Advanced Wireless Congestion Control Techniques: XCP-Winf and RCP-Winf

LUÍS BARRETO
Instituto Politécnico de Viana do Castelo
Escola Superior de Ciências Empresariais
Avenida Pinto da Mota, Valença
PORTUGAL
lbarreto@esce.ipvc.pt

Abstract: TCP, the most used congestion control protocol, was developed having in consideration wired networks characteristics. The proliferation of wireless mesh networks has put in evidence some of TCP drawbacks. In such networks, TCP experiences serious performance degradation problems, due to its congestion control mechanisms. In a wireless network packet loss is not, as in a wired network, strongly correlated to congestion, but also with medium related errors. A congestion control scheme for wireless networks should be based in accurate estimation of network characteristics, namely link capacity and available bandwidth, based on end to end measurements. We describe new explicit flow control protocols for wireless mesh networks, based in XCP and RCP. We name these protocols XCP-Winf and RCP-Winf. They are supported in a new method to estimate the available bandwidth and the path capacity over a wireless network path, denoted as rt-Winf. 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. Simulations with ns-2 show that XCP-Winf and RCP-Winf outperform TCP efficiency in wireless mesh networks.

Key–Words: congestion control, available bandwidth, path capacity, measurements, performance, wireless networks.

1 Introduction

Wireless networks are becoming very popular and are being installed almost everywhere. Reliable transport protocols such as TCP were developed to perform well in traditional wired networks where packet losses occur mostly because of congestion [1]. However, networks with wireless and other lossy links also suffer from significant losses due to bit errors and handoffs. TCP responds to all losses by invoking congestion control and avoidance algorithms. This results in degraded end-to-end performance in wireless systems and networks [2]. When a signal strength weakness or noise is inferred in a wireless network, burst errors can occur. TCP will infer it as a timeout and will reduce significantly network performance, as link level retransmissions will occur. Also, TCP implements a linear, Additive Increase Multiplicative Decrease (AIMD), control system. This control process also does not match the network congestion dynamics in wireless networks, where wireless channel congestions are pervasive. In wireless multi-hop networks, wireless channel congestions come and go within a time scale which is much smaller than TCP's congestion control delay. Consequently, TCP fails to resolve the congestions in wireless networks thus making congestion control in wireless networks an important issue.

As stated in [3] "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".

TCP, that relies on the OSI-based architecture has served well in the context of wired networks, where is applied the well know adage "everything runs over IP, and IP runs over everything", however the gains presented by this architecture diminish significantly when the same network stack is run over wireless networks. The performance degradation of TCP over a dynamic wireless networks, is a consequence of its minimal control information exchange between layers of the network stack. In wired networks the binary feedback, in terms of packet acknowledgement control information, works well since network topology in wired networks are inherently stable, and the underlying link reliability is exceptionally high; mak-

ing congestion the primary cause for loss of packets in wired networks. For wired TCP interprets the feedback of packet loss correctly as a signal of congestion, and forces sources to cut back their rates performing, in these types of networks, congestion control functionality efficiently. In wireless networks, however, the frequent change in network structure (due to mobility or link reliability), and the higher link error rates (as compared to wired networks), makes the binary packet acknowledgement control information exchange between layers inadequate. In a wireless network, apart from congestion, a packet could have been lost due to link unreliability, or due to collisions that might be caused by interference. Thus, due to lack of important information, such as link capacity and available bandwidth, TCP forces sources over a wireless network to have a an underachieved behavior, in terms of the flow rates that the sources can achieve.

New congestion control mechanisms that rely on network interaction have been proposed. The eXplicit Control Protocol (XCP) [4] and the Rate Control Protocol (RCP) [5] are two of those congestion control mechanisms. Both XCP and RCP are examples of explicit congestion control techniques. Their main idea is to generalize explicit congestion notification, where nodes inform each other about the degree of congestion. XCP, also as TCP a window based congestion control protocol, decouples channel utilization from fairness control. Since channel utilization is decoupled from fairness control it is important to estimate efficiently the aggregate traffic behavior, i.e both available bandwidth and link capacity. RCP is a congestion control algorithm whose main key is to finish flows as quickly as possible. RCP updates dynamically the rate assigned to the flows, approximating processor sharing in the presence of feedback.

As both XCP and RCP use explicit congestion control, and network interaction, their behavior should be more efficient, than traditional congestion control mechanisms, in shared medium systems, such as wireless mesh and ad hoc networks. However, as shown in [6] their behavior in such environments is not very good and effective. As, for their operation, they rely in important underlying network information, such as available bandwidth and link capacity, [6] shows that they are incapable to predict effectively capacity in wireless networks, and, also, that they are not very efficient and fair.

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 wireless networks, and to improve XCP and RCP behavior in such environments. It is, then, of major importance, in shared medium congestion control, to obtain accurately link capacity and available bandwidth and, then, use these parameters actively in congestion control. Factors such as handoffs, channel allocation and, of course, channel quality are directly related to link capacity. Being able to accurately monitor link capacity and available bandwidth and, then, use that information to congestion control and congestion monitoring is a main area of interest in the development of wireless congestion control.

Estimation of link capacity has been widely studied, and can be achieved through either active or passive measurement [7]. Tools that use active measurement work by injecting measurement probe packets into the network. While, accordingly to [7] "Passive measurement tools use the trace history of existing data transmission". Using active measurement tools can add excess overhead and end-to-end semantics may not always be maintained - these are some important drawbacks of such tools. Passive measurement tools can be less reliable as they only rely in sparse data. A new available bandwidth and link capacity evaluation mechanism was proposed in [8]. This mechanism measures accurately and passively link capacity and available bandwidth in wireless networks.

In [8] it is proposed the rt-Winf algorithm. rt-Winf is a new wireless inference mechanism, based on IdleGap [9], that is able to estimate available bandwidth and path capacity. rt-Winf uses the information included in RTS/CTS packets to measure the transmission time and obtain the link capacity and available bandwidth.

Knowing that being able to feed correct traffic information to both XCP and RCP would improve their behavior, it is proposed, in this paper, the integration of the rt-Winf [8] algorithm in their congestion control techniques, as both rely on available bandwidth and capacity estimation for their operations. rt-Winf obtained link capacity and available bandwidth is passed, through cross-layer techniques - implemented with ns-miracle [10] -, to XCP and RCP that use that information in their native congestion control techniques. This allows to considerably improve XCP and RCP performance in wireless networks. These new congestion control techniques are called XCP-Winf and RCP-Winf.

The rest of this paper is organized as follows. Next section, section 2, briefly presents the background and related work. Then, section 3 describes the rt-Winf algorithm and rt-Winf obtained results. In section 4 it is presented how rt-Winf was integrated with XCP and RCP. Section 5 describes and discusses the results obtained through simulation. Finally, section 6 presents the conclusions and future work.

2 Related Work

2.1 Capacity and Available Bandwidth Estimation

Link capacity has been widely studied in wired networks. Some examples of link estimation tools on wired environments are: AbGet [11], PathChirp [12], IPerf [13], Pathload [14], IGI/PTR [15], Pathchar [16], CapProbe [17]. AdHoc Probe [18], WBest [19] and IdleGap [9] are some developments of link estimation tools for wireless networks. It is important to mention that while WBest calculates both capacity and available bandwidth, AdHoc Probe provides only the path capacity of the wireless channel.

IdleGap is a recent mechanism for obtaining available bandwidth in wireless networks. IdleGap takes into consideration the CSMA Collision Avoidance (CSMA-CA) scheme of wireless networks. The idle nodes, which are waiting to transmit, use the Network Allocation Vector (NAV) [20]. The NAV shows how long other nodes allocate the link in the IEEE 802.11 MAC protocol. NAV value is used, in conjunction with Request To Send (RTS) / Clear to Send (CTS) handshake, to address the hidden node problem [21]. The hidden node problem arises when we are in presence of wireless nodes that are out of range of other wireless nodes, but can interfere in the communication process of other wireless nodes. For solving such problem a control handshake is done - RTS/CTS handshake. A node wishing to initiate a communication sends a RTS packet. The destination node replies with a CTS packet. Any other node receiving the RTS or CTS packets should contend and backoff from sending data for a random interval of time (solving the hidden node problem) and, also, update their NAV value. Even though a node is located at a place where it cannot reach other active nodes, the node can know whether another node is already using the wireless network by checking its NAV. The idle time in the wireless network can then be estimated from the NAV information.

IdleGap uses a very accurate approach to characterize the busy time and the total elapsed time, obtaining a very accurate *Idle Rate*. One of the main issues of IdleGap is that it uses the pre-defined IEEE802.11 header *DataRate* [22] value, which, as stated in [8], is not practical and real, thus leading to not very accurate and over-dimensioned estimation values. Thus, it is not realistic in the determination of link capacity. Another important issue of Idlegap is the introduction of a new sublayer in the Open System Interconnection (OSI) Model [23] stack, the *Idle Module*. Although rtWinf algorithm was based in this mechanism, which will be better described in section 3, does not suffer these constraints. IdleGap proposes the consideration of 3 different states for a wireless node: *Sender*, *Receiver* and *Onlooker*. These states are distinguished on the *Idle Module*, the module is used to determine the *Idle Rate*.

TSProbe [24] is a new capacity estimation tool based in AdHoc Probe, but focused in time-slotted connections such as bluetooth or WIMAX. TSProbe uses, for its operation, the interaction between several link layer properties to deploy an adaptive probing scheme. TSProbe employs an iterative probing scheme that utilizes payloads that vary in size. While accurate in time-slotted connections, it lacks of efficiency in dynamic wireless environments.

Other techniques, such as TCP with fast recovery defined in [25], try to improve TCP performance including bandwidth estimation. In this mechanism he TCP source tries to estimate the available bandwidth using an exponential averaging. All TCP operation principles are maintained except that, when a timeout or 3 duplicate ACKs occurs, the available bandwidth estimate is used to reset the TCP congestion window and the slow start threshold.

All the presented tools were defined with the main purpose to only estimate available bandwidth and link capacity in specific network conditions. The presented tools lack of some overall capabilities. Some can just estimate available bandwidth and others only link capacity. In a wireless network environment it is important to have a tool that can retrieve an accurate busy time and the total elapsed time between communications. It is also important to have a mechanism that uses, not only source information, but also receiver information, as this is only the way to have a precise network status. So, it is important to introduce the concept of network cooperation in network estimation tools. Another important parameter, to obtain accurate values, is the effective calculation of the actual data rate that is used by each communication process.

2.2 Congestion Control

Congestion control over network paths has been an active area of research, for all kinds of mediums and traffic [26]. There exists a variety of network applications such as video streaming and conferencing, voice over IP (VoIP), and video on demand (VoD). The number of users for these network applications is continuously growing hence resulting in congestion. The Transmission Control Protocol (TCP) [27] is the most used congestion control protocol in computer networks. TCP uses the *Additive Increase Multiplicative Decrease* (AIMD) algorithm and the *slow-start* mechanism [28]. 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 [29]. It is important to notice, that this is not a true statement in multihop wireless networks.

