CapEst: A Measurement-based Approach to Estimating Link Capacity in Wireless Networks

Apoorva Jindal, Konstantinos Psounis, *Senior Member, IEEE* Mingyan Liu, *Senior Member, IEEE*

Abstract—Estimating link capacity in a wireless network is a complex task because the available capacity at a link is a function of not only the current arrival rate at that link, but also of the arrival rate at links which interfere with that link as well as of the nature of interference between these links. Models which accurately characterize this dependence are either too computationally complex to be useful or lack accuracy. Further, they have a high implementation overhead and make restrictive assumptions, which makes them inapplicable to real networks.

In this paper, we propose CapEst, a general, simple yet accurate, measurement-based approach to estimating link capacity in a wireless network. To be computationally light, CapEst allows inaccuracy in estimation; however, using measurements, it can correct this inaccuracy in an iterative fashion and converge to the correct estimate. Our evaluation shows that CapEst always converged to within 5% of the correct value in less than 18 iterations. CapEst is model-independent; hence, it is applicable to any MAC/PHY layer and works with auto-rate adaptation. Moreover, it has a low implementation overhead, can be used with any application which requires an estimate of residual capacity on a wireless link and can be implemented completely at the network layer without any support from the underlying chipset.

Index Terms—Capacity Estimation, Wireless Mesh Networks, Rate Control.

1 Introduction

The capacity of a wireless link is defined to be the maximum sustainable data arrival rate at that link. Estimating the residual capacity of a wireless link is an important problem because knowledge of available capacity is needed by several different tools and applications, including predicting safe sending rates of various flows based on policy and path capacity [17], online optimization of wireless mesh networks using centralized rate control [24], distributed rate control mechanisms which provide explicit and precise rate feedback to sources [20], admission control, interference-aware routing [32], network management tools to predict the impact of configuration changes [17] etc.

However, estimating residual link capacity in a wireless network, especially a multi-hop network, is a hard problem because the available capacity is a function of not only the current arrival rate at the link under consideration, but also of the arrival rates at links which interfere with that link and the underlying topology. Models which accurately represent this dependence are very complex and computationally heavy and, as input, require the complete topology information including which pair of links interfere with each other, the capture and deferral probabilities between each pair of links, the loss probability at each link, etc [6], [13], [17], [19], [32]. Simpler models make simplifying assumptions [5], [15], [16], [23], [24] which diminish their accuracy in real networks. Moreover, model-based capacity estimation techniques [17], [24] work only for the specific MAC/PHY layer for which they were designed and extending them to a new MAC/PHY layer requires building a new model from scratch. Finally, none of these methods work with auto-rate adaptation at the MAC layer, which makes them inapplicable to any real network.

In this paper, we propose CapEst, a general, simple yet accurate and model-independent, measurement-based approach to estimating link capacity in a wireless network. CapEst is an iterative mechanism. During each iteration, each link maintains an estimate of the expected service time per packet on that link, and uses this estimate to predict the residual capacity on that link. This residual capacity estimate may be inaccurate. However, CapEst will progressively improve its estimate with each iteration, and eventually converge to the correct capacity value. Our evaluation of CapEst in Sections 4 and 5 shows that, first, CapEst converges, and secondly, CapEst converges to within 5% the correct estimate in less than 18 iterations. Note that one iteration involves the exchange of roughly 200 packets on each link. (See Section 5 for more details.)

Based on the residual capacity estimate, CapEst predicts the constraints imposed by the network on the rate changes at other links. CapEst can be used with any application because the application can use it to predict the residual capacity estimate at each link, and behave accordingly. By a similar argument, CapEst can be used

- *Apoorva Jindal is with Juniper Networks Inc, Sunnyvale, CA. E-mail: ajindal@juniper.net.*
- *Mingyan Liu is with the Department of Electrical Engineering and Computer Science, University of Michigan, Ann Arbor. E-mail: mingyan@eecs.umich.edu.*
- *Konstantinos Psounis is with the Department of Electrical Engineering, University of Southern California, Los Angeles. E-mail: kpsounis@usc.edu.*

by any network operation to properly allocate resources. As a show-case, in this paper, we use CapEst for online optimization of wireless mesh networks using centralized rate control. Later, Section 6 briefly illustrates how CapEst is used with two other applications: distributed rate allocation and interference-aware routing.

The properties which makes CapEst unique, general and useful in many scenarios are as follows. (i) It is simple and requires no complex computations, and yet yields accurate estimates. (ii) It is model-independent, hence, can be applied to any MAC/PHY layer. (iii) The only topology information it requires is which node interferes with whom, which can be easily collected locally with low overhead [24]. (iv) It works with auto-rate adaptation. (v) It can be completely implemented at the network layer and requires no additional support from the chipset. (vi) CapEst can be used with any application which requires an estimate of wireless link capacity, and on any wireless network, whether single-hop or multi-hop.

2 Related Work

Model-based Capacity Estimation: Several researchers have proposed models for IEEE 802.11 capacity/throughput estimation in multi-hop networks [5], [6], [13], [15], [16], [17], [18], [19], [23], [24], [29], [32]. Model-based capacity estimation suffers from numerous disadvantages. They are tied to the model being used, hence, cannot be extended easily to other scenarios, like a different MAC/PHY layer. Moreover, incorporating auto-rate adaptation would make the models prohibitively complicated, so these techniques do not work with this MAC feature. Finally, and most importantly, the models used tend to be either very complex and require complete topological information [6], [13], [17], [19], [32], or make simplifying assumptions on traffic [18], [23], topology [15], [16] or the MAC layer [5], [24], [29] which significantly reduces their accuracy in a real network. Finally, there exists a body of work which studies throughput prediction in wireless networks with optimal scheduling [11]. If these are applied directly to predicting rates with any other MAC protocol, it will lead to considerably overestimating capacity.

Congestion/Rate Control: Numerous congestion control approaches have been proposed to regulate rates at the transport layer, for example, see [1], [20], [27], [28], [30], [31] and references therein. These approaches either use message exchanges between interfering nodes or use back-pressure. All these approaches improve fairness/throughput; however, they do not allow the specification of a well-defined throughput fairness objective (like max-min or proportional fairness). Moreover, most of them require changes to either IEEE 802.11 MAC or TCP. Our work is complementary to these works. We focus on how to accurately estimate capacity irrespective of the MAC or the transport layer. This capacity estimate, amongst many applications, can also be used to build a rate control protocol which allows the intermediate routing nodes to provide explicit and precise rate feedback to source nodes (like XCP or RCP in wired networks). It can also be used to set long-term rates while retaining TCP, similar to the model-based capacity estimation techniques of [17] and [24], to get better end-to-end throughputs.

Asymptotic Capacity Bounds: An orthogonal body of work studies asymptotic capacity bounds in multi-hop networks [7], [8], [10]. While these works lend useful insight into the performance of wireless networks in the limit, their models are abstract by necessity and cannot be used to estimate capacity in any specific real network.

3 CapEst Description

CapEst is an iterative mechanism. During each iteration, each link measures the expected service time per packet on that link, and uses this measurement to estimate the residual capacity on that link. The application, which for this section is centralized rate allocation to get max-min fairness, uses the estimated residual capacity to allocate flow-rates. The estimate may be inaccurate; however, it will progressively improve with each iteration, and eventually converge to the correct value. This section describes each component of CapEst in detail and the max-min fair centralized rate allocator.

We first define our notations. Let V denote the set of wireless nodes. A link $i \rightarrow j$ is described by the transmitter-receiver pair of nodes $i, j \in V, i \neq j$. Let $\mathcal{L}$ denote the set of active links. Let $\lambda_{i\rightarrow j}$ denote the packet arrival rate at link $i \rightarrow j$. Let $N_{i\rightarrow j}$ denote the set of links which interfere with $i \rightarrow j$, where a link $k \rightarrow l$ is defined to interfere with link $i \rightarrow j$ if and only if either i interferes with node k or l, or j interferes with node k or l. For convenience, $i \rightarrow j \in N_{i\rightarrow j}$. ($N_{i\rightarrow j}$ is also referred to as the neighborhood of $i \rightarrow j$ [20].)

