Globally Fair Radio Resource Allocation for Wireless Mesh Networks

Ashish Raniwala
Computer Science Dept
Stony Brook University
Stony Brook, NY

Pradipta De
IBM Research
New Delhi, INDIA

Srikant Sharma
Symantec Corporation
Santa Clara, CA

Rupa Krishnan
Computer Science Dept
Stony Brook University
Stony Brook, NY

Tzi-cker Chiueh
Computer Science Dept
Stony Brook University
Stony Brook, NY

***Abstract*—Network flows running on a wireless mesh network (WMN) may suffer from partial failures in the form of serious throughput degradation, sometimes to the extent of starvation, because of weaknesses in the underlying MAC protocol, dissimilar physical transmission rates or different degrees of local congestion. Most existing WMN transport protocols fail to take these factors into account. This paper describes the design, implementation and evaluation of a *coordinated congestion control* (C3L) algorithm that guarantees fair resource allocation under adverse scenarios and thus provides end-to-end max-min fairness among competing flows[1]. The C3L algorithm features an advanced topology discovery mechanism that detects the inhibition of wireless communication links, and a general collision domain capacity re-estimation mechanism that effectively addresses such inhibition. A comprehensive ns-2-based simulation study as well as empirical measurements taken from an IEEE 802.11a-based multi-hop wireless testbed demonstrate that the C3L algorithm greatly improves inter-flow fairness, eliminates the starvation problem, and at the same time maintains high radio resource utilization efficiency.**

I. Introduction

An ideal transport protocol efficiently utilizes all available resource in a network and allocates them fairly among competing flows even when the underlying link-layer protocol may be inherently unfair. For wireless mesh networks (WMNs) in particular, there are additional issues, such as shortcomings in the MAC protocol and intricate inter-dependencies among neighboring wireless links, that greatly complicate the problem of fair resource allocation. Previously proposed transport protocols for WMNs either fail to provide end-to-end fairness, or rely upon proprietary MAC protocols to achieve local inter-flow fairness. Consequently, network flows running on existing WMNs could encounter partial failures in the form of either temporary serious throughput degradation or persistent starvation.

On the other hand, economies of scale make IEEE 802.11 an attractive technology for building wireless mesh networks. The MAC protocol of IEEE 802.11, however, introduces serious unfairness among competing nodes when used in multi-hop WMNs [1]. The well-known *hidden node problem* [20] causes one wireless link's transmission to be *inhibited* by another link, eventually leading to unfair resource allocation between the two. More specifically, while the RTS/CTS messages in IEEE 802.11's MAC protocol effectively stop a hidden node from interfering with an on-going communication transaction, they cannot prevent the hidden node from initiating its RTS/CTS sequence at inopportune times and subsequently suffering from long backoff delays. TCP exacerbates this unfairness problem because TCP senders further back off when their packets take a long time to get through the inhibited links. As a result, TCP flows traversing an inhibited link could be completely suppressed in the worst case. Another related fairness problem in multi-hop WMNs is that when two TCP flows share the same wireless link, the flow traversing fewer hops tends to acquire a higher share of network resource. In a multi-hop WMN, this translates into smaller resource share to flows whose end points are farther away from each other. Finally, existing transport protocols at best attempt to allocate a radio channel's resource fairly among flows from a single node, rather than among all flows from all nodes that share the radio channel. As a result, a flow emanating from a node with fewer flows tends to get a larger than fair share of channel bandwidth. Figure 1 qualitatively illustrates these unfairness problems with simple examples, whereas Figure 2 quantifies the extent of unfairness by presenting the throughput of the participating flows in each instance.

To protect network flows running on an IEEE 802.11-based WMN from any partial failures described above, we design and implement a *coordinated congestion control algorithm (C3L)* that performs global radio resource allocation and thus provides end-to-end flow-level max-min fairness despite the weaknesses in the MAC layer. We develop an advanced topology discovery algorithm that is able to identify not only all the usable wireless links between nodes, but also their interference relationships, including *inhibition* relationships due to the hidden node problem. C3L computes the max-min fair share of individual wireless links based on the latest traffic loads continuously collected at a central location. Unlike previously proposed solutions, C3L is designed to work with multi-hop flows and takes into account both inter-flow and intra-flow dependency. Furthermore, it incorporates a general collision domain capacity re-estimation algorithm that can effectively resolve the unfairness problem due to hidden nodes. Once a wireless link is assigned a radio resource share, flows sharing that link are in turn allocated their shares in a fair way.

C3L takes a *centralized traffic engineering* approach, which we believe is justified for the following reasons. First, individual WMN nodes do not move and routes between WMN nodes change relatively infrequently. In addition, the traffic load pattern, being aggregated from multiple end users, is not expected to fluctuate on a second-by-second basis. Finally,

[1]The actual algorithm can be modified to suit other definitions of fairness[20]. For clarity of discussion, we focus on max-min fairness.

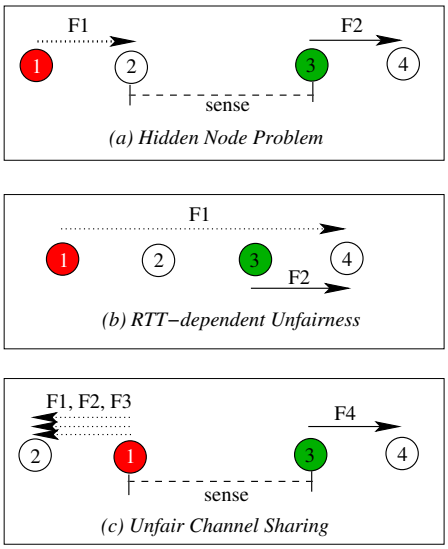

Fig. 1. Three scenarios in which significant unfairness among flows arises. The wireless node getting a lesser than fair share of bandwidth is marked as '1' (and colored in red or white), whereas the one getting a larger share is marked as '3' (and colored in green or black). (a) Node 1 lacks information about Node 3's transmissions, attempts its communication at inopportune times, and eventually backs off unnecessarily. (b) Flow F1 traverses more hops than Flow F2. Some transport protocols, such as TCP, give more bandwidth to flow F2. (c) Flow F1, F2, F3, and F4 all share the same channel, but most transport protocols allocate more bandwidth to F4 than to others.

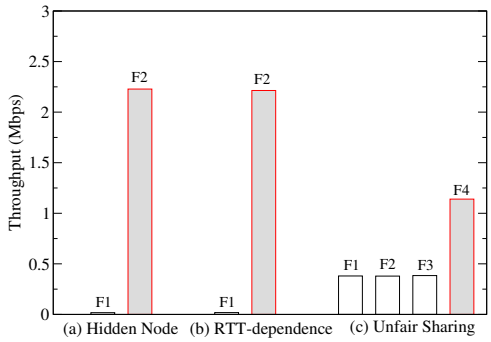

Fig. 2. TCP unfairness over 802.11 WMN for scenarios depicted in Figure 1. (a) TCP amplifies the MAC-layer unfairness due to the hidden node problem. (b) TCP gives a much higher than fair share of bandwidth to flows with smaller hop counts (especially one-hop flows [1]). (c) TCP does not allocate channel bandwidth equally among flows that share the same channel but traverse through nodes with different numbers of flows.

most of a WMN's traffic is directed to/from its gateway nodes connected to the wired backbones. These gateway nodes are natural candidates as a central coordinator that performs traffic engineering for attached WMN nodes. That is, C3L could be deployed on each WMN gateway and thus could readily scale with larger WMNs because the number of gateway nodes in them also increases.