TCP was designed to provide reliable end-to-end delivery of data over unreliable networks. In practice, TCP was deployed in the context of wired networks. Ignoring the properties of wireless links leads to TCP poor performance. In a wireless network, however, packet losses occur more often due to unreliable wireless links than due to congestion. When using TCP over wireless links, each packet loss on the wireless link results in congestion control measures being invoked at the source. As wireless channels have a broadcast nature, neighboring nodes in a wireless network can not transmit simultaneously. As TCP generates bursty traffic based on the current congestion window size, the packets of the multihop flow contend with each other for the channel at successive hops. This leads to self contention, thus increasing chances for dropping the packets, causing severe performance degradation.

As an adaptive transport protocol, TCP controls its offered load (through adjusting its window size) according to the available network bandwidth. It additively increases its congestion window in the absence of congestion and drastically reduces its window when a sign of congestion is detected. In the wired world, congestion is identified by packet loss, which results from buffer overflow events at the bottleneck router [30]. So, if available bandwidth is not correctly inferred TCP will not use the medium efficiently and the communication will suffer of poor performance. Also, as referred in [7] the slow-start algorithm of TCP requires the connection to be conservative and assumes that available bandwidth to the receiver is small. In wireless networks these unstable behavior compromises the entire network dynamics. In such kind of etworks it should be implemented a feedback mechanism that allows traffic sources to control their transmission rate.

Some congestion control mechanisms try to enhance TCP behavior in a wireless environment. Mechanisms like TCP-F [31], TCP-ELFN [32], TCP-BuS [33], ATCP [34] represent some examples of protocols for wireless networks in general. They concentrate on improving TCP's throughput by freezing TCP's congestion control algorithm during link-failure induced losses, especially when route changes occur. These TCP developments differ in the manner in which losses are identified and notified to the sender and in their details of freezing TCP's congestion control algorithm. TCP-ELFN explicitly notifies the TCP sender of routing failures, causing the sender to enter a standby mode. The sender re-enters the normal TCP mode on route restoration, identified using periodic probe messages. In ATP [35], a flow receives the maximum of the weighted average of the sum of the queuing and transmission delay at any node traversed by the flow. ATP uses the inverse of this delay as the sending rate of a sender. Even though these schemes do not recognize the need of congestion detection and signaling over a neighborhood, their congestion metric implicitly takes some degree of neighborhood congestion into account.

New mechanisms like imTCP (Inline measurement TCP) [36] and TCP-AP (TCP with Adaptive Pacing) [37] have been proposed. ImTCP introduces a new bandwidth measurement algorithm that can perform inline measurements. The algorithm is applied to a TCP sender. The ImTCP sender adjusts the transmission intervals of data packets accordingly to the estimation results of available bandwidth. The available bandwidth estimation results from the arrival intervals of ACK packets. In imTCp capacity estimation is not considered. TCP-AP was developed taking only into consideration multihop wireless environments. A TCP-AP sender adapts its transmission rate using an estimate of the 4-hop propagation delay and the coefficient of variation of recently measured round-trip times. Its main issue is just using, in whatever type of scenario, an estimate of the current 4-hop propagation delay. Recent research has recognized the importance of explicitly detect and signal congestion over a network. One example is the Explicit Wireless Congestion Control Protocol (EWCCP) [38]. This mechanism identifies the set of flows that share the channel capacity with flows passing through a congested node. EWCCP assumes that the achievable rate region of 802.11 is convex, thus being proportionally fair. It must be referred that EWCCP has not been yet tested in a real implementation.

More congestion control techniques for wireless networks have been proposed. TPA (Transport Protocol for Ad Hoc Networks) [39], COPAS [40] and LRED [41] try to address congestion control issues in ad hoc networks. WCP [42] and WCPCap [42] are congestion control mechanisms developed for wireless mesh networks. TPA congestion control mechanism is inspired by TCP, but optimized to minimize the number of required packet retransmissions. Packets are transmitted in blocks using a window-based scheme. A block with a fixed number of packets is transmitted reliably before any packet of the next block is transmitted. Packet retransmissions are only performed when every packet of a block has been

transmitted once. A block is transmitted in several rounds: first every packet is transmitted once, then not yet acknowledged packets of this block are retransmitted until every packet of the block has been delivered and acknowledged. COPAS proposes a route selection scheme that attempts to find disjoint paths for different flows by assigning weights to links proportional to the average number of backoffs on the link. LRED uses a exponential weighted moving average of the number of retransmissions at the MAC layer as a measure of congestion while marking packets.

WCP is a rate-based congestion control protocol for static multihop wireless mesh networks which use the 802.11 MAC. In WCP, for every flow, the source maintains a rate R which represents the long term sending rate for the flow. WCP is AIMD-based, so that the source additively increases R on every ACK reception and multiplicatively decreases R upon receiving a congestion notification from intermediate forwarding nodes. WCPCap estimates the available capacity within each neighborhood, and distributes this capacity to contending flows, using a distributed rate controller. WCPCap uses local information and can be implemented in a distributed manner.

An alternative to AIMD based schemes are schemes in which intermediate routers send explicit and precise feedback to the sources. Examples of such congestion control schemes, in wired networks, are the The eXplicit Control Protocol (XCP) [4] and the Rate Control Protocol (RCP) [5].

XCP was designed to extract congestion information directly from intermediate nodes (routers and/or switches). According to [43], "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 has its disadvantages: it is more difficult to deploy, since changes need to be made in all routers and end-systems in the network. A XCP network is composed of 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 best network conditions, that is, the maximum throughput. This feedback is accomplished by the use of a congestion header in each packet sent. Along the path, intermediate nodes update the congestion header. When the packet reaches the receiver, it copies the network information, obtained from the last intermediate router, into outbound packets of the same flow (normally acknowledgment packets).

The Rate Control Protocol (RCP) is part of 100x100 clean state project [44]. The mission of this project is to create blueprints for a network that goes beyond today's Internet [44]. 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. RCP intends to improve web users flows, distributed computing and distributed file-systems, making flows to finish close to the minimum possible. RCP uses the same feedback principle of XCP and tries to emulate processor sharing. However, it uses a different approach. Routers along the path don't determine incremental changes to the end-system's throughput, but determine the available capacity and the rate at which the end-system should operate.

More recently, and having into consideration RCP main properties, it has been proposed, for wireless sensor networks (WSN), the Wireless Rate Control Protocol WRCP [45] mechanism. WRCP uses explicit feedback based on capacity information to achieve a max-min fair rate allocation over a collection tree. In WRCP a receiver capacity model is applied. This model associates capacities with nodes instead of links. The receiver model is, also, used to develop and implement the explicit and distributed rate-based congestion control protocol for wireless sensor networks. WRCP tries to achieve, in WSN, max-min fairness.

As shown in [6] and [46], XCP and RCP don't behave as well as TCP in a WMNs. This is due to the fact that the available capacity at a wireless link depends on the link rates at the neighboring edges. Ignoring this dependence will overestimate the available capacity and lead to poor performance and to instability. The possibility of directly estimating the exact capacity of a link as a function of the link rates at the neighboring edges allows that an accurate XCP-like scheme can be implemented for wireless multi-hop networks. Thus, using rt-Winf information effectively, as XCP and RCP native operations use as parameters the link capacity and the available bandwidth, can significantly improve their behavior in a wireless network. WXCP [47] and XCP-b [48] are variants of XCP that measure indirect parameters such as queue sizes and number of link layer retransmissions, using for that very complex heuristics. The direct estimation of the link capacity will allow a more accurate XCP-like scheme to be implemented for wireless multi-hop networks.

2.3 Collision Probability

A wireless network is performance dependent on its medium access control scheme [49]. In CSMA-CA, a node is allowed to transmit only if it determines the medium to be idle. CSMA-CA, however, cannot prevent packet collisions caused by nodes that are

located within the transmission range of the receiver, but not of the sender (hidden nodes problem [21]). To prevent DATA packet collisions due to hidden nodes, IEEE 802.11 supports the RTS/CTS mechanism [22]. However, it must be noticed that in ad hoc networks, this assumption does not hold in general. Neighboring nodes are often unable to receive the control packets because they are masked by on-going transmissions from other nodes near them. This means that the RTS/CTS mechanism does not generally prevent DATA packet collisions, even under perfect operating conditions, such as negligible propagation delay, no channel fading and no node mobility. [50] states that if nodes are mobile, then a node that did not hear an RTS or CTS may migrate into the footprint of a receiver and destroy a DATA packet with its own transmission. The probability of such a scenario increases with the mobility of the nodes. [51] also shows that in an ad hoc network, a successful exchange of RTS and CTS is not sufficient to prevent DATA packet collisions.

Collisions in wireless networks have been a research topic ([30],[52], [53]). As mentioned before, the MAC IEEE 802.11 protocol cannot prevent hidden node collisions from happening. Some previous works - [54], [51], [55] - tried to infer collision probabilities through extensive mathematical formulas. It must be noticed, here, that collisions in the IEEE 802.11 networks occur before congestion. Since packets are lost when the queue size in some nodes exceeds a limited value. This threshold is far away from the congestion limit, making the system unstable [56].

3 rt-Winf

IdleGap [9] was the underlying basis for the development of rt-Winf. The main purpose of rt-Winf was to mitigate IdleGap main issues, being compatible with all systems and evaluating both the link capacity and the available bandwidth without overloading the network. rt-Winf does not affects the OSI Model and obtains all the necessary times to calculate the path capacity and available bandwidth. One of the main issues of IdleGap is that it uses the *DataRate* value of the IEEE802.11 header [22], rt-Winf effectively calculates the capacity, instead of using the value present on the IEEE802.11 header. The operational principles of rt-Winf allows it to rely in the Request To Send (RTS) / Clear To Send (CTS) handshake or in probe packets.

3.1 RTS/CTS Packets

rt-Winf with RTS/CTS control packets enabled relies, as IdleGap, on this handshake to correctly retrieve the NAV values. As IdleGap uses the *DataRate* value of the header it was necessary to evaluate its accuracy. It was, then, performed a large number of captures (~ 200) in a real wireless environment. With the data gathered, it was possible to conclude 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 to trigger the calculations.

The obtained captures also allowed to realize how each node state (as defined by IdleGap) managed the received packets. CTS, DATA and ACK packets are captured in the case of the *Sender* state. In the *Receiver* state, a node 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 occurred 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 the ACK packet with that information, from the *Receiver*, 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.

3.2 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 [22]. 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 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 packets. To be fully operational, both Sender and Receiver must be running the rt-Winf mechanism.

$$C = \frac{PacketSize}{TransferTime} \quad (1)$$

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

$$AB = 1 - \left(\frac{\sum TransferTime}{TotalElapsedTime}\right) \times C \quad (2)$$

In a normal VoIP call using G.711 codec [57], the overhead introduced by this mechanism is $\sim 1.66\%$.

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

Figure 1: rt-Winf Algorithm.