In this description, we make the following assumptions. (i) The retransmit limit at the MAC layer is very large, so no packet is dropped by the MAC layer. (ii) The size of all packets is the same and the data rate at all links is the same. Note that these assumptions are being made for ease of presentation, and in Sections 3.3 and 3.4, we present modifications to CapEst to incorporate finite retransmit limits and different packet sizes and data rates respectively.

3.1 Estimating Capacity

We first describe how each link estimates its expected service time. For each successful packet transmission, CapEst measures the time elapsed between the MAC layer receiving the packet from the network layer, and the MAC layer informing the network layer that the packet has been successfully transmitted. This denotes the service time of that packet. Thus, CapEst is completely implemented in the network layer. CapEst maintains the value of two variables $\overline{S}_{i\to j}$ and $K_{i\to j}$[1], which denote the estimated expected service time and a counter to indicate the number of packets over which the averaging is being done respectively, at each link $i \to j$. For each successful packet transmission[2], if $S^{last}_{i\to j}$ denotes the service time of the most recently transmitted packet, then the values of these two variables are updated as follows.

$$\overline{S}_{i\to j} \leftarrow \frac{\overline{S}_{i\to j} \times K_{i\to j} + S^{last}_{i\to j}}{K_{i\to j}+1} \quad (1)$$

$$K_{i\to j} \leftarrow K_{i\to j} + 1. \quad (2)$$

$1/\overline{S}_{i\to j}$ gives the MAC service rate at link $i \to j$. Thus, the residual capacity on link $i \to j$ is equal to $(1/\overline{S}_{i\to j}) - \lambda_{i\to j}$. Now, since transmissions on neighboring links will also eat up the capacity at link $i \to j$, this residual capacity will be distributed amongst all the links in $N_{i\to j}$. Note that the application using CapEst, based on this residual capacity estimate, will either reallocate rates or change the routing or admit/remove flows etc, which will change the rate on the links in the network. However, this rate change will have to obey the following constraint at each link to keep the new rates feasible.

$$\sum_{k\to l \in N_{i\to j}} \delta_{k\to l} \leq \left(1/\overline{S}_{i\to j}\right) - \lambda_{i\to j}, \forall i \to j \in \mathcal{L}, \quad (3)$$

where $\delta_{k\to l}$ denotes the rate increase at link $k \to l$ in packets per unit time. How exactly is this residual capacity divided amongst the neighboring links depends on the application at hand. For example, Section 3.2 describes a centralized methodology to distribute this estimated residual capacity amongst interfering links so as to obtain max-min fairness amongst all flows. Similarly, Section 5.7 describes how to divide this estimated capacity to get weighted fairness.

Note that all interfering links in $N_{i\to j}$ do not have the same effect on the link $i \to j$. Some of these interfering links can be scheduled simultaneously, and some do not always interfere and packets may go through due to capture affect (non-binary interference [19]). However, the linear constraint of Equation (3) treats the links in $N_{i\to j}$ as a clique with binary interference. Hence, there may be some remaining capacity on link $i \to j$ after utilizing Equation (3) to ensure that the data arrival rates remain feasible. CapEst will automatically improve the capacity estimate in the next iteration and converge to the correct capacity value iteratively.

We refer to the duration of one iteration as the *iteration duration*. At the start of the iteration, both the variables $\overline{S}_{i\to j}$ and $K_{i\to j}$ are initialized to 0. We start estimating the expected service time afresh at each iteration because the residual capacity distribution in the previous iteration may change the link rates at links in $N_{i\to j}$, and hence change the value of $\overline{S}_{i\to j}$. Thus, retaining $\overline{S}_{i\to j}$ from the previous iteration is inaccurate. We next discuss how to set the value of the iteration duration. Note that CapEst has no overhead; it only requires each link to determine $\overline{S}_{i\to j}$ which does not require any message exchange between links. This however does not imply that there is no constraint on the choice of the iteration duration. Each link $i \to j$ has to measure $\overline{S}_{i\to j}$ afresh. Thus, an iteration duration has to be long enough so as to accurately measure $\overline{S}_{i\to j}$. However, we cannot make the iteration duration too long as it directly impacts the convergence time of CapEst. Moreover, the application which will distribute capacity will have overhead as it will need message exchanges to ensure Equation (3) holds. The choice of the value of the iteration duration is further discussed in Section 5.

3.2 Distributing Capacity to Obtain Max-Min Fairness

To illustrate how CapEst will be used with a real application, we now describe a centralized mechanism[3] to allocate max-min fair rates to flows.

Each link determines its residual capacity through CapEst and conveys this information to a centralized rate allocator. This centralized allocator is also aware of which pair of links in $\mathcal{L}$ interfere with each other as well as the routing path of each flow. Let $\mathcal{F}$ denote the set of end-to-end flows characterized by their source-destination pairs. Let r_f denote the new flow-rate determined by the centralized rate allocator, and let $r^{allocate}_{i\to j}$ denote the maximum flow rate allowed on link $i \to j$ for any flow passing through this link.

Consider a link $i \to j$. Let $I(f, i \to j)$ be an indicator variable which is equal to 1 only if flow $f \in \mathcal{F}$ passes through a link $i \to j \in \mathcal{L}$, otherwise it is equal to 0. Based on the residual capacity estimates, the centralized allocator updates the value of $r_f, f \in \mathcal{F}$ according to the following set of equations.

$$\begin{aligned} r^{max}_{i\to j} &\leftarrow r^{allocate}_{i\to j} + \alpha \frac{1/\overline{S}_{i\to j} - \lambda_{i\to j}}{\sum_{k\to l \in N_{i\to j}} \sum_{f\in\mathcal{F}} I(f, k \to l)} \\ r^{allocate}_{i\to j} &\leftarrow \min_{k\to l\in N_{i\to j}} r^{max}_{k\to l} \\ r_f &\leftarrow \min_{i\to j \in P_f} r^{allocate}_{i\to j}, \end{aligned} \quad (4)$$

where $0 < \alpha \leq 1$ is a parameter which controls the proportion of residual capacity distributed, $r^{max}_{i\to j}$ denotes the maximum flow-rate allowed by link $i \to j$ on any link in $N_{i\to j}$ and the set P_f contains the links lying on the routing path of flow f. Thus, amongst the links a flow traverses, its rate is updated according to the link

1. We could have used an exponentially weighted moving average to estimate $\overline{S}_{i\to j}$ too. However, since each packet is served by the same service process, giving equal weight to each packet in determining $\overline{S}_{i\to j}$ should yield better estimates, which we verify through simulations.

2. CapEst uses the aggregate stream of packets to estimate capacity, not per flow stream of packets. Thus, all packets of all flows will be used to update $\overline{S}_{i\to j}$ and $K_{i\to j}$.

3. Note that the centralized nature of the algorithm described in this section is due to the centralized max-min allocator and not CapEst, which does not require centralized operation. In Section 6.1, we discuss a distributed rate allocation algorithm based on CapEst.

with the minimum residual capacity in its neighborhood. Note that the residual capacity estimate may be negative, which will merely result in reducing the value of r_f.

This mechanism will allocate equal rates to flows which pass through the neighborhood of the same bottleneck link. According to the proof presented in [21], for CSMA-CA based MAC protocols, this property ensures max-min fairness.

3.3 Finite Retransmit Limits

A packet may be dropped at the MAC layer if the number of retransmissions exceeds the maximum retransmit limit. Since this packet was dropped without being serviced, what is its service time? Should we ignore this packet and not change the current estimate of the expected service time, or, should we merely take the duration for which the packet was in the MAC layer and use it as a measure of its service time? Note that these lost packets may indicate that the link is suffering from severe interference, and hence, imply that flows passing through the neighborhood of this link should reduce their rates. Ignoring lost packets is thus not the correct approach. Moreover, merely using the duration the lost packet spent in the MAC layer as the packet's service time is not sufficient to increase the value of the expected service time by an amount which leads to a negative residual capacity.

