Janus: A Novel MAC Protocol for Full Duplex Radio

Jae Young Kim, Omid Mashayekhi, Hang Qu, Maria Kazandjieva, and Philip Levis
Stanford University
jykim1,omidm,quhang@stanford.edu, mariakaz,pal@cs.stanford.edu

Abstract

This paper presents Janus, a novel MAC protocol for full–duplex wireless networks. Unlike other full-duplex MACs, Janus allows partially interfering nodes to cooperate by finding the appropriate transmission rates based on interference levels, making better use of the channel. Computing the optimal schedule and transmission rates is NP-Complete, so Janus uses a cheaper heuristic approach. Janus also ensures that channel access time is shared fairly between all nodes. Janus has lower per-packet overhead compared to CSMA/CA because it eliminates random back-off and lets nodes transmit multiple packets with a single set of control packets. We show that for a setup with one access point and three nodes, Janus achieves 2.5X the throughput of half–duplex system based on CSMA/CA.

1 Introduction

Single–channel full–duplex wireless is a nascent and exciting field of research. Its feasibility has been shown at the physical layer with off-the-shelf components [3, 7, 8]. In an effort to build up the stack, there have been several proposals [9, 13] for modified 802.11 MAC protocols. Nevertheless, we believe that in order to make the most out of the unique characteristics of a full–duplex physical layer, it is important to design a clean–slate MAC layer.

This paper presents Janus, an AP-based MAC protocol, specifically designed for full–duplex access networks. The design and implementation of Janus are guided by several goals:

- **Identify all full–duplex opportunities.** Determining when nodes are hidden to each other is critical to identifying all the cases in which simultaneous transmissions can happen. This requires an accurate picture of interference in the network.
- **Schedule packet exchange to maximize throughput.** There could be multiple candidates to form full–duplex cooperations. Packet exchanges should be arranged to leverage simultaneous transmissions in a way that maximizes throughput.
- **Provide fairness.** Protocol needs to remain fair in assigning the opportunities.

Prior work [9, 13] has considered nodes to be fully hidden or fully conflicting, not taking into account the amount of interference the nodes experience. Janus takes a different approach and asks, can some interference be acceptable if packets are sent at a lower data rate? In fact, Janus considers the interference as a source that degrades the channel capacity, but still reliable communications are possible by decreasing the rate of transmission. Janus is the first full–duplex MAC protocol to take into account interference levels when determining the appropriate data rate of simultaneous transmissions. Now, partially interfering nodes can also benefit from concurrent transmissions.

The main job of every MAC protocol is to resolve the problems of channel contentions and collisions. Janus uses a centralized mechanism to schedule transmission in a way that eliminates collisions. The protocol operates in rounds; at the beginning of each round, the AP asks nodes what length packets they would like to send, and also collects information about the interference environment around each node. The latter is critical for Janus's first goal of identifying all full–duplex opportunities. Although collecting information adds overhead, it helps Janus to maximizes full–duplex cooperations and, improved overall performance of the system justifies this effort.

With interference and packet length data at hand, the AP can start matching outgoing and incoming packets in order to take advantage of the full–duplex PHY layer. For each transmission, the AP has to make several decisions – when to transmit, whether to use full–duplex or half–duplex, and what data rate to send packets at. We show that finding the optimal match of incoming and outgoing transmissions is an NP-complete problem. Therefore, the paper proposes a heuristic that yields throughput performance close to the best case. Janus introduces new intuitive metrics to use in the low complexity step-by-step decision making process to benefit the overall throughput.

Fairness should not be compromised while improving throughput. Two nodes could make great full–duplex pair and benefit the overall throughput, but they should not be allowed to starve other nodes. Janus has load control mechanism to enforce fairness, as well.

Our MAC protocol is implemented for and tested on the WARP board hardware [2]. We study the Janus's performance under increasing load, and compare it to the throughput that half–duplex based on CSMA/CA can achieve.

The evaluation results show that Janus achieves 2.5X the throughput of half–duplex for highly loaded systems; a number that reflect not only the gain from scheduling simultaneous transmissions but also the reduction in per-packet overhead. Janus meets the goal of low overhead operation by ensuring that only nodes with packets to send, participate in the control packet exchange. While this still introduces control packets, it eliminates the chance of collisions and so CSMA/CA-style back-offs become unnecessary.

The rest of the paper is organized as follows. The next section briefly discusses related MAC protocols and Section 3 gives an overview of Janus. Sections 4 and 5 present the two key Janus mechanisms, namely information collection and scheduling. Section 6 covers several implementation challenges. Lastly, Section 7 presents an evaluation.

2 Related Work

Janus is a centralized MAC protocol, in which the AP controls the rate and timing of packet transmissions. The idea of receiver-initiated MAC layers is not new and is common in the context of low-power wireless networks [5, 11]. Such protocols have been shown [6, 14] to better handle the hidden node problem. Since determining hidden node relationships accurately is critical to optimizing full–duplex transmissions, we choose a centralized MAC design over a distributed one.

Single–channel full–duplex communication is an emerging area, with its practical feasibility shown [3, 7, 8] only recently. There are two MAC protocols [9, 13] for full–duplex that set out goals similar those of Janus.

The FD-MAC protocol [9] is a distributed MAC for access networks, built on top of 802.11's CSMA/CA. The work proposes several new mechanisms in order to leverage full–duplex opportunities and to prevent node starvation. FD-MAC allows the AP to switch between full–duplex and half–duplex mode in order to ensure that all nodes get a chance to transmit. Unlike Janus, FD-MAC does not provide fairness guarantees. Since FD-MAC has no explicit topology discovery, nodes snoop on each others' transmissions to learn about hidden nodes. If a node N_1 overhears an ACK packet sent from a neighboring N_2, then N_1 knows that its transmissions will interfere at N_2, ruling out some full–duplex opportunities. The gains of the FD-MAC (with its associated physical layer) are 0.7X over a half–duplex setup, significantly lower than what Janus achieves.

ContraFlow [13] is another full–duplex MAC proposal that is not limited to AP-based networks. The contention problem is again addressed with a modified (for fairness) version of the 802.11 back-off procedure. Acknowledgments are send immediately after a packet, with the second node transmitting a busy tone in order to protect the ACK from hidden nodes. This is a tradeoff that ensures lower ACK latency at the price of wasted channel time. Janus takes a slightly different approach in which ACKs are sent at the end of a round of packets. If a round is only one packet time, Janus' approach is similar to that of ContraFlow. But batching all the ACK packets gives more flexibility to full–duplex scheduling, rather than force each ACK packet to be scheduled immediately after the corresponding data packet. And Janus can save bandwidth due to fewer ACK packets needed.

ContraFlow uses the history of successfully received packets from a node to infer the interference structure of the network. The level of interference is abstracted as the probability that the interference can corrupt simultaneous full–duplex packet receiving. However, this method of interference estimation only considers packets sent at the highest rate.

The ways in which FD-MAC and ContraFlow identify full–duplex opportunities is substantially different from Janus' approach. In the prior two mechanisms, nodes are either hidden or conflicting. Janus' contribution is to recognize that this binary classification is too coarse, limiting potential full–duplex gains. Instead, Janus uses interference measurements to determine whether nodes are hidden for some (lower) data rates but not for other, higher ones. This flexibility means that in cases in which prior MACs would have ruled out a full–duplex transmission, Janus can send additional packets, albeit at a lower rate.