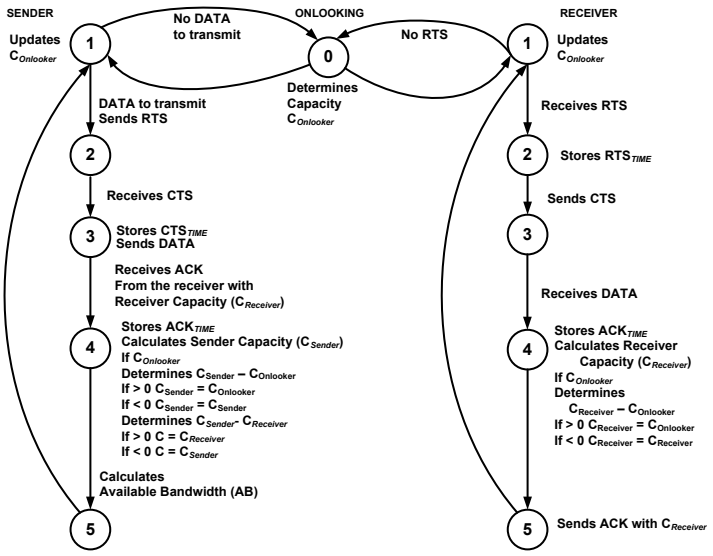


Figure 2: rt-Winf Sender, Receiver and Onlooking State Diagrams.

For a flow with more than 1Mbps, the overhead is less than $\sim 0.15\%$.

3.3 rt-Winf Results

We have implemented rt-Winf in the CMU Wireless Emulator [58] and the ns-2 simulator [59]. The three states defined by rt-Winf mechanism and the cooperation between them and between the nodes was developed in C language. 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. Path capacity and available bandwidth were evaluated in different scenarios.

For path capacity evaluation, rt-Winf results were

compared with AdHoc Probe tool results and maximum throughput (that represents the maximal theoretical throughput)in a simple 2 ad hoc nodes testbed. AdHoc Probe tool measures efficiently the path capacity in a wireless communication [18] when compared to other tools. It was used a simple 2 ad hoc nodes scenario. An UDP flow with Constant Bit Rate (CBR) of 64 Kbps was injected between the two nodes. Accordingly to [60], the maximal theoretical throughput is obtained through

$$TH_{80211b} = \frac{MSDU}{Delay\_per\_MSDU} \quad (3)$$

where MSDU is the MAC Service Data Unit.

The maximum throughput represents, in ideal conditions, the maximum achievable capacity. Figure 3 shows the path capacity results. As we are using a low CBR flow, the expected capacity should be less than the maximum throughput. The simulations validated that assumption, showing that rt-Winf results are close to the maximum throughput values. It is possible, then, to observe that rt-Winf uses efficiently the information present in the channel, in order to obtain the resulting capacity. This is because rt-Winf measures more accurately the channel occupation time, as it takes into consideration all traffic flows. Comparing with the AdHoc Probe method, and with a similar probing time, rt-Winf gathers more information to perform the desired calculations, thus being able to be statistically more precise and less sensitive to flow variations. AdHoc Probe only takes into consideration its probing packets, which with the network dynamics can suffer dispersion and collisions, introducing a negative impact in the capacity evaluation.

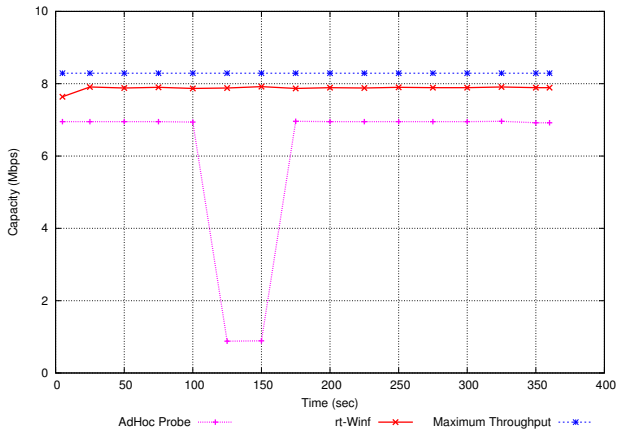


Figure 3: AdHoc Probe and rt-Winf Path Capacity Estimation.

Path capacity and available bandwidth evaluations were also conducted on a wireless mesh scenario (Figure 4). The two mobile nodes, *Mobile Node 1* and *Mobile Node 2*, communicate with each other through two mesh nodes responsible, that are responsible for the routing and link management. The mobile nodes are in such a distance that the traffic is always routed by the mesh nodes.

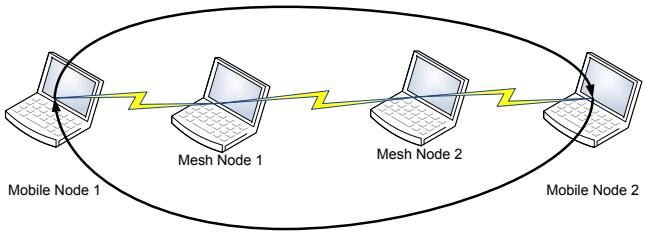


Figure 4: Wireless Mesh Scenario.

Path capacity results are shown in Figure 5, and available bandwidth results are shown in Figure 6. Figure 6 provides the results gathered with rt-Winf, IPerf UDP [13] and IdleGap. Maximum throughput values are also presented, being considered as an upper bound of the result, as described before. IPerf UDP results are considered the lower bound.

As observed in Figure 5, 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.

The results presented in Figure 6 allow 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, as it is possible to observe how the results vary through time. Those results are within an upper bound, the maximum theoretical throughput, and a lower bound, IPerf UDP.

To observe the impact of rt-Winf with probe packets in a wireless mesh scenario and to allow a valuable comparison between the emulator and simulator results, some simulations in the ns-2 simulator [59] were also conducted . As rt-Winf is based in IdleGap, the simulations also allow a baseline comparison of those tools. In the simulations it was used a FTP transfer from a source to a sink, with different simultaneous flows. The maximum throughput is calculated using ns-2 default values and using Equation 3. Figure 7 summarizes the obtained results. Each value is an average of 20 runs lasting 300 seconds of simulated time and nodes are stationary. As observed, IdleGap results are almost equal to the the maximal

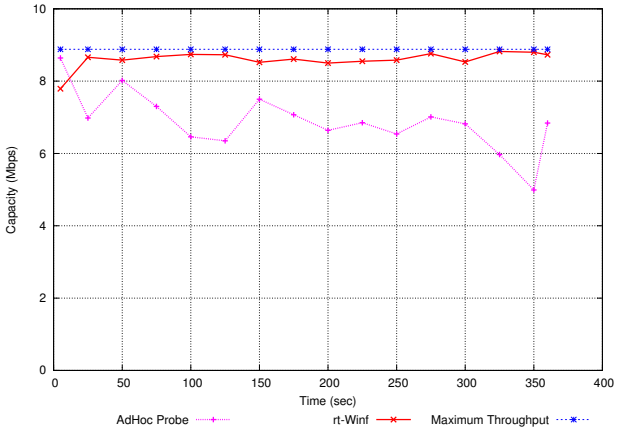


Figure 5: Wireless Mesh Scenario Path Capacity.

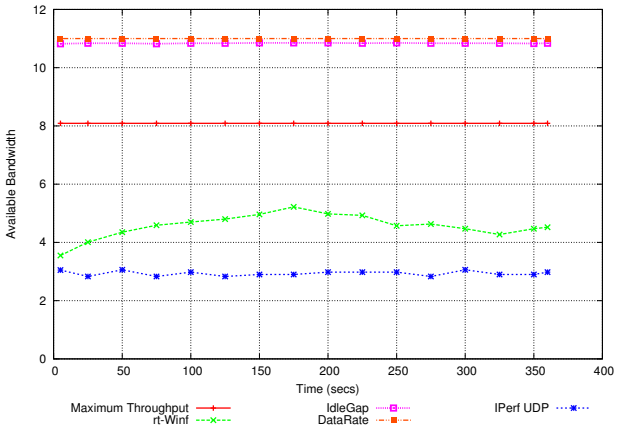


Figure 6: Wireless Mesh Scenario Available Bandwidth.

theoretical throughput, as it is using the IEEE802.11 Header *DataRate* value in the calculations. These results validate the ones obtained with the CMU Emulator, since the results for 1 flow in Figure 7 are similar to the ones of Figure 5. For the rt-Winf probe packets simulations, it was used packets with different sizes. With these simulations it is also possible to conclude that rt-Winf with probe packets is also efficiently measuring the capacity, and its values are very similar to the rt-Winf mechanism working with RTS/CTS control packets.

4 XCP-Winf and RCP-Winf

For improving the performance of congestion control techniques in dynamic wireless networks we define a new congestion control scheme, based in XCP and RCP. The solution proposed adopts the inherent explicit congestion control mechanisms of XCP and RCP updated with the interaction of a link and available bandwidth estimation mechanism. Both XCP and RCP take the link capacity at the interface to compute the rate feedback. That introduces capacity overestimation which will generate inflated feedback, the senders will send more than the link can transfer. The estimation mechanism used is rt-Winf. The senders, accordingly to the feedback and the estimation tool information, update their transmit rate. The estimation mechanism is integrated both at senders, receivers and onlookers nodes. These protocols are called XCP-Winf and RCP-Winf.

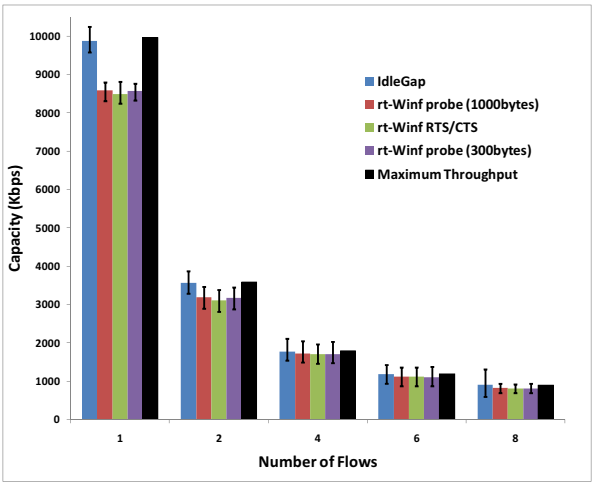


Figure 7: Ns-2 Capacity Results.

rt-Winf available bandwidth and link capacity measurements are used by XCP-Winf and RCP-Winf. As rt-Winf values are obtained in the MAC layer, this information has to be accessed by XCP and RCP. All operating principles of XCP and RCP are unchanged, the main difference is that the information on available bandwidth and capacity are obtained in the MAC layer. The rt-Winf information is sent to the network layer through a simple, but effective, cross layer communication process. For this communication 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 [61]. A generic XCP-Winf/RCP-Winf system relies on the main functioning principles of XCP and RCP and is represented in Figure 8.

rt- Winf inserts the available bandwidth and the link capacity information in the shared database and, then, XCP and RCP access/get that information and update their functions with the accessed information. Next we present, specifically for a XCP-Winf system, some of the operations conducted. A XCP sender that requests a desired throughput needs to obtain the $Delta\_Throughput$, i.e. the amount of throughput

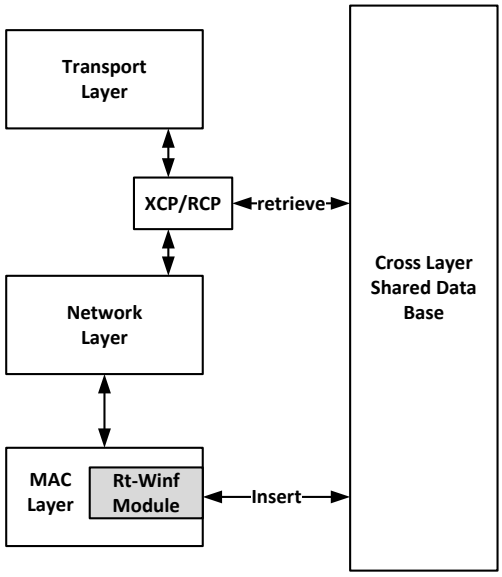


Figure 8: XCP-Winf/RCP-Winf System.

