CapEst: Estimating wireless link capacity in multi-hop networks

Apoorva Jindal
Juniper Networks
Sunnyvale, CA 94089
Email: ajindal@juniper.net

Konstantinos Psounis
University of Southern California
Los Angeles, CA, 90089
Email: kpsounis@usc.edu

Mingyan Liu
University of Michigan
Ann Arbor, MI 48109
Email: mingyan@eecs.umich.edu

***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, 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.

I. 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 [1], online optimization of wireless mesh networks using centralized rate control [2], distributed rate control mechanisms which provide explicit and precise rate feedback to sources [3], admission control, interference-aware routing [4], network management tools to predict the impact of configuration changes [1] 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 [1], [4]–[7]. Simpler models make simplifying assumptions [2], [8]–[10] which diminish their accuracy in real networks. Moreover, model-based capacity estimation techniques [1], [2] 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 III and IV 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 IV 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 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.

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 [2]. (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.

II. 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 $i \rightarrow j$ [3].)

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 II-C and II-D, we present modifications to CapEst to incorporate finite retransmit limits and different packet sizes and data rates respectively.

A. 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 $\overline{S}_{i\rightarrow j}$ and $K_{i\rightarrow j}$[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 $i \rightarrow j$. For each successful packet transmission, if $S^{last}_{i\rightarrow j}$ denotes the service time of the most recently transmitted packet, then the values of these two variables are updated as follows.

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

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

$1/\overline{S}_{i\rightarrow j}$ gives the MAC service rate at link $i \rightarrow j$. Thus, the residual capacity on link $i \rightarrow j$ is equal to $\left(1/\overline{S}_{i\rightarrow j}\right) - \lambda_{i\rightarrow j}$. Now, since transmissions on neighboring links will also eat up the capacity at link $i \rightarrow j$, this residual capacity will be distributed amongst all the links in $N_{i\rightarrow j}$. Note that the application using CapEst, based on this residual capacity estimate, will either re-allocate 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.

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

where $\delta_{k\rightarrow l}$ denotes the rate increase at link $k \rightarrow l$ 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 II-B describes a centralized methodology to distribute this estimated residual capacity amongst interfering links so as to obtain max-min fairness amongst all flows.

Note that all interfering links in $N_{i\rightarrow j}$ do not have the same effect on the link $i \rightarrow j$. 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 [6]). However, the linear constraint of Equation (3) treats the links in $N_{i\rightarrow j}$ as a clique with binary interference. Hence, there may be some remaining capacity on link $i \rightarrow j$ 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 $\overline{S}_{i\rightarrow j}$ and $K_{i\rightarrow j}$ 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 $N_{i\rightarrow j}$, and hence change the value of $\overline{S}_{i\rightarrow j}$. Thus, retaining $\overline{S}_{i\rightarrow j}$ 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 $\overline{S}_{i\rightarrow j}$ 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 $i \rightarrow j$ has to measure $\overline{S}_{i\rightarrow j}$ afresh. Thus, an iteration duration has to be long enough so as to accurately measure $\overline{S}_{i\rightarrow j}$. 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 IV.

B. Distributing Capacity to Obtain Max-Min Fairness

To illustrate how CapEst will be used with a real application, we now describe a centralized mechanism 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 $r^{allocate}_{i\rightarrow j}$ denote the maximum flow rate allowed on link $i \rightarrow j$ for any flow passing through this link.

Consider a link $i \rightarrow j$. Let $I(f, i \rightarrow j)$ be an indicator variable which is equal to 1 only if flow $f \in \mathcal{F}$ passes through a link $i \rightarrow j \in \mathcal{L}$, 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.

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

where $0 < \alpha \leq 1$ is a parameter which controls the proportion of residual capacity distributed, $r^{max}_{i\rightarrow j}$ denotes the maximum flow-rate allowed by link $i \rightarrow j$ on any link in $N_{i\rightarrow j}$ 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 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 [11], for CSMA-CA based MAC protocols, this property ensures max-min fairness.

C. 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?

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

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[2], the latter term is equal to $\frac{W_m/2+T_s}{1-p_{i\rightarrow j}^{loss}}$, where W_m is the largest back-off window value, T_s is the transmission time of a packet and $p_{i\rightarrow j}^{loss}$ is the probability that a DATA transmission on link $i \rightarrow j$ is not successful. This value 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[3].

D. 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 $k \rightarrow l \in N_{i\rightarrow j}$ 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 $k \rightarrow l \in N_{i\rightarrow j}$ as well as the residual capacity on $i \rightarrow j$ 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 $\overline{T}_{i\rightarrow j}$ represents the average packet transmission time at link $i \rightarrow j$, then the capacity estimation mechanism will impose the following constraint.

