How Real-time Bandwidth Inference Improves the Congestion Control in Wireless Mesh Networks

Luís Barreto[1], Bruno Rés[2], Susana Sargento[2]
[1] Instituto Politécnico de Viana do Castelo, Portugal
lbarreto@esce.ipvc.pt
[2] Instituto de Telecomunicações, Universidade de Aveiro, Portugal
{bruno.res,susana}@ua.pt

***Abstract*—This paper presents a new method to estimate the available bandwidth and the path capacity over a wireless network path, denoted as rt-Winf and its integration with XCP and RCP. The estimation is performed in real-time and without the need to intrusively inject packets in the network. This is accomplished by resorting to the CSMA-CA scheme with RTS/CTS packets to determine each node's channel allocation. rt-Winf repeatedly samples the available bandwidth of the network path, requiring little computation in each iteration, and being lightweight with respect to memory requirements. rt-Winf has been tested both in the CMU Wireless Emulator and the ns-2 simulator, using mesh networks scenarios. The obtained results show that rt-Winf obtains the available bandwidth and capacity estimation with the highest accuracy and without introducing overhead traffic in the network. The simulation results of the congestion control approaches, conducted in ns-2, also show that rt-Winf integration with XCP and RCP improves their behavior.**

***Index Terms*—available bandwidth, path capacity, measurements, performance, wireless networks, congestion control, transport protocol.**

I. Introduction

As stated in [1] "The deployment of wireless mesh networks (WMNs) reveals that despite the advances in physical-layer transmission technologies, limited capacity, and consequently available bandwidth, continues to be a major factor that limits the performance of WMNs and severe congestion collapses are pervasive within WMNs". A congestion control scheme which provides an efficient and accurate sharing of the underlying network capacity among multiple competing applications is crucial to the efficiency and stability of WMNs. Then it is of major importance to obtain accurately link capacity and available bandwidth and, then, use these parameters actively in WMNs congestion control.

In a network path we have a sequence of *H* store-and-forward links that transfer packets from a sender to a receiver. Each link i can transmit data at a rate C_i, referred as link capacity. Then, the wireless link end-to-end capacity can be defined as $C \equiv \min_{i=1,..H} C_i$. The available bandwidth can, thus, be defined as the fraction of the links capacity that has not been utilized during a period of time. If we extend this concept to the entire path, the end-to-end available bandwidth is the minimum available bandwidth among all links in the path.

Available bandwidth and path capacity estimation have been widely studied, but most of the mechanisms work in scenarios with wired and/or last hop wireless networks. Tools like [2][3] are some of those examples. These tools work sending out a series of various probe packets with different sizes and, for each probe, they measure the time an error packet is received. The bandwidth of each link and its latency are obtained through statistical analysis of those measurements. Other tools like AbGet [4] and PathLoad [5] rely on self induced congestion. AdHoc probe [6] is a wireless active measurement technique that uses packet pairs to measure the end to end path rate based on one way measurements. IdleGap [7] is different from the previous approaches as it is a new mechanism to infer the available bandwidth in a passive way, without the need to use probe packets.

In WMNs, packet loss is typically due to: wireless channel impairments causing bit errors, handoffs due to mobility and, of course, possibly congestion. The most used congestion control protocol Transmission Control Protocol (TCP) [8] assumes that a packet loss is always due to congestion in the network and, but not as often, of packet reordering. TCP does not respond well to packet loss due to bit errors and handoffs making TCP-based applications suffering of poor performance.

Having in mind the previous considerations, we propose an on-line capacity and available bandwidth technique, called rt-Winf, and its integration with the eXtensible Control Protocol (XCP) [9] and the Rate Control Protocol (RCP) [10] congestion control techniques. As XCP and RCP are two congestion control mechanisms that actively use link capacity and available bandwidth, the proposed technique is applied on them. rt-Winf is a novel available bandwidth and path capacity estimation tool based in IdleGap [7]. The obtained link capacity and available bandwidth are then passed, through cross-layer techniques, to XCP and RCP that use that information in their native congestion control techniques. We call these adapted protocols XCP-Winf and RCP-Winf.

The rest of this paper is organized as follows. Next section, section II, briefly presents the background and related work. Then, section III describes the rt-Winf algorithm. In section IV it is presented how rt-Winf was integrated with XCP and RCP. Section V describes and discusses the results obtained through simulation. Finally, section VI presents the conclusions and future work.