Algorithm 1: XCP-Winf Sender Algorithm.

```
/* Available Bandwidth and Capacity
   Estimation                                   */
```
$Desired\_Throughput$: senders desired change in throughput.
$Available\_Bandwidth_{Winf}$: rt-Winf obtained available bandwidth.
$Delta\_Throughput$: desired or allocated change, per packet, in throughput.

Access Cross Layer Shared Database;
Retrieve Available Bandwidth and Capacity;

```
/* Obtain Delta Throughput                      */
```
Throughput = C_{Winf};
$Desired\_Throughput <= Available\_Bandwidth_{Winf}$;
$Delta\_Throughput = \frac{Desired\_Throughput - C_{Winf} \times 1000}{C_{Winf} \times \frac{RTT}{MSS}}$;

```
/* Send a packet                                */
```
Update Congestion Header Delta Throughput Field;
Send Packet;

the sender wishes to change. In a XCP-Winf system the $Delta\_throughput$ is obtained by

$$Delta\_Throughput = \frac{Desired\_Throuhgput - C_{Winf} \times 1000}{C_{Winf} \times \frac{RTT}{MSS}} \quad (4)$$

where RTT is the current Round Trip Time and MSS is the Maximum Segment Size. C_{Winf} is the link capacity obtained from rt-Winf. The $Desired\_Throughput$ is a value that might be supplied by an application, or it might be the speed of the local interface. A XCP router can also obtain the aggregate feedback [4] based in rt-Winf information:

$$F_{Winf} = \alpha \times (C_{Winf} - Available\_Bandwidth_{Winf}) - \beta \times \frac{q}{d} \quad (5)$$

where α and β are constant parameters, and $\frac{q}{d}$ represents the persistent queue.
$Available\_Bandwidth_{Winf}$ is also obtained by the rt-Winf mechanism.

4.1 XCP-Winf Functions

This section briefly describes the XCP-Winf functions. The only functions, relatively to XCP, that are changed are the XCP Sender and XCP Router functions. The XCP Receiver is not changed as it operations remain the same. The XCP Sender uses the *Sender* state of the rt-Winf algorithm and the XCP Router uses the *Onlooker* state. Next, we present the corresponding algorithms for the XCP-Winf Sender and Router functions. The XCP-Winf Receiver is just the responsible for copying the $Delta\_Throughput$ value that arrives in a packet to the $Reverse\_Feedback$ field of outgoing packets. A XCP-Winf *Receiver* operates in a similar way as a XCP *Receiver*. When acknowledging a packet, the XCP-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.

When operating as a XCP-Winf Sender several calculations need to be done. The pseudo-code of a XCP-Winf Sender is presented in Algorithm 1. The XCP-Winf operations are basically the same as standard XCP, except that it uses rt-Winf to obtain the link capacity and available bandwidth and, then, obtain the $Delta\_Throughput$. If no additional capacity is needed, the $Desired\_Troughput$ will be equal to zero, and the packet will be immediately sent. If the value of $Delta\_Throughput$ exceeds $Available\_Bandwidth_{Winf}$, it is reduced to the current value of $Available\_Bandwidth_{Winf}$.

The XCP-Winf Router/Node system operations when in the *Onlooker* state are divided in four moments. Those moments are: when a packet arrives, when a packet departs, the control interval timeout packet and the assessment of the persistent queue. The pseudo-code for each of those moments are presented in Algorithms 2, 4, 3 and 5. Once more, rt-Winf available bandwidth and capacity are used in the calculations. In Algorithm 2 it is possible to see what parameters are updated when a packet arrives on a node in the *Onlooker* state. The parameter

$sum\_inv\_throughput$ is used for capacity allocation, thus it uses the capacity value obtained by rt-Winf, allowing to have more precise values when compared to standard XCP. Another important parameter is the $sum\_rtt\_by\_throughput$, this parameter is used to obtain the control interval and on its calculation, as it depends from the throughput of the link, it is used the rt-Winf capacity. This algorithm also checks if the round trip time of each flow is exceeding the maximum allowable control interval. If that is true, and for avoiding delays when new flows are started, the maximum allowable control interval is used in all next calculations.

In Algorithm 3 are shown the operations that are done when the control timer expires. This algorithm uses rt-Winf values to determine the aggregated feedback (F_{Winf}) and $input\_bw$. The parameter $input\_bw$ represents the average bandwidth of the arriving traffic. The parameter $shuffled\_traffic = max(0, 0.1 \times input\_bw - |F_{Winf}|)$ defines the amount of capacity that will be shuffled in the next control interval. Shuffling takes a small amount of the available capacity and redistributes it by adding it to both the positive and negative feedback pools. This allows new flows to acquire capacity in a full loaded system [4]. Both a and b are constants. a has the value of 0.4 and b 0.226.

When a packet departs the node has to calculate a per-packet capacity change, that will be compared to the $Delta\_Throughput$ value in the packet header. as stated in [4] "using the AIMD rule, positive feedback is applied equally per-flow, while negative feedback is made proportional to each flow's capacity". In Algorithm 4, the node verifies whether the packet is requesting more capacity (via the packet's $Delta\_Throughput$ field) than the node has allocated. If so, this means that the sender's desired throughput needs to be reduced and, also, verified against rt-Winf available bandwidth. If the node has allocated more capacity than the available bandwidth, the desired throughput is updated to the rt-Winf available bandwidth. If the allocated capacity is less than the available bandwidth, the $Delta\_Throughput$ field in the packet header is updated with the feedback allocation.

XCP-Winf, as XCP, needs to calculate a queue that does not drain in a propagation delay, that is the persistent queue. The operations needed to obtain that queue are represented in Algorithm 5. This queue is intended to be the minimum standing queue over the the estimation interval. Each time a packets departs the current instantaneous queue length ($queue$) is checked and the minimum queue size is calculated. T_q represents the queue estimation timer. When this timer expires the persistent queue is equal to the minimum queue value over the last T_q interval. For obtaining the duration of the T_q interval it is used the capacity value of rt-Winf. Comparing XCP-Winf with XCP, it is possible to conclude that both use the same principles but differ in the way link capacity and available bandwidth are obtained and used.

4.2 RCP-Winf Functions

RCP-Winf updates RCP operations, using rt-Winf available bandwidth and capacity values. The RCP rate update equation is related to link capacity. In RCP-Winf the same equation is related to rt-Winf link capacity evaluation. A RCP-Winf Receiver operates the same way as a standard RCP Receiver. The RCP-Winf Receiver just updates $rcp\_reverse\_bottleneck\_rate$ in the ACK packet and sends it to the sender. When operating as a sender RCP-Winf needs to do some operations that allow it to modulate the congestion window. Algorithm 6 shows the pseudo-code of a RCP-Winf Sender. The RCP-Winf Sender will evaluate the $rcp\_bottleneck\_rate$ with the rt-Winf obtained link capacity, gathered through the get method of the cross-layer communication process. Accordingly to the evaluation it will update, or not, $rcp\_bottleneck\_rate$ value. Then, it modulates the congestion window and calculates the pacing interval, that will be used to send packets from the queue.

When the rate timer expires a RCP-Winf Router, i.e. a node on the *Onlooker* state, performs the operations presented in Algorithm 7. The node first gets the rt-Winf available bandwidth and capacity values. Then, assumes that the aggregate incoming traffic is defined by rt-Winf available bandwidth. Next, obtains the average round-trip time of the traffic that has arrived in the estimation interval. After that, the node updates the RTT estimate (avg_rtt). The node, then, updates, using rt-Winf capacity, the rate that will be offered to the flows. The node tests the rate value and updates it. The rate value cannot be under the MIN_RATE or above a weighted (ETA) link capacity value. After that, the node decides the length of the next rate estimation interval. Before finishing, the node resets the variables and restarts the timer. ETA controls the target link-utilization and can be

Algorithm 2: XCP-Winf Router/Onlooker Packet Arrival Operations.

Pkt_size: Packet size form the IP header.
$input\_traffic$: volume of data that arrives during a control interval.
$MAX\_INTERVAL$: maximum allowable control interval.
C_{Winf} : rt-Winf obtained Capacity.

foreach *Packet Arrival* **do**
  $input\_traffic \mathrel{+}= Pkt\_size$;
  $sum\_inv\_throughput \mathrel{+}= \frac{Pkt\_size}{C_{Winf}}$;
  if *(Rtt < MAX_INTERVAL)* **then**
    $sum\_rtt\_by\_throughput \mathrel{+}= \frac{Rtt \times Pkt\_size}{C_{Winf}}$;
  else
    $sum\_rtt\_by\_throughput \mathrel{+}= \frac{MAX\_INTERVAL \times Pkt\_size}{C_{Winf}}$;

Algorithm 3: XCP-Winf Router/Onlooker Control Interval Timeout Operations.

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

any value in the range $0.95 < ETA < 1$. It is important to choose a value inferior to 1 as it allows some comfort to drain excess traffic before building up a queue. $ALPHA$ and $BETA$ are respectively the stability and performance constant factors. It is recommended to use $ALPHA$ values between 0.4 and 0.6, $BETA$ values between 0.2 and 0.6. When in extreme congestion scenarios the minimum rate value allowed is

$$\frac{(00.1 \times MTU)}{averageRTT} \tag{6}$$

, this will be the considered value for MIN_RATE. RCP-Winf keeps unchanged the standard operations performed by RCP routers, when a packet arrives and departures. It mus be noticed that in its operation RCP-Winf only relies in link capacity, while XCP-Winf relies in both link and available bandwidth.

4.3 Collision Probability

MAC IEEE 802.11 uses the Distribution Coordination Function (DCF) access method. The DCF access method is based on the CSMA-CA principle in which a host wishing to transmit senses the channel, waits for a period of time and then transmits if the medium is still free. If the packet is correctly received, the receiving host sends an ACK frame after another fixed period of time. If the ACK frame is not received by the sending host, a collision is assumed to have occurred. The sending host attempts to send the packet again when the channel is free for the period augmented with a random interval of time. If there are multiple hosts attempting to transmit, the channel may be sensed busy and in this case hosts enter the collision avoidance phase. It is possible to conclude that each access to the medium is independent from the previous one and when a collision occurs, a transmitting host waits a random number of slots distributed geometrically. The transmitting node is responsible for all collision dynamics, thus, all information regarding collision probability can be driven from the a transmitting node.