3 Janus Overview

Janus is an AP-based MAC protocol for full–duplex wireless networks. The main challenge in full–duplex is to determine how to schedule simultaneous transmission so throughput is maximized. There are two scenarios in which simultaneous transmissions can happen. In first case, the AP and another node exchange packets at the same time. This is always feasible, as long as the nodes have data to exchange; it will not cause any harmful interference.

In second case, the AP sends packets to one node, say N_1, while another node, N_2, sends packets to the AP at the same time. There is a potential problem since the packets originating at N_2 might corrupt the packets that N_1 is receiving.

A simplistic way to solve the problem is to only allow the second scenario when N_1 and N_2 are completely hidden to each other. However, the wireless environment is not binary, and so the level of interference could be very high or very low. Janus takes advantage of the continuous nature of interference by scheduling some packets to be transmitted at a lower rate, so the interference can be tolerated at the overhearing node.

The rest of this section presents an overview of the main Janus components. Sections 4 and 5 cover individual aspects of the protocol in more detail.

AP Information Collector: The AP initiates each cycle of the protocol by sending a *probe* packet. This packet signals all the nodes registered under the current AP to send a set of information in a predefined order to AP. This information is two fold. First, the length of the transmission (in bytes) that nodes intend to send to the AP. Second, interference that it senses from those nodes in the network that it can

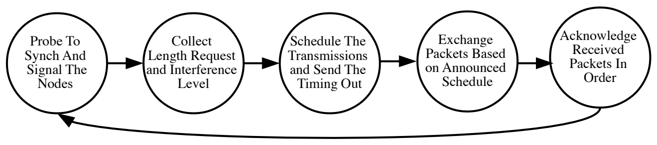

Probe To Synch And Signal The Nodes
Collect Length Request and Interference Level
Schedule The Transmissions and Send The Timing Out
Exchange Packets Based on Announced Schedule
Acknowledge Received Packets In Order

Figure 1: The state machine of Janus. AP initiates each cycle of protocol by sending a probe.

possibly form a simultaneous transmission with (if the interference from a node is severe it will not bother to send the data according to that node). Janus forms a table with collected interference values called *Conflict Map*. In fact, Janus is an AP-based protocol. AP collects information and serves as a central decision unit.

Full Duplex Scheduler: The collected data from nodes plus the outgoing queue of the AP (downlink traffic) is provided to the Janus scheduler. This module uses the information to determine which packets will be transmitted concurrently and at what data rate.The detailed schedule of transmission times are sent to the nodes. Then, the packet exchange will start based on the scheduled timings. If there is any other metric that needs to be enforced (e.g. fairness, latency, QoS) Janus could reflect it in the scheduler. Specifically, we use the Deficit Round Robin technique [12] to achieve fairness and control latency.

Acknowledging Packets: At the end of each cycle, there are slots allocated again by the scheduler to each node to acknowledge the possibly received packets during the cycle. Because of the interleaved packets from of uplink and downlink traffic, nodes cannot acknowledge received packets immediately. If they did, it could cause a collision with an ongoing transmission. Janus, postpones all acknowledgments until after all packet exchanges have terminated.

Figure 1 shows the state machine of Janus. Nodes are waiting for the AP to signal the beginning of each cycle by a probe. Then, they send the uplink transmission request, and the interference level they sense from other nodes to the AP. The scheduling is done at the AP to leverage any possible full–duplex opportunity and enforce fairness. Then, packet exchange occurs based on the announced timings followed by acknowledgements. The next sections describe each of the Janus components in more detail.

4 Collecting Information at the AP

Since Janus is a centralized MAC protocol, the access point (AP) collects all the necessary traffic and interference information. Based on that, Janus can make full–duplex scheduling decisions that are both fair and maximize throughput. Although a distributed MAC protocol, like CSMA/CA, is simpler and could have less overhead, it is unlikely to achieve good performance with only local information. This is the case because in order to maximize full–duplex transmissions, Janus needs interference information at non-AP nodes.

The challenge of information collection is two-fold. First, the information exchange mechanism should be efficient with minimum overhead. In particular, a node should not cause overhead unless it is active and has packets to transmit. Second, the interference information should be measured and calculated accurately, so the AP can make the best scheduling decisions.

4.1 Control Packet Exchanges

Janus exchanges control, data, and ACK packets in rounds coordinated by the AP, as shown in Figure 2. Each round consists of three periods: the scheduling preparation period for collecting necessary scheduling information and broadcasting the schedule result, the packet exchange period for data packet transmission, and the acknowledgement period.

4.1.1 Scheduling Preparation Period

During the scheduling preparation period, the AP exchanges scheduling information with the rest of the nodes associated with it. The information gathering has two stages and idle nodes without packets to send do not participate in the second stage, minimizing overhead. Before a node participates in the information exchange, it registers itself with the AP, and is assigned a distinct ordered ID. The registration step will be discussed in more detail later.

In the first stage of the information exchange, the AP queries all nodes whether they have packets to send. This is done via a Probe Request packet. Each node that wants to participate in this round of full–duplex transmissions responds in order (based on ID) with a packet. The responses are labeled Request Flags in Figure 2

The duration for flag transmission lasts for fixed number of flag packet slots, e.g., 64. And each distinct flag corresponds to one of the slots. A better way to implement the flag transmission is to use the single tone scheme, decreasing overhead. Single tones have been previously used for frequency back-off [10] due to their ability to conserve bandwidth. The single tones from different nodes can be sent simultaneously, and only several OFDM symbol periods are needed for the AP to identify which single tones are on or off. The current Janus implementation, however, does not implement this single tone scheme due to the resource limitation on the WARP board.

As soon as the AP knows which nodes want to be involved in the next round of full–duplex communication, the AP sends a Request Information packet (RI) that gives the nodes the order in which they should transmit their replies. Each node sends a Reply Request Information packet (RRI) at its predetermined slot. Determining when to transmit is straightforward, since the data rate and packet length of the RRI packets are fixed. Since each node knows the order of transmissions, it can calculate when to send its own reply packet.

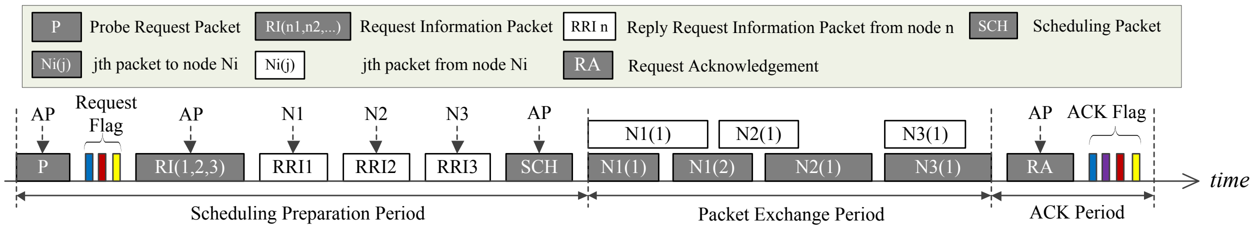


Figure 2: One round of Janus MAC packet exchanges.