II. Existing Flow Fairness Mechanisms

Luo et al. [6] pointed out the trade-off between maintaining strict fairness among flows and maximizing the spectrum reuse efficiency. They presented a service model that guarantees a minimum bandwidth allocation to each participating flow while maximizing the aggregate radio resource usage. Huang and Bensaou proposed algorithms to compute the max-min fair share to each wireless link in a multi-cell ad hoc network [2]. Nandagopal et al. [3] designed an analytical framework that given an arbitrary fairness model specified as a utility function, can generate a contention resolution algorithm to achieve the specified fairness requirement. All these efforts consider only single-hop flows, and treat each multi-hop flow as multiple independent single-hop flows. As shown in [5], single-hop fairness does not translate into multi-hop fairness because of the traffic dependency among different hops of a multi-hop flow. The algorithm presented in this paper provides end-to-end max-min fairness to multi-hop flows.

Tassiulas and Sarkar [4] proposed algorithms to achieve max-min fairness across multi-hop flows. In their interference model, transmissions for multiple flows can proceed simultaneously as long as they do not share any common node. Li [5] presented centralized and distributed algorithms to achieve end-to-end flow fairness in multi-hop ad hoc networks. He first defined the notion of a contending flow group that includes all flows which contend with one another directly or indirectly, and then ensured fairness by equally dividing the channel capacity among members of each contending flow group. However, their definition of contending flow group is transitive. For instance, consider 3 flows F1, F2, and F3, where F1 and F2 are in one collision domain, and F2 and F3 are in another collision domain. Li's algorithm groups these three flows into the same contending flow group, and assigns each of them B/3, where B is the channel capacity of a single collision domain. In contrast, we are able to achieve much stronger max-min fairness among multi-hop flows and assign B/2 to each flow in this case. Finally, Yi and Shakkottai [22] presented a hop-by-hop congestion control mechanism over wireless multi-hop networks, which imposes a channel access time constraint on each flow. All the solutions described above require MAC layer modification, in contention resolution or packet scheduling, and thus cannot be directly applied to IEEE 802.11-based WMNs. In contrast, the algorithm proposed in this paper is specifically designed to work with unmodified 802.11 MAC.

The bandwidth estimation techniques employed by existing wireless transport protocols can be broadly classified into two categories: (1) Probing-based and (2) Measurement-based. TCP employs *AIMD probing* to fill the network pipe while avoiding congestion. The behavior of *TCP* and its variants over multi-hop ad hoc networks has been extensively studied [12], [19], [10], [16]. TCP has several weaknesses when operating on wireless multi-hop networks [16]. Firstly, TCP's ACK clocking does not work well due to ACK bunching, which occurs because of bursty transmission of packets from the same flow, and leads to upward skewing of RTT estimation and destroys TCP's self clocking strategy [16], [21]. In [18], authors propose sender-side traffic smoothing to alleviate this burstiness. Secondly, TCP confuses wireless channel errors, a frequent occurrence in wireless networks, with congestion-related losses and triggers its congestion control mechanism unnecessarily. Many implicit [17] and explicit [13] mechanisms have been proposed to enable TCP to discriminate between these two types of losses and to trigger its congestion control only when necessary. Finally, TCP's fairness is not RTT-independent: Flows with smaller RTT get a larger share of channel bandwidth than those with larger RTT.

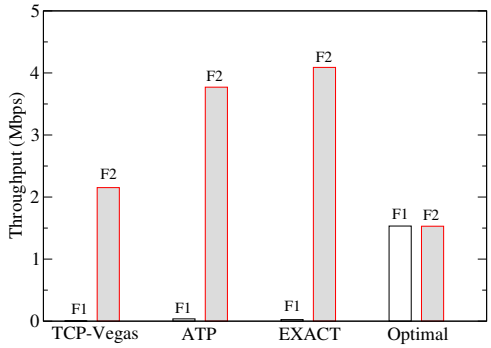

Fig. 3. Amplification of the hidden node problem (Fig 1 (a)) in existing wireless transport protocols.

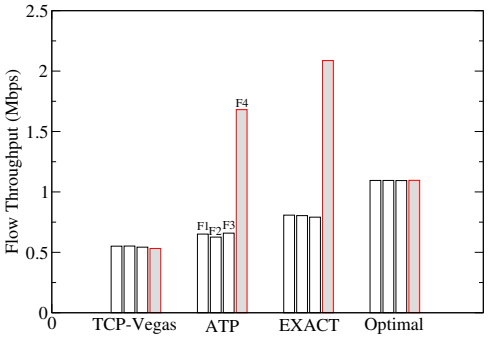

Fig. 4. While existing transport protocols work effectively when the participating flows share some common intermediate node, they fail to allocate bandwidth fairly when flows share a common radio channel (Fig 1 (c)). The optimal resource allocation was achieved by exhaustively trying different sending rates. Note that although TCP-Vegas allocates bandwidth fairly among the flows, its overall channel utilization is much lower than the optimum.

Many transport protocols measure statistics such as inter-packet delay to estimate the available network bandwidth to individual flows. WTCP [14] measures the ratio of inter-packet spacing on the receiver and that on the sender, to determine whether to increase or decrease the sending rate. This approach assumes that all flows in the network are serviced in a strict round-robin fashion at the bottleneck links. This assumption does not hold in general on 802.11-based WMNs, because packet transmissions on wireless links tend to be bursty, and traffic bursts arriving at a bottleneck link may be serviced without any interleaving. In ATP [16], every intermediate node measures the queuing and transmission delays for each packet transmitted on a wireless link. The sum of exponentially averaged queuing and transmission delays yields the *average packet service time*, which reflects the ideal dispatch interval for every flow sharing the link assuming each flow has exactly one packet in the link's queue, the equilibrium condition for ATP. The sending rate of a multi-hop flow is set to the inverse of the maximum of the average packet service times associated with the intermediate hops. One weakness with ATP is that it couples the queue-size management with rate estimation, which leads to substantial rate fluctuation and eventually non-optimal estimation of channel bandwidth [9]. Finally, EXACT [7] is another explicit rate-based flow control scheme. EXACT routers measure the available bandwidth as the inverse of per-packet MAC contention and transmission time. Each router then runs a proportional max-min fair bandwidth sharing algorithm to divide this measured bandwidth among the flows passing through it.

None of the above algorithms directly addresses the unfairness problems that occur in IEEE 802.11-based WMNs. Our experiments show that these protocols indeed produce unfair resource allocation across flows. Figure 3 depicts the behavior of different transport protocols in the presence of a hidden node, and Figure 4 shows the unfair bandwidth sharing among multiple flows for each of these protocols.

III. Coordinated Congestion Control

A. Overview

A resource allocation is *max-min fair* if one cannot increase the resource allocation to any flow without reducing the resource allocation of another flow with an already smaller share [4]. That is, given a resource allocation vector $B = [b_i]$, where b_i is the resource allocation to flow f_i, B is max-min fair if increasing any b_i to $b_i+\delta$ leads to reduction in allocation b_j of some flow, where $b_j < b_i$. It can be proved that the C3L algorithm proposed in this paper is max-min fair. Proofs are left out owing to space constraint.

