Design and Evaluation of a new MAC Protocol for Long-Distance 802.11 Mesh Networks*

Bhaskaran Raman†
Department of CSE
IIT Kanpur
Kanpur, INDIA 208016
braman AT iitk DOT ac DOT in

Kameswari Chebrolu
Department of EE
IIT Kanpur
Kanpur, INDIA 208016
chebrolu AT iitk DOT ac DOT in

ABSTRACT

802.11 has been used well beyond its original intended use of WLANs. Of particular interest to us in this paper is its use in long-distance mesh networks being designed/used for low-cost rural connectivity. We describe in detail a new MAC protocol, called 2P, that is suited for such networks in terms of efficiency. A significant challenge here is the implementation of this protocol on top of off-the-shelf 802.11 hardware, to preserve the cost benefits. We show how this can be achieved, by exploiting the flexibilities available within Prism2-based chipsets. We then present the dependence of 2P on the network topology, and show that it is indeed possible to design in practice, network topologies compatible with 2P. We describe experimental as well as simulation-based evaluations of 2P, and show that 2P achieves significant performance improvement (as much as 20 times more throughput) over 802.11 CSMA/CA in long-distance mesh networks.

Categories and Subject Descriptors

C.2.1 [**Network Architecture and Design**]: Wireless communication; C.2.1 [**Network Architecture and Design**]: Network topology

General Terms

Design, Performance

Keywords

802.11 mesh networks, MAC design, Signal-to-Interference Ratio, Network topology design

*This work was supported by Media Lab Asia, IIT Kanpur.
†Part of this work was done when the author was a visiting researcher at CalIT2, UCSD.

MobiCom'05, August 28–September 2, 2005, Cologne, Germany.

1. INTRODUCTION

Although originally designed to be a wireless replacement of LANs, IEEE 802.11 [6] has been used for other purposes involving multi-hop mesh networks [2, 14]. Of specific interest to us in this paper are those which involve long-distance links, envisioned for providing *low-cost* Internet access to rural locations [3, 10, 9].

The 802.11 CSMA/CA MAC was designed to resolve contention in indoor conditions. It is inefficient in mesh networks, especially those with long-distance links. CSMA/CA suffers from unnecessary contention resolution in a point-to-point link, delays due to the larger round-trip time, and even ACK timeouts on the long-distance links [10, 9]. However, use of other technologies such as wire-line, cellular, or 802.16 for such access networks is currently prohibitive in terms of cost [10]. We seek to retain the cost advantages of 802.11, and improve its performance by redesigning the MAC.

The mesh networks we consider are as in [3, 10, 9]. Fig. 1 depicts a schematic of such networks. These networks have three prominent characteristics of relevance to us: (a) multiple radios per node, (b) use of high-gain directional antennae, and (c) long-distance point-to-point links (several kilometres). These factors distinguish these networks from most networks considered in prior literature on multi-hop 802.11 networks. Furthermore, in these networks, we have one node (or perhaps more) which has wired Internet access, called the *landline node*. The landline node provides Internet connectivity to the other nodes (village locations) in the wireless mesh.

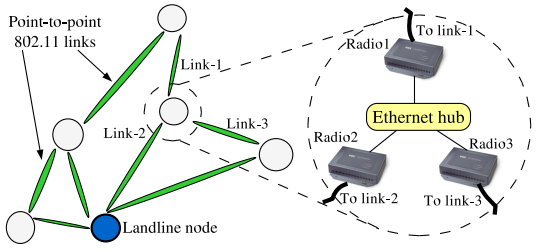

Figure 1: 802.11 mesh: nodes have one radio per link

The presence of multiple radios per node allows the use of separate, non-overlapping channels for the various links. However, we consider the use of a single channel, for several

related reasons. (1) First, it is quite convenient administratively, if we were to reserve a channel for the wireless mesh, and have the other channels be freely used for any local access. This is especially convenient for operation since this would prevent the wireless mesh backbone from being affected by "RF pollution" in the long run. Such an arrangement would also be suited for large municipality (or campus) managed WiFi networks [17], to avoid or alleviate RF pollution. (2) Second, in some developing countries, the 802.11b/a bands are licensed for outdoor use [10]. In such cases, use of more number of channels may translate directly to higher operational cost. (3) Finally, if we use 802.11b, then we have only 3 independent channels, and it is quite likely that the wireless mesh is not 3-edge-colourable. Or, while operating in a region already having 802.11 "RF pollution" (e.g. a university campus), it may not be possible to use some channels for the long-distance links at some locations. In such cases a contiguous portion of the mesh network would have to operate in the same channel. And we consider this most generic case in this paper.