We use the following approach. We define the service time of a lost packet to be equal to the sum of the duration spent by the packet in the MAC layer and the expected additional duration required by the MAC layer to service the packet if it was not dropped. Assuming independent losses[4], the latter term is equal to $\frac{E[\text{Time to transmit a packet}]}{1-p^{loss}_{i\to j}}$, where $p^{loss}_{i\to j}$ is the probability that a DATA transmission on link $i \to j$ is not successful. This quantity depends on the specific MAC layer at hand. For example, for IEEE 802.11, this quantity can be approximated by $\frac{W_m/2+T_s}{1-p^{loss}_{i\to j}}$, where W_m is the largest back-off window value and T_s is the transmission time of a packet. Note that $p^{loss}_{i\to j}$ can be directly monitored at the network layer by keeping a running ratio of the number of packets lost to the number of packets sent to the MAC layer for transmission[5] after incorporating that each lost packet is transmitted as many times as the MAC retransmit limit.

3.4 Differing Packet Sizes and Data Rates

The methodology to estimate the expected service time (Equations (1) and (2)) remains the same. What changes is how to distribute this residual capacity amongst neighboring links, which is governed by the constraint of Equation (3). The objective of this constraint is to ensure that the sum of the increase in the proportion of time a link $k \to l \in N_{i\to j}$ transmits should be less than the proportion of time the channel around i and j is empty. Thus, if each link has a different transmission time, then the increase in rates on $k \to l \in N_{i\to j}$ as well as the residual capacity on $i \to j$ has to be scaled so as to represent the increase in airtime being consumed and the idle airtime around i and j respectively. Hence, if $\overline{T}_{i\to j}$ represents the average packet transmission time at link $i \to j$, then the capacity estimation mechanism will impose the following constraint.

$$\sum_{k\to l\in N_{i\to j}} \delta_{k\to l}\frac{\overline{T}_{k\to l}}{\overline{T}_{i\to j}} \le \left(1/\overline{S}_{i\to j}\right) - \lambda_{i\to j}, \forall i \to j \in \mathcal{L}. \quad (5)$$

Equation (5) merely normalizes rates to airtime. Note that Equation (3) is a special case of Equation (5) if the packet sizes and data rates at all links are the same.

Finally, the following equation states how the value of $\overline{T}_{i\to j}$ is estimated.

$$\overline{T}_{i\to j} \leftarrow \frac{\overline{T}_{i\to j} \times K_{i\to j} + \frac{P^{last}_{i\to j}}{D^{last}_{i\to j}}}{K_{i\to j}+1}, \quad (6)$$

where $P^{last}_{i\to j}$ and $D^{last}_{i\to j}$ denote the packet size of the last packet transmitted and the data rate used to transmit the last packet respectively for link $i \to j$.

4 Convergence of CapEst

In this section, we attempt to analytically understand the convergence properties of CapEst with the max-min fair centralized rate allocator under a WLAN topology, where all nodes can hear each other, all nodes are homogeneous, and each link has the same packet arrival rate. Let λ_{fin} denote the arrival rate at each edge such that the expected service time at each edge is equal to $1/\lambda_{fin}$. If $\lambda < \lambda_{fin}$, then the system is stable and the input arrival rate can be supported. Hence, the service rate $1/\overline{S}$ is larger than the arrival rate and the arrival rate is increased in the next iteration. On the other hand, $\lambda > \lambda_{fin}$, then the system is unstable and the service rate is smaller than the arrival rate. This leads to a reduction in the arrival rate in the next iteration. Thus, there is always a push to move λ towards λ_{fin}. However, it is not obvious whether this process would converge to $\lambda = \lambda_{fin}$ or keep oscillating.

The following theorem states a sufficient condition for the process to converge for a homogeneous WLAN topology.

Theorem 4.1: For a WLAN topology where all nodes can hear each other, with homogeneous nodes having small buffers, and with each link having the same packet arrival rate, λ_{fin} always exists and is unique, and CapEst, with the max-min fair centralized rate allocator with $\alpha < \frac{1}{2b_0}$, under the following assumptions: (i)

4. A more complex model accounting for correlated losses can be easily incorporated, however, our simulation results presented in Section 5.8 show that this simple independent loss model yields accurate results.

5. Actually, the correct approach is to use the probability of loss seen at the PHY layer, but measuring this quantity requires support from the chipset firmware. If the firmware supports measuring this loss probability, it should be used.

$nb_0 >> 1$, (ii) $b_0^2 >> 1$, (iii) $b_0 > 4$, (iv) $n > 3$ and (v) $b_0 \leq \frac{(n-2)T_s}{(n-1)\sigma}$, where n is the number of nodes, b_0 is the first average backoff window value, T_s is the packet transmission time and σ is the slot duration, with the initial rate chosen to be larger than $\frac{1}{nb_0\sigma+(n-1)\sigma+nT_s}$, always converges to λ_{fin}.

The details of the proof are given in the Appendix. We give a sketch of the proof here. We first determine that the set of equations representing the system is a fixed point formulation of the form $\lambda = \Psi(\lambda)$ where λ denotes the arrival rate at each node. CapEst iteratively solves this fixed point formulation. We show that the fixed point equation is continuous, hence by Brouwer fixed point theorem, the fixed point must exist. We then show that $\Psi(\lambda)$ is monotonically decreasing and concave, and then use the proof technique of [33] to prove that this implies that the fixed point is unique. To derive the convergence properties of CapEst, we next use induction to show that at iteration $k \geq 1$, with CapEst, $\Psi(\lambda_k) \geq \lambda_k$, which implies that at the congested edge $i \rightarrow j$, the value of $|\frac{1}{\bar{S}_{i\rightarrow j}} - \lambda_{i\rightarrow j}|$ reduces with each iteration. This together with a proof to show that the value of $\lambda_k, k > 1$ is upper bounded, proves that CapEst converges to the unique fixed point.

Note that the choice of α and the initial rate stated in the theorem only yield sufficient conditions for the WLAN topology. Later, through simulations, we observe that, with general multi-hop topologies, with fading and shadowing, heterogeneous nodes and different arrival rates for each link, even with $\alpha = 1$ and irrespective of the initial arrival rate, CapEst always converged to the correct value. And we observe that the convergence in all these simulations exhibit the property derived in the proof, that with each iteration, at the congested edge $i \rightarrow j$, $|\frac{1}{\bar{S}_{i\rightarrow j}} - \lambda_{i\rightarrow j}|$ continues to decrease. Thus, the modulus of the residual capacity estimate keeps on reducing with each iteration, and hence, $|\lambda - \lambda_{fin}|$ decreases with each iteration which ensures the convergence of CapEst.

5 Performance

In this section, we demonstrate through extensive simulations that CapEst not only converges quickly but also converges to the correct rate allocation. Thus, we verify both the correctness and the convergence of CapEst.

We evaluate CapEst with the max-min fair centralized rate allocator with $\alpha = 1$ over a number of different topologies, for different MAC protocols and with finite retransmit values at the MAC layer. We observe that CapEst *always* converges to within 5% of the optimal rate allocation in less than 18 iterations. Note that even though the evaluation of CapEst in this section is only through the max-min rate allocator application discussed in Section 3.2, any other application can as well be used to conduct a similar evaluation. Section 6 discusses two such applications.

Packet Payload	1024 bytes
MAC Header	34 bytes
PHY Header	16 bytes
ACK	14 bytes + PHY header
RTS	20 bytes + PHY header
CTS	14 bytes + PHY header
Channel Bit Rate	11 Mbps
Propagation Delay	1 μs
Slot Time	20 μs
SIFS	10 μs
DIFS	50 μs
Initial backoff window	31
Maximum backoff window	1023
α	1

TABLE 1
Simulation parameters.