When all wireless links interfere with one another and hence belong to a single collision domain, the bandwidth share of each link is proportional to the number of flows going over it. If n_i is the number of flows going over the i^{th} link, then the *link's* fair share should be $C_{max} * n_i / \sum_i n_i$, where C_{max} is the i^{th} link's raw capacity. The above allocation works if the effective link capacity, C_{eff}, indeed equals C_{max}. Even in a *symmetric* collision domain, where every packet transmission is visible to all senders, C_{eff} could be less than C_{max} because the MAC-layer overheads, such as backoffs and interference-induced bit errors, reduce the effective channel capacity. In an *asymmetric* collision domain, where not every sender can see every packet transmission, some nodes unnecessarily slow down their transmission rates, which leads to a reduction in the effective channel capacity.

To estimate C_{eff}, we first assume that C_{est}, the estimated channel capacity of the collision domain, equals C_{max}. If $C_{est} > C_{eff}$, then some of the links in the collision domain would not be able to support their incoming traffic load, and their queues would be built up. From queue build-up one can infer that the current C_{est} is higher than C_{eff}. On the other hand, if none of the links in the collision domain has its queue built up despite all nodes sending at their assigned rates, then C_{est} is lower than C_{eff}.

The same logic can be used to detect capacity mis-estimation in an asymmetric collision domain, such as that in Figure 1(a). In this case, unlike in a symmetric collision domain, where allocating $C_{max}/2$ to each sender would result in queue build-up at both senders, here only the inhibited sender's queue is built up. The inhibiting link's sender, on the other hand, can transmit data at a rate up to C_{max} without experiencing any queue build-up. Figure 5 shows this effect. In essence, the inhibited sender perceives an inflated picture of the inhibitor's traffic. Therefore, simply decreasing the inhibitor's rate to $C_{max}/2$ does not help the inhibited sender. One way to address this starvation problem is to artificially decrease an asymmetric collision domain's C_{eff}. If at least one of the senders in a collision domain sees its queue built up, the channel capacity estimate is decreased to

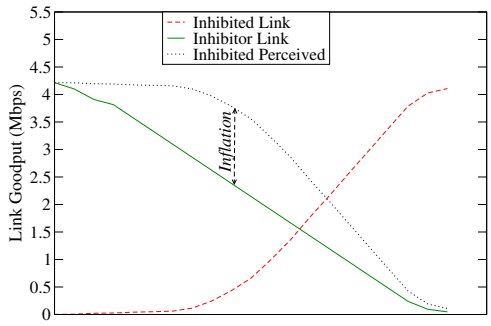

Fig. 5. Reducing the rate allocation to inhibiting link gives a fairer ground for the inhibited sender to contend for the channel.

$C_{est} - \delta$. However, if none of the senders in a collision domain sees its queue built up, the capacity estimate is increased to $C_{est} + \delta$. Unlike TCP, where each flow does this probing independently, C3L probes a collision domain's capacity for all flows associated with that domain.

With multiple collision domains, each link could participate in multiple overlapped collision domains, and thus receives a resource allocation from each domain it participates in. The most constrained collision domain is the one that assigns the smallest resource allocation and hence dictates the allocation to the link. Furthermore, because the ultimate bandwidth share assigned to a multi-hop flow is the resource allocation it receives at its most constrained hop, this inter-hop dependency within a flow further complicates the resource allocation problem.

Figure 6 depicts the various components of the proposed coordinated congestion control algorithm (C3L), and the data flow among them. Topology discovery algorithm is used by each node to detect its (1) direct communication neighbors, (2) symmetric interference neighbors, and (3) inhibitor links. Based on the result of topology discovery, the central controller constructs a conflict graph of the network [23], and finds all cliques (fully connected maximal subgraphs) in the conflict graph. Each clique in the conflict graph corresponds to a collision domain in the physical network.

C3L takes a centralized traffic engineering approach. WMN nodes continuously profile the flows passing through them, and periodically report to a central controller the set of transit flows and their routes. Based on these traffic profiles, the central controller uses the C3L algorithm to compute the resource allocation to individual links, and distributes this allocation information to the network nodes. The sender of each link takes the per-link allocation and divides it fairly across all the flows passing through the link. In addition, each node monitors the queues associated with its wireless links, and modifies the rate of injecting traffic to a wireless link if its queue starts to build up. Overloaded links are also reported to the central controller, which uses this information in the next run to adjust the capacity estimate of related cliques. The centralized traffic engineering approach is reasonable in WMNs because individual nodes do not move and routes change relatively infrequently. In addition, the traffic pattern, being aggregated from multiple end users, does not change on a second-by-second basis. Finally, most of the traffic is directed to/from the gateway nodes that are connected to the wired backbones. These gateway nodes are natural candidates

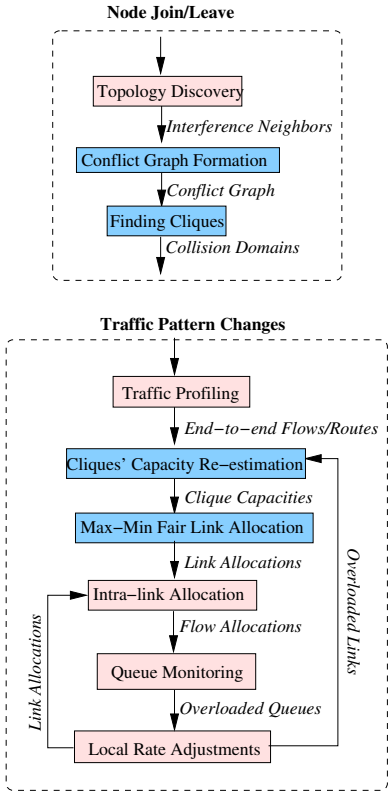

Fig. 6. The overall flowchart describing various control mechanisms in the C3L algorithm. The blocks in blue (colored dark) are executed on the central controller, while the ones in pink (colored light) are executed on individual nodes.

as a central coordinator that performs traffic engineering for attached WMN nodes.

B. Advanced Topology Discovery

The goal of a typical topology discovery process is to determine the usable communication links in the network. Our topology discovery in addition determines the interference relationship between these links, including finding out which links inhibit any particular link. This information is essential to the construction of the conflict graph and to find all collision domains in the network.

Each node periodically broadcasts a HELLO message. The HELLO messages are received by its communication neighbors, thereby allowing the neighbors to discover the node. Furthermore, the fraction of HELLO messages received by each neighbor is used to determine the delivery ratio of the corresponding wireless link, and infer whether the link is usable or not.

The discovery of interfering neighbors is slightly more involved, as it may not be possible to directly communicate with an interfering neighbor at default settings. Our experiments with a real wireless testbed show that it is indeed possible to communicate with an interfering neighbor if (1) the link-layer encoding is switched to lowest possible (1 Mbps for 802.11b and 6 Mbps for 802.11a), and (2) the transmit power is increased by 3 dB. This is because the lowest link-layer encoding is resilient enough to communicate packets with distant neighbors as long as the received signal quality is above 3 dB. Therefore, changing the link-layer encoding and

increasing the transmit power by 3 dB ensures that a node can communicate directly with interfering neighbors.