With only a single channel available, 802.11 CSMA/CA MAC is all the more ill-suited for the wireless mesh. Specifically, it would not be possible to operate the multiple links at a node simultaneously. This is so even with the use of directional antennae, since we have "side-lobes" which cause interference among the links at a node. The well known expose node problem surfaces as the exposed interface problem in our setting.

In prior work [15], we have shown that even in the presence of side-lobes, it is possible to (a) transmit along all links of a node simultaneously (SynTx), or (b) receive along all links of a node simultaneously (SynRx). We have verified through experiments that if we are "careful" about the power settings of each of the links, such operation is possible. However, the CSMA/CA MAC and 802.11 DCF operation inherently do not allow either SynTx or SynRx (Sec. 2). We propose an alternative MAC protocol called 2P, based on SynTx/SynRx. The protocol is simple: we switch between a transmission phase and a reception phase at each node. This keeps all the links of a node operational in one of either directions at all times, making full use of the available capacity. We briefly outline the protocol in Sec. 2.

In this paper, we address issues under three main heads: (a) The design details of 2P; specifically the challenge of achieving 2P on off-the-shelf 802.11 hardware, thus preserving the cost advantages. (b) Dependence of 2P on the network topology, and the transmission power settings of the radios in the network. And, (c) a performance analysis of 2P, and comparison with CSMA/CA, through experimentation as well as simulations.

We start with a background description in Sec. 2, followed by a detailed design of the 2P state machine (Sec. 3), along with our implementation strategy on top of off-the-shelf hardware. As proof of concept, we have implemented 2P using driver-level modifications to the Linux open-source HostAP [4] driver v0.2.4 for Prism2 chipsets.

Next, we present in detail the topological constraints for 2P to be feasible in a given network (Sec. 4). We show how the constraints can be expressed in terms of a linear system of equations, with the variables being the powers of transmission along each link at the different nodes. We then build on this framework, and present simple topology creation heuristics, to plan a wireless mesh deployment in which 2P can operate.

In Sec. 5, we present detailed evaluations of 2P. We first evaluate the topology creation heuristics, on real village maps as well as on random topologies. We then present detailed ns-2 based simulations of 2P, and comparisons with CSMA/CA. Our simulations show that 2P achieves close to *optimal* throughput by keeping each link active at all times. And in comparison with CSMA/CA, we have observed up to 20 times better throughput. Finally, we present an evaluation of the performance of our prototype driver-level implementation of 2P. Our experiments indicate that 2P operation is feasible, achieving parallel operation of the links at a node, while CSMA/CA clearly does not allow this.

We present concluding discussions in Sec. 6. Specifically, we introduce the notion of *morphing*, which allows us to build fault tolerance in the topology, and more importantly, also allows us to save power.

2. BACKGROUND

In this section, we briefly summarize our prior work related to SynRx/SynTx [15]. We describe the outline of the 2P MAC protocol based on SynRx/SynTx. This will help set the context for presenting further details of 2P operation, its dependences, and its evaluation.

The mesh networks of interest to us have a separate radio for each link at a node, and point-to-point links with directional antennae. A schematic of this is shown in Fig. 1. However, even with multiple radios per node, the 802.11 MAC still allows the operation of only one link at a time (i.e. under single channel operation). This is due to *side-lobes* of the directional antennae. For example, consider the simple scenario shown in Fig. 2, with two point-to-point links involving three nodes A, N, and B. We show three cases: *SynRx*, *SynTx*, and *Mix-Rx-Tx* – the terminology is with respect to data transmission along the two links at node N. (Note that the labels R_i and T_i are different in each case.)

