section, we validate our proposed scheme via ns-2 simulation with the IEEE 802.11 g standard and TCP Reno. The basic access scheme is used. The simulation parameters are listed in Table 1. We repeated each experiment 30 times with a different random seed in each trial. Each run lasts for 50 s, and quantities of interest are averaged over all the runs.

5.1. Throughput allocation problem

We first evaluate the accuracy of the solution of the feasibility problem for the network with $|L| = |M_x| = |M_y| = |R| = 1$. We assume that the throughput requirement on flows 2 and 3 is 2 Mbps and we denote the requirement on flows 1 and 4 as c. We start with $c = 0.5$ Mbps and gradually increase the requirement each time by 1 Mbps and we stop the increment when the problem becomes infeasible. We divide the test cases into two subgroups where the maximum transmission limit k is set to 7 and 14 respectively. In each set of throughput requirements, we conduct simulations with two different RTT_{as} settings which are $RTT_{as} = 1$ ms and $RTT_{as} = 50$ ms.

Fig. 7(a) shows the results when $k = 7$ and Fig. 7(b) shows the results when $k = 14$. Since the requirements on flows 1 (2) and 4 (3) are identical in each test case, the throughput of flows 1 (2) and 4 (3) are nearly the same in the simulation results. Therefore, for more clarity, we just report the average value of flows 1 (2) and 4 (3). A good matching can be observed in all cases. The average error in throughput allocation is 4% when $k = 7$ and is 5% when $k = 14$. By comparing Fig. 7(a) and (b), we can see that the maximum achievable throughput increases from 3.5 Mbps to 6.5 Mbps by changing k from 7 to 14 as higher reliability per TCP packet transmission is provided when $k = 14$ and fewer packets are dropped in the view of the TCP sender. However, the tradeoff is that the RTT between clients and APs is larger. We performed a similar simulation where $k = 21$ but we find the maximum achievable throughput is still 6.5 Mbps.

Table 5 compares the model prediction and simulation results of successful transmission probability which is equal to $1 - P$ where P is the conditional collision probability. We consider four cases in our previous simulation as examples. They are

(a) $k = 7$, $RTT_{as} = 1$ ms, $c = 3.5$ Mbps,
(b) $k = 7$, $RTT_{as} = 50$ ms, $c = 3.5$ Mbps,
(c) $k = 14$, $RTT_{as} = 1$ ms, $c = 6.5$ Mbps,
(d) $k = 14$, $RTT_{as} = 50$ ms, $c = 6.5$ Mbps.

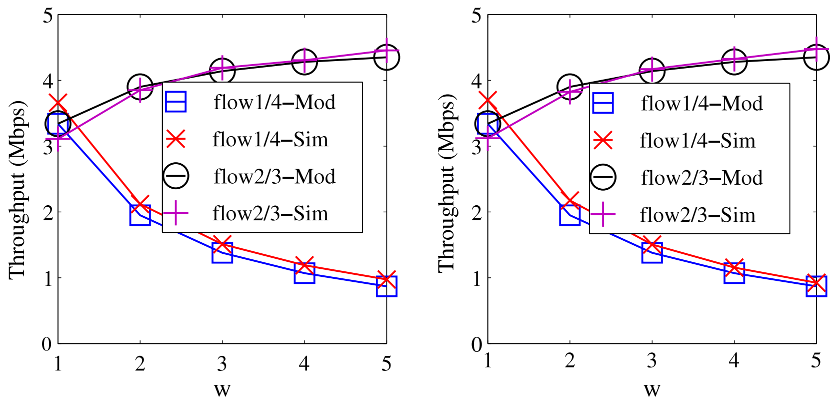
We define the TCP ACK flow from the node in set L to AP x as flow 5, from the node in set M to AP x as flow 6, from the node in set M to AP y as flow 7 and from the node in set R to AP y as flow 8. We can observe an excellent match between our predictions and the simulation results.

Table 5
Successful transmission probability.

Flow	Case a		Case b		Case c		Case d	
	Ana	Sim	Ana	Sim	Ana	Sim	Ana	Sim
1	1.00	1.00	1.00	1.00	1.00	0.99	1.00	0.99
2	0.64	0.61	0.64	0.63	0.41	0.36	0.41	0.36
3	0.64	0.62	0.64	0.63	0.41	0.36	0.41	0.39
4	1.00	1.00	1.00	1.00	1.00	0.99	1.00	0.99
5	0.91	0.94	0.91	0.95	0.88	0.93	0.88	0.93
6	0.93	0.95	0.93	0.95	0.85	0.90	0.85	0.90
7	0.93	0.96	0.93	0.95	0.85	0.89	0.85	0.91
8	0.91	0.94	0.91	0.94	0.88	0.92	0.88	0.93

Table 6
Confidence interval α (Mbps) of the average throughput.