Based on the data collected during topology discovery, the central controller forms the conflict graph for the network. In a conflict graph, every usable directed communication link is represented by a vertex, and there is an edge between two vertices if the corresponding communication links interfere with each other (can not do successful simultaneous transmission). Further, the controller finds all the cliques in the conflict graph using the algorithm proposed by Bron and Kerbosch [25]. A clique is a maximal fully connected subgraph. Physically, a clique represents a collision domain: a set of links, only one of which should be active at a time. Although this algorithm finds all the cliques, it can be modified to do incremental clique finding each time the network topology changes. Finally, for each wireless link, C3L determines if it is an *inhibited* link, which is characterized by the following condition: its sender lies outside the interference range of a node that is within the communication range of its receiver. For example, in Figure 1, F1 is an inhibited link inhibited by F2, because Node 1 cannot see all packets transmitted by Node 3, and end up increasing its backoff delay to a large value.

C. Max-Min Fair Resource Allocation among Links

Based on the traffic profile obtained from the network, the controller periodically performs a max-min fair resource allocation among all wireless links in the network. This algorithm goes through each clique in the conflict graph in the decreasing order of constraints imposed on the participating flows in the clique. The measure of constraint is the ratio of a clique's residual capacity and the number of unassigned one-hop sub-flows traversing the clique's links. The most constraining clique (the one with the lowest ratio) is visited first. Upon visiting any clique, the residual capacity is equally divided among all the unassigned sub-flows. The bandwidth assignment to each sub-flow is propagated to all its upstream and downstream links. This propagation modifies the residual capacity of all the cliques these newly assigned flows pass through. The algorithm terminates once all the flows have been assigned their fair share. Note that the order of visiting cliques cannot be pre-determined, as the constraint of a clique is a dynamic measure, and may change after some of its sub-flows get their bandwidth assignment from other more constrained cliques.

D. Clique Capacity Re-estimation Algorithm

The capacity of a clique is dependent upon the presence of hidden nodes, exact packet transmission scheduling at MAC layer, amount of time nodes spend in backoff, and packet errors. Because of these factors, it is hard to determine a clique's capacity analytically. We therefore take a probing approach to estimate the clique's capacity.

For any given clique, the central controller starts with a capacity estimate C_{est} equal to C_{max}. If a clique's effective capacity $C_{eff} < C_{est}$, some of the senders in the clique are unable to forward their incoming traffic, and their transmission queues build up. The build-up of a wireless link's queue beyond a certain threshold serves as an indication that the controller has overestimated the capacity of one or more of the cliques that the wireless link participates in. Because queues

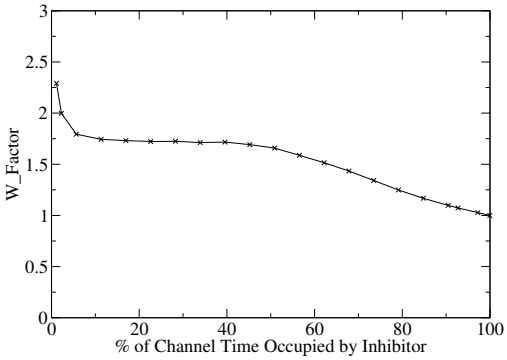

Fig. 7. W_FACTOR: The ratio of perceived inhibitor's channel usage to the actual inhibitor's load on channel. W_FACTOR varies between 1 and 2, with a typical value around 1.7

may also build up because of short-term traffic burstiness, a large threshold value is therefore required to identify long-term queue build-up. To pinpoint the clique with capacity overestimate, C3L estimates the channel time usage of all the cliques that a link participates in. The clique with the maximum channel time usage is the one experiencing maximal contention with respect to the link in consideration. Because an inhibited link perceives an inflated usage of the associated inhibitor links, each inhibitor link's channel time usage is multiplied by an adjusting factor, called W_FACTOR, before being added to the clique usage. The value of W_FACTOR is determined empirically. Once the clique with capacity overestimate is pinpointed, its capacity estimate is decremented. Figure 7 shows the variation of W_FACTOR based on graphs in Figure 5. W_FACTOR varies mostly between 1 and 2, with a typical value around 1.7

If, on the other hand, none of the senders in the clique experiences any queue build-up despite all the senders sending at their assigned rate, then we may have underestimated the clique's capacity. If so, the capacity estimate of the clique is incremented.

The increments to clique's capacity are done in small steps of δ_i, and the decrements are done with exponentially increasing δ_r. Every time a clique's capacity is incremented, δ_r is initialized to δ_i. Each time the previous decrement is insufficient and the queues still build up, the δ_r is multiplied by 2. The rationale of such exponentially increasing decrement step is that the previous estimate of C_{est} did not result in queue build-up, while the $C_{est}+\delta_i$ did. Therefore, the clique's effective channel capacity lies within the previous estimate and current estimate: $C_{est} <= C_{eff} <= C_{est} + \delta_i$. However, sometimes the channel capacity may decrease substantially. If so, we increment the δ_r exponentially to arrive at the correct bandwidth estimate. This clique capacity estimation algorithm is shown in Algorithm 1.

E. Per-Link Resource Allocation among Flows

Given the resource allocated to a wireless link, the link's sender node runs a second-level local max-min fair allocation over all the flows traversing the link. The sender ensures that the sum of resources allocated to flows on a link never exceeds the link's first-level allocation. Each of the flow's data packet header carries the minimum allocation to the flow on any of its hops. Each time a packet reaches a hop that assigns a smaller

Clique Capacity Re-estimation Algorithm

Let $Overloaded$ be the list of overloaded links. r_{nid} is the average number of retransmissions on link nid.
Let Cap_{cid}^{prev} and Cap_{cid}^{new} are the previous and new estimated capacity of clique cid.
Let O_{nid}^{prev} be the previous bandwidth assigned to link nid.
Let δ_i^{cid} and δ_r^{cid} be the increment and decrement terms respectively for clique cid's capacity.

```
for each link nid in Overloaded do
  for each clique cid that nid participates in do
    Chnl_nid^cid ← Estimate_Channel_Usage(cid,nid)
  end for
  Find clique-id mcid that has max_cid(Chnl_nid^cid)
  Cap_mcid^new ← Cap_mcid^prev − δ_r^mcid
end for
for each clique cid do
  if no link nid in clique is overloaded then
    Cap_cid^new ← min(C_max, Cap_cid^prev + δ_i^cid)
    δ_r^cid = δ_i^cid
  else if clique's capacity is decreased then
    δ_r^mcid = δ_r^mcid * 2
  end if
end for
```

Procedure: Estimate_Channel_Usage(cid,nid)

```
usage ← 0
for each link lid in clique cid do
  if Hidden(nid, lid) then
    usage ← usage + O_lid^prev * r_lid * W_FACTOR
  else
    usage ← usage + O_lid^prev * r_lid
  end if
end for
return usage
```

resource allocation than is stamped on the packet, the stamp is changed to the local bandwidth assignment on the hop. This information is relayed via the flow's receiver back to the sender in form of periodic control packets sent every *epoch*. These control packets are prioritized on every node to provide timely feedback to the senders.