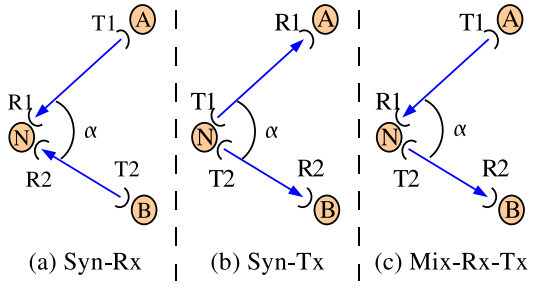

Figure 2: SynRx, SynTx, and Mix-Rx-Tx

Now, Mix-Rx-Tx is not feasible, since R_1's reception is affected by the transmission of T_2. This is due to (1) physical proximity, and (2) presence of side-lobes in the directional antennae. We have confirmed this through experimentation. We used an outdoor testbed with long-distance links, as depicted in Fig. 3. The heights of the towers in the three village nodes, and the angle between the two links, is as given in the diagram. The two antennae at N were mounted at heights of $15m$ and $30m$ respectively, for the links towards A and B. We used 24dBi parabolic grid directional antennae [5].

With respect to the other two cases, SynRx and SynTx,

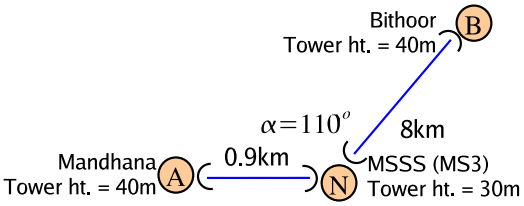

Figure 3: Testbed for SynRx, SynTx, and Mix-Rx-Tx

each link acts as interference for the other, due to the side-lobes. However, our initial empirical analysis of the SINR (Signal to Noise and Interference Ratio), showed that both SynRx and SynTx should be feasible, even in the presence of side-lobes of the parabolic grid antennae in use. We omit the details of this analysis here and we present a more detailed analysis, for a generic topology in Sec. 4. For now, it suffices to know that the SINR analysis depends on: (a) the signal level, i.e. transmission power as well as the path loss, and (b) the interference level, which in turn depends on the side-lobe level *at the* particular angular separation α between the two links. The BER (Bit Error Rate) in each link depends on its SINR value. In the case of both SynRx as well as SynTx, with appropriate power settings, it is possible to have the SINR of each link at a high enough level to achieve negligible BER ($< 10^{-6}$).

Now, even though SynRx and SynTx are potentially feasible, the 802.11 MAC does not allow either of them as explained below. In the case of SynRx, say R_2 finishes receiving first. Then an immediate ACK would be sent (after the SIFS interval), which would interfere with the ongoing reception at R_1. It thus reduces SynRx to a case of Mix-Rx-Tx. SynTx is also not possible: say T_1 starts transmission first. The other interface T_2 would carrier-sense T_1's transmission, and backoff.

The above problem with the 802.11 MAC is essentially due the presence of side-lobes, in the (imperfect) antennae being used. Doing away with the side-lobes (or reducing them) is definitely possible, and constitutes a PHY layer approach to addressing the problem. However, there are two issues. First, reducing side-lobes requires paying a lot of attention to electro-mechanical detail, and such antennae are currently very expensive (U.S. $600+ as opposed to the $20-50 parabolic grid antennae). Second, apart from the *side-lobes*, the antennae also have a near-field effect (i.e. the directionality comes into play only at longer distances, not when the two antennae at N are mounted close to each other). Hence it is not clear if complete isolation is possible at close physical proximity even with high-precision antennae.

Our paper essentially presents a *low-cost*, MAC layer approach to addressing the issue. In prior work, we have experimentally verified that SynRx, SynTx are potentially feasible, under "appropriate" power settings, with parabolic grid directional antennae. We used the same testbed as in Fig. 3 for this. To show this, we had to disable the offending immediate ACKs, and the carrier sense mechanisms. We achieved this through appropriate settings in the off-the-shelf hardware – we explain these settings when we present our MAC protocol implementation.

SynRx/SynTx together are known as SynOp: synchronous operation of the links at a node. Our MAC protocol, termed *2P*, is based on SynOp. The *2P* MAC protocol is simple in concept: it operates by switching each node between the *two phases* SynRx and SynTx. When a node switches from SynRx to SynTx, its neighbours have to switch from SynTx to SynRx, and vice versa. The keen reader may have noted that this requires that the network topology be bipartite; we return to this issue in Sec. 4. The operation of 2P for a simple 4 node topology is depicted in Fig. 4. Transmissions are shown above the time-line and receptions are shown below the time-line for the various links.