The registration step is incorporated into the two stages. A new node tries to register and get an assigned request flag slot after capturing a Probe Request packet showing which request flag slots are used. Then, the node chooses an unused slot randomly, and participates in the second stage to declare that it has packets to send, using the chosen request flag. The registration succeeds if the node finds itself allowed to transmit data in the Scheduling packet. This is to prevent the case that multiple unregistered nodes chooses the same request flag slot. In this case, the Reply Request Information packet corrupts, and the AP will not allow any of them to transmit data after detecting the collision. So the nodes have to retry the registration.

The Reply Request Information packet contains two sets of data. The first is the lengths of all packets the node wants to transmit. The second is information about the interference the node experiences from its neighboring nodes. During the scheduling step the AP uses the packet length and interference data to determine the best full–duplex transmission schedule. That schedule is broadcast to all nodes in a packet, labeled 'SCH' in Figure 2.

After gathering all the scheduling information, Janus's scheduler will schedule the transmission as discussed in Section 5. The schedule result is broadcast in Scheduling packet, and nodes and the AP will transmit data packets accordingly.

4.1.2 Acknowledgement Period

Janus uses a batch acknowledgement mechanism specially designed for full–duplex communication. We choose this approach because sending an ACK immediately after receiving a data packet, like CSMA/CA does, adds additional constrains on Janus's full–duplex scheduling, and thus eliminates full–duplex transmission opportunities.

Janus introduces an acknowledge period at the end of each round, as shown in Figure 2. First, the AP broadcasts a Request Acknowledgement packet (RA), including the sequence number of all the packets that AP wants to acknowledge. After receiving the RA packet, a node can decide whether the packet it sends in the round is acknowledged and send ACK flags representing the packets it wants to acknowledge. In each round, each packet transmission from AP to node is assigned with a flag slot which is only valid for that round. The flag slot of a packet is assigned by AP, and indicated in the sequence number area of the data packet header. The single tone scheme can also be used here to save bandwidth, as in the scheduling preparation period.

Before discussing the scheduling portion of the Janus protocol, we describe how interference is measured and used.

4.2 Conflict Map

Both full–duplex opportunity and rate selection depends on interference information. A conflict map that describes the level of interference one node experiences when another node (including itself) is transmitting at the same time.

Former solutions describe interference in a coarse way. In CSMA, packets are allowed to transmit if the sender detects no interference. Otherwise, packet transmission is blocked. CSMA treats interference in a binary way, it is either complete or non-existent. In ContraFlow [13], the level of interference is abstracted as the probability that the interference can corrupt simultaneous full–duplex packet receiving. The probability is only measured in highest rate situation

Janus uses signal-to-interference ratio (SIR) to quantify the level of interference. So Janus can make scheduling decision according to the best understanding of underlying interference. CSMA or ContraFlow can measure SIR, but cannot always identify which node generates the interference since the interfering packet might be corrupt. Janus, as an AP-based MAC protocol, achieves this through AP indicating where the interference comes from.

Janus treats the conflict map as a two–dimension SIR matrix, $\{C_{j,k}\}$. $C_{j,k}$ refers to the ratio of the signal strength that Node j receives from AP and the interference strength that Node k generates. Assume N_1 receives packets from AP, and N_2 sends packets to AP at the same time, introducing interference. In this case, the SIR of the packet N_1 received is $C_{1,2}$. A special case is that the diagonal elements of C refer to the SIR when a node is sending and receiving at the same time, reflecting the full–duplex interference cancellation capability.

The conflict map offers sufficient information for Janus to choose full–duplex packet rates. To choose packet rates given SIR is a classical problem, and the result is just a mapping function from SIR to rate as shown in Table 2. For the mentioned case, N_1 chooses packet rate through mapping $C_{1,2}$ to the appropriate rate. Sometimes, there might be multiple nodes generating interference when N_1 receives packets. For example, if N_3 also sends packets to AP simultaneously, N_1 uses the minimum SIR of $C_{1,2}$ and $C_{1,3}$ for mapping, since the packet rate should be chosen to tolerate the highest interference.

Janus transforms every SIR element in the conflict map to the corresponding rate, and constructs a rate matrix. This rate matrix is equivalent to the conflict map for scheduling, but is easy to calculate and transmit. Janus's scheduler only uses rate matrix rather than processing conflict map directly.

Janus constructs the conflict map through the cooperation of nodes and AP in the second stage of information exchange, and doesn't introduce additional packets as overhead. First, each node listens to the Request Information Packet. It measures the signal strength, and identify when other nodes send in the following time slots. Second, each node listens to the Reply Request Information Packet of every other nodes. It measures the interference strength from them. Last, the node can calculate SIR, dividing the signal strength by the interference strength. After each node calculates the line of the conflict map corresponding to it, all the lines are gathered at the AP side in the Reply Request Information Packet of next round as mentioned in Section 4. The conflict map is measured in every round since the conflict map might change over time.

5 Full-Duplex Scheduling

This section discusses the challenges of designing a full–duplex scheduler and how Janus addresses them. The Janus scheduler aims to maximize full–duplex transmission opportunities, while maintaining fair channel access for all nodes. To this end, Janus has two separate mechanisms to guarantee fairness and bounded latency, while improving throughput by leveraging simultaneous transmissions. First, there is a Load Controller Unit (LCU), which uses the collected packet length requests to determine how long each node can send for. Then, the Rate-Timing Allocator (RTA) uses the conflict map and rate matrix to determine the order and data rates at which nodes will transmit.

5.1 Load Controller Unit (LCU)

The Load Controller Unit (LCU) is responsible for enforcing fairness and for providing guaranteed bounds for the overall latency of the system. Janus's metric for fairness is channel access time; the amount of data that each node will be able to transmit will depend on the channel quality between the AP and that node. In prior MAC protocols based on CSMA/CA, each node cooperates in achieving fairness by using random back-offs. Since decision making in Janus is centralized at the AP, channel access time can be accurately divided based on the global traffic information.

Janus uses the idea of Deficit Round Robin (DRR) [12] which is a more practical version of Fair Queuing [4]. In fair queuing, the completion time of head-of-the-line packets in all queues is calculated and the packet with the shortest completion time will get the channel. This results in perfect fairness, at the cost of high computation overhead. Instead, DRR serves all queues in a round robin manner and in each cycle allocates a deficit to each queue. Any transmission would be deducted from that deficit, and a queue's access share cannot exceed its deficit. If the deficit is unused due to an empty queue, it expires. However, if it is unused due to a long packet that did not fit in the slot, the deficit would be transferred to the next round; an effort to resemble the behavior of fair queuing in postponing longer transmissions.

The cyclic behavior of DRR perfectly matches Janus's. However, conventional DRR works based on the number of bits served from each queue, while we need to take into account the effect of channel conditions in the design. In other words, the fact that one of the nodes has a poor channel condition should not suffer all the other nodes in the network. So, Janus enforces fairness based on the channel access time.