5.1 Methodology

We use Qualnet version 4.0 as the simulation platform. All our simulations are conducted using an unmodified IEEE 802.11(b) MAC (DCF). RTS/CTS is not used unless explicitly stated. We use the default parameters of IEEE 802.11(b) (summarized in Table 1) in Qualnet. Unless explicitly stated, auto-rate adaptation is turned off, the link rate is set to 11 Mbps, the packet size is set to 1024 bytes and the maximum retransmit limits are set to a very large value. This setting allows us to first evaluate the performance of the basic CapEst mechanism without the modifications for auto-rate adaptation and finite retransmit limit. Later, we include both to evaluate the mechanisms to account for them. We run bulk transfer flows till $10,000$ packets per flow have been delivered. (Section 5.11 discusses the impact of having short-flows in the network.) Finally, to determine the actual max-min rate allocation, we use the methodology proposed by [20].

The implementation of CapEst and the max-min fair centralized rate allocator closely follows their description in Section 3. We choose the iteration duration to be 200 packets, that is, each link resets its estimate of expected service time after transmitting 200 packets[6]. Finally, to be able to correctly distribute capacity, the centralized rate allocator also needs to be aware of which links interfere with each other. We use the binary LIR interference model described in [24] to determine which links interfere. The link interference ratio (LIR) is defined as $LIR = \frac{c_{31}+c_{32}}{c_{11}+c_{22}}$ where c_{11}, c_{22} and c_{31}, c_{32} are UDP throughputs when the links are backlogged and transmit individually and simultaneously respectively. $LIR = 1$ implies no interference, with lower LIR's indicating a higher degree of interference. Similar to the mechanism in [24], links with $LIR > 0.95$ are classified as non-interfering.

6. Note that based on the convergence time requirements and overhead of the application, one can decide to choose a longer iteration duration too.

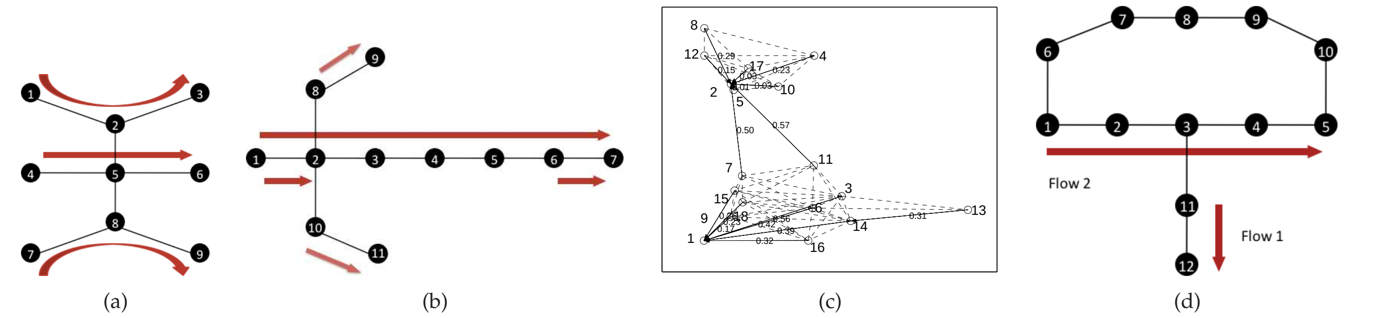


Fig. 1. (a) Flow in the Middle Topology. (b) Chain-cross Topology. (c) Deployment at Houston. (d) Twin-Flow topology.

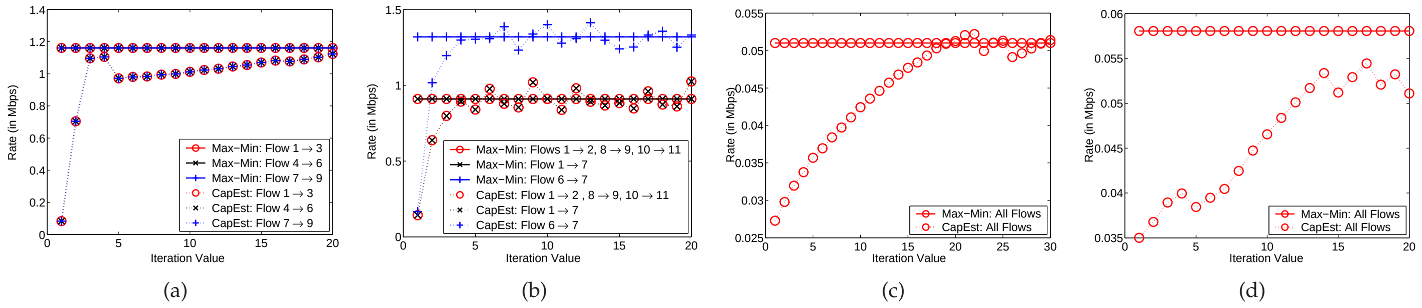


Fig. 2. Performance of CapEst. (a) Flow in the Middle. (b) Chain-Cross. (c) Random topology: 30 nodes, 5 flows. (d) Random topology: 100 nodes, 10 flows.

5.2 Commonly Used Multi-hop Topologies

In this section, we evaluate CapEst with two different, commonly used topologies. Simulations on these two topologies are conducted with zero channel losses, although packet losses due to collisions do occur. We adjusted the carrier sense threshold to reduce the interference range to be able to generate these topologies.

5.2.1 Flow in the Middle Topology

Figure 1(a) shows the topology. This topology has been studied and used by several researchers to understand the performance of different rate control and scheduling protocols in mesh networks [13], [20], [31]. Figure 2(a) plots the evolution of the rate assigned to each flow by the centralized rate allocator. We observe that the mechanism converges to within 5% of the optimal max-min rate allocation in less than 16 iterations.

5.2.2 Chain-Cross Topology

Figure 1(b) shows the topology. This topology was proposed by [20] to understand the performance of rate control protocols in mesh networks. This topology has a flow in the middle which goes over multiple hops ($1 \rightarrow 7$) as well as a smaller flow in the middle ($1 \rightarrow 2$). Figure 2(b) plots the evolution of the rate assigned to each flow by the centralized rate allocator. We see that the mechanism converges to within 5% of the optimal in less than 5 iterations.

5.3 Randomly Generated Topologies

We generate two topologies by distributing nodes in a square area uniformly at random. The source-destination pairs are also randomly generated. The first random topology has 30 nodes and 5 flows, while the second has 100 nodes and 10 flows. We use the two-ray path loss model with Rayleigh fading and log-normal shadowing [22] as the channel model in simulations. The carrier-sense threshold, the noise level and the fading and shadowing parameters are set to their default values in Qualnet. We use AODV to set up the routes. Figures 2(c) and 2(d) plot the evolution of the rate assigned to each flow by the centralized rate allocator. For the smaller random topology, the mechanism converges to within 5% of the optimal within 18 iterations.

For the larger topology, the mechanism converges to a rate smaller than the optimal. The reason is as follows. In this topology, the link which is getting congested first, or in other words, has the least residual capacity remaining at all iterations of the algorithm, is a high-loss link[7] (loss rate $> 40\%$). Hence, the iteration duration has to be larger than 200 packets to obtain an accurate estimate. If we increase the iteration duration to 500 packets, as shown in Figure 4(b), the mechanism converges to within 5% of the optimal within 15 iterations. Note that, in general, either routing schemes like ETX [4] will avoid the use of such high-loss links for routing, or auto-rate

7. A link which suffers from a large number of physical layer losses without including losses due to collisions is referred to as high-loss link.

adaptation will reduce the data rate to reduce the link loss rate, and hence, for most cases, an iteration duration of 200 packet suffices.

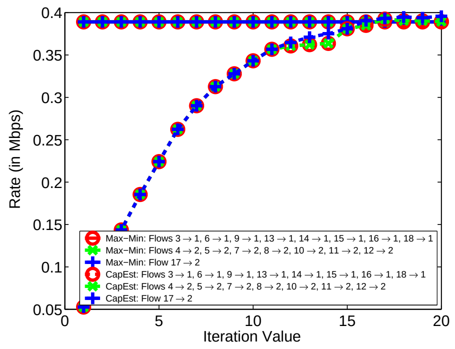


Fig. 3. Performance of CapEst: Deployment at Houston.