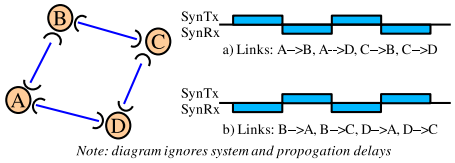

Figure 4: 2P Illustration

While in outline 2P is simple, significant questions remain: (1) how to achieve 2P on off-the-shelf hardware, to preserve the cost benefits, (2) how to achieve 2P without tight node synchronization across the nodes of the network, (3) what are the conditions (network topology), under which SynOp/2P is feasible, and (4) what is the performance of 2P in comparison with CSMA/CA. We seek to answer these questions in this paper. We begin with the detailed description of 2P, with reference to our prototype implementation on Prism2-based 802.11 hardware.

3. 2P: DESIGN AND PROTOTYPE IMPLEMENTATION

2P is an alternative to the 802.11 CSMA/CA MAC. Given that our choice of 802.11 is driven primarily by low-cost considerations, the immediate question is how to achieve 2P without losing the cost-benefits of off-the-shelf 802.11 hardware. The 802.11 MAC is usually implemented in firmware. Hence 2P is ideally implemented with firmware modifications. Such modifications involve little additional cost so long as the firmware is separate from the main 802.11 chipset (this is the architecture in many current implementations). For proof of concept, we have explored the approach of driver-level implementation. (The firmware is usually proprietary, and we do not have access to any firmware source).

In this section, we present our solution to addressing the various challenges in realizing 2P. In Sec. 3.1, we describe how we achieve SynTx and SynRx individually, on off-the-shelf hardware. Subsequently, Sec. 3.2 describes 2P operation on a single isolated link, by switching between SynTx and SynRx. Finally, Sec. 3.3 presents 2P operation in the general case where we have more than one interface at the various nodes.

3.1 Achieving SynOp

In the previous section, we saw that SynOp was not possible in 802.11 DCF due to two offending factors: (1) immediate ACKs sent by the MAC, and (2) carrier-sense based

backoff, as implemented by the hardware. These two functionalities must be *turned-off* to enable SynOp. To get rid of immediate ACKs, we do two things. (1) We use IBSS (Independent Basic Service Set, also knows as ad-hoc) mode of operation for all the interfaces, with a separate SSID (Service Set IDentifier) for each link. (2) We convert IP unicast packets to MAC broadcast packets at the driver level. The reverse is done at the other end of the link.

Now, 802.11 specifies that there are no ACKs for broadcast (and multicast) packets. Note that this is true only for IBSS mode of operation. (In infrastructure mode, there are ACKs for broadcast packets sent from a client to an access-point.) We thus avoid immediate ACKs. For our purposes, broadcast packets are equivalent to unicast packets since the link is point-to-point. (Some off-the-shelf hardware are configured by default to send broadcast packets at a lower rate than the maximum possible 11Mbps, but this can be changed easily).

Note that although we have gotten rid of the ACKs sent by the hardware, we can have ACKs sent by the driver. That is, a separate LLC can be implemented by the driver. The ACKs can even be piggybacked on data packets in the opposite direction (say, by prepending to the MAC payload). This also allows for implementing a window-based link-layer ACK mechanism – this makes better sense for the long-distance point-to-point link than the immediate-ACK mechanism due to the long round-trip. (We have incorporated this in our extensions to the ns-2 simulator, but not in the actual HostAP-based implementation.)

Next, to get rid of carrier-sense backoffs during SynTx, we make use of a feature provided by Intersil Prism chipsets (and some others too). Many 802.11 radio cards have two antenna connectors for diversity, say LEFT and RIGHT (these are called LOW and HIGH respectively, in Intersil Prism terminology). We are allowed to select the receiving antenna, at the driver level, using an *antsel_rx* command. We connect the actual antenna to say LEFT. During transmission, the receiving antenna is set to RIGHT (which is unconnected). This forces the carrier-sense to happen on the unconnected connector RIGHT, which sees only (negligible) noise; this essentially avoids carrier-sense backoff.

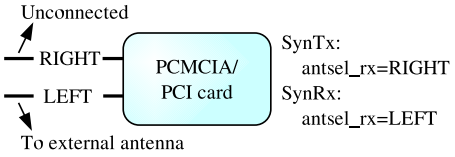

Figure 5: Using *antsel_rx* to avoid carrier-sensing

However note that we have to switch the receiving antenna to LEFT before switching from SynTx to SynRx. This is shown in Fig. 5. This is thus an overhead in switching between the two phases SynTx and SynRx.