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

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.

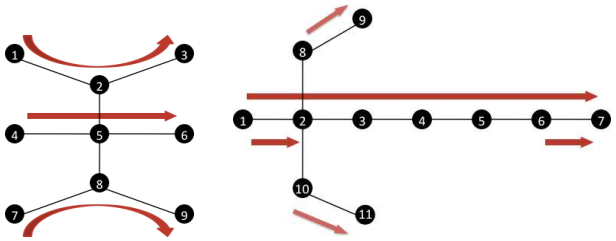
Finally, the following equation states how the value of $\overline{T}_{i\rightarrow j}$ is estimated.

$$\overline{T}_{i\rightarrow j} \leftarrow \frac{\overline{T}_{i\rightarrow j} \times K_{i\rightarrow j} + \frac{P_{i\rightarrow j}^{last}}{D_{i\rightarrow j}^{last}}}{K_{i\rightarrow j}+1}, \quad (6)$$

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

III. 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.



(a) (b)

Fig. 1. (a) Flow in the Middle Topology. (b) Chain-cross Topology.

The following theorem states a sufficient condition for the process to converge for a homogeneous WLAN topology. Please refer to [12] for the proof.

Theorem 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) $nb_0 >> 1$, (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}.

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 intial arrival rate, CapEst always converged to the correct value. And this convergence exhibits the following property which is also observed in the proof of Theorem 1: *with each iteration, at the congested edge $i \rightarrow j$, $|\frac{1}{\overline{S}_{i\rightarrow j}} - \lambda_{i\rightarrow j}|$ keeps on reducing.*

IV. 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, with finite retransmit values at the MAC layer and with auto-rate adaptation. We observe that CapEst *always* converges to within 5% of the optimal rate allocation in less than 18 iterations.

A. Methodology

We use Qualnet version 4.0 as the simulation platform. All our simulations are conducted using an unmodified IEEE 802.11(b) MAC

[2]A more complex model accounting for correlated losses can be easily incorporated, however, our simulation results presented in Section IV-G show that this simple independent loss model yields accurate results.

[3]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.

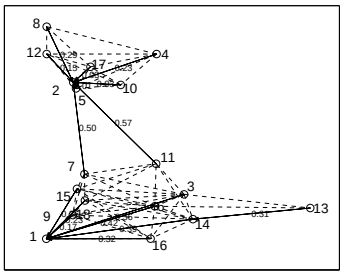


Fig. 2. Deployment at Houston.

(DCF). RTS/CTS is not used unless explicitly stated. We use the default parameters of IEEE 802.11(b) in Qualnet. Unless explcitly 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. Finally, to determine the actual max-min rate allocation, we use the methodology proposed by [3].

The implementation of CapEst and the max-min fair centralized rate allocator closely follows their description in Section II. We choose the iteration duration to be 200 packets, that is, each link resets its estimate of expected service time after transmitting 200 packets. 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 [2] 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 simultaneouly respectively. $LIR = 1$ implies no interference, with lower LIR's indicating a higher degree of interference. Similar to the mechanism in [2], links with $LIR > 0.95$ are classified as non-interfering.

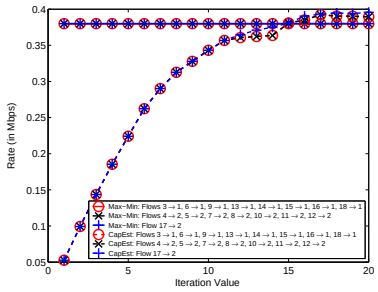


Fig. 4. Performance of CapEst: Deployment at Houston.

B. 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.

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 [3], [5], [13]. Figure 3(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.

2) Chain-Cross Topology: Figure 1(b) shows the topology. This topology was proposed by [3] 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 3(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.

C. 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 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 3(c) and 3(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[4] (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 5(b), the mechanism converges to within 5% of the optimal within 15 iterations. Note that, in general, routing schemes like ETX [14] will avoid the use of such high-loss links for routing, and hence, for most cases, an iteration duration of 200 packet suffices.