5.4 A Real Topology: Deployment at Houston

The next topology is derived from an outdoor residential deployment in a Houston neighborhood [2]. The node locations (shown in Figure 1(c)) are derived from the deployment and fed into the simulator. The physical channel that we use in the simulator is a two-ray path loss model with log-normal shadowing and Rayleigh fading. The ETX routing metric [4] (based on data loss in absence of collisions) is used to set up the routes. Nodes 1 and 2 are connected to the wired world and serve as gateways for this deployment. All other nodes route their packets toward one of these nodes (whichever is closer in terms of the ETX metric). The resulting topology as well as the routing tree is also shown in Figure 1(c).

Figure 3 plots the evolution of the rate assigned to each flow by the centralized rate allocator. Again, the mechanism converges to within 5% of the optimal in less than 12 iterations.

5.5 Impact of a Smaller Iteration Duration

In this section, we evaluate the impact of using a smaller iteration duration on performance. We plot the evolution of the rate assigned to each flow in Figure 4(a) for the flow in the middle topology with one iteration duration = 75 packets. We observe that the rate converges to a value smaller than the optimal. (Note that we had made a similar observation for the 100 node random topology in Section 5.3.) In general, for all topologies we studied, we observed that using an iteration duration not sufficiently large to allow the estimated expected service time to converge leads to the mechanism converging to a rate smaller than the optimal, however, the mechanism still always converged and did not suffer from oscillations.

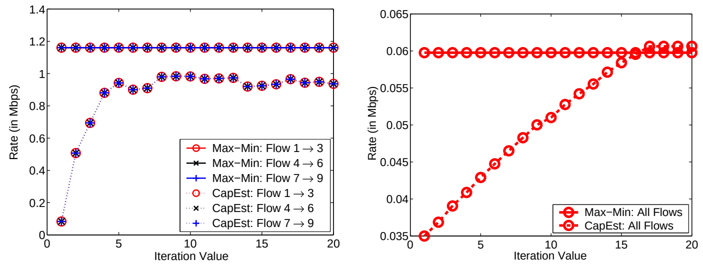


Fig. 4. Performance of CapEst with a different iteration duration. (a) Flow in the Middle with iteration duration = 75 packets. (b) Random topology: 100 nodes, 10 flows with iteration duration = 500 packets.

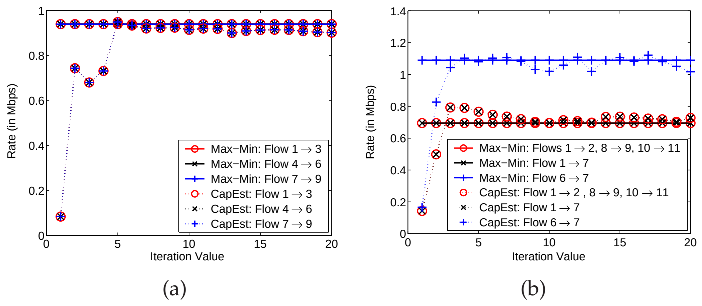


Fig. 5. Performance of CapEst with IEEE 802.11 with RTS/CTS. (a) Flow in the Middle. (b) Chain-cross.

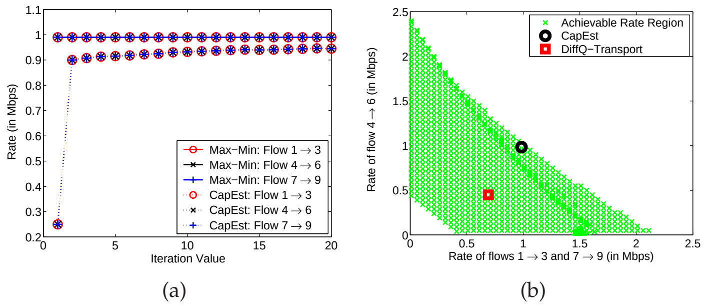


Fig. 6. (a) Performance of CapEst with DiffQ-MAC: Flow in the Middle. (b) CapEst vs DiffQ-Transport over DiffQ-MAC.

5.6 Different MAC layers

An attractive feature of CapEst is that it does not depend on the MAC/PHY layer being used. Hence, in future, if one decides to use a different medium access or physical layer, CapEst can be retained without any changes. In this section, we evaluate the performance of CapEst with two different medium access layers.

5.6.1 With RTS/CTS

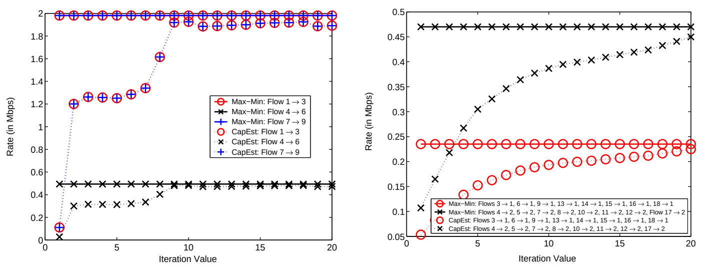
We first evaluate the performance of CapEst with IEEE 802.11 DCF with RTS/CTS. Figures 5(a) and 5(b) plot the evolution of the assigned flow rates for the flow in the middle topology and the chain-cross topology respectively. For both the topologies, the mechanism converges to within 5% of the optimal within 6 iterations.

5.6.2 Back-pressure MAC

We next evaluate the performance of CapEst with a back-pressure-based random-access protocol [9], [14], [30]. The fundamental idea behind back-pressure based medium access is to use queue sizes as weights to determine which link gets scheduled. Solving a max-weight formulation, even in a centralized manner, is NP-hard [26]. So, multiple researchers have suggested using a random access protocol whose channel access probabilities inversely depend on the queue size [9], [14], [30]. This ensures that the probability of scheduling a packet from a larger queue is higher. The most recent of these schemes is *DiffQ* [30]. DiffQ comprises of both a MAC protocol as well as a rate control protocol. We refer to them as DiffQ-MAC and DiffQ-Transport respectively. The priority of each head of line packet in a queue is determined by using a step-wise linear function of the queue size, and each priority is mapped to a different AIFS, CWMin and CWMax parameter in IEEE 802.11(e).

Back-pressure medium access is very different from the traditional IEEE 802.11 DCF in conception. However, CapEst can still accurately measure the capacity at each edge. Figure 6(a) plots the evolution of the assigned flow rates for the flow in the middle topology with DiffQ-MAC. The different priority levels as well as the AIFS, CWMin and CWMax values being used are the same as the ones used in [30]. Again, the mechanism converges to the optimal values within 5% of the optimal within 12 iterations.

This set-up also demonstrates the advantage of having a rate control mechanism which does not depend on the MAC/PHY layers. Optimal rate-control protocols for a scheduling mechanism which solves the max-weight problem at each step are known. However, if we use a distributed randomized scheduling mechanism like DiffQ-MAC, these rate control protocols are no longer optimal. But, using a rate allocation mechanism based on CapEst, which makes no assumption on the MAC layer, ensures convergence to a rate point close to the optimal. For example, Figure 6(b) plots the achievable rate region (or the feasible rate region) for DiffQ-MAC for the flow in the middle topology. We plot the rate of the middle flow against the rate of the outer two flows. (Using symmetry to assume that the rate of the outer flows is equal simplifies the figure as it becomes a two-dimensional figure instead of three-dimensional one.) Figure 6(b) also plots the throughput achieved by CapEst after 15 iterations of the algorithm as well as the throughput achieved by DiffQ-Transport (originally proposed by [3] and shown to be optimal with centralized max-weight scheduling). The figure shows that CapEst allocates throughput within 5% of the optimal while DiffQ-Transport achieves only 55% of the optimal throughput. Thus, the decentralized rate control of DiffQ-Transport loses its optimality with DiffQ-MAC as it was designed to be optimal for the centralized max-weight scheduling scheme. On the other hand, CapEst which is designed to achieve optimal throughput independent of the scheduling scheme achieves near-optimal throughput.



(a) (b)

Fig. 7. Performance of CapEst with weighted fairness. (a) Flow in the Middle. (b) Deployment at Houston.