II. Background and Related Work

A. Capacity and Available Bandwidth Estimation

Link capacity estimation has been widely studied in wired networks. IPerf [11], CapProbe [3] and Pathchar [2] are some examples. AbGet [4] and Pathload [5], are some examples of available bandwidth estimation methods. There are also developments with respect to wireless networks, such as AdHoc Probe [6], WBest [12] and IdleGap [7]. AdHoc Probe provides only the path capacity of the wireless channel. WBest calculates both capacity and available bandwidth.

WBest contains an algorithm that is divided in two phases. In the first one, it uses packet pair techniques in order to determine the capacity. In the second phase, it uses packet train techniques in order to determine the available bandwidth. In fact, in this phase, packets are sent at the rate obtained in the first phase. It means that in this period of time the WBest tool is being very intrusive, causing undesired problems in the network.

IdleGap is a recent mechanism for obtaining available bandwidth in wireless networks. IdleGap is focused in the CSMA Collision Avoidance (CSMA-CA) scheme of wireless networks. It takes Network Allocation Vector (NAV) [13] into consideration, that is then used by the idle nodes which are waiting to transmit. It uses a very accurate approach to characterize the busy time and the total elapsed time, obtaining a very accurate *Idle Rate*. However, IdleGap uses the pre-defined IEEE802.11 header *DataRate* [13] value, which is not practical and real, thus leading to not very accurate and over-dimensioned estimation values. It is not realistic in the determination of link capacity and introduces a new sublayer in the model stack.

The authors of IdleGap propose the consideration of 3 different states for a wireless node: *Sender*, *Receiver* and *Onlooker*. These states are distinguished on the *Idle Module*, which is the module used to determine the *Idle Rate*. The introduction of the *Idle Module* has an important disadvantage, that is the modification of the OSI Model, by the introduction of a new sublayer. rt-Winf algorithm will use some of the concepts of IdleGap, but it will not change the OSI model.

B. Congestion Control

The Transmission Control Protocol (TCP) [8] is the most used congestion control protocol in computer networks. TCP uses the *Additive Increase Multiplicative Decrease* (AIMD) algorithm and the *slow-start* mechanism [14]. It is able to also provide TCP congestion avoidance and recovery. Due to its AIMD strategy, TCP is known to have some limitations: unstable throughput, increased queuing delay, limited fairness. TCP assumes that, in its operation and with today's network improvements, the probability of a lost packet is higher than the one of a corrupted packet [15]. It is important to notice, that this is not a true statement in WMNs. Some new and specific congestion control mechanisms try to enhance TCP behavior in WMNs. Mechanisms like TCP-F [16], TCP-ELFN [17], TCP-BuS [18], ATCP [19] represent some examples of protocols for wireless networks in general.

The eXtensible Control Protocol was designed to extract congestion information directly from routers. According to [20], "XCP achieves fairness, maximum link utilization and efficient use of bandwidth". XCP is also scalable, as per-flow congestion state is carried in packets. However, XCP requires changes to be made in all routers and end-systems in the network. A XCP network is composed by XCP sender hosts, receiver hosts and intermediate nodes where queuing from the sender to the receiver occurs. XCP uses a feedback mechanism to inform the sender about the network conditions, that is, the maximum throughput.

The Rate Control Protocol (RCP) is part of 100x100 clean state project [21]. The mission of this project is to create blueprints for a network that goes beyond today's Internet [21]. RCP, similarly to XCP, is a congestion control algorithm. The main goal of RCP is to deliver fast flow-completion times or download times. RCP was also designed having in mind typical flows of typical users in today's Internet (traffic bursts).

III. rt-Winf

The rt-Winf mechanism [22] was developed inspired by IdleGap [7], but with the purpose to mitigate the problems previously mentioned, being compatible with all systems and determining both the link capacity and available bandwidth without overloading the network. These characteristics overstep AdHoc Probe and WBest inherent limitations and problems. rt-Winf does not introduce any change to the OSI Model, as opposed to IdleGap, being able to obtain all the necessary times to calculate the path capacity and available bandwidth. Another important aspect of rt-Winf, relatively to IdleGap, is the effective calculation of the capacity, instead of using the *DataRate* value of the IEEE802.11 header [13].

A. RTS/CTS Packets

rt-Winf with RTS/CTS control packets enabled relies on this handshake to correctly retrieve the NAV values. In order to evaluate the accuracy of the duration field on the IEEE802.11 header, we performed a large number of captures ($\sim$ 200). We concluded that the duration value on data packets is not reliable, because different sized packets have always the same duration. The RTS/CTS packets have accurate duration values, which can be used in the calculations.