The intra-link resource allocation algorithm works as follows. Flows associated with a link are first sorted in an increasing order of their sending rate. Flows with the smallest sending rate are considered first. Let C_{alloc} be the first-level allocation to the link, and n be the number of flows passing through the link. Initially, the residual capacity $C_{residual}$ equals C_{alloc}, and number of unassigned flows n_{unasgn} equals n. If a flow's sending rate f_s is smaller than $C_{residual}/n_{unasgn} - \delta$, then it is allocated $f_s + \delta$, and the residual capacity is reduced by that amount. If, on the other hand, f_s is greater than $C_{residual}/n_{unasgn} - \delta$, then the flow is allocated $C_{residual}/n_{unasgn}$. This algorithm ensures that flows with smaller requirements are assigned based on their requirement, while the remaining bandwidth is divided equally among the remaining flows. The extra δ allocation ensures that a finite-requirement flow has a chance to grow. As long as a flow uses δ less than its actual assignment, it is considered to be a finite-requirement flow with a requirement of f_s. If the flow indeed consumes the extra δ bandwidth, the estimated requirement of the flow is reset to infinity.

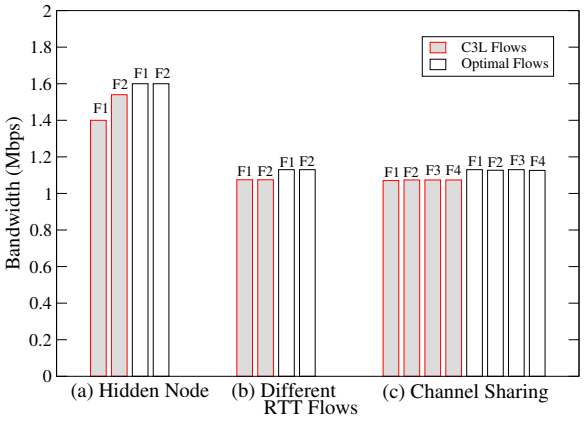

Fig. 8. Performance of C3L in different scenarios of Figure 1. (a) C3L accounts for inhibition relationships that arise from hidden-node scenarios when performing fair allocation of bandwidth. (b) C3L's fairness is end-to-end and does not depend upon the number of hops a flow traverses. (c) C3L takes spatial sharing into account and fairly allocates radio resource among interfering flows, even when they do not share a common node.

IV. Performance Evaluation

We implemented the coordinated congestion control algorithm (C3L), and studied its performance through both comprehensive NS-2 simulations as well as testbed experiments. This section presents the results of this simulation and empirical study. We first compare the fairness of C3L with existing congestion control mechanisms. We then look at the convergence properties of C3L. Next, we evaluate its computation and communication overheads. Finally, we present the empirical results obtained from the testbed.

Unless otherwise specified, the default settings for all the simulations were as follows. The RTS/CTS mechanism was enabled. δ_i was set to 5% of C_{max}. W_FACTOR was set to 1.7. The cycle time for running the centralized max-min fair allocation was 8 seconds, while the $epoch$ for sending response from a flow's receiver to its sender was 1 second. Finally, the routing protocol used for all the simulations was Destination-Sequenced Distance Vector Routing (DSDV).

A. Fairness in Specific Network Configurations

We first compared the performance of C3L with the optimum in three different scenarios depicted in Figure 1. The optimum in each case was identified by exhaustively trying out various sending rate combinations in the network. Figure 8 shows the results. Scenario (a) corresponds to the hidden node case, where Flow F1's link is inhibited by Flow F2's link. As shown in Figure 8(a), despite this inhibition C3L achieves an almost equal allocation to F1 and F2. Further, the allocation is close to optimal. The fair allocation comes from explicit rate control of inhibiting link in C3L. This is in contrast to the extremely unfair allocation done by TCP, ATP and EXACT in the same scenarios (Figure 3).

Scenario (b) corresponds to the case, where a 3-hop flow F1 shares a hop with a 1-hop flow F2. Recall that TCP is unfair to flows traversing more hops (Figure 2). C3L's fairness is

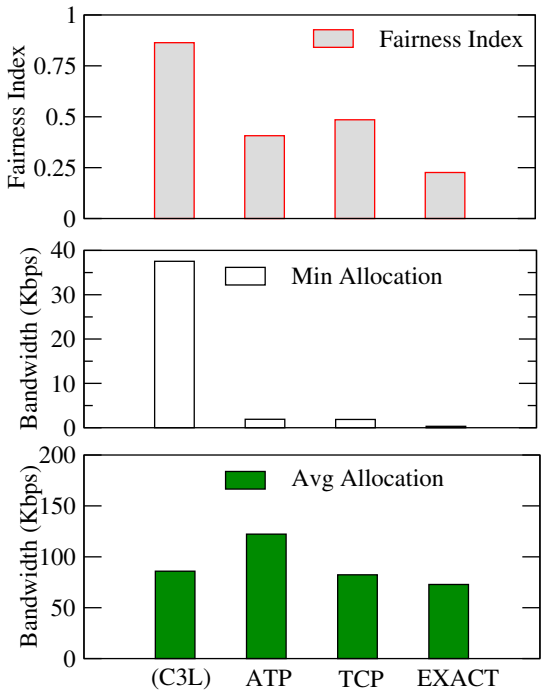

Fig. 9. Performance of C3L in a 64-node grid network with 20 end-to-end flows. The fairness index of C3L flows is closest to 1 suggesting a fair allocation of bandwidth with use of C3L. This result is also strengthened by the second histogram that shows that unlike ATP, TCP, and EXACT, no C3L flow is starved. Finally, C3L achieves this fairness without sacrificing the efficiency, as seen from its comparable performance in terms of average resource allocation to each flow.

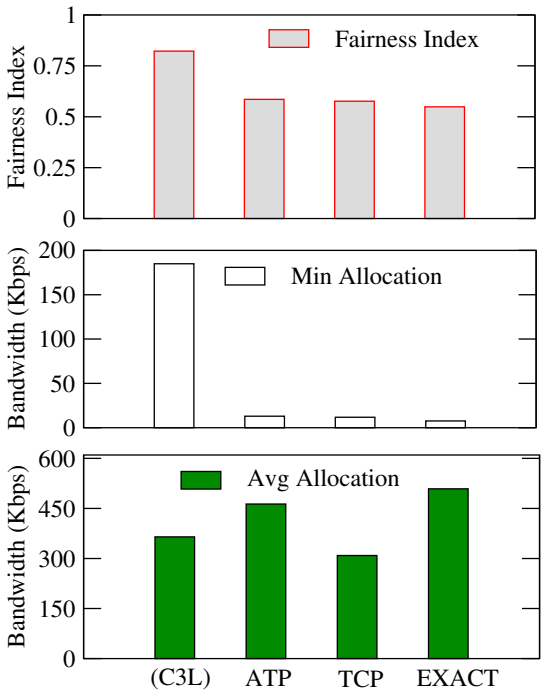

Fig. 10. Performance of C3L in a 64-node random mesh network with 4 gateways and 15 flows each destined to the closest gateway. The bandwidth distribution of C3L flows is most fair with no starvation.

end-to-end and is independent of the number of hops a flow traverses, as shown in Figure 8(b).

Finally, scenario (c) corresponds to the case of flows sharing a common radio channel, but not a common wireless node. Most existing transport protocols give more bandwidth to a flow emanating from node with fewer flows (Figure 4). In contrast, C3L recognizes this special channel sharing pattern and appropriately allocates bandwidth to competing links in proportion to the number of flows passing through them. This mechanism results in a fairer resource allocation among flows that share the same radio channel, as shown in Figure 8(c).