5.7 Weighted Fairness

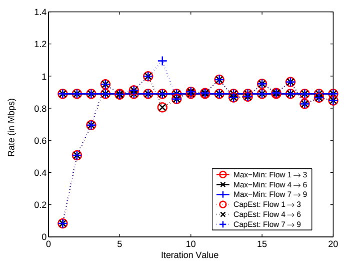
Section 3.2 describes how to use CapEst to obtain max-min fairness. However, the same mechanism can easily be adapted to obtain weighted fairness, which is defined as follows. Let $w_f > 0, \forall f \in \mathcal{F}$ denote the weight of a flow. Then, weighted fairness will allocate rates $r_f, \forall f \in \mathcal{F}$ such that $r_{f_1}/r_{f_2} = w_{f_1}/w_{f_2}, \forall f_1, f_2 \in F$, the allocated flow rates are feasible, and the rate of no flow can be increased without reducing the rate of any other flow.

Given $w_f, \forall f \in \mathcal{F}$, we now describe a modification of the methodology proposed in Section 3.2 to achieve weighted fairness. The central allocator will now update the value of $r_{i\to j}^{max}$ according to the following equation: $r_{i\to j}^{max} = r_{i\to j}^{allocate} + \frac{1/E[S_{i\to j}]-\lambda_{i\to j}}{\sum_{k\to l \in N_{i\to j}} \sum_{f\in\mathcal{F}} w_f I(f,k\to l)}$. The equation to update the value of $r_{i\to j}^{allocate}$ remains the same, and the new flow rates are updated as $r_f^{new} = \min_{i\to j \in P_f} w_f r_{i\to j}^{allocate}$.

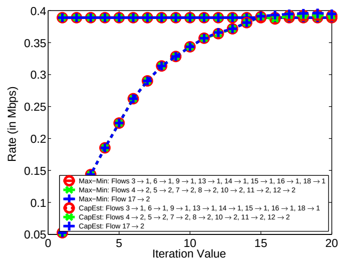
Figures 7(a) and 7(b) plot the evolution of flow rates with CapEst for the flow in the middle topology and the deployment at Houston respectively where $w_{1\to 3} = w_{7\to 9} = 4$ and $w_{4\to 6} = 1$ for the flow in the middle topology, while for the deployment at Houston, $w_f = 1$ for all flows f being routed towards gateway 1 and $w_f = 2$ for all flows f being routed towards gateway 2. CapEst converges to within 5% of the optimal within 18 iterations of the algorithm.

5.8 Finite Retransmit Limits

In this section, we set the IEEE 802.11 retransmit limits to their default values. Thus, packets may be dropped at the MAC layer. We use the methodology proposed in Section 3.3 to update the estimate of the expected service time for lost packets. Figures 8(a), and 8(b) show the evolution of allocated rates for the flow in the middle topology and the deployment at Houston respectively. There is slightly more variation in the allocated rates, however, not only does CapEst converge but also the



(a)



(b)

Fig. 8. Performance of CapEst with finite MAC retransmit limits. (a) Flow in the Middle. (b) Deployment at Houston.

convergence time remains the same as before. We evaluate CapEst for all the other scenarios described above with the default retransmit values for IEEE 802.11, and our observations remain the same as before.

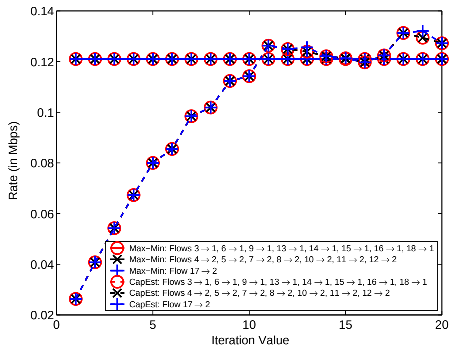
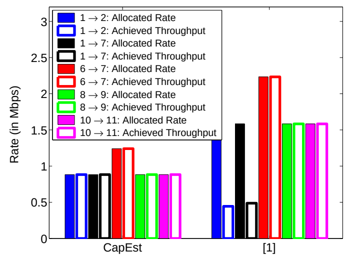


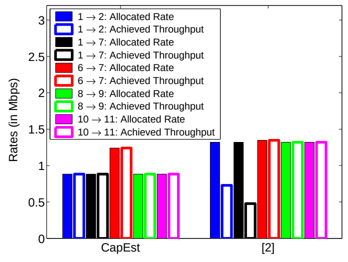
Fig. 9. Performance of CapEst with auto-rate adaptation: Deployment at Houston.

5.9 Auto-Rate Adaptation

In this section, we switch on the default auto-rate fallback mechanism of Qualnet. We use the methodology proposed in Section 3.4 to constrain the rate updates. Figure 9 show the evolution of allocated rates for the deployment at Houston. Again, not only does CapEst converge to the correct rate allocation but also the convergence time remains the same as before.

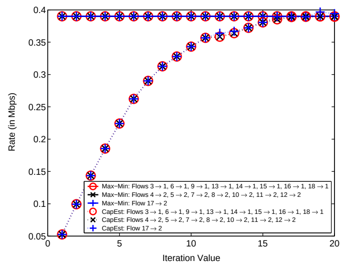


(a)

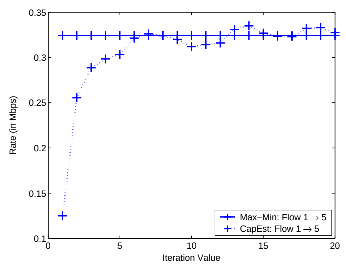


(b)

Fig. 10. (a) CapEst vs [17]. (b) CapEst vs [24].



(a)



(b)

Fig. 11. Performance of CapEst. (a) With short flows; the deployment at Houston topology. (b) With external interference; Twin-flow topology.

5.10 Comparison with Prior Works

Two prior works have discussed centrally allocating max-min fair rates for a given topology [17], [24]. [17] assumes that the entire topology is precisely known, and then sets up a convex optimization problem based on an approximate model for IEEE 802.11 DCF without RTS/CTS to determine the exact flow rates. [24] assumes that only which node interferes with whom is known, and then sets up a linear program based on an approximate model for IEEE 802.11 DCF to determine the exact flow rates. Note that the model used by [17] is more complex as well as more accurate than [24]. Both these methods are model-based, and hence suffer from all the shortcomings any model-based capacity estimation technique suffers from. Additionally, since they are based on approximate models, they will yield either an over-estimate or an under-estimate of the actual rates, and as reported by [17], [24], this over or under estimation can be more than 40% of the actual value.

Figure 10(a) compares the rates assigned as well as the actual throughput achieved by CapEst after 15 iterations and the method proposed in [17] (set-up to obtain max-min fairness) while Figure 10(b) compares the same for CapEst and the method proposed in [24] (set-up to obtain max-min fairness) for the chain-cross topology. We observe that the methodologies of both the prior works overestimate capacity which leads to an infeasible rate allocation leading to a lot of packet drops and a much lower actual throughput value for the two flows in the middle ($1 \rightarrow 2$ and $1 \rightarrow 7$).

Note that the initial flow rates can be assigned based on the solution of either of these two methods. However, after assigning these initial rates, the CapEst mechanism can be used to converge to the correct flow rates. In other words, instead of starting CapEst from near-zero rates, its initial starting rate allocation can be determined using either one of these two methods.

5.11 Short Flows

The 200 packets in an iteration duration can belong to any flow, long-term or short-term. Hence, CapEst does not need any additional support when numerous short flows are present in a network. For example, the centralized rate allocation mechanism can easily determine the rate of a new short flow based on the previous residual capacity estimate. If we introduce 50 short flows of 10 packets each in the deployment at Houston topology, at randomly generated times after the 10^{th} iteration, in addition to the 16 long flows, and choose the source of each short flow randomly, we observe that each of these short flows are allocated adequate capacity allowing them to complete their data transfer quickly within 200 ms, without any discernible impact on the rates allocated to the long-term flows, as observed in Figure 11(a).