3.2 2P Operation on a Single Link

With these above two modifications, SynOp is feasible. That is, SynRx and SynTx are individually feasible. But 2P also requires switching between the two phases. We now describe how this works on a single isolated point-to-point link.

We term one end of a point-to-point link to be an interface, abbreviated as *ifa*. With respect to a particular interface, we term the other end of the link to be its *link neighbour*, abbreviated as *link-nbr*. The state diagram for 2P operation is shown in Fig. 6. We first focus on the part of the state-diagram in "bold", which represents the operation of an isolated link. This diagram is from the point of view of a single ifa at a given node.

2P operates on the link by having each interface coordinate its switch between SynRx and SynTx with its link-nbr. In steady state, an interface sends B bytes in SynTx, and switches to SynRx, during which it receives B bytes. Thus we achieve bidirectional traffic on the half-duplex link. We define the combination of a pair of SynRx+SynTx phases to be a *round*.

For ease of implementation, we have a *marker* packet sent at the end of the regular packets in a SynTx phase. The receiving end switches from SynRx to SynTx on receiving a marker packet. Before each switch, *antsel_rx* is changed as described in the previous subsection. The usage of the marker packet, as well as the switching of *antsel_rx* add to the per-round overhead.

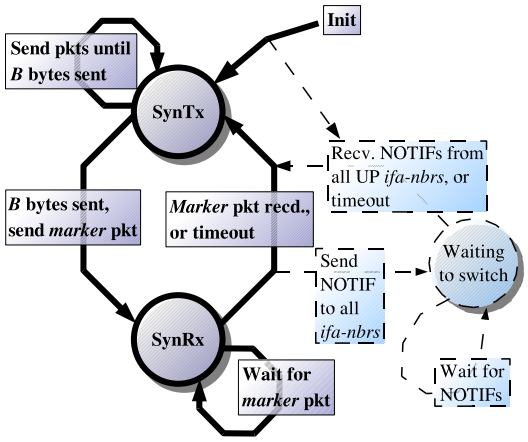

Figure 6: 2P state diagram

The marker packet acts as a "token" which is passed from one end to another, and we are guaranteed that at any time, exactly one end of the link is transmitting, and the other receiving. Under such steady-state operation, the two ends of the link are said to be in *loose synchrony*.

We need to handle the following two cases: (a) temporary loss of synchrony, and (b) link recovery after a failure (i.e. link initialization). Temporary loss of synchrony could happen due to the loss of the marker packet (this is equivalent to loss of token in a token passing system). And during link initialization, there is no (loose) synchrony in the system. Thus both cases are treated similarly.

In 2P, we handle the above cases using a *timeout* mechanism. On entering SynRx, an ifa starts a timer. The timer is for the ifa to switch from SynRx back to SynTx, in case such a switch does not happen because of a lost marker packet. This ensures that the ifa is not in SynRx indefinitely waiting for its link-nbr's marker packet. This is shown in Fig. 6.

The timeout value used is the same across all ifas and

all the nodes. An example of link resynchronization (link-resync) is shown schematically in Fig. 7. The two ends of the link are N_1 and N_2. Transmissions and receptions are shown above and below the time-line respectively, with shading. Idle times are shown without any shading. In the SynTx phase from N_2 to N_1, which ends at t_0, the marker packet is lost. The two links are out of synchrony with each other temporarily. N_1's timer expires at time t_1, and it starts SynTx. At t_1, when N_2 hears from N_1, its timer is *put on hold*, expecting N_1 to finish its SynTx with a marker packet. On getting the marker packet, the N_2 will switch to SynTx, and the link is now back in synchrony. Of course, if the marker packet from N_1 is missed, then N_2's timer will trigger, and the resync would happen in the next round.