For improving efficiency and reliability of XCP-Winf and RCP-Winf available bandwidth and link capacity inference mechanism, it is of extreme importance to account the collision probability. As rt-Winf makes a real time analysis of the network status, it is important to know the extra time introduced when a node is waiting to transmit as a result of collisions. When a node wants to transmit and enters contention mode, due to collision detection, this will affect link evaluation, leading to over estimated capacity and available bandwidth values. For obtaining more real

Algorithm 4: XCP-Winf Router/Onlooker Packet Departure Operations.

pos_fbk: positive allocation.
neg_fbk: negative allocation.

foreach *Packet Departure* **do**
  $pos\_fbk = C_p \times Pkt\_size/C_{Winf}$;
  $neg\_fbk = C_n \times Pkt\_size$;
  $feedback = pos\_fbk - neg\_fbk$;
  if $(Delta\_Throughput > feedback)$ **then**
    if $(feedback > Available\_Bandwidth_{Winf})$ **then**
      $Delta\_Throughput = Available\_Bandwidth_{Winf}$;
    else
      $Delta\_Throughput = feedback$;
    $neg\_fbk = min(residue\_neg\_fbk, neg\_fbk + (feedback - Delta\_Throughput))$;
    $pos\_fbk = Delta\_Throughput + neg\_fbk$;
  $neg\_fbk = min(residue\_neg\_fbk, neg\_fbk + (feedback - Delta\_Throughput))$;
  $pos\_fbk = Delta\_Throughput + neg\_fbk$;
  $residue\_pos\_fbk = max(0, residue\_pos\_fbk - pos\_fbk)$;
  $residue\_neg\_fbk = max(0, residue\_neg\_fbk - neg\_fbk)$;
  if $(residue\_pos\_fbk <= 0)$ **then**
    $C_p = 0$;
  if $(residue\_neg\_fbk <= 0)$ **then**
    $C_n = 0$;

Algorithm 5: XCP-Winf Router/Onlooker Queue Determination Operations.

$queue$: the persistent queue
T_q: queue estimation timer
$ALLOWED\_QUEUE$:standing queue size that we are willing to maintain, it is recommended a nominal value of 2ms

On packet departure do:
$min\_queue = min(min\_queue, inst\_queue)$

When the queue computation timer expires do:
$queue = min\_queue$;
$min\_queue = inst\_queue$;
$T_q = max(ALLOWED\_QUEUE, (avg\_rtt - inst\_queue/C_{Winf})/2)$;
$queue_timer.reschedule(T_q)$;

values for the available bandwidth and, of course, link capacity it is important to take in consideration all possible collisions in a period of time [62]. When a sender, due to medium sense, cannot transmit due to collision the backoff mechanism is activated. This

Algorithm 6: RCP-Winf Sender Algorithm.

/* Available Bandwidth and Capacity Estimation */
snd_wnd: sender congestion window.
$rcp\_bottleneck\_rate$: receiver rate feedback.
rcp_rtt: round trip time.
C_{Winf}: rt-Winf Capacity.
MSS: the maximum segment size.

Access Cross Layer Shared Database;
Retrieve Available Bandwidth and Capacity;
/* Evaluates $rcp\_bottleneck\_rate$ */
if $C_{Winf} - rcp\_bottleneck\_rate <= 0$ **then**
  $rcp\_bottleneck\_rate = C_{Winf}$
/* Calculates congestion window and pacing interval */
$snd\_wnd = \frac{rcp\_bottleneck\_rate * rcp\_rtt}{(MSS + RCP\_HEADER\_SIZE + IP\_HEADER\_SIZE)}$;
$packet\_pacing\_interval = \frac{MSS}{rcp\_bottleneck\_rate}$;
/* Send a packet */
Update Congestion Header;
Send Packet;

Algorithm 7: RCP-Winf Router/Onlooker-Rate Estimation Timer Timeout Operations.

$Available\_Bandwidth_{Winf}$: rt-Winf available bandwidth.
C_{Winf}: rt-Winf capacity.
Tr: Rate estimation interval.
sum_rtt_Tr: Sum of round-trip time values seen in an interval Tr.
avg_rtt_Tr: The average of round-trip time values seen over all RCP packets in an interval Tr.
avg_rtt: The moving average of the round-trip time
$input\_traffic\_Bytes$: is the aggregate amount of incoming RCP traffic.
$input\_traffic\_rate$: The incoming available traffic bandwidth.
$num\_pkts\_with\_rtt$: The number of packets in interval Tr that carry valid round-trip time values.
rcp_rate: Bandwidth offered to a flow and is updated periodically once every Tr.
Q_Bytes: The buffer occupancy at the output interface in Bytes.
$link\_rate$: Link bandwidth measured in Bytes/ms.
$MAX\_RATE\_ESTIMATION\_INTERVAL$: time interval between rcp_rate updates.

On rate estimation timer timeout do:
$input\_traffic\_rate = Available\_Bandwidth_{Winf}$;
$avg\_rtt\_Tr = \frac{sum\_rtt\_Tr}{num\_pkts\_with\_rtt}$;
if $(avg\_rtt\_Tr >= avg\_rtt)$ **then**
  $rtt\_sample\_weight = \frac{Tr}{avg\_rtt}$;
else
  $rtt\_sample\_weight = \frac{rcp\_rate}{C_{Winf}} \times \frac{avg\_rtt\_Tr}{avg\_rtt} \times \frac{Tr}{avg\_rtt}$;
$avg\_rtt = rtt\_sample\_weight \times avg\_rtt\_Tr + (1 - rtt\_sample\_weight) \times avg\_rtt$;
$rcp\_rate = rcp\_rate \times (1 + [\frac{(\frac{Tr}{avg\_rtt}) \times (ALPHA \times (ETA \times C_{Winf} - input\_traffic\_rate) - BETA \times \frac{Q\_Bytes}{avg\_rtt})}{(ETA \times C_{Winf})}])$;
if $(rcp\_rate < MIN\_RATE)$ **then**
  $rcp\_rate = MIN\_RATE$;
else if $(rcp\_rate > ETA \times C_{Winf})$ **then**
  $(rcp\_rate = ETA \times C_{Winf})$;
$Tr = min(avg\_rtt, MAX\_RATE\_ESTIMATION\_INTERVAL))$;
$input\_traffic\_Bytes = 0$;
$num\_pkts\_with\_rtt = 0$;
$sum\_rtt\_Tr = 0$;
$schedule\_rate\_timer(Tr)$;

mechanism is also consuming bandwidth. Being K the extra bandwidth consumed due to backoff, as defined by equation 7.

$$K = \frac{DIFS + backoff}{T(m)} \tag{7}$$

where DIFS represents the IEEE 802.11 Distribution Coordination Function Interframe Space obtained by $SIFS + (2 \times Slottime)$ [22], $backoff$ is the medium backoff time and $T(m)$ is the time between the transmission of two consecutive packets. Whenever a station wants to transmit and the medium is busy, it has to wait some time before sensing again the medium. This time, defined in the IEEE 802.11 standard, is the $backoff$ time and is always known by the sender. When the RTS/CTS handshake is enabled, $T(m)$ can be obtained through:

$$T(m) = 2 \times DIFS + 3 \times SIFS + backoff + CTS + RTS + DATA \tag{8}$$

The collision probability (Pc) can, then, be defined as $1 - k$. Applying this result to the available bandwidth (AB) inference mechanism we have:

$$\begin{aligned} AB = Pc \times AB_{Winf} \Rightarrow AB = (1-k) \times AB_{Winf} \\ \Rightarrow AB = (1 - \frac{DIFS + backoff}{T(m)}) \times AB_{Winf} \end{aligned} \tag{9}$$

Equation 9 is only used when a node is in the *Sender* state of XCP- and RCP-Winf. It is only possible to control and obtain the $T(m)$ and $backoff$ times in the *Sender* state. Using the information available in the *Sender* improves queue management, leading to less packet losses and, also, to queue management improvement in the intermediate nodes. A node in the *Onlooker* state only updates its NAV, not being capable of determining the overall transmission side collision probability as it is not aware of some important time control information. In the *Receiver* state, a node can only account with updated NAVs, that are important for determining the Idle Rate, which do not consider the collision of the sender side. As collision can only occur when a node is sending packets, the information obnatined by the sender is the one that can improve collision control. As rt-Winf is a cooperative inference mechanism, the sender being able to infer collision probability and, then, use that information against the received information, allows rt-Winf to use more precise information in its decision process, resulting in a more fair and accurate rate control.

5 Simulation Results

In order to monitor, observe an measure the performance of XCP- and RCP-Winf we performed several simulations. This section presents and discusses the results of the simulations carried out.

5.1 XCP-Winf and RCP-Winf Results

This section introduces the simulation setup to evaluate the performance of the proposed congestion control mechanisms and presents the simulation results. The results were obtained using the ns-2 simulator. To evaluate the performance three metrics were used: throughput, delay and number of received packets. Different wireless mesh and ad hoc scenarios were used. The simulations were repeated using different ns-2 seed values. The configured default transmission range is 250 meters, the default interference range is 500 meters, and the channel data rate is 11 Mbps. For the data transmissions, it was used an FTP application with packets of 1500 bytes or a Constant Bit Rate (CBR) application. For mobility the ns-2 *setdest* tool is used. This tools 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, with the help of *trace2stats* scripts [63] adapted to our own needs. The routing protocol used was the Destination-Sequence Distance-Vector (DSDV) [64]. The presented results show the average aggregated value and the 95% confidence interval.

Next we present, analyze and compare the mesh topologies results. The mesh topologies defined were: a grid of 5, 9, 12 and 16 fixed mesh nodes. In all mesh topologies, it was used a combination of 3, 4, 5, 6 and 7 mobile nodes. Figure 9 represents a mesh topology of 5 mesh nodes and 5 mobile nodes. The mobile nodes were, simultaneously, sources and sinks. The results show throughput, delay and the number of received packets.

Figure 10(a), Figure 10(b) and Figure 10(c) show the previously referred performance metrics for five different scenarios. In each scenario was used a fixed number of 16 mesh nodes and a variable number, from 3 to 7, of mobile nodes. Each mobile node, as previously stated, is simultaneously sending and receiving data.