RTT	1 ms				50 ms			
k	$k = 7$		$k = 14$		$k = 7$		$k = 14$	
c/flow	1/4	2/3	1/4	2/3	1/4	2/3	1/4	2/3
0.5	0.0020	0.0045	0.0024	0.0060	0.0023	0.0039	0.0029	0.0050
1.5	0.0051	0.0070	0.0043	0.0050	0.0054	0.0076	0.0060	0.0070
2.5	0.0069	0.0043	0.0073	0.0060	0.0134	0.0070	0.0114	0.0069
3.5	0.0096	0.0057	0.0085	0.0040	0.0199	0.0081	0.0136	0.0065
4.5	–	–	0.0099	0.0051	–	–	0.0217	0.0078
5.5	–	–	0.0114	0.0057	–	–	0.0190	0.0073
6.5	–	–	0.0156	0.0052	–	–	0.0440	0.0144



(a) $RTT_{as} = 10$ ms. (b) $RTT_{as} = 20$ ms.

Fig. 8. Simulation and analysis results of the throughput optimization problem.

To study whether our simulation has reached the steady state, we calculated the 99.98% confidence interval on the average throughput. The confidence interval is equal to $\bar{t} \pm \alpha$ where $\bar{t}$ is the average throughout obtained from the simulation results. Table 6 reports the value of α for the experiment. We can see that α is of little significance compared with the average throughput, which indicates the steady state is reached.

5.2. *Throughput optimization problem*

In this subsection, we evaluate the performance of the TCP optimization in (28). We first study the network performance when there is only one client in each set, i.e., $|L| = |M_x| = |M_y| = |R| = 1$, and then we study the situation of multiple clients.

5.2.1. *Single client per set*

We perform the TCP optimization for the example network in Fig. 1. We fix $W(L)$ and $W(R)$ as 1 and set $W(M_x) = W(M_y) = w$ where $w = \{1, 2, 3, 4, 5\}$. To achieve higher maximum end-to-end throughput, we set $k = 14$. Fig. 8(a) shows the results when $RTT_{as} = 10$ ms and Fig. 8(b) shows the results when $RTT_{as} = 20$ ms. We can observe an excellent agreement between the model prediction and simulation results. Flows 2 and 3 can achieve higher throughput than flows 1 and 4 because of the weight settings. The average error in prediction is 6% in both cases. The results show that the maximum

Table 7
Successful transmission probability.

Flow	$w = 1$		$w = 2$		$w = 3$		$w = 4$		$w = 5$	
	Ana	Sim	Ana	Sim	Ana	Sim	Ana	Sim	Ana	Sim
1	1.00	1.00	1.00	1.00	1.00	0.99	1.00	0.99	1.00	0.99
2	0.39	0.38	0.39	0.35	0.39	0.34	0.39	0.38	0.39	0.33
3	0.39	0.38	0.39	0.36	0.39	0.35	0.39	0.38	0.39	0.32
4	1.00	1.00	1.00	0.99	1.00	0.99	1.00	0.99	1.00	0.99
5	0.81	0.88	0.85	0.91	0.87	0.92	0.89	0.93	0.90	0.94
6	0.91	0.93	0.87	0.91	0.84	0.89	0.82	0.88	0.81	0.88
7	0.91	0.93	0.87	0.90	0.84	0.89	0.82	0.88	0.81	0.88
8	0.81	0.88	0.85	0.91	0.87	0.92	0.89	0.93	0.90	0.94

Table 8
Confidence interval α (Mbps) of the average throughput.

w	RTT (ms)	Flow 1/4	Flow 2/3	w	RTT (ms)	Flow 1/4	Flow 2/3
1	10	0.0035	0.0106	1	20	0.0343	0.0112
2	10	0.0109	0.0065	2	20	0.0216	0.0065
3	10	0.0148	0.0098	3	20	0.0057	0.0084
4	10	0.0223	0.0103	4	20	0.0233	0.0113
5	10	0.0218	0.0101	5	20	0.0021	0.0098

Table 9
Confidence interval α (Mbps) of the average throughput.