To this end, Janus AP sets a value for the time share in each round n, say $T_{share}(n)$, and advertises it in probe. Then, each node will update its current deficit counter, $T_{deficit}(n)$, with the following equation:

$$T_{deficit}(n) = T_{share}(n) + T_{deficit}(n-1) \times I(n-1) \tag{1}$$

where $I(n)$ is 0 if the queue becomes empty in round n, or 1 if the deficit goes idle since the packet could not fit in the slot. Then, nodes start to deduct as many packet transmission times as possible from the allocated deficit and inform the AP about these packet lengths (as discussed in Section 4). Specifically, assume that P_{ij} is the length of the j^{th} packet in bytes from the head of the queue of the i^{th} node, and R_i is the transmission rate between AP and node i based on the previous round estimation. If

$$\frac{1}{R_i}\sum_{j=1}^{k} P_{ij} \quad < T_{deficit}(n) < \quad \frac{1}{R_i}\sum_{j=1}^{k+1} P_{ij} \tag{2}$$

then, node will announce $\{P_{i1}, P_{i2}, \ldots, P_{ik}\}$ to the AP and updates the deficit accordingly:

$$T_{deficit}(n) = T_{deficit}(n) - \frac{1}{R_i}\sum_{j=1}^{k} P_{ij} \tag{3}$$

Similarly, the AP performs the same calculations for the outgoing queues.

Note that in full–duplex PHY one node may suffer from the interference caused by another node since they are scheduled to transmit/receive simultaneously. Janus makes sure that channel access time for each node is allocated based on the best rate that it could have transmitted if the channel were allocated exclusively to that node.

Last, it is worth mentioning that AP can control latency by tuning the $T_{share}(n)$. The amount of time it takes to complete each round of packet exchange is directly related to the transmission times allocated to each queue. So, if the $T_{share}(n)$ is small the network would be more agile in responding to newly arrived packets. On the other hand, it is desirable to have long packet exchange period compared to the overhead of each cycle of protocol. This trade-off between throughput and latency is discussed more in later sections.

5.2 Rate-Timing Allocator (RTA)

After the Load Controller Unit (LCU) has determined the packets that should be served from each queue, it is the job of Rate-Timing Allocator (RTA) to select the order in which queues will be served. Moreover, since simultaneous transmissions could cause interference at the secondary receiver, this unit finds the appropriate rate for the communications based on possible conflicts. The goal is to find the best matches with potential full–duplex cooperations to maximize the overall throughput.

5.2.1 Problem Statement

Assume that there are k nodes registered with the AP. Let I_j and O_j for $1 \leq j \leq k$ be the aggregate length of packets in bytes that the AP should receive from and send to node j in current round, respectively. Janus serves all packets from one queue back-to-back and so, all packets in I_j (O_j) are considered as a single entity. This itself is enough to achieve acceptable gain without scheduling each packet individually. The intuition behind it is that the LCU tends to make the transmission time of I_js and O_js comparable based on allocated deficit and so simultaneous transmissions tends to have comparable length. To simplify the discussion in the following, transmissions from AP to nodes and from nodes to AP are called *outgoing* and *incoming*, respectively.

Since the total bytes to transmit in each round is fixed, shortest completion time results in maximum throughput. RTA tries to decrease the completion time by forming the best full–duplex matches from the incoming and outgoing queues. Finding the shortest completion time is an optimization problem with following constraints:

- The transmission rate of I_j is fixed and is equal to the maximum available bandwidth from node j to the AP as if channel was exclusively allocated. This is possible since the AP can fully cancel its own self–interference.
- The transmission rate of O_j is determined by the highest interference caused from matched I_j's. Note that one queue could be matched with multiple queues.

This optimization problem is NP-complete; Appendix A shows a derivation from the Hamiltonian cycle problem. Therefore, even for a moderate number of nodes it is infeasible to get the optimal solution in a reasonable time. We propose a heuristic algorithm which converges fast, has low computational complexity, and results in gains almost as good as the best case (Section 7.)

5.2.2 Janus Scheduler

Janus matches incoming and outgoing queues for simultaneous transmissions step by step. It is possible to have a case in which full–duplex is an option, but the data rate would be so low that its performance will be worse than a higher rate half–duplex transmission. Thus, the Janus scheduler considers both half– and full–duplex candidates for each queue.

In order to simplify the explanation of the decision making process a couple of metrics are introduced. Janus proposes a new metric called *Lingering Factor (LF)* as

$$LF(O_m) = O_m \left[\frac{1}{R_{new}} - \frac{1}{R_{old}} \right] \tag{4}$$

where, R_{new} and R_{old} refer to the new and old rate of the outgoing queue, O_m, involved in current cooperation, respectively. This factor calculates how longer the outgoing queue, O_m is made because of the interference caused by the current match from incoming queue.

As mentioned earlier, we are also interested in the difference in completion time $\Delta T_{completion}$ resulting from forming a full–duplex match. This value can be computed as:

$$\Delta T_{completion} = T_{full-duplex} - LF(O_m) \tag{5}$$

where, $T_{full-duplex}$ is the amount of time the incoming and outgoing channel overlap due to current match. The value could be positive or negative for a full–duplex candidate.

Although the ultimate goal is to minimize the overall, $\Delta T_{completion}$, it is not good to aim for the best $\Delta T_{completion}$ in each round. In fact, since this optimization is local, there could be decisions that harm the overall performance. While, one candidate could results in more reduction in completion time in current step, it could degrade an output queue rate severely (poor *Lingering Factor (LF)*), or prevent forming a better match in later step. The Janus scheduler tries to address the deficiencies in this local optimization.

The next step is to schedule the queues. Starting from all unscheduled queues, RTA fixes one of the incoming queues to be served first. Initially completion time is calculated with the assumption that all queues are scheduled using half–duplex and transmitted with the highest available rate. At every step, the total completion time and rates are modified, resulting in one of the following situations:

- **The incoming channel is longer than outgoing.** Calculate $\Delta T_{completion}$ for all unscheduled outgoing queues. If none of them is positive then let the current incoming queue finish with no more matches. Otherwise, among those with positive $\Delta T_{completion}$ choose one with the lowest LF and schedule it with appropriate rate.
- **The outgoing channel is longer than incoming.** Calculate $\Delta T_{completion}$ for all the unscheduled incoming queues. If none of them is positive then let the current outgoing queue finish with no more match. Otherwise, make a list of candidates with positive $\Delta T_{completion}$. For

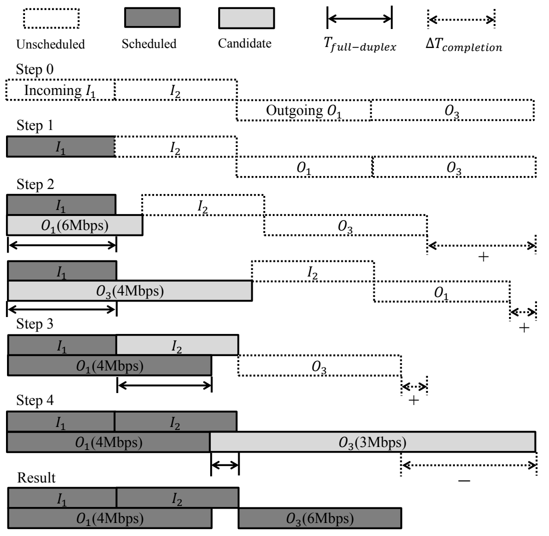


Figure 3: An example of the scheduling algorithm. $\Delta T_{completion}$ and $T_{full-duplex}$ are the difference in completion time and the amount of overlap between incoming and outgoing channel caused by current full–duplex candidate, respectively.

each incoming queue in the list consider the interference it could induce on current outgoing queue. If it can induce a less amount of interference on an unscheduled outgoing queue, then eliminate it from the list. This is an attempt to save better matches for future. If the list is empty, let the current outgoing queue finish with no more matches. Otherwise choose one with the lowest LF and schedule it. Also update the rate of current outgoing queue according to the new interference level.