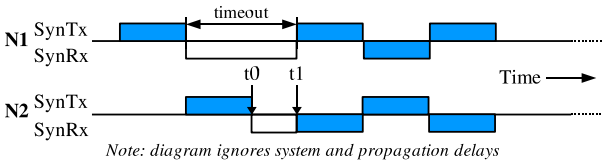

Figure 7: The 2P link-resync process

We note two things. First, the link-resync process is quick; it takes only one round, assuming that there are no further packet losses. Next, we note that CRC errors of packets other than the marker packet do not trigger a timeout, or any asynchrony. This is because 2P does not look inside the MAC frame anyway. In implementation, we instruct the hardware to pass on errored packets also up to the driver (as allowed by the HostAP driver). The timeout mechanism is triggered only on link failure, or on complete packet loss (no signal received) due to any reason. For reliability of the marker packet, it is sent at a lower encoding rate (1Mbps). This does not cost much since the marker packet is only 1-byte long in our implementation.

There is one final corner case to be mentioned in the link-resync process. We could have a case where the two ends of the link get out of synchrony, and timeout for each other at about the same time. The SynTx phases of the two ends would coincide, and thus they would not hear each others' marker packets. This would cause repeated timeouts. In practice, if the timers trigger within the duration of the one-way propagation delay (plus any system delay), such repeated timeouts would result. To avoid this, we add some random perturbation to the timeout value each time. That is, a node delays (bumps) its SynTx by a random amount. We term this as *bumping*. This is shown in Fig. 8. N_1 chooses a random bumping time of b_1, which happens to be smaller than the value b_2 chosen by N_2. Hence N_1 starts its SynTx first, at t_2, on hearing which N_2 puts its timer on hold. From the next round, the two ends are in sync.

There are two parameters we have introduced in the above description of 2P. The first is B, the number of bytes sent in each phase. The second is the *timeout* parameter. We discuss the choice of these in turn.

2P has a fixed overhead per round. This includes the change of $antsel\_rx$, and the transmission of the marker packet. Hence the larger the value of B, the smaller the effect of the per-round overhead on the achievable throughput. However, a large value of B means more latency. The worst

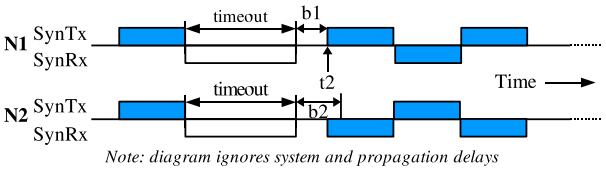

Figure 8: Bumping to avoid repeated timeouts

case excess latency, in comparison to the latency that would be suffered by a packet on a 11Mbps half-duplex (5.5Mbps full-duplex) link would be one phase duration (transmission time of B bytes). This is because, in the worst case, after arriving at the head of the queue at one end of the link, say N_1, the packet may have to wait for a further phase duration (transmission in the opposite direction, from N_2 to N_1) before being transmitted. For $B = 10Kbytes$, the per-round overhead is about 6%, and the excess latency is about $13ms$. We feel that this is alright for TCP as well as real-time connections, so long as the number of such wireless hops is small (3-4). If a lower latency is desired, the value of B can be appropriately reduced; the per-round overhead is about 11% for $B \simeq 4.5Kbytes$.

The timeout value has a lower bound: the phase duration. With reference to Fig. 7, N_1 would want to wait for at least its SynRx phase duration before deciding that it has missed the marker packet from N_2. The lower the value of the timeout, the quicker the link re-sync after a marker packet loss, or after link failure recovery. In our simulation, we have used a value of timeout to be 1.25 times the phase duration (this value is about $1.6ms$). However, in our implementation, we have used timeout values ranging from $25ms$ to $50ms$. This is to allow for the kernel's scheduling jitter, which is $10ms$ for Linux 2.4.

3.3 Communication Across Interfaces at a Node

We now discuss how 2P operates in the general case, when there are multiple interfaces at a node. We term the multiple interfaces at a node to be *interface neighbours* of one another, abbreviated as *ifa-nbr(s)*.

The important additional functionality here is that the ifa-nbrs at a node need to coordinate with one another to synchronize their switch from SynRx to SynTx (if all of them do not switch to Tx simultaneously, then it would not be SynTx!). Conceptually, this coordination is simple, and is shows as part of the 2P state diagram in Fig. 6. Once an interface decides that it is ready to switch to Tx (either after receiving a marker packet from its link-nbr, or after a timeout), it sends a *notification (NOTIF)* to its ifa-nbrs. In turn, it waits for a NOTIF from its ifa-nbrs before switching to SynTx. The notifications are broadcast/multicast to all ifa-nbrs, and the last such notification would trigger all of the ifa-nbrs to enter SynTx simultaneously (subject to notification delays, assumed to be small).