With the obtained captures, it was possible to realize how each state managed the received packets. In the case of the *Sender* state, the node was able to capture the CTS, DATA and ACK packets. A node in the *Receiver* state was able to capture the RTS and the DATA packets, while a node in the *Onlooker* state was able to capture the complete set of packets: RTS, CTS, DATA and ACK. This different knowledge implied the conception of different algorithms for each state. Then, we propose that each node state uses a different method to determine the *Idle Rate*. In the case of the *Sender*, it is considered the NAV of the CTS packets on the available bandwidth calculation. For the capacity calculation, it is considered the time that the channel is busy, that is, the difference between ACK time, CTS time and the duration of the occured Short Inter-Frame Spacing - SIFS (where ACK time is the actual clock time when the ACK packet is received, and CTS time is the clock time when CTS packet is received). The *Receiver* uses the NAV of the RTS packets to obtain

the *Idle Rate*, and the difference between the DATA time, RTS time and 3 times SIFS to obtain the capacity (where DATA and RTS times are, respectively, the clock time when DATA packet is received and RTS packet is received). The *Onlooker* uses the NAV value according to the existence, or not, of the RTS packet to obtain both the available bandwidth and capacity. If a node in the *Onlooker* state captures a CTS packet of a communication without capturing the RTS packet, this implies that the communication is suffering from the *hidden nodes* problem. Thus, the algorithm will only use the NAV from the CTS packet to retrieve the correct values. The total elapsed time represents the difference between the last captured ACK time and the initial time. The packet size considered is the DATA packet size. Figure 1 shows the different approaches for each state while Figure 2 represents the state diagram of the rt-Winf tool. It is possible to observe each state's transitions. When a node is not transmitting or receiving packets it is on the *Onlooker* state. In this state, the node calculates the onlooking capacity. Thus, it can use this information, when changing to the *Sender* or *Receiver* state. The onlooking capacity is obtained as described in Figure 1. When a CTS packet is captured by the *Sender*, it starts to evaluate the available bandwidth and capacity, while the *Receiver* starts this process when a RTS packet is received. The *Receiver* sends the calculated available bandwidth and capacity in an ACK packet to the *Sender*. When the *Sender* receives, from the *Receiver*, the ACK packet with that information, it compares it with the available bandwidth and capacity that it has previously calculated. If the information received through the ACK packet is lower than the obtained, the *Sender* will use the available bandwidth and capacity received in the ACK packet. Otherwise, the *Sender* will transmit using the available bandwidth and capacity calculated before. This cooperation is a great improvement when compared to IdleGap.

B. Probe Packets

If RTS/CTS packets are not present, rt-Winf can use probe packets in order to retrieve the transfer time values. Probe packets can be sent between nodes. These must be UDP generated packets with altered Frame Control IEEE 802.11 header: Type Data and Subtype Reserved. We used packets with Frame Control Type set to 10 (data) and Subtype to 1001 (Reserved). This way the *Sender* and the *Receiver* can successfully differentiate these packets from the ordinary data packets. IEEE802.11 standard defines that, for each successfully received packet, it must be sent a MAC ACK packet [13]. The whole process is very similar to the one with the RTS/CTS handshake.

The generated packets are used to retrieve the capacity and available bandwidth values, according to Equation 1 and Equation 2. These packets are only sent before a node wants to start a transmission and in the absence of traffic. This allows the system to initially determine the available bandwidth and capacity. Then, the existing traffic and the MAC layer ACK will be used to trigger the calculations. As NAV values are not correctly defined in DATA packets, rt-Winf uses clock time information to determine the busy time. So, NAV values are not considered in this specific implementation with probe

State	Available Bandwidth	Capacity
On-looking	Captured RTS Packet? YES: $AB=C(1-\frac{\sum \text{NAV}_{\text{RTS}}}{\text{Total elapsed time}})$ NO: $AB=C(1-\frac{\sum \text{NAV}_{\text{CTS}}}{\text{Total elapsed time}})$	$C=\frac{\text{Packet Size}}{ACK_{Time}-CTS_{Time}-2\text{SIFS}}$
Sender	$AB=C_{Sender}(1-\frac{\sum \text{NAV}_{\text{CTS}}}{\text{Total elapsed time}})$	$C_{Sender}=\frac{\text{Packet Size}}{ACK_{Time}-CTS_{Time}-2\,SIFS}$
Receiver	$AB=C_{Receiver}(1-\frac{\sum \text{NAV}_{\text{RTS}}}{\text{Total elapsed time}})$	$C_{Receiver}=\frac{\text{Packet Size}}{DATA_{Time}-RTS_{Time}-3\text{SIFS}}$

Figure 1. rt-Winf Algorithm.