The obtained results show that the integration of rt-Winf in XCP and RCP improves significantly their

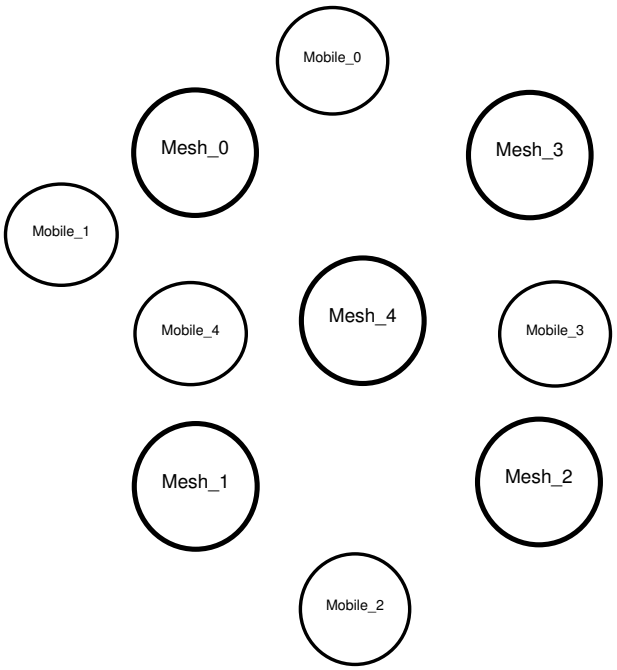
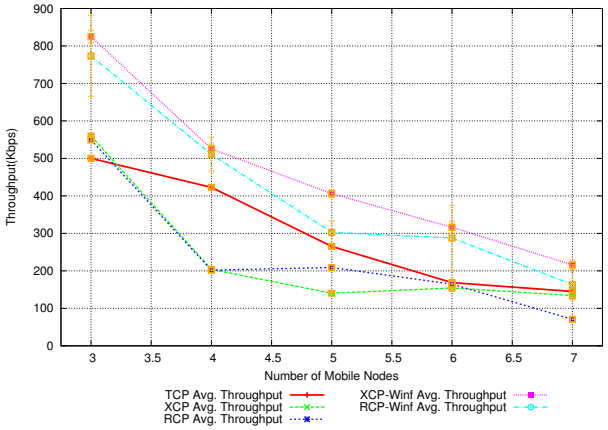


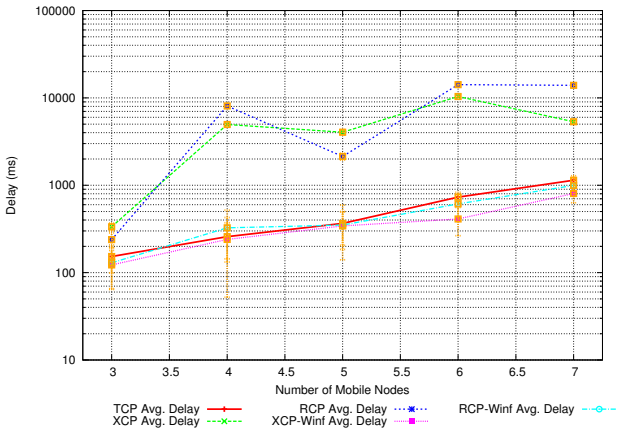
Figure 9: Topology 5 Mesh Nodes - 5 Mobile Nodes.

behavior. The available bandwidth and capacity evaluation of rt-Winf, and the cross-layer information, are important and make XCP and RCP behave more efficiently and with better channel utilization, this also leads to less channel losses (more received packets). The use of rt-Winf in the the mesh nodes (onlooking state) makes the feedback mechanism more accurate, as all nodes in the network can determine available bandwidth and capacity, and send that information to the other nodes that are participating in the communication.

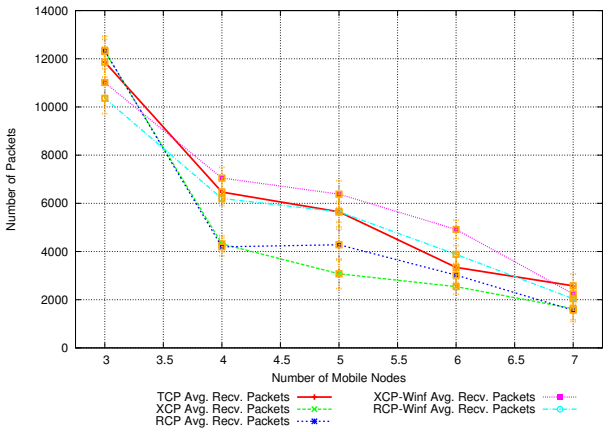
For evaluating XCP-Winf and RCP-Winf against CBR Applications, it was configured an UDP application (simulating a VoIP application), for the 16 mesh nodes scenario and variable number of mobile nodes. Figure 11 shows the obtained results for the 16 mesh nodes scenario and variable number of mobile nodes. Without rt-Winf enabled, XCP obtains better results than RCP for a lower number of mobile nodes. This is due to the fact that RCP was developed having in mind for Internet bursts traffic. With less mobile nodes changing information, the number of collisions is lower what means less re-transmissions and less burst traffic in the network. It is, also, possible to conclude that both XCP and RCP are not evaluating correctly the link capacity and don't have the necessary mechanisms to overcome this situation. With rt-Winf, the throughput results are considerably better, but, still, reflect the problems that XCP and RCP have in controlling congestion when the traffic is UDP. Once more, RCP-Winf reflects its base development for bursty traffic. The CBR application is sending data at a constant rate but with more mobile nodes send-



(a) Throughput.



(b) Delay.



(c) Received Packets.

Figure 10: 16 Mesh Nodes - Variable Number of Mobile Nodes Results.

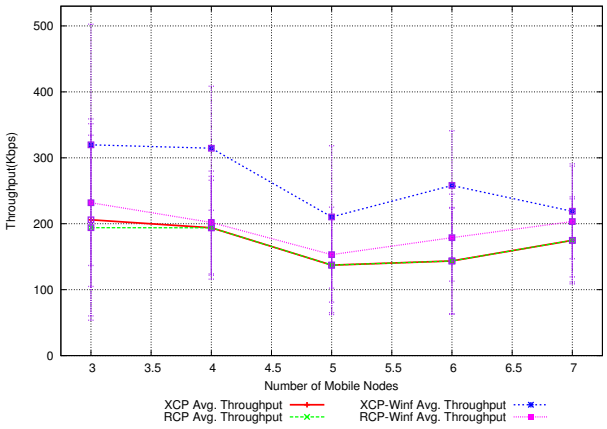
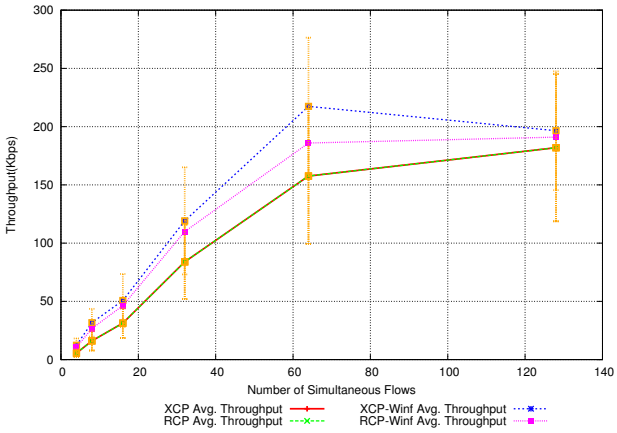


Figure 11: 16 Mesh Nodes - Variable Number of Mobile Nodes, CBR Throughput.

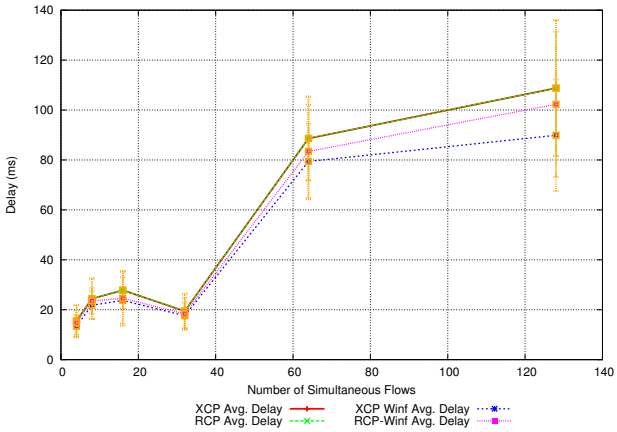
ing data, more collisions will occur and more bursts of traffic will be present in the network. This situation will allow RCP to react more precisely and, with more mobile nodes, to have better throughput results.

The same metrics were studied in several ad hoc networks scenarios using CBR UDP 64 Kbps flows. The scenarios were composed of 8, 16, 32, 64, 128 and 256 nodes, and for each scenario we had 4, 8, 16, 32, 64 and 128 simultaneous flows. The flows were randomly generated through the ns-2 *gencbr.tcl* tool. The mobility was also dynamically generated through different seed values. The obtained results are presented in Figure 12(a), Figure 12(b) and Figure 12(c).

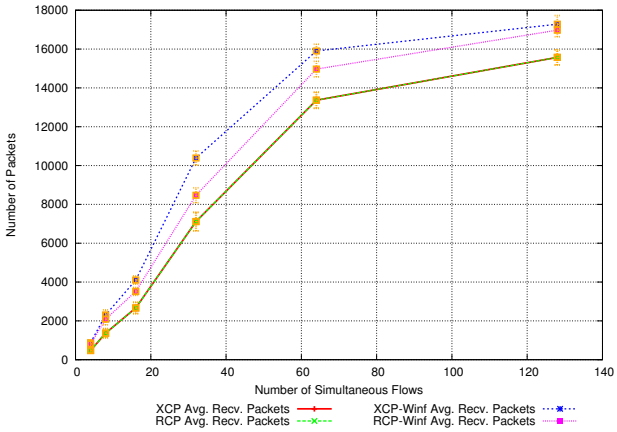
From the analysis of those results it is possible to conclude that standard XCP and RCP react the same way when the traffic is UDP, while the integration of rt-Winf makes them react differently as they both use the information from the MAC sublayer differently, that is available bandwidth and capacity. 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 node queue management. It is possible, the, to hava higher rate and less losses. As more packets are transmitted,more throughput is obtained and the medium is better used,it is possible to infer that both XCP-Winf and RCP-Winf are more stable and fair. In the same conditions, it is possible to send more information with a higher rate. It mus be noticed that the results also reflect the mobility randomness, where it is possible to conclude that we have situations when more nodes are in each other influence area. Another factor that is influencing the results is the routing information and the exchanged routing messages that as flows increase also increases collisions and delay on the net-



(a) Throughput.



(b) Delay.



(c) Received Packets.

Figure 12: Ad Hoc Network with Variable Number of Flows Results.

work, being also reflected in throughput values.

For a more real evaluation of XCP-Winf and RCP-Winf it was defined a new mesh network scenario. The parameters of this new scenario were defined to simulate a public building, with public services and a public garden (Figure 13). Table 1 shows all the parameters defined for the simulation. In this scenario mobile nodes start to transmit in a random way and their transmission last 240 seconds. The mesh nodes position was defined, also, randomly in the inside area. It was also defined that a mesh node, randomly choosed, was shutdown for 100 seconds. Two types of flows were used a light traffic flow and a heavy traffic flow, as defined in Table 1.