Set	Case 1	Case 2	Case 3	Case 4
L	0.0039	0.0071	0.0052	0.0031
M_x	0.0052	0.0044	0.0058	0.0043
M_y	0.0049	0.0086	0.0030	0.0074
R	0.0095	0.0045	0.0037	0.0121

throughput of flows 2 and 3 are bounded by 4.5 Mbps even when most of the bandwidth is allocated to flows 2 and 3. We believe this is due to the hidden terminal problem. The collision between TCP data packets of flows 2 and 3 limits their maximum achievable throughput.

Table 7 compares the model predictions and simulation results of successful transmission probability in the previous TCP optimization problem. We consider the case with an $RTT_{as} = 20$ ms in our previous simulation as an example. We can again observe an excellent matching between the prediction and the simulation. Table 8 reports the 99.98% confidence interval. We can again see that steady state is reached in our simulation.

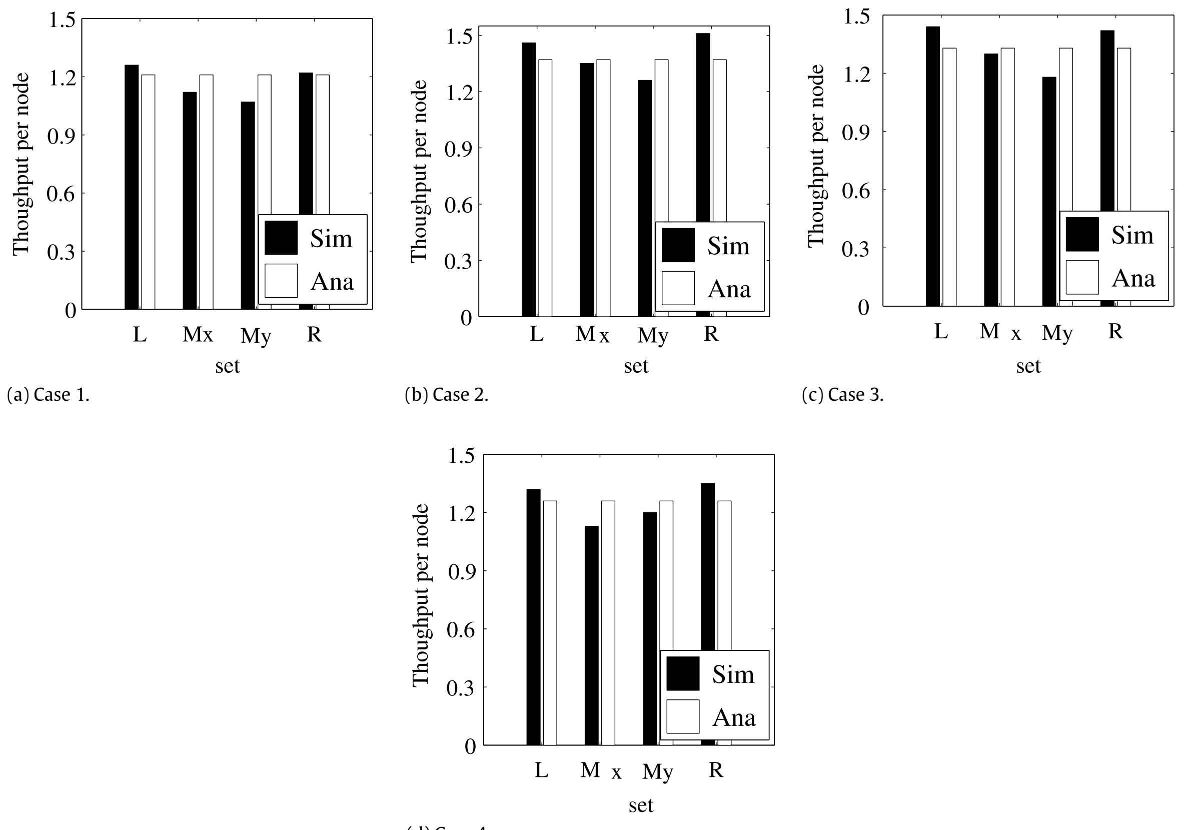
5.2.2. Multiple clients

Finally, we validate the TCP performance optimization when $|L|$, $|M_x|$, $|M_y|$ and $|R|$ is larger than 1. We randomly set $|L|$, $|M_x|$, $|M_y|$ and $|R|$ to an integer between 1 and 5. We set $RTT_{as} = 10$ ms, $k = 14$, $W(L) = W(R) = W(M_x) = W(M_y) = 1$ which is max–min throughput allocation. We repeated our simulation four times with the following random settings:

(1) $|L| = 4$, $|M_x| = 2$, $|M_y| = 3$, $|R| = 1$;
(2) $|L| = 2$, $|M_x| = 4$, $|M_y| = 1$, $|R| = 5$;
(3) $|L| = 3$, $|M_x| = 2$, $|M_y| = 3$, $|R| = 3$;
(4) $|L| = 5$, $|M_x| = 2$, $|M_y| = 2$, $|R| = 1$.

Fig. 9 compares the model predictions and simulation results of the average throughput of a client in different sets. The starvation problem is completely removed and each node is able to receive more than 1 Mbps end-to-end throughput. The average prediction error for cases 1 to 4 are 7%, 6%, 7% and 9% respectively and Jain's fairness index is 0.994, 0.997, 0995 and 0.992 respectively. The 99.98% confidence intervals of the average throughput shown in Table 9 indicates the convergence of the simulation. In all test cases, it takes less than 5 s to compute the optimum solution for both the feasibility and optimization problems.

We also simulate these scenarios with the original IEEE 802.11 g basic access scheme without tuning the windows. To have a fair comparison, we also set $k = 14$. The average throughput of each node is shown in Table 10. In all cases, the individual throughput of nodes in sets M_x and M_y is upper bounded by 0.2 Mbps. Similar to the example in Fig. 1, nodes in sets M_x and M_y starve throughout the whole simulation except the first few seconds. Most of the wireless bandwidth is allocated to the nodes in sets L and R. The standard deviation of their average throughput is shown in Table 11.



(a) Case 1. (b) Case 2. (c) Case 3. (d) Case 4.

Fig. 9. Simulation and analysis results of the optimization problem (Mbps).

Table 10
Average throughput per-node under IEEE 802.11 g (Mbps).

Set	Case 1	Case 2	Case 3	Case 4
L	4.48	8.88	6.05	3.83
M_x	0.164	0.027	0.089	0.061
M_y	0.101	0.068	0.075	0.045
R	16.16	3.94	5.79	18.43

Table 11
Standard deviation of per-node throughput under IEEE 802.11 g (Mbps).

Set	Case 1	Case 2	Case 3	Case 4
L	2.18	3.49	2.61	1.87
R	5.94	1.85	2.76	4.32

6. Related work

Most of the analytical models for IEEE 802.11 DCF are designed for single-cell WLANs. In the first wave of papers, analysis of the DCF with a simplified backoff procedure is done in [19,20]. In the seminal paper [21] and its generalization [22], the DCF is modeled as a two-dimensional Markov chain and a renewal process. The throughput of N saturated senders (including the AP) is determined by solving a set of fixed point equations. All the models in [21,19,20,22] do not consider the transport layer protocol.

The lack of joint TCP and MAC layer analysis in [21,19,20,22] makes all these models only apply to some specific practical situations. To model the interaction between the TCP and the MAC, a Markov chain model that includes the number of active TCP flows in a single-cell WLAN is proposed in [12]. Both uplink and downlink traffic are considered in this model and the model keeps track of the number of data packets in each TCP sender's buffer and the number of ACK packets in each TCP receiver's buffer through a two-dimensional random process. The stationary probability distribution of the buffer occupancy at the AP and wireless stations can be evaluated through the proposed Markov Chain. Similar Markov models are adopted in [14,15]. In [15], the Markov chain is extended to be multi-dimensional in which each state contains the information of the numbers of stations with a particular number of TCP packets in their buffer. In [13], Bruno et al. propose a Markov model that can compute the distribution of active stations when the traffic is a mix of long-lived TCP traffic and finite-load UDP traffic. They show that the active TCP connections and the aggregate UDP traffic can be modeled as two saturated flows. The main drawback of these Markov models is the state space explosion when the number of TCP flows is large as the model has to keep track on the activities of each TCP flow in terms of number of packets and ACKs.