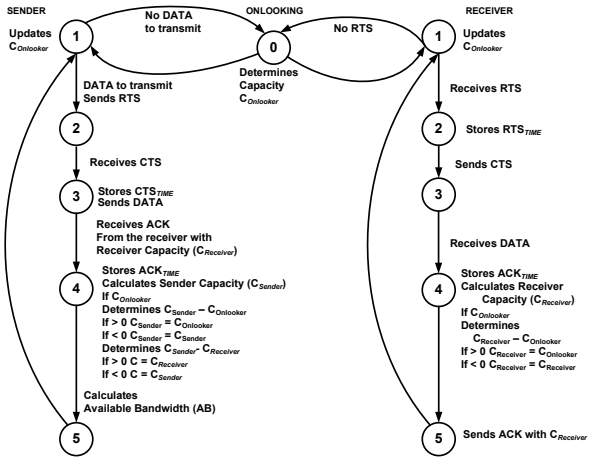

Figure 2. rt-Winf Sender, Receiver and Onlooking State Diagrams.

packets. To be fully operational, both Sender and Receiver must be running the rt-Winf mechanism. After each transfer Equation 1 and Equation 2 are used to obatin, respectively, the C and the AB. It is used a sliding window technique, where for each $t < t_{max}$ all values are set to its initial state, allowing to reflect the variation that may occur.

$$C = \frac{PacketSize}{TransferTime} \tag{1}$$

where *TransferTime* is equal to $ACKTime - DataTime$.

$$AB = 1 - \left(\frac{\sum_{t=0}^{t<t_{max}} TransferTime}{TotalElapsedTime} \right) * C \tag{2}$$

The overhead is the result of introducing, before each complete flow, a packet with a maximum size of 1500 bytes. No other packtes, beyond the DATA ones, are needed. In a normal VoIP call, the overhead introduced by this mechanism is $\sim 1.66\%$. For a flow with more than 1Mbps, the overhead is less than $\sim\ 0.15\%$.

IV. XCP-WINF AND RCP-WINF

As rt-Winf obtains available bandwidth and capacity values in the MAC layer, this information can be transferred to congestion control mechanisms, like XCP and RCP, for improving their behavior. We then integrated rt-Winf with XCP and RCP. In the integration, all operating principles of XCP and RCP are unchanged, the main difference is that the information on available bandwidth is obtained in the MAC layer. The rt-Winf information is sent to the network layer through a cross layer communication process. For this communication

Algorithm 1: XCP-Winf Router/Onlooker Control Interval Timeout Operations.

$Available\_Bandwidth_{Winf}$: rt-Winf obtained available bandwidth.
avg_rtt: average rtt value, used to determine the control interval.
F_{Winf}: Aggregated Feedback, uses rt-Winf values.
C_p: positive feedback scale factor.
C_n: negative feedback scale factor.
$residue\_pos\_fbk$: pool of available positive capacity a router has to allocate.
$residue\_neg\_fbk$: pool of available negative capacity a router has to allocate.
$MIN\_INTERVAL$: propagation delay on link, value between 5 and 10 ms.

On estimation control timeout do:

$avg\_rtt = \frac{sum\_rtt\_by\_throughput}{sum\_inv\_throughput}$;
$input\_bw = Available\_Bandwidth_{Winf}$;
$F_{Winf} = a \times (C_{Winf} - input\_bw) - b \times \frac{queue}{avg\_rtt}$;
$shuffled\_traffic = max(0, 0.1 \times input\_bw - |F_{Winf}|$;
$residue\_pos\_fbk = shuffled\_traffic + max(F_{Winf}, 0)$;
$residue\_neg\_fbk = shuffled\_traffic + max(-F_{Winf}, 0)$;
$C_p = \frac{residue\_pos\_fbk}{sum\_inv\_throughput}$;
$C_n = \frac{residue\_neg\_fbk}{input\_traffic}$;
$input\_traffic = 0$;
$sum\_inv\_throughput = 0$;
$sum\_rtt\_by\_throughput = 0$;
$ctl\_interval = max(avg\_rtt, MIN\_INTERVAL)$;
$timer.reschedule(ctl\_interval)$;

Algorithm 2: RCP-Winf Router/Onlooker Rate Estimation Timer Operations.

rcp_rate: the bandwidth offered to a flow.
MIN_RATE: the minimum value for rcp_rate.
ETA: a constant value.
C_{Winf} : rt-Winf obtained Capacity.