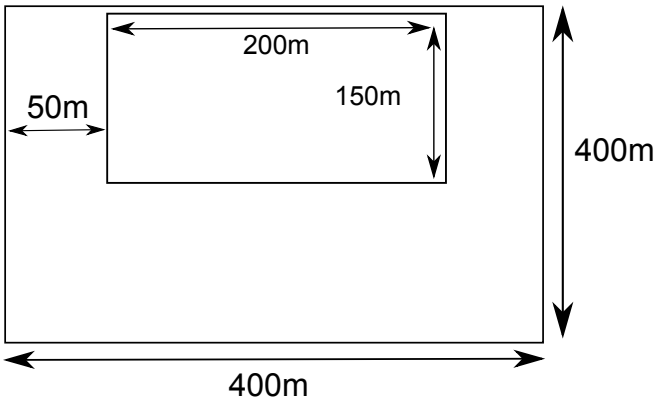
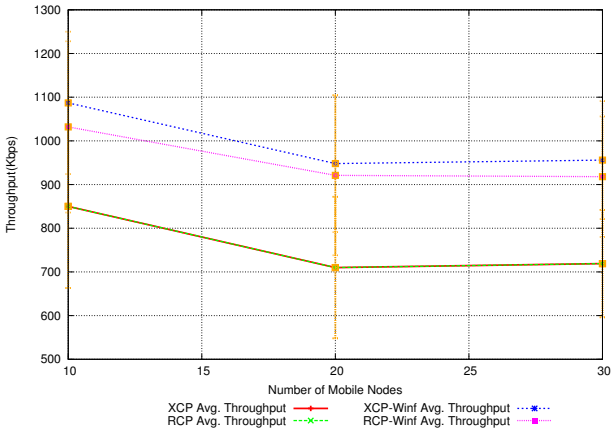


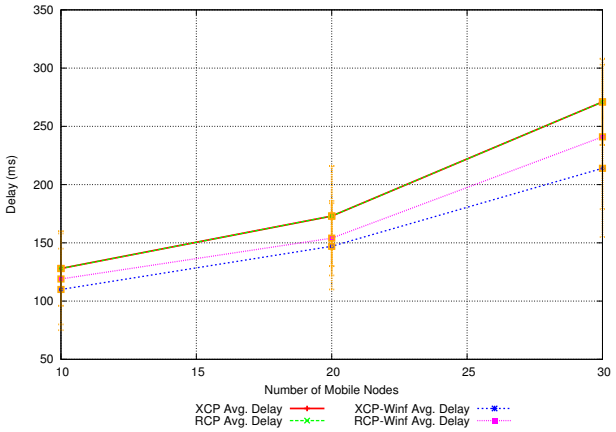
Figure 13: Building Layout Simulation.

The obtained performance results for the light traffic flows are shown in Figure 14(a), Figure 14(b) and Figure 14(c); for the heavy traffic flows results are presented in Figure 15(a), Figure 15(b) and Figure 15(c). As can be observed from the results the integration of rt-Winf in both XCP and RCP improve their standard behavior. As nodes that are not participating in the communication enter the *Onlooker* state and, also, evaluate the network performance, is is possible to have a state by state and rate by rate overall performance evaluation. As rt-Winf uses, in its operation, three different states and network cooperation, it is possible to have a more effective and efficient hop by hop performance evaluation. This results in a more efficient evaluation and use of the channel capacity. As XCP-Winf and RCP-Winf also have more capability and facility to adapt to the changing conditions of the network this is expressed in better transmitting rates and better channel usage.

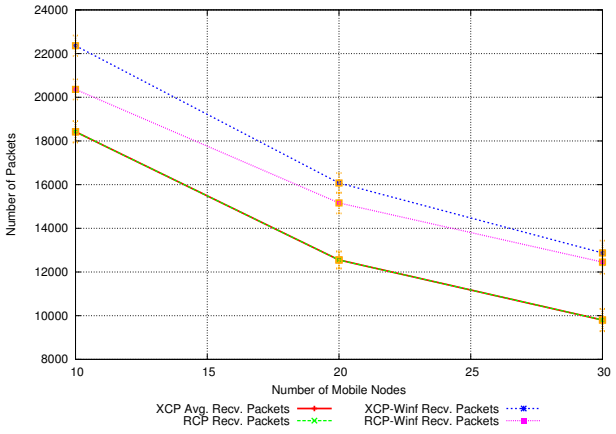
For understanding how node queue management is affected by the XCP-Winf and RCP-Winf cooperation mechanisms, one of the mobile nodes queue



(a) Light Flow Throughput.



(b) Light Flow Delay.



(c) Light Flow Received Packets.

Figure 14: Public Building Light Flow Simulation Results.

length was actively measured. The obtained values are represented in Figure 16. Compared to the standard versions, XCP-Winf and RCP-Winf are using more efficiently the queue, allowing a more stable occupation of the queue, that is then reflected in the net-

Simulation Parameters	
Topology Area	400m x 400m
Simulation Time	600 sec.
Simulation repetition	30 times
Number of Mobile Nodes	10, 20, 30
Number of Mesh Nodes	6
Mesh Nodes Position	Random
Path Loss Model	Two Ray
Mobility Model	Random Way Point
Maximum Movement Speed	30 m/s
Light Traffic	4 CBR 64 Kbps Flows, sent each 400 ms
Heavy Traffic	4 CBR 128 Kbps Flows, sent each 100 ms
Mac layer	IEEE 802.11
Propagation Model	Two Ray Ground
Routing Protocol	DSDV

Table 1: Simulation Environment.

work performance values obtained. Is is possible to see that standard XCP and RCP have high variations of queue length, that is, surely, making them behave more unstable. We can conclude that XCP-Winf and RCP-Winf can manage more efficiently queue nodes, as rely on global network information.

Finally, we analyzed the influence of the collision probability in the performance evaluation. We carried out a set of new simulations, in the building simulation layout, changing the speed of the nodes. The new simulations were defined with no mobility and with a maximum mobility speed of 100 m/s, with the purpose to ensure extreme mobility. Figure 17 shows the obtained results for XCP-Winf and Figure 18 for RCP-Winf. It is possible to conclude that with increased speed and more traffic in the network, the collision probability is improving the performance. This is due as with more speed and more traffic more collisions will occur, making the collision rate an important factor in the performance evaluation. Using the collision probability parameter allows, thus, to achieve better medium usage that is, then, reflected in more efficient available bandwidth and capacity evaluation. Collision probability improves throughput performance results from $\sim$ 6.5 % in RCP-Winf to $\sim$ 8.5 % in XCP-Winf when nodes are moving very quickly and the traffic is saturating the network. In such conditions the collision probability rate is an important factor, having more influence in the wireless link capacity evaluation.

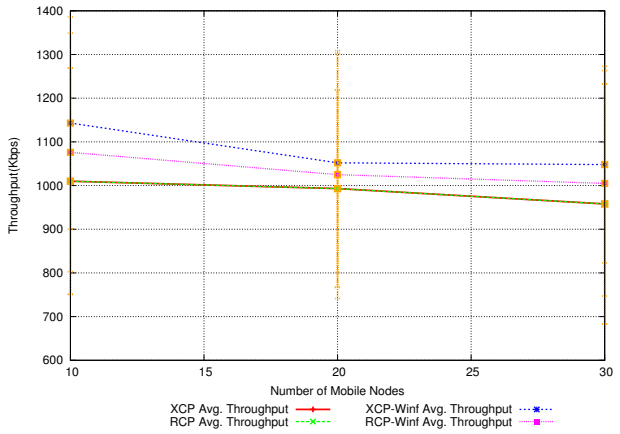
6 Conclusions

In this paper we presented a study that demonstrates that using MAC layer information, through a cross layer communication process, in congestion control can be an important factor of network performance improvement.

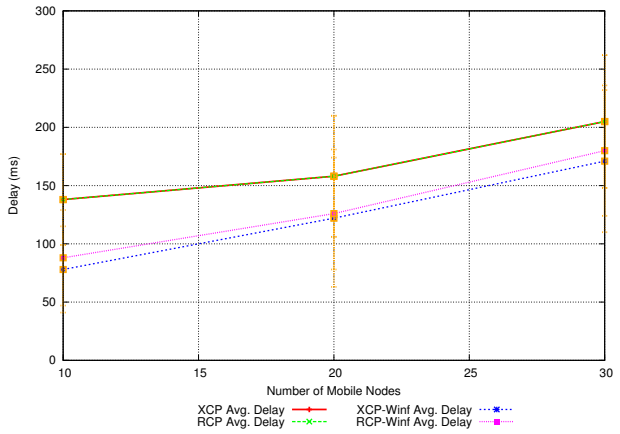
A passive monitoring tool, rt-Winf, for the measurement of wireless capacity and available bandwidth was first introduced. rt-Winf uses information already available on the network: it can rely on the CTS/RTS/ACK messages handshake or on small probes. These packets provide time information, allowing to know each node's channel allocation. Then, it was presented the cross layer approach and the integration of rt-Winf in both standard XCP and RCP. These new congestion control protocols are designated XCP-Winf and RCP-Winf.

rt-Winf evaluation results, using CMU emulator and ns-2 simulator, show that 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.

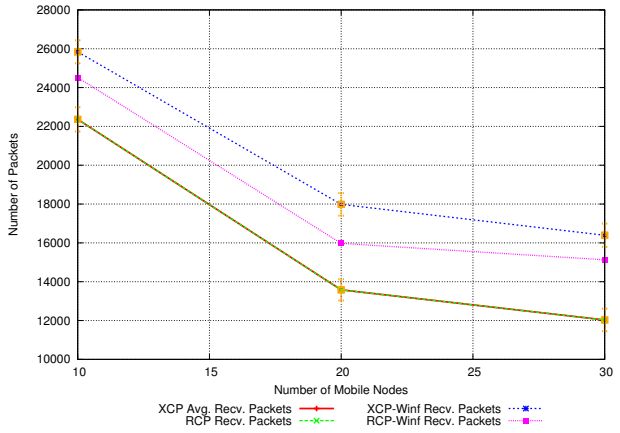
The evaluation results of XCP-Winf and RCP-Winf, obtained through ns-2 simulations, show that the rt-Winf algorithm improves significantly XCP and RCP behavior, making them more efficent and stable. To obtain the available network capacity, both XCP and RCP need that all nodes in the network cooperate, which increases network overhead, specially when



(a) Heavy Flow Throughput.



(b) Heavy Flow Delay.



(c) Heavy Flow Received Packets.

Figure 15: Public Building Heavy Flow Simulation Results.

dealing with special wilreless environments, such as wireless mesh networks and ad hoc networks. Using rt-Winf, that works in the MAC layer, it is possible to perform link capacity and available bandwidth calculations without interfering in the network dynamics,

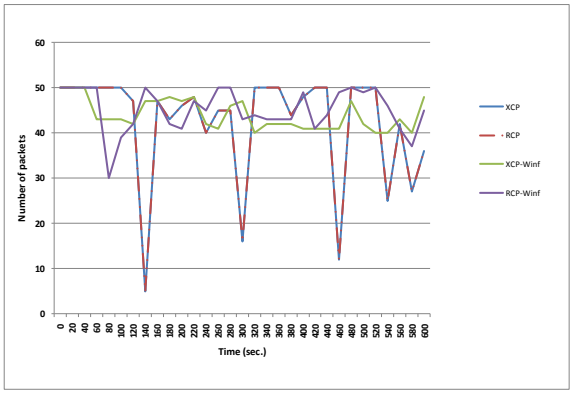


Figure 16: Mobile Node Avg. Queue Length.