A relatively small amount of work has been done on modeling multi-cell WLANs. In [23], Bonald et al. analyze the download traffic capacity of multi-cell WLANs by modeling each cell as a queue with state-dependent serving rate. The analysis starts from a theoretical point of view such that protocol details like uplink TCP ACK traffic and TCP rate control is not considered in the analysis. In [24], Bonald et al. extend the work in [23] and propose an opportunistic scheduling algorithm to handle the head of line blocking due to RTS/CTS exchanges in multi-cell WLANs. The traffic capacity of the proposed scheduling algorithm is evaluated through a fluid limit approach. Starting from a different objective, our proposed scheme is to optimize the network throughput under the operation of TCP and IEEE 802.11 DCF. Details like TCP ACK upstream flows, reliability on each TCP packet transmission and maximum number of transmissions per-MAC frame are included in our model.

Our analysis is somewhat related to those in CSMA multi-hop networks [9,6,10] as the hidden terminal problem arises frequently in multi-hop networks. In these works, the activation of different links or flows is modeled as a continuous time Markov chain. The throughput of each link is computed based on the steady probability distribution.

7. Conclusion

In this paper, we studied the throughput optimization problem of downlink TCP traffic in multi-cell WLANs under the influence of hidden terminals. We first identified a throughput starvation problem in such multi-cell WLANs. To resolve this problem, we proposed an analytical model that captures real world complexity like packet transmission limits, upstream flow of TCP ACK packets and hidden node collisions. Given different throughput requirements, we formulated the throughput allocation problem. Starting from this problem, we further formulated a throughput optimization problem subject to max–min fairness or weighted max–min fairness constraints. The solution of these two problems yields the optimum contention window setting of each node. By applying these optimal settings, the throughput unfairness problem can be eliminated without any modification at the transport layer. We evaluated and validated our scheme extensively through simulation. In the future, we intend to implement our proposed scheme on a real system and evaluate its performance.

References

[1] IEEE standard for information technology -telecommunications and information exchange between systems-local and metropolitan area networks-specific requirements - part 11: Wireless LAN Medium Access Control (MAC) and physical layer (PHY) specifications, June 12 2007.

[2] ns-2. http://www.isi.edu/nsnam/ns/.

[3] W.R. Stevens, TCP/IP Illusrated Volume 1: The Protocols, Addison-Wesley Publishing Company, 1994.

[4] V. Shrivastava, N. Ahmed, S. Rayanchu, S. Banerjee, S. Keshav, K. Papagiannaki, A. Mishra, CENTAUR: Realizing the full potential of centralized WLANs through a hybrid data path, in: ACM MobiCom, 2009.

[5] Iperf. http://iperf.sourceforge.net/.

[6] M. Garetto, T. Salonidis, E. Knightly, Modeling per-flow throughput and capturing starvation in CSMA multi-hop wireless networks, in: IEEE INFOCOM, 2006.

[7] S. Liew, C. Kai, J. Leung, B. Wong, Back-of-the-envelope computation of throughput distributions in CSMA wireless networks, IEEE Transactions on Mobile Computing 9 (9) (2010) 1319–1331.

[8] A. Leon-Garcia, Probability and Random Processes for Electrical Engineering, Addison-Wesley Publishing Company, 1994.

[9] R. Boorstyn, A. Kershenbaum, B. Maglaris, V. Sahin, Throughput analysis in multihop CSMA packet radio networks, IEEE Transactions on Communications 35 (3) (1987) 267–274.

[10] X. Wang, K. Kar, Throughput modelling and fairness issues in CSMA/CA based ad-hoc networks, in: IEEE INFOCOM, 2005.

[11] A. Kumar, Comparative performance analysis of versions of TCP in a localnetwork with a lossy link, IEEE/ACM Transactions on Networking 6 (4) (1998) 485–498.

[12] R. Bruno, M. Conti, E. Gregori, Performance modeling and measurements of tcp transfer throughput in 802.11-based WLANs, in: ACM MSWiM, 2006.

[13] R. Bruno, M. Conti, E. Gregori, Throughput analysis and measurements in IEEE 802.11 WLANs with tcp and udp traffic flows, IEEE Transactions on Mobile Computing 7 (2) (2008) 171–186.

[14] S. Choi, K. Park, C.-K. Kim, Performance impact of inter-layer dependence in infrastructure WLANs, IEEE Transactions on Mobile Computing 5 (7) (2006) 829–845.

[15] J. Yu, S. Choi, Modeling and analysis of TCP dynamics over IEEE 802.11 WLAN, in: WONS, 2007.