On rate estimation timer timeout do:
.....
if *($rcp\_rate < MIN\_RATE$)* **then**
  $rcp\_rate = MIN\_RATE$;
else if *($rcp\_rate > ETA \times C_{Winf}$)* **then**
  $(rcp\_rate = ETA \times C_{Winf})$;
.....

system, it was used a shared database architecture, with a set of methods to get/insert information in a database accessible by all protocol layers. One example of such architecture is the MobileMan cross-layered network stack [23]. After obtaining the available bandwidth and the link capacity, rt-Winf inserts that information in the shared database and then, XCP and RCP access that information and update their functions with the accessed information.

A. XCP-Winf and RCP-Winf Functions

This section briefly describes the XCP/RCP-Winf functions. Compared to base XCP and RCP, the only functions that are changed are the XCP/RCP Sender and XCP/RCP Router functions. The XCP/RCP Receiver is not changed as its operations remain the same. When acknowledging a packet, the XCP/RCP-Winf Receiver copies the congestion header from the data packet to the corresponding acknowledgment packet and acknowledges the data packet in the same way as a TCP receiver.

Next, we present the corresponding algorithms for some of the XCP/RCP-Winf Router functions. In Algorithm 1 it is presented one of the phases of the *Onlooker* operations for a XCP-Winf Router system, which is the control interval timeout packet. Algorithm 2 shows some of the per-packet operations performed by a RCP-Winf router when the rate estimation timer expires.

V. Simulation Results

This section shows simulation results of our proposed mechanisms. The results are obtained using the CMU Wireless Emulator [24] and the ns-2 simulator [25]. Although the emulator provides more realistic results than a traditional network simulator, we also present ns-2 simulations for comparison purposes. Moreover, ns-2 is used for the simulation of the congestion control approaches. In base rt-Winf, the system is configured with enabled RTS/CTS/ACK handshake packets. In rt-Winf probe, RTS/CTS/ACK handshake is not enabled, and probe packets are implemented; the maximum achievable data rate is set to 11 Mbps. Nodes are placed in such a distance that the path loss effect is considered negligible. The three states defined by rt-Winf mechanism and the cooperation between them and between the nodes was developed in C language. Several scenarios were used, varying the number of nodes and the traffic load.

A. rt- Winf Wireless Mesh Network Results

In the evaluation of available bandwidth and path capacity, the testbed used in the CMU emulator is composed by two mobile nodes communicating with each other through two mesh nodes responsible for the routing and link management. The mobile nodes are in such a distance that the traffic is routed by the mesh nodes. In this scenario there is only a Constant Bit Rate (CBR) 64 Kbps UDP traffic from and to the mobile nodes, with RTS/CTS enabled. Each simulation was conducted independently. The residual traffic on the network is introduced by each mechanism.

The obtained results of path capacity are shown in Figure 3, and the results of available bandwidth are shown in Figure 4. In this figure, we show the results provided by rt-Winf, IPerf UDP and IdleGap. Maximum throughput values are also presented, being considered as an upper bound of the results. The lower bound is the IPerf UDP result [22].

As observed in Figure 3, rt-Winf is less sensitive to variations when compared to AdHoc Probe. This is because rt-Winf is taking into consideration all packets in the network and is measuring the channel occupation time of each packet, while AdHoc Probe is only considering the packets that it generates, thus, being more sensitive to flow variations.

Through the results in Figure 4, it is possible to observe how IdleGap is not effectively measuring the available bandwidth. IdleGap values have a small variation, but are near the *DataRate* value, which is also higher than the maximum achievable throughput, and is not taking into consideration the network conditions. As opposed to IdleGap, rt-Winf provides more real results; those results are within an upper bound, the maximum theoretical throughput, and a lower bound, IPerf UDP.