- **The incoming and outgoing channels are equal:** If there are still unscheduled incoming queues, schedule one randomly. Otherwise, schedule all the remaining outgoing queues using half–duplex.

This process iterates until all queues are scheduled. Next, we walk through an example of the scheduling algorithm.

5.2.3 Example Scenario

In the example in Figure 3, Janus has two incoming queues, I_1, I_2, and two outgoing queues, O_1, O_3 to serve. To be simple, all the incoming queues are assumed to have the same transmission rate of R_{in}. The half–duplex transmission rate, R_{half}, is also chosen to be the same for all queues. $R_{O_j \leftarrow I_k}$ is the maximum transmission rate of outgoing queue O_j when incoming queue I_k is transmitted simultaneously. Table 1 shows the value of the parameters used in this example. At each step, queues could have three different states:

- **Scheduled:** A queue is scheduled, but the rate of a scheduled outgoing queue could change later due to possible simultaneous incoming transmissions;

	R_{in}	R_{half}
Rate(Mbps)	6	6

	$R_{O_1 \leftarrow I_1}$	$R_{O_1 \leftarrow I_2}$	$R_{O_3 \leftarrow I_2}$	$R_{O_3 \leftarrow I_2}$
Rate(Mbps)	6	4	4	3

	I_1	I_2	O_1	O_3
Queue Length (Bytes)	800	900	1000	1200

Table 1: The parameter setting of the example. R_{in} is the rate of incoming queues. R_{half} is the rate for half–duplex transmissions. $R_{O_j \leftarrow I_k}$ is the maximum transmission rate of outgoing queue O_j when incoming queue I_k is transmitted simultaneously.

- **Unscheduled:** The queue has not been considered yet. The scheduler assumes the queue is transmitted by half–duplex just to keep track of the worst transmission completion time up to this step;
- **Candidate:** The queue is being considered in the current step.

At the very beginning of the round, all queues are unscheduled in Step 0. In Step 1, the scheduler chooses one incoming queue, e.g., I_1, randomly, and schedules it to transmit at the start of the round.

In Step 2, the scheduler is faced by three choices, choosing O_1 or O_3 as the next outgoing transmission by full–duplex, or letting I_1 finish without any match. Since, $\Delta T_{completion}$ cause by both full–duplex choices is positive then, half–duplex option is ignored. In this case, $LF_{step2}(O_1) = 0$, and $LF_{step2}(O_3) = 800\mu s$, and so O_1 is scheduled as the next outgoing queue.

In Step 3, now the outgoing transmission is longer than the incoming one and the scheduler can either choose to transmit I_2 by full–duplex or let O_1 finish without any match. Again, since $\Delta T_{completion}$ caused from matching I_2 is positive, it could be a candidate. Janus further investigate other possible matches for I_2. Since $R_{O_1 \leftarrow I_2} > R_{O_3 \leftarrow I_2}$, we decide to schedule I_2 in current round. Notice, the rate of outgoing queue O_1 is adjusted here, since it is influenced by the simultaneous transmission of I_2.

Step 4 is similar to Step 3. But this time, $\Delta T_{completion}$ is negative for the full–duplex option so, the scheduler chooses to let the rest of I_2 finish without match and starts transmitting O_3 using half–duplex, afterward. After Step 4, all the queues are fixed and the final result is achieved.

6 Implementation

This section discusses several Janus implementation challenges. Janus runs on the WARP v2 platform [2] from Rice University. The physical and MAC layers are based on the v.16.01 real-time OFDM reference design with channel coding support. It provides 10Mhz bandwidth with WiFi-like packet format. The design provides Viterbi decoder with

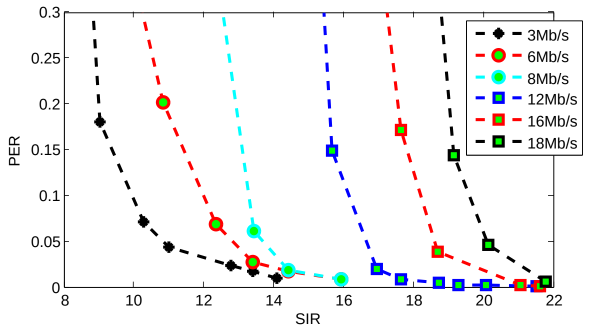


Figure 4: Packet Error Rate vs SIR with 50000 iterations: packet length is 1400 bytes and interfering packet length is 100 bytes back-to-back transmitted with 20 us interval.

Rate (Mbps)	3	6	8	12	16	18
SIR (dB)	10	12.3	13.4	16.2	18.3	19.6

Table 2: SIR to rate matrix conversion table:SIR thresholds are determined at PER value of 0.1**.**

three code rates and supports up to nine data rates with combination of three modulations. Among them, Janus chooses six data rates for rate adaptation, shown in Figure 4.

Physical Layer. Since this paper focuses on the MAC layer, Janus assumes an existing real-time full–duplex physical layer. To mimic that, we use separate frequency bands for the transmission (TX) and receive (RX) channels and assume that residual self-interference is negligible. However, nodes cannot overhear packet transmissions from neighboring nodes if the RX and TX frequencies are different. In order to have real interference in the network during experiment, this problem must resolved.

The solution is to let a node transmit packets on both the TX and RX channels simultaneously only when the node does not do full–duplex with AP. Let's assume AP and N_1 are doing full–duplex transmission and reception. After finishing AP transmission, if N_1 is still transmitting packets, AP can send packets to N_2 according to the scheduling map. During AP and N_1 full–duplex, transmission on RX channel should be muted. As soon as AP ends its transmission, N_1 should turn it on to give interference on N_2 receiving. Since AP have all the necessary information when nodes should mute or unmute transmission on the RX channel, it only needs to add muting and unmuting time for each node at the scheduling packet.

Single Tone Detection: Janus proposes single tone transmission and detection scheme for control packets and acknowledgements. However, single tone scheme is not implemented in Janus by the following reason. Single tone scheme does not provide much reduction of MAC overhead compared to simple alternative ones for small number of nodes. Current experimental setup described at the Section 7.1 uses three nodes which is small enough not to give much differentiation. However, single tone scheme will significantly reduce MAC overhead as the number of clients increases. We let single tone implementation as future work to evaluate Janus scalability by increasing the number of clients.

Simple alternative acknowledgement scheme is used for Janus implementation. When AP sends a Request ACK (RA) packet, RA contains not only sequence numbers for acknowledgements to nodes, but also list of node numbers AP wants to get acknowledgements from. Each node sends its ACK packet on the order of the list. Since ACK packets from nodes has fixed data rate and length, nodes can easily calculate transmission time of ACK. This scheme increases MAC overhead compared to single tone scheme but, it is still more efficient acknowledgement scheme than individual packet acknowledgements like CSMA/CA.