B. Fairness in General Network Configurations

The ability to effectively deal with the above three specific scenarios renders C3L a fairer resource allocation protocol for general networks. For this subsection, we evaluated the effectiveness of C3L for several large mesh networks. Here we present the results of two instances. For each protocol, we show the fairness index, the minimum-allocation flow's end-to-end bandwidth, and the average end-to-end flow bandwidth.

The fairness index indicates the degree of fairness in resource allocation across flows. If X_i is the end-to-end bandwidth achieved by flow i and n is the number of flows in the network, then the fairness index is computed using the following formula [26] –

$$FairnessIndex = \frac{(\sum_i X_i)^2}{n * \sum_i (X_i)^2} \quad (1)$$

The closer the fairness index is to 1, the fairer the associated resource allocation. The max-min fairness does not mean exactly equal allocation, and even the optimal fairness index may not reach 1 in many cases.

Figure 9 shows the fairness achieved by different congestion control mechanisms in a 64-node *8x8 grid network* with 20 randomly chosen flows. The fairness index for C3L is much better than other congestion control mechanisms, suggesting a more uniform allocation of bandwidth across flows by C3L. The second histogram shows the bandwidth achieved by minimum allocation flow in all the cases. C3L again does a much fairer allocation of bandwidth when compared with other protocols that end up starving some flows. Finally, the last histogram shows the average allocation across flows. The comparable average allocation of C3L suggests that C3L achieves fairness while also maximizing the utilization of the network. C3L is thus not just fair but also efficient. The average bandwidth is somewhat smaller for C3L because it gives more bandwidth to some of the flows with a larger hop count. Allocating bandwidth to a flow with a larger hop count consumes more radio resource, and therefore decreases the average flow bandwidth.

Histograms in Figure 10 show the same results for a 64-node *random mesh network* with 4 gateway nodes distributed across the network. The traffic profile comprises 15 flows originating from randomly chosen nodes, and ending at the closest gateway. Again, C3L performs a fairer and efficient allocation of bandwidth across all the flows. Figure 11 shows the actual distribution of end-to-end bandwidth achieved by different congestion control mechanisms. TCP, ATP, and EXACT give much higher bandwidth to some of the flows, while starving others. The allocation of C3L is most uniform with no obvious starvation.

To dissect the performance improvements due to different mechanisms of C3L, we took a 28-node random mesh network

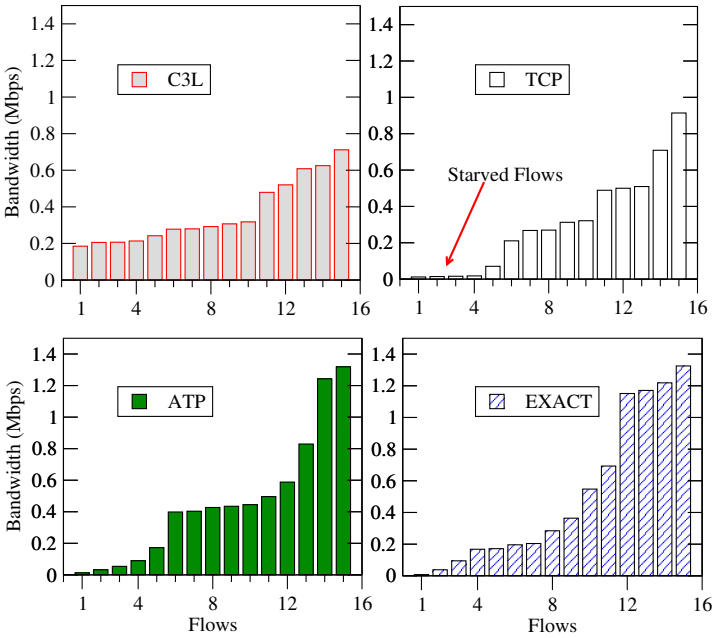

Fig. 11. Actual end-to-end bandwidth distribution across flows for results in Figure 10. C3L has the most uniform distribution of bandwidth across flows. All TCP, ATP, and EXACT lead to starvation of some of the flows.

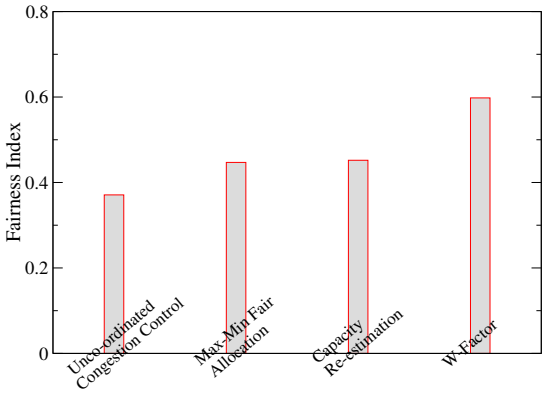

Fig. 12. Contribution of different control mechanisms in C3L to inter-flow fairness in a 28-node random mesh network with 5 end-to-end flows. The control mechanisms are introduced in the following order: Max-min fair allocation, Capacity re-estimation, and W_FACTOR.

and ran 5 flows through the network. Figure 12 shows the improvement in fairness index as we turn on each control mechanism of C3L. The leftmost bar corresponds to the case where each node estimates the bandwidth of its links locally in a way similar to EXACT. This causes some flows to starve leading to a small value of fairness index. In the second bar, we introduce max-min fairness but not the capacity re-estimation algorithm. The fairness index indeed improves with the introduction of the max-min fairness algorithm. Introducing capacity re-estimation algorithm alone does not help much. However, the introduction of capacity re-estimation along with W_FACTOR leads to a substantial improvement in fairness index. 802.11 MAC layer introduces inefficiencies and unfairness that lead to capacity reduction of some of the collision domains. Accurate estimation of each collision domain's capacity is therefore an important component of fair resource allocation.

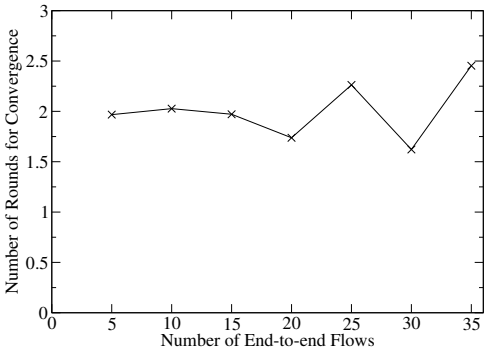

Fig. 13. Algorithm convergence speed versus number of flows. Most of the flows converge within 2 rounds of the proposed algorithm.

C. Algorithm Convergence Speed

We next evaluated the convergence properties of the C3L algorithm. We took a 64-node grid network and two randomly generated traffic profiles T_1 and T_2 of 20 flows each. For the first half of each experiment, the traffic load was exerted based on the traffic profile T_1. Then the traffic load was switched to the traffic profile T_2 in the second half and the convergence time measured. The network was assumed to be converged when 90% of the flows reach 90% of their eventual average bandwidth. The experiment was repeated with different number of flows in T_1 and T_2. The graph in Figure 13 shows that most flows converge within 2 rounds of the proposed algorithm.

One way to reduce the convergence time is to reduce the cycle time, i.e. the periodicity at which the proposed algorithm is run. Table I shows the average number of rounds it takes when different cycle times are used. At very small cycle time, the intra-link bandwidth allocation does not converge completely, hence the number of rounds required for convergence increases. In the end we choose 8 seconds, which optimizes the convergence time as well as the communication overhead, as the cycle time for all our experiments.