In order to compare the values of the emulator with the ones of the simulator, and also to investigate the behavior of

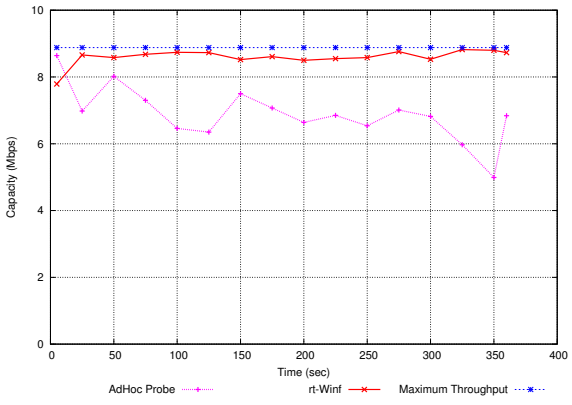

Figure 3. Wireless Mesh Scenario Path Capacity

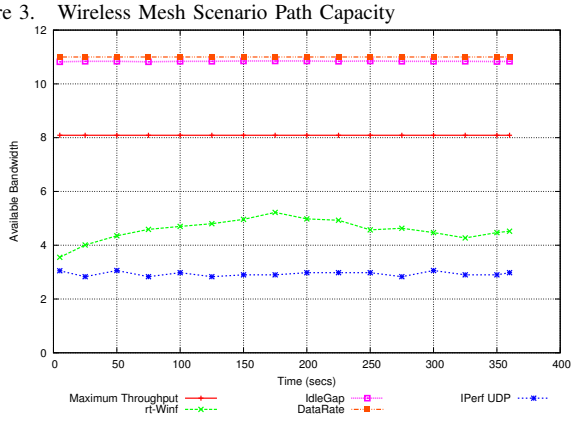

Figure 4. Wireless Mesh Scenario Available Bandwidth

rt-Winf with probe packets in a wireless mesh scenario, we also conducted some simulations in the ns-2 simulator [25]. As rt-Winf is based in IdleGap, the simulations also compares those two tools. Each simulation consists of a FTP transfer from a source to a sink, with different simultaneous flows. The maximum throughput is calculated, using ns-2 default values, applying the method described in [26].

Figure 5 summarizes the obtained results. Each value is represented by an average and a 95% confidence interval, which results from 20 runs lasting 300 seconds of simulated time; the nodes are stationary. As observed, IdleGap results are very similar to the the maximal theoretical throughput, as it is using in the calculations the IEEE802.11 Header *DataRate* value. These results validate the ones obtained with the CMU Emulator, since the results for 1 flow in Figure 5 are similar to the ones of Figure 3. It is also possible to conclude that rt-Winf with probe packets (different sizes were used) is also efficiently measuring the capacity, and its values are very similar to the rt-Winf mechanism working with RTS/CTS control packets.

B. XCP-Winf and RCP-Winf Results

This section shows the simulation results of XCP-Winf and RCP-Winf network performance evaluation. The network performance is analyzed by two important parameters: throughput and the number of received packets. The results are obtained using the ns-2 simulator [25]. In the simulations we used various mesh topologies scenarios: a grid of 5, 9, 12 and 16 fixed mesh nodes. In all mesh topologies, it was used a

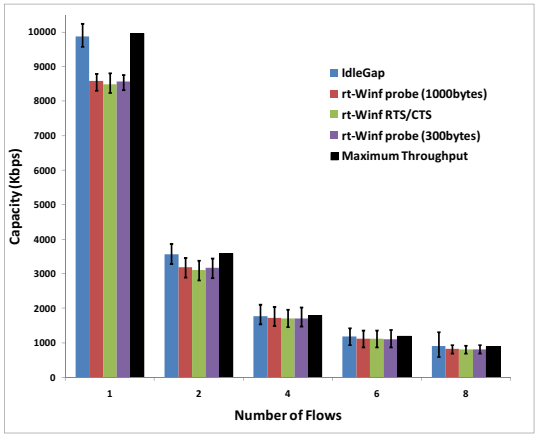

Figure 5. Ns-2 Capacity Results.

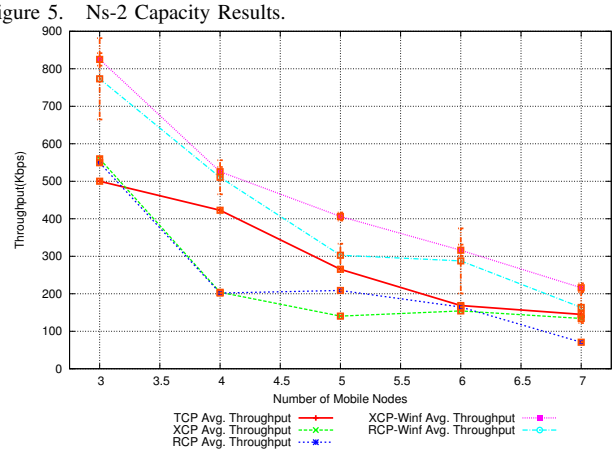

Figure 6. 16 Mesh Nodes - Variable Number of Mobile Nodes, Throughput.

combination of 3, 4, 5, 6 and 7 mobile nodes. The mobile nodes were, simultaneously, sources and sinks. The routing protocol used was the Destination-Sequence Distance-Vector (DSDV) [27].