D. A Real Topology: Deployment at Houston

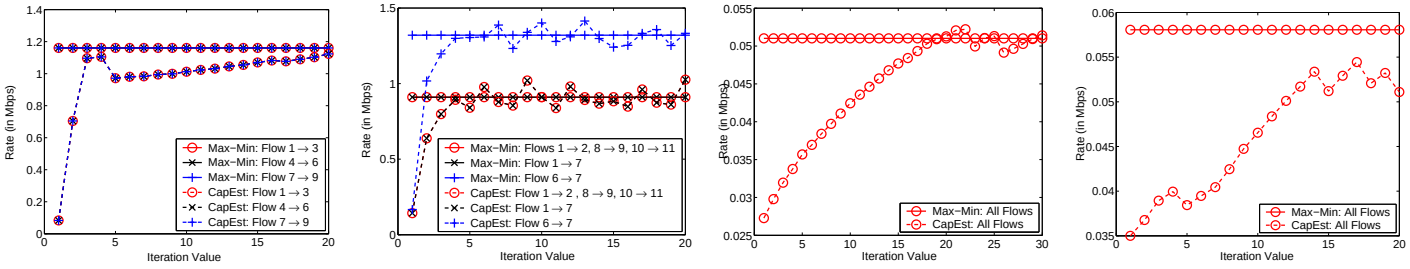
The next topology is derived from an outdoor residential deployment in a Houston neighborhood [15]. The node locations (shown in Figure 2) are derived from the deployment and fed into the simulator. The physical channel that we use in the simulator is a two-ray path loss model with log-normal shadowing and Rayleigh fading. The ETX routing metric (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 2.

Figure 4 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.

E. Impact of a Smaller Iteration Duration

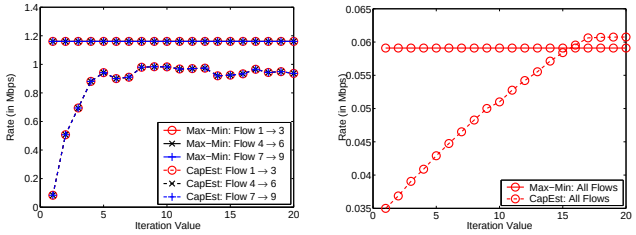
In this section, we evaluate the impact of using a smaller iteration duration on performance. We plot the evolution of the rate assigned to each flow in Figure 5(a) for the flow in the middle topology with one iteration duration $= 75$ packets. We observe that the rate converges to a value smaller than the optimal. (Note that we had made a similar observation for the 100 node random topology in Section IV-C.) In general, for all topologies we studied, we observed that using

[4]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.



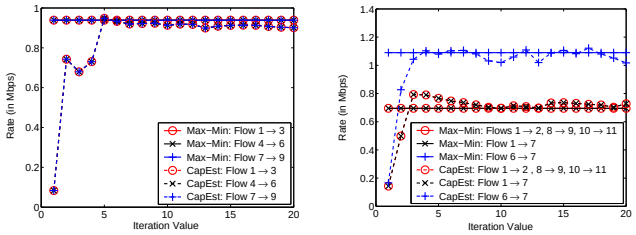
(a) (b) (c) (d)

Fig. 3. 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.



(a) (b)

Fig. 5. 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.



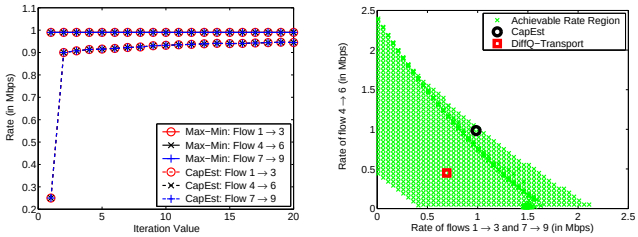
(a) (b)

Fig. 6. Performance of CapEst with IEEE 802.11 with RTS/CTS. (a) Flow in the Middle. (b) Chain-cross.

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.

F. 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.



(a) (b)

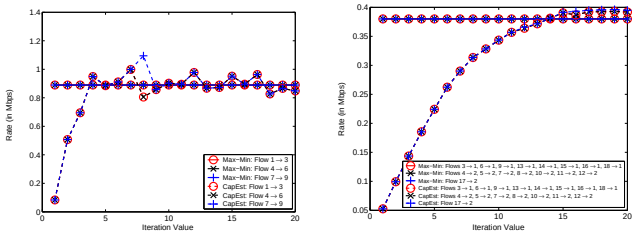
Fig. 7. (a) Performance of CapEst with DiffQ-MAC: Flow in the Middle. (b) CapEst vs DiffQ-Transport over DiffQ-MAC.

1) With RTS/CTS: We first evaluate the performance of CapEst with IEEE 802.11 DCF with RTS/CTS. Figures 6(a) and 6(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.

2) Back-pressure MAC: We next evaluate the performance of CapEst with a back-pressure-based random-access protocol [16]–[18]. 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 [19]. So, multiple researchers have suggested using a random access protocol whose channel access probabilities inversely depend on the queue size [16]–[18]. This ensures that the probability of scheduling a packet from a larger queue is higher. The most recent of these schemes is *DiffQ* [16]. 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).