D. Protocol Overheads

The proposed algorithm requires additional computation for topology discovery and max-min resource allocation. Topology discovery requires incrementally finding all the cliques and is only executed when a node is added or deleted from the network. Because topology changes in a WMN are fairly infrequent, the computational cost of the incremental clique finding algorithm is less of a concern. Further, because of the physical proximity of radio interference, the degree of each conflict graph node is limited, and the number of cliques in a network tends to be linear with respect to the number of its nodes (Figure 14).

The run-time computation overhead of the algorithm that determines max-min fair allocation of flows is more critical to the overall network performance because it needs to be invoked frequently. We measure the computation time of each algorithm run with respect to the changing number of flows (Figure 15). Even with 100 flows on a 64-node grid network, this computation only takes about 1.3 msec on a standard 1.7 GHz CPU.

Cycle Time	Convergence (Number of Rounds)
2	3.31
4	3.70
8	1.34
16	1.49

TABLE I
NUMBER OF ROUNDS FOR CONVERGENCE VERSUS CYCLE TIME FOR EXECUTING THE PROPOSED CENTRALIZED MAX-MIN FAIR ALLOCATION ALGORITHM.

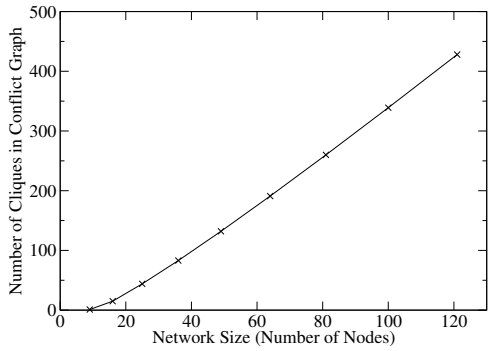

Fig. 14. Number of cliques increases linearly with the network size.

The other performance overhead associated with the proposed algorithm is the communication cost required to collect the traffic profile. This needs to be done once in every round of the centralized algorithm. In our simulations, each node sends one packet containing information about its links, flows, and routes, to the central controller. As the diameter of the network increases as $\sqrt{N}$ times the network size (N is the number of nodes), the overall communication overhead increases as $N * \sqrt{N}$. Our measurements show that with 100 flows on a 64-node grid network, the communication traffic imposed by C3L is less than 0.5% of network bandwidth.

E. C3L Evaluation Using A Miniaturized WMN Testbed

To evaluate the performance of C3L in real world, we built a miniaturized 802.11a testbed similar to MiNT [27]. The testbed consists of 4 Routerboard RB-230 nodes each equipped with an Atheros-based IEEE 802.11a/b/g wireless NIC operating in 802.11a mode. We connected each NIC to a 2 dBi antenna through a 22 dB hardware attenuator.

We reproduced the hidden terminal problem (Figure 1 (a)) in the testbed and evaluated its impact on the fairness properties of different congestion control mechanisms. Figure 16 shows

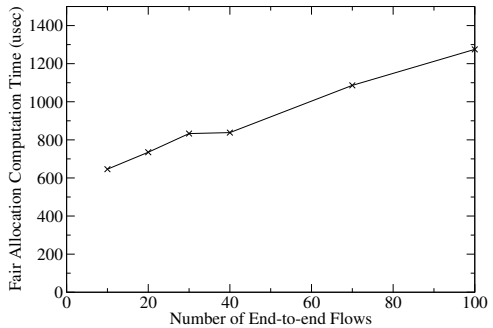

Fig. 15. Fair allocation computation time increases linearly with number of flows, but stays below 1.5 msec even with 100 flows.

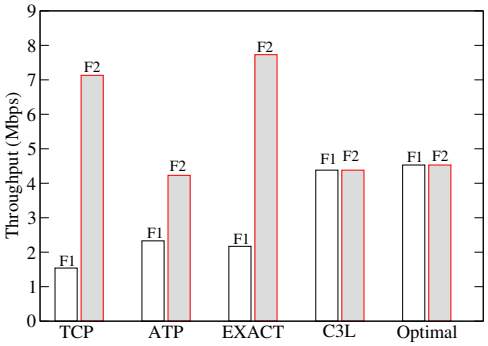

Fig. 16. Impact of hidden node problem on different congestion control mechanisms as observed on the testbed. The testbed topology is similar to the one shown in Figure 1 (a).

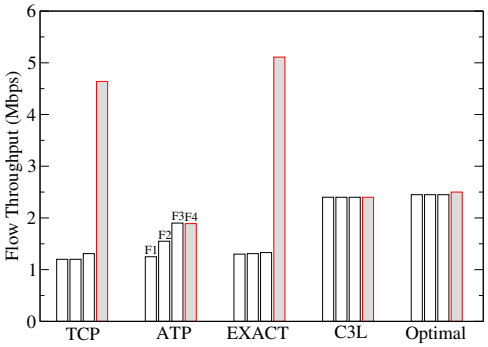

Fig. 17. Fairness of different congestion control mechanisms for channel space sharing scenario as observed on the testbed. The topology of the testbed was set in accordance with Figure 1(c).

the results. In contrast with simulation results (Figure 3), the inhibited flow does not suffer from starvation with any congestion mechanism. This is because of the differences in the NIC implementation of the backoff algorithm. Specifically, our experiments suggest that in commercially available 802.11a/b/g NICs, the maximum backoff window is set to a small value in order to achieve better performance. Similar to simulations, existing transport protocols demonstrate substantial unfairness. C3L not only remedies this unfairness, but achieves a performance close to optimal.

We similarly reproduced general channel space sharing scenario (Figure 1 (c)) in the testbed. Figure 17 shows the fairness of different congestion control mechanisms in this setup. Although all flows are sharing the same channel space, existing transport protocols allocate less bandwidth to flows F1, F2 and F3 than flow F4. C3L not only achieves fairness in this scenario, but also achieves close to optimum fairness.

A limitation of C3L is that it assumes two links are either interfering all the time or not interfering at all. While this 0-1 interference assumption works in simulations, in reality the interference between two links could be partial. Specifically, temporal variations in channels could lead to time-dependent interference, where two links sometimes interfere and sometimes not. To evaluate the impact of this 0-1 interference assumption on C3L performance, we placed 4 nodes such that link N1-N2 is on the boundary of interference zone of link N3-N4 leading to partial interference scenario, and sent two flows Flow-1 and Flow-2 over them . Figure 18 demonstrates the temporal variations by plotting the throughput of Flow-

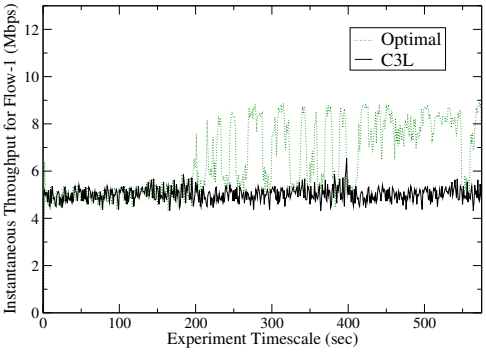

Fig. 18. Temporal variations in link interference results in throughout variations. Here, two flows Flow-1 and Flow-2 were sent over two interfering hops N1-N2 and N3-N4 respectively. The graph shows the instantaneous bandwidth achieved by Flow-1 for both cases. The average bandwidth achieved by Flow-1 and Flow-2 was 4.85 Mbps and 4.90 Mbps respectively using C3L, and 6.43 Mbps and 6.52 Mbps respectively using the optimal algorithm.