All simulations last for 300 seconds. The simulations were repeated 10 times with different ns-2 seed values, and both mean and 95% confidence values are presented. We used ns-2 default transmission range and interference range, the channel data rate is 11 Mbps. For the data transmissions, it is configured an FTP application with packets of 1500 bytes. In the mobile nodes, the ns-2 *setdest* tool is used. This tool generates a random node movement pattern. We configure *setdest* with a minimum speed of 10 m/s, a maximum speed of 30 m/s and a topology boundary of 1000x1000 meters. All results were obtained from ns-2 trace files.

Figure 6 and Figure 7 show the previously referred performance metrics for five different scenarios. In each scenario it was used a fixed number of 16 mesh nodes and a variable number, from 3 to 7, of mobile nodes. Figure 6 shows how throughput is improved in XCP and RCP with rt-Winf; it is also possible to see that the new results are much better than the ones obtained with TCP. These results represent an improvement in throughput. The throughput values of XCP-Winf are $\sim$ 47% to $\sim$ 60% better than the ones with TCP, while with the base XCP throughput values were worse than TCP. For RCP-Winf, the percentages when compared to TCP

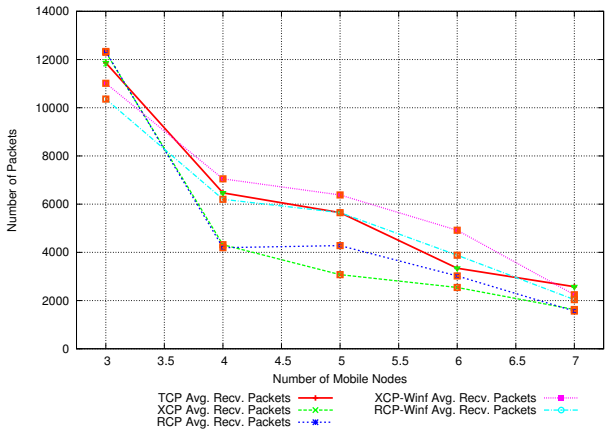

Figure 7. 16 Mesh Nodes - Variable Number of Mobile Nodes, Received Packets.

are between $\sim 17\%$ and $\sim 56\%$. In terms of received packets, as observed in Figure 7, it is also possible to see that with rt-Winf integrated, both XCP and RCP can receive more packets, which reflects a lower rate of lost packets. This is due to the fact that XCP-Winf and RCP-Winf, with accurate link capacity and available bandwidth, are using more efficiently the medium and improving each nodes queue management. Then, the nodes, and of course the network, can transmit with a higher rate and less losses.

The results show that the integration of rt-Winf in XCP and RCP improves significantly their behavior. The available bandwidth and capacity evaluation of rt-Winf, and the cross-layer information, are important and surely make XCP and RCP behave more consistently and with better channel utilization (this also leads to less channel losses).

VI. Conclusions and Future Work

This paper presented a new mechanism, based in Idle-Gap, for the measurement of wireless capacity and available bandwidth and its integration in XCP and RCP. rt-Winf uses information already available in the network: it can rely on the CTS/RTS/ACK messages handshake or on small probes. These packets provide time information, allowing to know each node's channel allocation. rt-Winf can be supported by any existing wireless equipment without the need to change the wireless NIC drivers.

The obtained results, conducted in wireless mesh networks, show that rt-Winf efficiently performs the desired calculations, providing accurate results without the need to negatively influence the network. rt-Winf can be used in a passive way, measuring the existing traffic of the wireless links, without the need to introduce more traffic in the network.

The performance evaluation study of both XCP and RCP integrated with rt-Winf clearly shows that the rt-Winf algorithm improves significantly XCP and RCP behavior, making them more stable and fair. Using rt-Winf working in the MAC layer, it is possible to perform link capacity and available bandwidth calculations without interfering in the network dynamics, allowing to significantly improve XCP and RCP performance.

As future work, we plan to compare our congestion control mechanism with other protocols, namely some of the developed TCP enhancements for WMNs, with the support of a real experimental setup.

References

[1] I. F. Akyildiz, X. Wang, and W. Wang, "Wireless mesh networks: a survey," *Computer Networks*, vol. 47, no. 4, pp. 445 – 487, 2005.

[2] V. Jacobson, "Pathchar: A tool to infer characteristics of internet paths," *MSRI talk*, April 1997.