Note that no such coordination is required for the switch from SynTx to SynRx. Since all ifa-nbrs start Tx at the same time, and since they all transmit the same amount of data, they would finish their Tx at the same time, and all are now ready to receive from their respective link-nbrs.

The NOTIFs are easily implemented, using shared mem-

ory, if the multiple interfaces could be in the same physical machine. The more generic case is that the interfaces are on different machines. In this case, the communication can be via ethernet messages, assuming that the multiple interfaces are connected via an ethernet hub. Or, a simple special-purpose hardware can be designed for this. In our implementation, we have implemented the most generic ethernet-based communication between the ifa-nbrs.

When an interface comes up, it needs to first synchronize itself with its ifa-nbrs before starting to communicate with its link-nbr (see Fig. 6). This is important to achieve SynTx.

Each interface maintains UP/DOWN state with respect to each of its ifa-nbrs. This is so that they need not wait for NOTIFs from ifa-nbrs which are marked down. An interface, say I_1, marks its ifa-nbr, say I_2, as being UP, after receiving a NOTIF from I_2. Timeouts are used to detect an ifa-nbr doing down.

In our implementation, we have not built in any reliability in the ethernet message based NOTIFs. Hence the timeouts also can be caused by lost ethernet messages. And we stipulated that three consecutive timeouts must be seen before declaring an ifa-nbr to be down. We have used a timeout value of $50ms$ here, but this value is not very critical to the performance of 2P, since ethernet message losses are rare.

3.4 Some Remarks

With the above mechanisms, 2P operates by having each node switch between SynTx and SynTx. There is no tight synchronization across the nodes in the network, only loose synchronization between each node and its neighbours. With this, each link essentially behaves like a half-duplex link of fixed capacity.

We make a few important remarks with respect to the above description of 2P. First, in 2P operation, when there is no data from the (IP) layer above, we send dummy filler bytes instead, to maintain synchrony. These are of course discarded at the other end. One possible concern here is the wastage of power for such transmission of dummy bytes. Although power savings are an important concern, the transmission power is negligible in comparison to other parts of the system in our setting. In concrete terms, the transmission power of the radio interface is hardly 0.1-0.2 Watts. In comparison, the power consumption of say, a single board computer used to drive the radio interface (such as soekris: www.soekris.com) is at least 4-6 Watts. The power consumption of commercial Access Points is similar. If a village node were to be using a PC, the power consumption would be much higher (75-100 Watts or more). Hence, in engineering terms, the transmission power of the radio itself is a negligible concern.

Our second remark is related to the durations of the two phases SynTx and SynRx. We have described these to be equal, of B bytes each. That is, we divide the available throughput in the half-duplex link in the ratio 50%:50% across the two directions. A possible line of thought is to make these durations unequal, say $B + x$ and $B - x$ bytes, to accommodate asymmetry in traffic along the two directions (e.g. more download than upload). This is certainly possible in 2P, with the following two constraints: (a) all ifas at a node should have the same duration of SynRx (or SynTx), and (b) if an ifa has SynTx of duration $B+x$ bytes, then its link-nbr should have the same SynRx duration, and vice versa. These constraints effectively mean that if a node is two hops or more away from the landline, its download bandwidth is $min(B+x, B-x)$ bytes per round. And if this has to be maximized, we must have $x = 0$, which is what we have chosen.

On first glance, the above may seem to be a shortcoming of 2P. But this is not so. In Fig. 2, since Mix-Rx-Tx is not possible (with any MAC), we can never have data flowing across a two-hop connection, say $N_1 \rightarrow N_2 \rightarrow N_3$, continuously in the same direction. In other words, when N_2 is receiving from N_1, it cannot be sending to N_3. Hence the maximum achievable throughput on a 2-hop link is half that in a single hop.

Of course, if 2P is used in a single-hop network (star configuration with the land-line node at the centre), then having unequal phase durations could be beneficial.

Our final remark is related to a practical aspect. In SynOp/2P, we have multiple radios placed in the vicinity of each other at a node. In such a setting, one has to pay attention to the *leakages* from the connectors between the radio and the external antenna. In our experience so far, we have found that with two radios at a node, placing them 4-5m apart usually suffices. In general, with many radios at a node, carefully engineered connectors may be required to minimize leakages.