5.12 External Interference

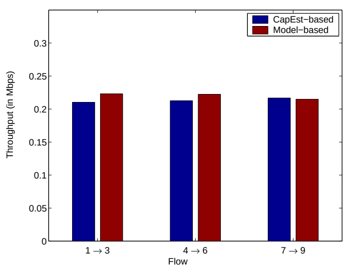
The presence of external interference merely increases the expected service time at links, which reduces the residual capacity estimate. Hence, unlike model-based capacity estimation techniques, CapEst does not need any additional support or mechanisms to account for external interference. Consider the twin-flow topology shown in Figure 1(d). Let flow $11 \rightarrow 12$ be a CBR flow sending data at the rate of 3 Mbps. Let flow $1 \rightarrow 5$ be routed along the path $1 \rightarrow 2 \rightarrow 3 \rightarrow 4 \rightarrow 5$. If CapEst with the max-min centralized rate allocator is used to set the rate of flow $1 \rightarrow 5$, then flow $11 \rightarrow 12$ will serve as a source of external interference. Figure 11(b) plots the evolution of the rate allocated to flow $1 \rightarrow 5$ in this scenario, and we observe that the rate allocated converges to within 5% of the optimal within 6 iterations.

6 Other Applications

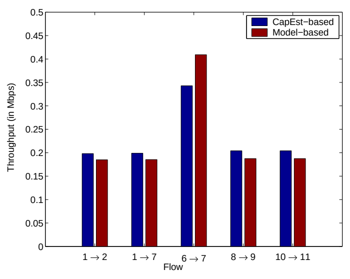
Accurately estimating capacity is an important tool which can be used in many different applications. So far we have only used centralized rate allocation as an example to study the properties of CapEst. As discussed in the Introduction, CapEst can also be used with a number of other applications. To illustrate how CapEst will get modified when used with these applications, in this section, we will explore how CapEst fits into the following two applications: distributed rate allocation and interference-aware routing.

6.1 Distributed Rate Allocation

WCPCap is one of the recent distributed rate control algorithms for mesh networks which uses capacity estimation to allow the intermediate router nodes to explicitly and precisely tell the source nodes the rate to transmit at [20]. It comprises of two parts. The first part estimates residual capacity at each link using a complex model for IEEE 802.11 with RTS/CTS in multi-hop networks, and the second part divides this estimated capacity amongst end-to-end flows in a distributed manner to achieve max-min fairness. Since WCPCap uses a specific model to estimate capacity, it suffers from all the drawbacks which any model-based capacity estimation technique suffers from.



(a)



(b)

Fig. 12. Performance of CapEst implemented within WCPCap. (a) Flow in the middle topology. (b) Chain-cross topology.

To illustrate the utility of CapEst in distributed rate allocation, we combine CapEst and WCPCap. CapEst replaces the first part of WCPCap to estimate capacity, while the second part which divides this estimated capacity is retained as such. We evaluate the performance of CapEst with WCPCap using Qualnet simulations. Default parameters of IEEE 802.11 in Qualnet (summarized in Table 1) are used. Retransmit limits are set to their default values and auto-rate adaptation is switched off[8]. The WCPCap parameters are set to their values used in [20]. Figures 12(a) and 12(b) compare the performance of model-based WCPCap and CapEst-based WCPCap for the flow in the middle and the chain cross topologies respectively. We observe that WCPCap with CapEst is slightly fairer than the model-based WCPCap as model-based WCPCap is only as accurate as the underlying model. For example, for the chain-cross topology, CapEst allocates a higher rate to flows $1 \rightarrow 2$, $1 \rightarrow 7$, $8 \rightarrow 9$ and $10 \rightarrow 11$ by reducing the rate of flow $6 \rightarrow 7$.

Implementation Overhead: Implementation overhead for distributed rate allocation mechanisms which estimate capacity, like WCPCap, tends to be prohibitive [20] because of the following two reasons. (i) Complete topology information has to be collected and maintained to seed the model. (ii) The number of messages exchanged between links is a linear function of the number of interfering links each of the two links have. With CapEst, both these overheads are reduced significantly. First, as discussed earlier, the only topological information needed by a node is its neighboring nodes, which can be collected with very low overhead [24]. Second, the size of messages required is smaller, and similarly to prior works [12], these messages can be encoded in the IP header in the following fields: Type-of-Service, Identification, Flags and Fragmentation Offset. (This strategy is based on the observation that the amount of fragmented IP traffic is negligible [25].)

8. As stated earlier, model-based capacity estimation techniques, like model-based WCPCap, become prohibitively expensive with auto-rate adaptation.

6.2 Interference-aware Routing

To determine routes offering maximum capacity, Gao *et al.* [32] discussed a model-based approach. Given a path in a multi-hop network, the end-to-end throughput is determined by the minimum link capacity of the path. So if one can figure out each link's capacity in a given path, then the minimum one is the end-to-end throughput capacity of this path. And amongst multiple paths between a source and a destination, the path which offers the maximum throughput is the best route.

Since CapEst can be used to determine the capacity at each link, it can easily be used to determine the best route. We use the same topology as used in [32] (Figure 1(d)) to evaluate CapEst. Flow 2 (between 1 and 5) has two candidate paths. Path 1 is $1 \to 2 \to 3 \to 4 \to 5$, and path 2 is $1 \to 6 \to 7 \to 8 \to 9 \to 10 \to 5$. Assuming default IEEE 802.11 parameters, summarized in Table 1, data rate of 11 Mbps, packet size of 1024 bytes and the rate of existing flow 1 (between 11 and 12) to be 3 Mbps, we evaluate the capacity offered by both paths by sending 200 packets back-to-back along each path, to be 0.36 Mbps and 1.28 Mbps respectively. Thus, path 2 is a much better route even though it has more hops than path 1. Using CapEst, we arrive at the same conclusion as [32] with a much smaller overhead and without requiring any complex computations or any assumptions on the MAC/PHY layer being used.

7 Limitations

In this section, we discuss two limitations which CapEst suffers from.

Iterative Mechanism: The initial capacity estimate provided by CapEst can be inaccurate, however, CapEst progressively improves its estimate with each iteration. Thus, CapEst is an excellent capacity estimation tool for applications which allow their behavior to be updated based on the new and more accurate capacity estimate. For example, applications which allocate resources, like rate allocation and interference-aware routing, can update the resources allocated to different flows based on the new estimate without affecting the network operation, and hence, are perfect candidates for using CapEst. However, applications which are required to provide an answer straightaway and are not allowed to change/update it later can not use an iterative mechanism. For example, admission control which has to provide a yes/no answer on whether to admit a new flow can not later update its answer to no after having admitted a flow.

Convergence Time: From the simulations, we observe that CapEst converges to within 5% of the optimal rate within 18 iterations where each iteration requires the exchange of 200 packets per link. This takes 2 seconds for the flow in the middle topology and 8 seconds for the deployment at Houston. Thus, the convergence time may become too large for some applications. However, note that, for all topologies we study, CapEst converges to 60% of the final rate within 6 iterations (500 ms for the flow in the middle topology and 2.5 s for the deployment at Houston), and then it takes a number of iterations after that to converge to within 5% of the final rate[9]. This allows the max-min rate allocator to utilize most of the capacity within the first few iterations. Also, the convergence time depends on the initial flow-rates. In all simulations, we start with a small initial flow-rate. If one can get a better estimate for the initial rates, the convergence time will be smaller. For example, as discussed in Section 5.10, model-based techniques like the ones proposed in [17], [24] can be used to obtain a good initial rate allocation. Another issue arises if we start from a very small initial rate allocation. Then, waiting for 200 packets to be exchanged before correcting the initial rate allocation will waste bandwidth. To solve this problem, we propose making the iteration duration for link $i \to j$ to vary inversely with $1/\overline{S}_{i\to j} - \lambda_{i\to j}$. This will ensure that when there is a lot of residual capacity in the network, it will get eaten up fast, hence the initial increase in rates will be fast. When the network gets closer to the optimal operating point, the iteration duration will increase to allow a more accurate estimate.