[16] S. Boyd, L. Vandenberge, Convex Optimization, Cambridge University Press, 2003.

[17] N. Ahmed, U. Ismail, S. Keshav, D. Papagiannaki, Online estimation of RF interference, in: ACM CoNEXT, 2008.

[18] N. Ahmed, S. Keshav, SMARTA: A self-managing architecture for thin access points, in: ACM CoNEXT, 2006.

[19] F. Cali', M. Conti, E. Gregori, IEEE 802.11 wireless LAN: Capacity analysis and protocol enhancement in: IEEE INFOCOM, 1998.

[20] H.S. Chhaya, S. Gupta, Performance modeling of asynchronous data transfer methods of IEEE 802.11MAC protocol, Wireless Networks 3 (1997) 217–234.

[21] G. Bianchi, Performance analysis of the IEEE 802.11 distributed coordination function, IEEE Journal on Selected Areas in Communications 18 (3) (2000) 535–547.

[22] A. Kumar, E. Altman, D. Miorandi, M. Goyal, New insights from a fixed-point analysis of single cell IEEE 802.11 WLANs, IEEE/ACM Transactions on Networking 15 (3) (2007) 588–601.

[23] T. Bonald, A. Ibrahim, J. Roberts, Traffic capacity of multi-cell WLANs, in: ACM SIGMETRICS, 2008.

[24] T. Bonald, A. Ibrahim, J. Roberts, Enhanced spatial reuse in multi-cell WLANs, in: IEEE INFOCOM, 2009.

Ka-Lok Hung received his B.Eng., M.Phil. and Ph.D. degree in Computer Science in 2004, 2006 and 2011 respectively from the Hong Kong University of Science and Technology (HKUST). His research interests include cross-layer optimization, mathematical modeling and performance evaluation in multi-cell WLANs and multi-hop wireless networks. He served on the technical program committees of IEEE GLOBECOM 2007, IEEE WCNC 2010, IEEE INFOCOM 2011 Workshop on Cognitive and Cooperative Networks and IEEE ICCT 2011, and has reviewed extensively for Annals of Telecommunications, IEEE Communications Letters, Elsevier Computer Networks and Performance Evaluation. In 2009, he received the Professor Samuel Chanson Teaching Assistant Award at HKUST. He is a student member of IEEE.

Brahim Bensaou received an Engineering Degree in Computer Science (with distinction) from the University of Science and Technology of Algiers, Algeria in 1982, and a DEA degree from University of Paris 11 in Computer Science in 1998. He enrolled in a Ph.D. program in 1990 and earned his Doctorate degree in Computer Science from the University of Paris 6 in late 1993. From 1990 to 1994, he was a research assistant at France Telecom Research labs near Paris, where he contributed to European research projects COST224 and COST242 on the design and performance evaluation of the then emerging ATM technology.

In mid 1995 he joined the Hong Kong University of Science and Technology (HKUST) as a Research Associate where he spent almost 2 years working on various problems in congestion and traffic control. In 1997 he joined the National R&D Center for Wireless Communications in Singapore (now known as the Institute of Infocomm Research, I2R A-Star) as a Member of Technical Staff, where he worked as System Architect on the design of Quality of Service (QoS) enabled MAC protocols and scheduling algorithms in Wireless ATM prototype project. His designs have been implemented in real prototype network cards in collaboration with Ericsson AB, Sweden. In 1998 he was promoted to a Senior Member of Technical Staff, and was instrumental in forming a small R&D group in the area of wireless networking. He led the group for a year and half then moved to Academia at HKUST in the fall of 2000, where he is now an Associate Professor of Computer Science and Engineering. He has published more than 100 research papers, received numerous research grants, graduated more than 20 postgraduate students and holds 3 US patents out of which one has been licensed in 2006.

Prof. Bensaou served as an Associate editor of IEEE Communications Letters, technical program co-chair of the 11th and 12th and 13th ACM Conferences on Modeling Analysis and Simulation of Wireless and Mobile Networks and technical program co-chair of the IEEE International Symposium on Wireless Pervasive Computing 2011. He served on the program committees of many international conferences including IEEE INFOCOM, IEEE GLOBECOM and IEEE ICC. He was a guest editor of a feature issue of OSA Journal of Optical Networking, Elsevier Computer Networks and Performance Evaluation. Professor Bensaou is a Senior member of IEEE and a member of ACM.