Single tone reply for Probe is also implemented as control packet exchanges. After nodes receive a Probe packet, each node sends a control packet with the packet request indicator flag. If nodes have packets to send, they set this flag to be one. AP will include node numbers replying with one in a RI packet.

Packet Detection

The Janus MAC requires nodes to be able to detect AP packets even under interfering node transmission. Since interfering packets have the correct packet format, a node cannot distinguish them before the header is decoded. This false detection of an interfering packet will cause the node to miss the desired packet destined for it. For example, if the interfering packets come first and desired packets next, nodes will detect interfering packets and start to decode them. When the real packet comes, the node will not detect it since it is in the middle of a packet reception.

Since the AP scheduler does not allow any packet transmissions under severe interference, we can assume that the packet coming from the AP is always stronger than that of interfering packet. If packet detector is smart enough to track higher packet detection, packet missing rate from false detection could be significantly reduced.

Again, since our focus is not PHY, we chose simple solution. We fixed AGC (Automatic Gain Control) with fixed value. With proper antenna placements, desired signal strength could be in good range of reception. By adjusting tx powers of interfering node, its received power can be adjusted. Finally, set the packet detector threshold to the value which is high enough to reject interfering signal. Since our experiment environment is highly static, channel variation is small enough to have static packet detector threshold during the experiment.

7 Evaluation

This section examines the performance of Janus under several different topology scenarios and traffic patterns. The two metrics used to evaluate the MAC protocol are MAC throughput and fairness.

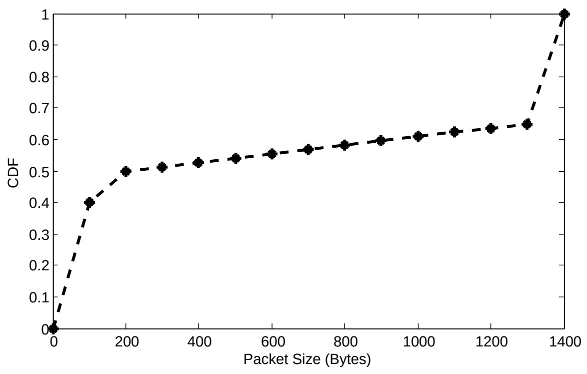


Figure 6: Packet length distribution for traffic generation.

	N_1	N_2	N_3
TR1 incoming	100%	100%	100%
TR1 outgoing	100%	100%	100%
TR2 incoming	100%	0%	50%
TR2 outgoing	0%	100%	50%
TR3 incoming	80%	20%	60%
TR3 outgoing	20%	80%	40%

Table 3: Three Different Traffic Types

The setup of all experiments involves one node acting as the access point (AP) and three others acting as clients. Each node is a WARP board running the Janus implementation.

7.1 Experimental Setup

The amount of interference between nodes affects Janus's full–duplex packet scheduling. In other words, depending on the non-zero values on off-diagonal terms in the conflict map, the degree of freedom on full–duplex packet scheduling changes. Therefore, we carefully select several different scenarios to show distinctive scheduling gain.

Three scenarios are chosen with different levels of interference. Figure 5 (a) depicts weak interference between N_1 and other nodes. It makes possible to assign high data rates when scheduler makes full–duplex pair with N_1 and N_2 or N_1 and N_3. For example, if N_2 can receive 18 Mbps packets from AP, scheduler can make N_1 and N_2 be full–duplex pair by assigning same 18Mbps for N_2 packets or, at least, next highest rate such as 16Mbps. Figure 5 (b) is different from (a) in that interference level between N_1 and N_2 increases. Due to increased interference, scheduler is expected to less choose N_1 and N_2 for full–duplex pair which leads to reduced throughput. In the Figure 5 (c), every nodes interfere to each other with significant level of interference. For the example above, scheduler cannot make N_1 and N_2 for full–duplex pair without significantly lowering data rate of N_2 packets.

The second ingredient to the Janus experiments is the traffic setup. Figure 6 shows packet size distribution derived from cumulative IPv4 packet traces [1]. Whenever a new packet is generated from AP or nodes, packet size is generated by the distribution shown in Figure 6. Another important factor for traffic generation is traffic loading. 100% traffic loading means transmit buffers are always fully queued and there's no limit to transmit packets. By changing traffic loading, buffers can be queued only with fractional time of the experiments. Different traffic loading is important to see how scheduler smartly makes full–duplex pairs between different nodes because full–duplex between AP and node is not always possible. Table 3 summarizes the loading factors for the three traffic types. TR1 is conventional full queue model balanced between outgoing and incoming packets. TR2 and TR3 give asymmetric packet loading between outgoing and incoming traffics for N_1 and N_2.

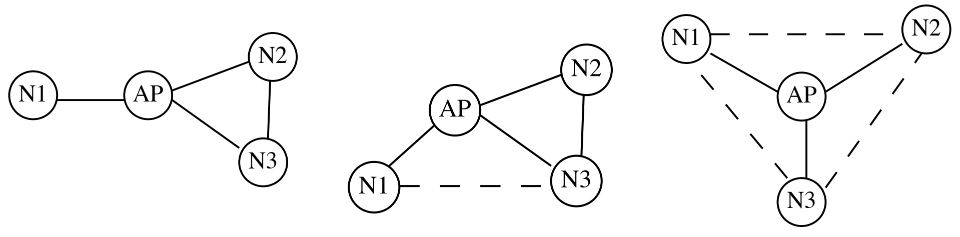
Finally, in order to decide what is the optimal data rate for given SIR, packet error rates (PER) for each data rate are measured under presence of interfering packets. Figure 4 shows packet error rate for different data rates when there are underlying interfering packet transmissions. Interfering packets are transmitted back to back with approximately 20us interval with 100 bytes payload. From experiment, PER increases when interfering packet length is shorter. We can guess that higher frequency of appearance and disappearance of interfering packets gives more negative effects to decoding desired packets. Transmitted desired packet length is 1400 bytes and also from experiment, longer packet length increases packet error rates. Since packet length of traffic patterns throughout all the evaluation of Janus ranges from 100 to 1400 bytes, Figure 4 shows worst performance of PER on WARP platform. For stable operation of Janus, rate matrix conversion is based on this PER curve. Target PER for each data rate is set to be 0.1 and each SIR threshold for rate conversion is shown in Table 2.

	Overhead	Per Packet Overhead
CSMA/CA	$279us$	$279us$
Janus	$2530us$	$69us$

Table 4: One round of MAC overhead time for Janus and CSMA/CA: overhead for CSMA/CA includes backoff and packet acknowledgement time. For Janus, MAC overhead counts time spent for sending control packets like Probe, RI, RRI, scheduling packet and acknowledgements. Each value is averaged over 1000 iterations.