8 Conclusions

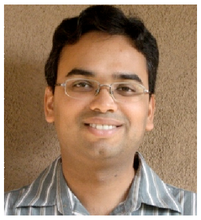
In this paper, we propose CapEst, a mechanism to estimate link capacity in a wireless network. CapEst yields accurate estimates while being very easy to implement and does not require any complex computations. CapEst is measurement-based and model-independent, hence, works for any MAC/PHY layer. CapEst can be easily modified to work with any application which requires an estimate of link capacity. Also, the implementation overhead of CapEst is small and it does not lose accuracy when used with auto-rate adaptation and finite MAC retransmit limits. Finally, CapEst requires no support from the underlying chipset and can be completely implemented at the network layer.

References

[1] U. Akyol, M. Andrews, P. Gupta, J. Hobby, I. Saniee, and A. Stolyar. Joint scheduling and congestion control in mobile ad hoc networks. In *"Proceedings of IEEE INFOCOM"*, 2008.
[2] J. Camp, J. Robinson, C. Steger, and E. Knightly. Measurement driven deployment of a two-tier urban mesh access network. In *Proceedings of ACM MOBISYS*, 2006.
[3] L. Chen, S. Low, M. Chiang, and J. Doyle. Cross-layer congestion control, routing and scheduling design in ad-hoc wireless networks. In *Proceedings of IEEE INFOCOM*, 2006.
[4] D. Couto, D. Aguayo, J. Bicket, and R. Morris. A high-throughput path metric for multi-hop wireless routing. In *Proceedings of ACM MOBICOM*, 2003.
[5] M. Durvy, O. Dousse, and P. Thiran. Border effects, fairness, and phase transition in large wireless networks. In *Proceedings of IEEE INFOCOM*, 2008.

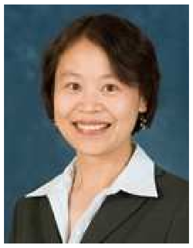
9. Congested links (operating at close to full utilization) govern the end-to-end flow rates. Since, these links are high-traffic links, they almost always have a packet to send. This ensures that the 200 packets are quickly exchanged.

[6] M. Garetto, T. Salonidis, and E. Knightly. Modeling Per-flow Throughput and Capturing Starvation in CSMA Multi-hop Wireless Networks. In *Proceedings of IEEE INFOCOM*, 2006.
[7] M. Gastpar and M. Vetterli. On the capacity of wireless networks: The relay case. In *Proceedings of IEEE INFOCOM*, 2002.
[8] M. Grossglauser and D. N. C. Tse. Mobility increases the capacity of ad hoc wireless networks. *IEEE/ACM Transactions on Networking*, 10(4), 2002.
[9] A. Gupta, X. Lin, and R. Srikant. Low-complexity distributed scheduling algorithms for wireless networks. In *Proceedings of IEEE INFOCOM*, 2007.
[10] P. Gupta and P. Kumar. Capacity of wireless networks. *IEEE Transactions on Information Theory*, 46(2), 2000.
[11] K. Jain, J. Padhye, V. Padmanabhan, and L. Qiu. Impact of interference on multi-hop wireless network performance. In *Proceedings of ACM MOBICOM*, 2003.
[12] K. Jang, R. Govindan, and K. Psounis. Simple yet efficient transparent airtime allocation in wireless mesh networks. Technical Report 915, University of Southern California, July 2010.
[13] A. Jindal and K. Psounis. The Achievable Rate Region of 802.11-Scheduled Multihop Networks. *IEEE/ACM Transactions on Networking*, 17(4):1118–1131, 2009.
[14] C. Joo and N. Shroff. Performance of random access scheduling schemes in multi-hop wireless networks. In *Proceedings of IEEE INFOCOM*, 2007.
[15] A. Kashyap, S. Ganguly, and S. Das. A measurement-based approach to modeling link capacity in 802.11-based wireless networks. In *Proceedings of ACM MOBICOM*, 2007.
[16] A. Kumar, E. Altman, D. Miorandi, and M. Goyal. New insights from a fixed point analysis of single cell IEEE 802.11 wireless LANS. In *Proceedings of IEEE INFOCOM*, 2005.
[17] Y. Li, L. Qiu, Y. Zhang, R. Mahajan, and E. Rozner. Predictable performance optimization for wireless networks. In *Proceedings of ACM SIGCOMM*, 2008.
[18] K. Medepalli and F. A. Tobagi. Towards Performance Modeling of IEEE 802.11 Based Wireless Networks: A Unified Framework and Its Applications. In *Proceedings of IEEE INFOCOM*, 2006.
[19] L. Qiu, Y. Zhang, F. Wang, M. Han, and R. Mahajan. A general model of wireless interference. In *Proceedings of ACM MOBICOM*, 2007.
[20] S. Rangwala, A. Jindal, K. Jang, K. Psounis, and R. Govindan. Understanding congestion control in multi-hop wireless mesh networks. In *Proceedings of ACM MOBICOM*, 2008.
[21] S. Rangwala, A. Jindal, K. Jang, K. Psounis, and R. Govindan. Neighborhood-centric congestion control for multi-hop wireless mesh networks. *Under submission at IEEE/ACM Transactions on Networking*, 2010.
[22] T. Rappaport. *Wireless Communications: Principles and Practice*. Prentice Hall, 2 edition, 1996.
[23] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan. Measurement-based models of delivery and interference. In *Proceedings of ACM SIGCOMM*, 2006.
[24] T. Salonidis, G. Sotiropoulos, R. Guerin, and R. Govindan. Online Optimization of 802.11 Mesh Networks. In *Proceedings of ACM CONEXT*, 2009.
[25] C. Shannon, D. Moore, and K. Claffy. Beyond folklore: Observations on fragmented traffic. *IEEE/ACM Transactions on Networking*, 10(6), 2002.
[26] G. Sharma, R. Mazumdar, and N. Shroff. On the complexity of scheduling in wireless networks. In *Proceedings of ACM MOBICOM*, 2006.
[27] K. Sundaresan, V. Anantharaman, H. Hsieh, and R. Sivakumar. ATP: A Reliable Transport Protocol for Ad Hoc Networks. *IEEE Transactions on Mobile Computing*, 2005.
[28] K. Tan, F. Jiang, Q. Zhang, and X. Shen. Congestion control in multihop wireless networks. *IEEE Transactions on Vehicular Technology*, 56:863–873, Mar. 2007.
[29] X. Wang and K. Kar. Throughput Modeling and Fairness Issues in CSMA/CA based Ad-hoc Networks. In *Proceedings of IEEE INFOCOM*, 2005.
[30] A. Warrier, S. Janakiraman, S. Ha, and I. Rhee. DiffQ: practical differential backlog congestion control for wireless networks. In *Proceedings of IEEE INFOCOM*, 2009.
[31] K. Xu, M. Gerla, L. Qi, and Y. Shu. TCP unfairness in ad-hoc wireless networks and neighborhood RED solution. In *Proceedings of ACM MOBICOM*, 2003.
[32] Y.Gao, D.Chiu, and J. Lui. Determining the end-to-end throughput capacity in multi-hop networks: methodology and applications. In *Proceedings of ACM Sigmetrics*, 2006.
[33] Q. Zhao, D. Tsang, and T. Sakurai. A simple and approximate model for nonsaturated IEEE 802.11 DCF. *IEEE Transactions on Mobile Computing*, 8(11), 2009.



Apoorva Jindal is a Member of Technical Staff at Juniper Networks. He was a Post-Doctoral Research Fellow at the University of Michigan, Ann Arbor. He received his MS and PhD degrees from the University of Southern California.

Konstantinos Psounis Konstantinos Psounis is an associate professor of EE and CS at the University of Southern California. He received his first degree from NTUA, Greece, in 1997, and the M.S. and Ph.D. degrees from Stanford in 1999 and 2002 respectively.



Mingyan Liu Mingyan Liu is an associate professor of EECS at the University of Michigan, Ann Arbor. She received her Ph.D. Degree in electrical engineering from the University of Maryland, College Park, in 2000.