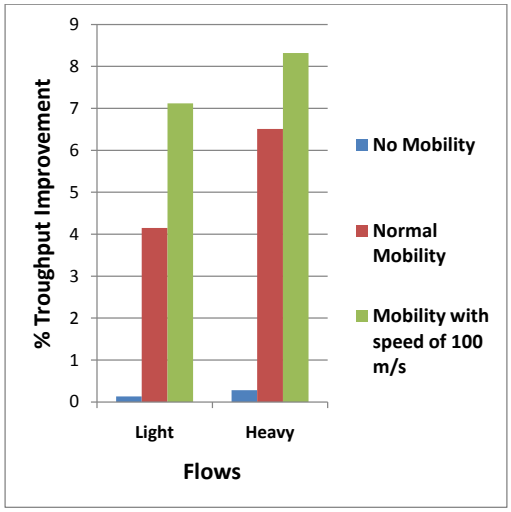


Figure 17: XCP-Winf Collision Probability.

allowing to significantly improve XCP and RCP performance. 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.

References:

[1] S. Vangala, , S. Vangala, and M. A. Labrador, "Performance of TCP over wireless networks with the snoop protocol," in *In Proceedings of the 27th Annual IEEE Conference on Local Computer Networks (LCN 2002*, 2002.

[2] H. Balakrishnan, V. N. Padmanabhan, S. Seshan, and Y. H. Katz, "A comparison of mechanisms for improving TCP performance over wireless

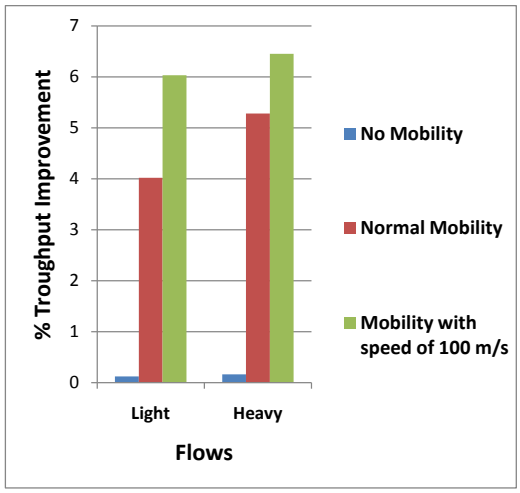


Figure 18: RCP-Winf Collision Probability.

links," *IEEE/ACM Transactions on Networking*, vol. 5, pp. 756–769, 1997.

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

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

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

[6] L. Barreto and S. Sargento, "TCP, XCP and RCP in wireless mesh networks: An evaluation study," in *15th IEEE Symposium on Computers and Communications (IEEE ISCC'10)*, Riccione, Italy, 6 2010.

[7] S. M. Buhari, M. H. Habaebi, and B. M. Ali, "A new congestion control algorithm for active networks," *Pertanika Journal of Science and Technology*, pp. 187–211, April 2005.

[8] 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, 2010.

[9] 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.

[10] "NS-MIRACLE: Multi- InteRfAce Cross-Layer Extension library for the Network Simulator," 2008. [Online]. Available: http://www.dei.unipd.it/wdyn/?IDsezione=3966

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

[12] V. J. R. et al., "Pathchirp: Efficient available bandwidth estimation for network paths," *Passive and Active Measurement (PAM) Workshop*, April 2003.

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

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

[15] N. H. et al., "Evaluation and characterization of available bandwidth probing techniques," *IEEE Journal on Selected Areas in Communications*, vol. 21, no. 6, August 2003.

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

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

[18] 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.

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

[20] M. S. Gast, *802.11 Wireless Networks - The Definitive Guide*, 4th ed. O'reilly, 2002.

[21] F. Tobagi and L. Kleinrock, "Packet switching in radio channels: Part ii–the hidden terminal problem in carrier sense multiple-access and the busy-tone solution," *Communications, IEEE Transactions on*, vol. 23, no. 12, pp. 1417 – 1433, Dec. 1975.

[22] *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

[23] H. Zimmermann, "OSI reference model- the ISO model of architecture for open systems interconnection," *IEEE Transactions on Communications*, vol. 28, no. 4, April 1980.

[24] L.-J. Chen, C.-W. Sung, H.-H. Hung, T. Sun, and C.-F. Chou, "Tsprobe: A link capacity estimation tool for time-slotted wireless networks," in *Wireless and Optical Communications Networks, 2007. WOCN '07. IFIP International Conference on*, July 2007, pp. 1 –5.

[25] C. C. Mario, M. Gerla, S. S. Lee, S. Mascolo, and M. Sanadidi, "TCP with faster recovery," in *In IEEE Milcom*, 2000.

[26] D. E. Chandra and B. Subramani, "A survey on congestion control," *Global Journal of Computer Science and Technology*, vol. 9, no. 5, pp. 82 – 87, 2010.

[27] J. Postel, "Transmission control protocol," RFC 793, Defense Advanced Research Projects Agency, September 1981. [Online]. Available: http://www.rfc-editor.org/rfc/rfc793.txt

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

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

[30] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. Gerla, "The impact of multihop wireless channel on TCP throughput and loss," 2003, pp. 1744–1753.

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

[32] 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.US: ACM, 1999.

[33] 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.

[34] 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.

[35] K. Sundaresan, V. Anantharaman, H.-Y. Hsieh, and A. R. Sivakumar, "Atp: a reliable transport protocol for ad hoc networks," *Mobile Computing, IEEE Transactions on*, vol. 4, no. 6, pp. 588–603, nov. 2005.

[36] C. L. T. Man, G. Hasegawa, and M. Murata, "ImTCP: TCP with an inline measurement mechanism for available bandwidth," *Comput. Commun.*, vol. 29, pp. 1614–1626, June 2006.

[37] S. M. ElRakabawy, A. Klemm, and C. Lindemann, "TCP with adaptive pacing for multihop wireless networks," in *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, ser. MobiHoc '05. ACM, 2005, pp. 288–299.

[38] K. Tan, F. Jiang, Q. Zhang, and X. Shen, "Congestion control in multihop wireless networks," *Vehicular Technology, IEEE Transactions on*, vol. 56, no. 2, pp. 863–873, mar. 2007.

[39] G. Anastasi, E. Ancillotti, M. Conti, and A. Passarella, "Tpa: a transport protocol for ad hoc networks," in *Computers and Communications, 2005. ISCC 2005. Proceedings. 10th IEEE Symposium on*, June 2005, pp. 51 – 56.

[40] C. Cordeiro, S. Das, and D. Agrawal, "Copas: dynamic contention-balancing to enhance the performance of TCP over multi-hop wireless networks," in *Computer Communications and Networks, 2002. Proceedings. Eleventh International Conference on*, Oct. 2002, pp. 382 – 387.

[41] Z. Fu, H. Luo, P. Zerfos, S. Lu, L. Zhang, and M. Gerla, "The impact of multihop wireless

channel on TCP performance," *Mobile Computing, IEEE Transactions on*, vol. 4, no. 2, pp. 209 – 221, April 2005.

[42] S. Rangwala, A. Jindal, K.-Y. Jang, K. Psounis, and R. Govindan, "Understanding congestion control in multi-hop wireless mesh networks," in *Proceedings of the 14th ACM international conference on Mobile computing and networking*, ser. MobiCom '08. ACM, 2008, pp. 291–302.

[43] 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

[44] "100x100 clean state project." [Online]. Available: http://100x100network.org/

[45] A. Sridharan and B. Krishnamachari, "Explicit and precise rate control for wireless sensor networks," in *SenSys '09: Proceedings of the 7th ACM conference on Embedded network sensor systems*. ACM, 2009.

[46] G. Nychis, G. Sardesai, and S. Seshan, "Analysis of XCP in a wireless environment," Carnegie Mellon University, Tech. Rep., 2006. [Online]. Available: http://www.andrew.cmu.edu/user/gnychis/xcp_wireless.pdf

[47] Y. Su and T. Gross, "Wxcp: Explicit congestion control for wireless multi-hop networks," in *In IWQoS '05: Proceedings of the 12th International Workshop on Quality of Service*, 2005.

[48] F. Abrantes and M. Ricardo, "A simulation study of XCP-b performance in wireless multi-hop networks," in *Proceedings of the 3rd ACM workshop on QoS and security for wireless and mobile networks*, ser. Q2SWinet '07. ACM, 2007.

[49] R. Rom and M. Sidi, *Multiple Access Protocols: Performance and Analysis (Telecommunication Networks and Computer Systems)*. Springer. [Online]. Available: http://www.amazon.com/exec/obidos/redirect?tag=citeulike07-20&path=ASIN/0387972536

[50] Z. Haas, "On the performance of a medium access control scheme for the reconfigurable wireless networks," in *MILCOM 97 Proceedings*, vol. 3, Nov. 1997, pp. 1558 –1564 vol.3.

[51] S. Ray, J. B. Carruthers, and D. Starobinski, "Evaluation of the masked node problem in ad-hoc wireless lans," *IEEE Transactions on Mobile Computing*, vol. 4, pp. 430–442, 2004.

[52] S. M. ElRakabawy, A. Klemm, and C. Lindemann, "TCP with adaptive pacing for multihop wireless networks," in *Proceedings of the 6th ACM international symposium on Mobile ad hoc networking and computing*, ser. MobiHoc '05. ACM, 2005, pp. 288–299.

[53] K. Xu, M. Gerla, and S. Bae, "How effective is the ieee 802.11 rts/cts handshake in ad hoc networks," in *Global Telecommunications Conference, 2002. GLOBECOM '02. IEEE*, vol. 1, Nov. 2002, pp. 72 – 76 vol.1.

[54] H. Khalife and N. Malouch, "Interaction between hidden node collisions and congestions in multihop wireless ad-hoc networks," 2006. [Online]. Available: www-rp.lip6.fr/ khalife/extended.pdf

[55] G. Bianchi, "Performance analysis of the ieee 802.11 distributed coordination function," *Selected Areas in Communications, IEEE Journal on*, vol. 18, no. 3, pp. 535 –547, Mar. 2000.

[56] M. Franceschetti, M. Migliore, and P. Minero, "The capacity of wireless networks: Information-theoretic and physical limits," *Information Theory, IEEE Transactions on*, vol. 55, no. 8, pp. 3413 –3424, Aug. 2009.

[57] "Recommendation ITU-T G.711.0," ITU, 2005. [Online]. Available: http://www.itu.int/rec/T-REC-G.711/

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

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

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

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

[62] "Accurate available bandwidth estimation in ieee 802.11-based ad hoc networks," *Computer Communications*, vol. 32, no. 6, pp. 1050 – 1057, 2009.

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

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