7.2 Janus vs Half-Duplex

In this section, Janus and half–duplex throughput are compared. Figure 7 shows throughput under increasing traffic loads. Traffic type used in this experiment gives same traffic loading to incoming and outgoing packets. For 100% loading, it is same as TR1. Under symmetric traffics like TR1, Janus does not show significant throughput changes between different scenarios and ,thus, only throughput for S3 is dis-



(a) Scenario 1 (S1) (b) Scenario 2 (S2) (c) Scenario 3 (S3)

Figure 5: Different Scenarios for conflicting nodes. The dashed lines means high interference. without much lowering data rate, full–duplex communication is not possible. The solid line shows no full–duplex communication is possible even with lowest data rate. If there is no line between two nodes then the interference is negligibly weak.

played in the Figure 7. To compare MAC overhead between Janus and half–duplex, two separate throughput values for each system are provided: One is throughput considering MAC overhead time and for the other, MAC overhead time is removed for throughput calculation. In the half–duplex CSMA/CA case, the overhead is time wasted to back-offs and acknowledgements. In Janus' case, it is the control packets that are transmitted at the beginning and end of each round.

Janus' throughput at 100% loading reaches 22.847 Mbps, while half–duplex counterpart is 12.138 Mbps if MAC overhead time is not considered. Throughput gain over half–duplex for this case is 1.9 times that of half–duplex. Since incoming and outgoing packets for each node-AP pair are symmetrically available for each round, scheduler can easily make full–duplex matches with high channel utilization. Interesting results are throughput including MAC overhead time. Janus throughput with MAC overhead is 2.5 times half–duplex counterpart. It can be justified by comparing per-packet overhead for Janus and CSMA/CA. Second column of the Table 4 shows average single round overhead time measured from the experiment in Figure 7. From the Table 4, single round of Janus gives higher overhead than CSMA/CA because of RRI transmission from nodes and a scheduling packet from AP. However, if this is converted to per-packet overhead time, third column of the Table 4 shows Janus gives much smaller overhead than CSMA/CA. For each round of Janus, multiple packet transmissions are available from nodes and AP according to $T_{share}(n)$, whereas single packet transmission for each round of CSMA/CA. With fine tuning of $T_{share}(n)$ discussed at Section 5.1, Janus can maintain low MAC overhead with tolerable latency.

7.3 Scheduler Gain

This section compares Janus throughput under different scenarios and different traffic types. Five throughput values for each traffic type in the Figure 8 come from different scenarios. S1,S2 and S3 are scenarios described in the Figure 5. There are two additional scenarios for upper and lower bound of throughput for each traffic type. The Best describes

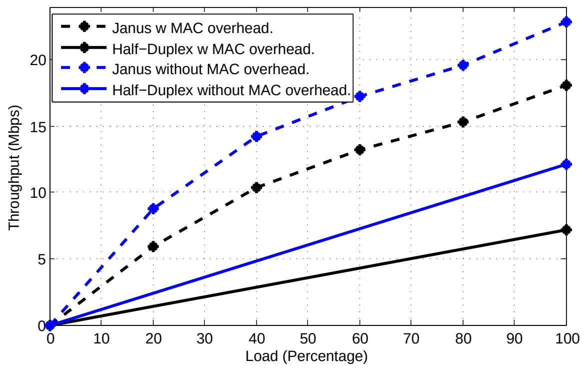


Figure 7: Load vs throughput curve comparing Janus and Half-Duplex System: for each system, two throughput values are displayed in the figure. One is the throughput including MAC overhead and the other is not considering MAC overhead.

a scenario when every node does not give any interference to others and they are completely hidden to each other. On the contrary, the Worst represents every node gives severe interference to others and no full–duplex communication between nodes is possible. Off-diagonal terms in conflict map is zero for the Worst and highest possible rate for the Best.

Left five throughputs in the Figure 8 are measured under TR1 traffic. As mentioned at the Section 7.2, Janus scheduler mostly makes full–duplex matches between AP and node when incoming and outgoing traffic loading is balanced. Since no need for extra efforts to find further full–duplex opportunities, scheduler does same scheduling for all the scenarios which lead to almost same throughput.

However, each scenario shows significantly different throughput under TR2 and TR3. N_1 only sends packets and N_2 always receives packets for TR2, which makes full–duplex between AP and N_1 or AP and N_2 not possible. Unless scheduler finds full–duplex matches with other nodes, every transmission involving N_1 and N_2 should be half–duplex. Due to weak interference between N_1 and N_2 for S1, scheduler can easily construct full–duplex pairs between them and gives almost as good throughput as the Best sce-

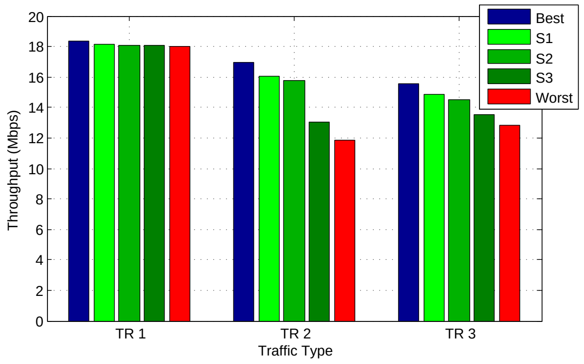


Figure 8: Throughput comparison between different scenarios under three traffic types.

nario.

For the S2, one change from S1 is increased interference between N_1 and N_3 which makes scheduler less choose them as a full–duplex match. Since N_1 and N_2 are dominant half–duplex candidates in TR2, interference increasing between N_1 and N_3 does not give much impacts on the throughput. Figure 8 shows that throughput values between S1 and S2 for TR2 does not have much difference as expected.

For S3, in order to form full–duplex pairs between different nodes, data rate for outgoing packets should be significantly decreased to combat high interference. One thing to note is that S3 throughput has still 10% gains over the Worst scenario which does not allow any full–duplex matches between different nodes. It suggests that even under high interference environment, leveraging full–duplex opportunities by matching different nodes still give additional gains.

Throughput comparisons under TR3 show same trends as TR2 with reduced throughput for all five scenarios. Reduced throughput is mainly caused by decreased traffic loadings of N_1 incoming and N_2 outgoing traffics which makes scheduler difficult to construct a full–duplex pair between N_1 and N_2.

7.4 Fairness

This section evaluates the fairness metric in the system under different traffic types. As described in Section 5.1, the LCU enforces fairness among queues. The deficit constant, T_{share}, is set to be 3ms for TR1 and 6ms for TR2 and TR3, respectively. Table 5 shows channel access time ratios between three nodes in S3.

It is worth mentioning two points. First, every queue has potentially equal channel access time to use regardless of their traffic loads or channel conditions. The fact that how much of this time is used depends on the available traffic in the queue. Second, the LCU guarantees that the amount of packets in bytes transmitted in each time share is calculated based on the highest available rate. RTA attempt in leveraging full–duplex opportunity could cause an outgoing queue to have a lower rate and so longer access time. However, the actual number of bytes have been already determined by LCU and they are only transmitted in a lower rate.

	N_1	N_2	N_3
TR 1 incoming	33.3%	33.4%	33.3%
TR 1 outgoing	33.3%	33.4%	33.3%
TR 2 incoming	67.0%	0%	33.0%
TR 2 outgoing	0%	60.2%	39.8%
TR 3 incoming	47.1%	15.8%	37.1%
TR 3 outgoing	11.2%	67.5%	21.3%

Table 5: Percentage of channel access time for different queues under various traffic types in S3.

Let's take a look at the results. TR1 provides balanced 100% traffic loadings over all the queues and so fair scheduler should share the channel access time equally among them–consistent with the result in Table 5.