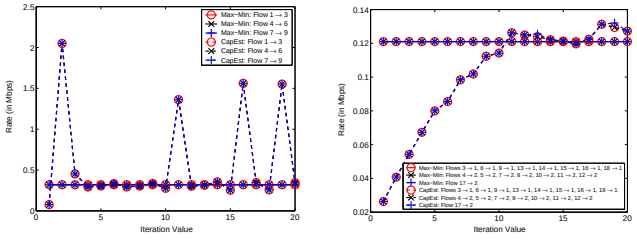
Back-pressure medium access is very different from the traditional IEEE 802.11 DCF in conception. However, CapEst can still accurately measure the capacity at each edge. Figure 7(a) plots the evolution of the assigned flow rates for the flow in the middle topology with DiffQ-MAC. The different priority levels as well as the AIFS, CWMin and CWMax values being used are the same as the ones used in [16]. 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 7(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. Figure 7(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 [20] 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.



(a) (b)

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



(a) (b)

Fig. 9. Performance of CapEst with auto-rate adaptation. (a) Flow in the Middle. (b) Deployment at Houston.

G. 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 II-C 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 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.

H. Auto-Rate Adaptation

In this section, we switch on the default auto-rate fallback mechanism of Qualnet. We use the methodology proposed in Section II-D to constrain the rate updates. Figures 9(a) and 9(b) show the evolution of allocated rates for the flow in the middle topology and the deployment at Houston respectively. Again, not only does CapEst converge to the correct rate allocation but also the convergence time remains the same as before. However, the variation in the rates being assigned is larger than before. This is due to the fact that at lower rates, fewer collisions are observed, hence the data rate at each link is higher. Thus, the residual capacity estimate observed will be larger and hence, the variation in rates allocated will be larger. In other words, the larger variation in rates is due to a larger variation in data rates caused by auto-rate fallback and not CapEst. Also, this variation is less prominent in the real scenario, the deployment at Houston, where due to fading, there is significant channel losses in absence of collisions and the expected service time estimate varies less due to collisions. We evaluate CapEst for all the other scenarios described above with the default auto-rate fallback mechanism of Qualnet, and our observation remains the same.

V. 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] Y. Li, L. Qiu, Y. Zhang, R. Mahajan, and E. Rozner, "Predictable performance optimization for wireless networks," in *Proceedings of ACM SIGCOMM*, 2008.

[2] T. Salonidis, G. Sotiropoulos, R. Guerin, and R. Govindan, "Online Optimization of 802.11 Mesh Networks," in *Proceedings of ACM CONEXT*, 2009.

[3] 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.

[4] 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.

[5] A. Jindal and K. Psounis, "The Achievable Rate Region of 802.11-Scheduled Multihop Networks," *IEEE/ACM Transactions on Networking*, vol. 17, no. 4, pp. 1118–1131, 2009.

[6] L. Qiu, Y. Zhang, F. Wang, M. Han, and R. Mahajan, "A general model of wireless interference," in *Proceedings of ACM MOBICOM*, 2007.

[7] 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.

[8] M. Durvy, O. Dousse, and P. Thiran, "Border effects, fairness, and phase transition in large wireless networks," in *Proceedings of IEEE INFOCOM*, 2008.

[9] 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.

[10] C. Reis, R. Mahajan, M. Rodrig, D. Wetherall, and J. Zahorjan, "Measurement-based models of delivery and interference," in *Proceedings of ACM SIGCOMM*, 2006.

[11] 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.

[12] A. Jindal, K. Psounis, and M. Liu, "Capest: A measurement-based approach to estimating link capacity in wireless networks," ArXiv, Tech. Rep. arXiv:1007.4724v1, 2010. [Online]. Available: http://arxiv.org/abs/1007.4724

[13] 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.

[14] 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.

[15] 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.

[16] A. Warrier, S. Janakiraman, S. Ha, and I. Rhee, "DiffQ: practical differential backlog congestion control for wireless networks," in *Proceedings of IEEE INFOCOM*, 2009.

[17] A. Gupta, X. Lin, and R. Srikant, "Low-complexity distributed scheduling algorithms for wireless networks," in *Proceedings of IEEE INFOCOM*, 2007.

[18] C. Joo and N. Shroff, "Performance of random access scheduling schemes in multi-hop wireless networks," in *Proceedings of IEEE INFOCOM*, 2007.

[19] G. Sharma, R. Mazumdar, and N. Shroff, "On the complexity of scheduling in wireless networks," in *Proceedings of ACM MOBICOM*, 2006.

[20] 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.