[3] R. K. et al., "Capprobe: A simple and accurate capacity estimation technique." *ACM SIGCOMM*, vol. 34, no. 4, 2004.

[4] D. A. et al., "Available bandwidth measurement as simple as running wget," *Passive and Active Measurement (PAM) Workshop*, 2006.

[5] J. M. et al., "End-to-end available bandwidth: Measurement methodology, dynamics and relation with TCP throughput," *ACM SIGCOMM*, August 2002.

[6] L.-J. C. et al., "Adhoc probe: Path capacity probing in wireless ad hoc networks," *Wireless Networks*, vol. 15, no. 1, pp. 111–126, 2009.

[7] H. K. L. et al., "Bandwidth estimation in wireless LANs for multimedia streaming services," *Advances in Multimedia*, vol. 2007, no. 1, pp. 9–9, 2007.

[8] J. Postel, "Transmission control protocol," RFC 793, Defense Advanced Research Projects Agency.

[9] D. Katabi, M. Handley, and C. Rohrs, "Congestion control for high bandwidth-delay product networks," *ACM SIGCOMM*, August 2002.

[10] N. Dukkipati and N. McKeown, "Why flow-completion time is the right metric for congestion control," *ACM SIGCOMM*, 2006.

[11] "IPerf." [Online]. Available: http://dast.nlanr.net/Projects/Iperf/

[12] M. L. et al., "Wbest: a bandwidth estimation tool for ieee 802.11 wireless networks," *IEEE LCN*, October 2008.

[13] *IEEE Std 802.11 - IEEE Standard for Information technology — Telecommunications and information exchange between systems — Local and metropolitan area networks — Specific requirements. Part 11: Wireless LAN Medium Access Control (MAC) and Physical Layer (PHY) Specifications*, IEEE Computer Society Standard, 2007. [Online]. Available: http://standards.ieee.org/getieee802/download/802.11-2007.pdf

[14] V. Jacobson and M. J. Karels, "Congestion avoidance and control," *ACM SIGCOMM Computer Communication Review*, August 1988.

[15] B. A. Fourouzan, *TCP/IP Protocol Suite*, 2nd ed. McGraw-Hill Higher Education, 2002.

[16] K. Chandran, S. Raghunathan, S. Venkatesan, and R. Prakash, "A feedback-based scheme for improving tcp performance in ad hoc wireless networks," 2001.

[17] G. Holland and N. Vaidya, "Analysis of tcp performance over mobile ad hoc networks," in *MobiCom '99: Proceedings of the 5th annual ACM/IEEE international conference on Mobile computing and networking*. New York, NY, US: ACM, 1999.

[18] D. Kim, C.-K. Toh, and Y. Choi, "Tcp-bus: improving tcp performance in wireless ad hoc networks," vol. 3, 2000, pp. 1707 –1713 vol.3.

[19] J. Liu and S. Singh, "Atcp: Tcp for mobile ad hoc networks," *Selected Areas in Communications, IEEE Journal on*, vol. 19, no. 7, pp. 1300 –1315, jul. 2001.

[20] A. Falk and D. Katabi, *Specification for the Explicit Control Protocol (XCP), draft-falk-xcp-spec-03.txt*, Information Sciences Institute Internet-Draft. [Online]. Available: http://www.isi.edu/isixcp/docs/draft-falk-xcp-spec-00.html

[21] "100x100 clean state project." [Online]. Available: http://100x100network.org/

[22] B. Res, L. Barreto, and S. Sargento, "rt-winf: Real time wireless inference mechanism," in *IEEE Globecom 2010 Workshop on Mobile Computing and Emerging Communication Networks (MCECN 2010)*, Miami, Florida, USA.

[23] M. Conti, G. Maselli, G. Turi, and S. Giordano, "Cross-layering in mobile ad hoc network design," *Computer*, vol. 37, pp. 48–51, 2004.

[24] "CMU wireless emulator." [Online]. Available: http://www.cs.cmu.edu/emulator/

[25] "The network simulator - ns-2," 2001. [Online]. Available: http://www.isi.edu/nsnam/ns/

[26] Željko Ilić et al., "Optimal MAC packet size in wireless LAN," *MIPRO*, 2005.

[27] C. E. Perkins and P. Bhagwat, "Highly dynamic destination-sequenced distance-vector routing (DSDV) for mobile computers," *ACM SIGCOMM*, 1994.

[28] M. Fiore, "trace2stats." [Online]. Available: http://www.tlc-networks.polito.it/fiore/index.html