Traffic types TR2 and TR3 are more interesting. Since traffic loadings of incoming and outgoing for TR2 and TR3 are different for each queue, final channel access time should be proportional to the loading ratios. As shown in Table 5, the incoming channel for both TR2 and TR3 is divided among the different queues almost proportional to the loading ratios; 67%,0% and 33% for TR2 and 50%,12.5% and 37.5% for TR3.

However, the outgoing channel share is a little bit different than the ideal result in both cases. This minor difference comes from the fact that the rate of outgoing queues could be affected by the RTA and some transmissions could be lingered due to lower assigned rate in the scheduling. But, by no means it means that the queue with more access tie is getting more than its fair share. The fair share has been already enforced from LCU.

8 Conclusion and Future Work

Janus is a novel AP–based MAC protocol, specifically designed for full–duplex wireless networks. Its centralized nature allows the AP to collect interference information from nodes and schedule transmissions in a way that removes collisions. Based on the interference data, the Janus scheduler chooses from multiple available packet data rates in order to maximize full–duplex opportunities. Experiments with three nodes and one AP, on the WARP hardware, show that Janus can achieve 2.5X throughput improvement over half–duplex, due to simultaneous transmissions and reduced per-packet overhead.

Janus is under active development and testing, with future work including larger, more diverse experiments. We are studying the effect of the Janus round length on latency and throughput. Additionally, increasing the size of the network and trying different interference topologies and traffic types can further our understanding of MAC protocol performance under full–duplex conditions.

A Full-duplex scheduling NP-C proof

The NP-completeness(NP-C) proof of Janus's full–duplex scheduling problem is derived from the problem of determining the existence of a Hamiltonian cycle in a given undirected bipartite graph, which has been shown to be NP-C. Let us denote this problem as P_1.

Starting from P_1, we claim that the problem of determining the existence of a Hamiltonian path with fixed start vertex and end vertex in a given undirected bipartite graph, say P_2, is NP-C. This can be easily proved by seeing that given an edge in a bipartite graph, there exists a Hamiltonian path traversing from one vertex of the edge to the other, if and only if there exists a Hamiltonian cycle with the edge in it.

Then, we show that a special case of Janus's full–duplex scheduling problem to maximize throughput is equivalent to P_2. This proves that the scheduling problem is also N-PC.

For a bipartite graph with N nodes $u_1, u_2, \cdots, u_N$ in one bipartition U, and $N+1$ nodes $v_1, v_2, \cdots, v_{N+1}$ in the other bipartition V, we form an equivalent case of Janus's full–duplex scheduling problem as follow. Assume Janus has N incoming queues, $I_1, I_2, \cdots, I_N$, and $(N+1)$ outgoing queues, $O_1, O_2, \cdots, O_{N+1}$, to schedule in a round. The length of an incoming queue is $(N+1)$ bytes, and the length of an outgoing queue is N bytes. The rates of all the outgoing packets without simultaneous incoming packet transmission, and the rates of all the incoming packets are 1 byte per second. Packets in the queue I_k and O_j can be transmitted simultaneously in full–duplex with rate 1 byte per second for the outgoing packets in queue O_j, if there is an edge connecting u_k and v_j. Otherwise I_k and O_j cannot be transmitted simultaneously.

It can be verified that there is a Hamiltonian path traversing from v_1 to v_{N+1} if and only if Janus can schedule all the packets within $[(N+1) \cdot N]$ seconds with O_1 scheduled at the beginning and O_{N+1} at the end.

A deviation here is that we assume one bipartition of graph has exactly one more vertex than the other, and the first outgoing packet and the last one are fixed in Janus's schedule. But it can be easily fixed by adding dumb vertexes and enumerating the first and the last scheduled packets.

In conclusion, Janus's full–duplex scheduling problem to maximize throughput is NP-C.

References

[1] Packet size distribution comparison between Internet links in 1998 and 2008. `www.caida.org/research/traffic-analysis/pkt_size_distribution/graphs.xml`.

[2] Rice University, WARP Project. `http://warp.rice.edu`.

[3] J. I. Choi, M. Jain, K. Srinivasan, P. Levis, and S. Katti. Achieving Single Channel, Full Duplex Wireless Communication. In *Proceedings of the 16th Annual International Conference on Mobile Computing and Networking, MobiCom'10*. ACM, 2010.

[4] A. Demers, S. Keshav, and S. Shenker. Analysis and Simulation of a Fair Queueing Algorithm. In *ACM SIGCOMM Computer Communication Review*, volume 19, pages 1–12. ACM, 1989.

[5] P. Dutta, S. Dawson-Haggerty, Y. Chen, C.-J. M. Liang, and A. Terzis. Design and Evaluation of a Versatile and Efficient Receiver-initiated Link Layer for Low-power Wireless. In *Proceedings of the 8th ACM Conference on Embedded Networked Sensor Systems*, pages 1–14. ACM, 2010.

[6] J. J. Garcia-Luna-Aceves and A. Tzamaloukas. Reversing the Collision-avoidance Handshake in Wireless Networks. In *Proceedings of the 5th International Conference on Mobile Computing and Networking (MobiCom'99)*, pages 120–131, 1999.

[7] S. S. Hong, J. Mehlman, and S. Katti. Picasso: flexible rf and spectrum slicing. In *Proceedings of the ACM SIGCOMM 2012 conference on Applications, technologies, architectures, and protocols for computer communication*, pages 37–48. ACM, 2012.

[8] M. Jain, J. Choi, T. Kim, D. Bharadia, S. Seth, K. Srinivasan, P. Levis, S. Katti, and P. Sinha. Practical, Real-time, Full duplex Wireless. In *Proceedings of the 17th Annual International Conference on Mobile Computing and Networking, MobiCom'11*, pages 301–312. ACM, 2011.

[9] A. Sahai, G. Patel, and A. Sabharwal. Pushing the Limits of Full-duplex: Design and Real-time Implementation. *arXiv preprint arXiv:1107.0607*, 2011.

[10] S. Sen, R. Roy Choudhury, and S. Nelakuditi. No Time to Countdown: Migrating Backoff to the Frequency Domain. In *Proceedings of the 17th annual international conference on Mobile computing and networking*, pages 241–252. ACM, 2011.

[11] S. Sen, R. Roy Choudhury, and S. Nelakuditi. CSMA/CN: Carrier Sense Multiple Access with Collision Notification. *Networking, IEEE/ACM Transactions on*, 20(2):544–556, 2012.

[12] M. Shreedhar and G. Varghese. Efficient Fair Queueing Using Deficit Round Robin. In *ACM SIGCOMM Computer Communication Review*, volume 25, pages 231–242. ACM, 1995.

[13] N. Singh, D. Gunawardena, A. Proutiere, B. Radunovic, H. Balan, and P. Key. Efficient and Fair MAC for Wireless Networks with Self-interference Cancellation. In *Modeling and Optimization in Mobile, Ad Hoc and Wireless Networks (WiOpt), 2011 International Symposium on*, pages 94–101. IEEE, 2011.

[14] O. G. Y. Sun and D. B. Johnson. RI-MAC: A Receiver-initiated Asynchronous Duty Cycle MAC Protocol for Dynamic Traffic Loads in Wireless Sensor Networks. In *Proceedings of the 6th International Conference on Embedded Networked Sensor Systems (Sensys'08)*, pages 1–14, 2008.