1 for the optimal case and the C3L case. In all, the optimal algorithm which instantly adapts the sending rates of two flows based on changes in interference between them could achieve 6.43 Mbps and 6.52 Mbps for Flow-1 and Flow-2 respectively. In contrast, C3L made a conservative assumption that the two links always interfere, and on an average achieved 4.85 Mbps and 4.90 Mbps respectively. This result does not conflict with Theorem 1, as here the capacity of the collision domain is indeed changing over time.

V. Conclusion

From an individual WMN user's standpoint, a realistic partial failure threat on an IEEE 802.11-based WMN is severe throughput degradation due to unfair radio resource allocation, which in turn results from weaknesses in the MAC and transport protocol. Existing solutions to this problem either cannot effectively eliminate the root cause of unfairness, or require modifications at the MAC layer. At the same time, congestion control schemes in state-of-the-art transport protocols for WMNs, such as TCP, ATP and EXACT, actually exacerbate this unfairness problem and end up allocating a WMN's resource in an unfair way among flows sharing the same WMN. In this paper, we show that it is possible to support inter-flow fairness over IEEE 802.11's MAC protocol through purely transport-layer mechanisms. Specifically, we propose a *coordinated congestion control* algorithm that, through real-time measurements, delivers significantly better inter-flow fairness than previously proposed solutions. To the best of our knowledge, C3L is the first research effort that successfully eliminates fairness-related partial failures over IEEE802.11-based WMNs. C3L takes a centralized traffic engineering approach, and continuously adapts the resource allocation to individual wireless links as well as to individual flows sharing the same link. Unlike some of the previous proposals, C3L ensures end-to-end flow fairness by taking into account both intra-flow and inter-flow dependencies. Moreover, C3L includes a general capacity re-estimation algorithm that can effectively mitigate the fairness problem due to the IEEE 802.11 MAC protocol.

The overall goal of this research project is to architect a transport protocol that can work well over 802.11-based wireless mesh networks. We separate the reliability and congestion control aspects of the transport protocol, and specifically focus on the latter in this paper. For reliability, we are investigating cross-layer optimization techniques to determine packet delivery status and perform localized lost packet retransmissions. For congestion control, we are developing a distributed version of the proposed coordinated congestion control algorithm that can achieve the same level of end-to-end flow fairness.

References

[1] S. Xu, T. Saadawi "Does IEEE 802.11 Work Well in Multi-hop Wireless Network?" IEEE Communications Magazine, Vol. 39, No. 6., 2001.
[2] X. Huang, B. Bensaou; "On Max-min Fairness and Scheduling in Wireless Ad-Hoc Networks: Analytical Framework and Implementation," Proc of the ACM Mobihoc, 2001.
[3] T. Nandagopal, T. Kim, X. Gao, V. Bharghavan; "Achieving MAC Layer Fairness in Wireless Packet Networks," Proc of the ACM MobiCom, 2000.
[4] L. Tassiulas, S. Sarkar; "Maxmin fair scheduling in wireless networks," Proc of IEEE Infocom, 2002.
[5] B. Li; "End-to-End Fair Bandwidth Allocation in Multi-hop Wireless Ad Hoc Networks," Proc of the ICDCS, 2005.
[6] H. Luo, S. Lu, V. Bharghavan; "A New Model for Packet Scheduling in Multihop Wireless Networks," Proc of the ACM Mobicom, 2000.
[7] K. Chen, K. Nahrstedt, N. Vaidya; "The Utility of Explicit Rate-Based Flow Control in Mobile Ad Hoc Networks," Proc of the IEEE WCNC, 2004.
[8] M. Kazantzidis, M. Gerla; "End-to-end versus Explicit Feedback Measurement in 802.11 Networks, " Proc of the 7th ISCC, 2002.
[9] R. Jain, S. Kalyanaraman, R. Viswanatha; "Rate Based Schemes: Mistakes to Avoid," ATM Forum/94-0882, September, 1994.
[10] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, M. Gerla,; "On TCP performance in multihop wireless networks," UCLA WiNG TR, 2002.
[11] A. Bakre, B. R. Badrinath; "I-TCP: Indirect TCP for Mobile Hosts," Proc of the ICDCS, 1995.
[12] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, R. H. Katz; "A Comparison of Mechanisms for Improving TCP performance over Wireless Links," Proc of ACM SIGCOMM, 1996.
[13] H. Balakrishnan, R. H. Katz; "Explicit Loss Notification and Wireless Web Performance," Proc of IEEE Globecom, 1998.
[14] P. Sinha, N. Venkitaraman, R. Sivakumar, V. Bharghavan; "WTCP: A Reliable Transport Protocol for Wireless Wide-Area Networks," Proc of ACM Mobicom, 1999.
[15] P. Bhagwat, P. Bhattacharya, A. Krishma, S. Tripathi; "Using channel state dependent packet scheduling to improve TCP throughput over wireless LANs," Wireless Network, Vol 3, No 1.
[16] K. Sundaresan, V. Anantharaman, H. Y. Hsieh, R Sivakumar; "ATP: A Reliable Transport Protocol for Ad Hoc Networks," Proc of the ACM Mobihoc, 2003.
[17] S. Biaz, N. H. Vaidya; "Discriminating Congestion Losses from Wireless Losses using Inter-Arrival Times at the Receiver," IEEE Symposium ASSET, 1999.
[18] H. Balakrishnan, V. Padmanabhan, R. Katz; "The Effects of Asymmetry on TCP Performance," Mobile Computing and Networking, 1997.
[19] G. Holland, N. Vaidya.; "Analysis of TCP performance over mobile ad hoc networks," Proc of the ACM Mobicom, 1999.
[20] V. Gambiroza, B. Sadeghi, E. Knightly; "End-to-end performance and fairness in multihop wireless backhaul networks," Proc of the ACM Mobicom, 2004.
[21] R. Chakravorty, A. Clark, I. Pratt; "GPRSWeb: Optimizing the Web for GPRS Links," Proc of the USENIX Mobisys, 2003.
[22] Y. Yi, S. Shakkottai; "Hop-by-hop congestion control over a wireless multi-hop network," Proc of the IEEE Infocom, 2004.
[23] K. Jain, J. Padhye, V. N. Padmanabhan, L. Qiu; "Impact of interference on multi-hop wireless network performance," Proc of the ACM Mobicom, 2003.
[24] A. Raniwala, T. Chiueh; "Architecture and Algorithms for an IEEE 802.11-based Multi-channel Wireless Mesh Network," Proc of the IEEE Infocom, 2005.
[25] C. Bron, J. Kerbosch; "Algorithm 457: finding all cliques of an undirected graph," ACM Communications, Vol 16, Issue 9, Sept 1973.
[26] R. Jain, A. Durresi, G. Babic; "Throughput Fairness Index: An Explanation," ATM Forum/99-0045, February 1999.
[27] P. De, A. Raniwala, S. Sharma, T. Chiueh; "MiNT: A Miniaturized Network Testbed for Mobile Wireless Research," Proc of the IEEE Infocom, 2005.