We have implemented 2P as a driver modification, with the HostAP open-source Linux driver for Intersil Prism based 802.11 chipsets. We have worked with v0.2.4 of the driver, and experimented with various PCMCIA as well as PCI prism-based devices. Experimentally, although we have been observed overheads in the prototype implementation, we still perform better than CSMA/CA. We evaluate the various performance overheads and the overall performance of 2P operation, in Section 5. We now discuss the issue of topology construction, in the context of 2P.

4. TOPOLOGY CONSTRUCTION

In our discussion so far, we have come across two dependences of 2P on the network topology. The first dependence is simple to express: the topology should be bipartite. This is because, at any time, if a node is in SynTx, all its neighbours have to be in SynRx, and vice-versa. This implies that the topology should not have any odd cycles; i.e. it should be bipartite. The second dependence is more intricate to express. This is the feasibility of SynOp at each node, and the related set of SINR equations – for each interface, all transmissions except that of its link-neighbour are seen as interference.

We term the first constraint as the *bipartition constraint*, and the second constraint as the *power constraint*. The power constraint is modeled as a linear set of equations – we describe this in Sec. 4.1. With the ability to express both the constraints, we then look at the issue of how to design a topology where both constraints are satisfied. This is the topic of Sec. 4.2.

4.1 Power Constraints in Topology Construction

The power constraint has to do with setting the appropriate power levels for each transmission in the network (the Prism2 hardware in our implementation allows us to configure the tx-power anywhere between 0dBm and about 20dBm with a 1dB granularity or less). For a generic bipartite topology, we can write a set of such equations to be satisfied for

SynOp to be feasible, as follows.

Denote by V, the set of village nodes, and by A, the set of antennae in a given topology. Each antenna at a node corresponds to a 802.11 radio interface. Let $|V| = N_V$ and $|A| = N_A$. N_V is thus the number of nodes, and $N_A/2$ the number of links in the topology. Let (V_1, V_2) be the bipartition of V. Let A_i denote the set of antennae at nodes in V_i, $i = 1, 2$. Let the antennae be $a_1, a_2 ... a_{N_A}$, such that a_{2i-1} in A_1 and a_{2i} in A_2 are opposite ends of the same link, $i = 1, 2 ... N_A/2$.

Denote the power of transmission from a_i to be P_i, $i = 1, 2 ... N_A$ (this is valid when the node corresponding to P_i is in SynTx mode). Also, let $d(i, j)$ denote the distance between (the nodes corresponding to) antennae a_i and a_j. Now, every receiving a_j gets non-zero power from every transmitting a_i, even though these may not be facing each other. Let $g(i, j)$ denote the effective gain corresponding to when a_i is transmitting and a_j is receiving. Note that $g(i, j) = g(j, i)$, and is the product (or sum, in dB space) of the transmission and reception gains. This is depicted in Fig. 9. The gain depends on the side-lobe levels of the two antennae, as well as the angles involved.

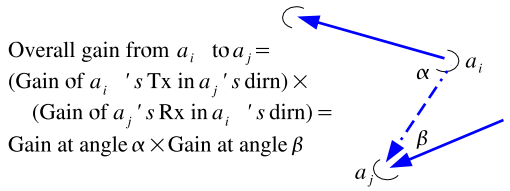

Figure 9: Illustrating gain from a_i to a_j

For a given topology, P_i's are variables, but $d(i, j)$ and $g(i, j)$ are known. We can now write a set of equations to be satisfied, for choosing the P_i values.

Consider the case when a_{2k} is transmitting, and a_{2k-1} is receiving. For proper reception, we need two conditions to be satisfied: (1) the signal level should be above a certain threshold, and (2) the signal should be above the interference by SIR_{reqd}. The first condition can be written as:

$$\frac{P_{2k} \times g(2k, 2k-1)}{PL[d(2k, 2k-1)]} \geq P_{min} \quad (1)$$

where P_{min} denotes the minimum required power level to work above the ambient noise level. P_{min} has a value of about $-85dBm$ for 11Mbps reception, for commercial 802.11b receivers [1]. $PL[d]$ denotes the path-loss at distance d. If we were to assume free-space path-loss, $PL[d] = (4 \times \pi \times f \times d)^2$, where f is the frequency in Hz, and d is in metres. We use the on-field, measured result from [13]. Here the path loss in the long-distance 802.11b links, expressed in dB terms is given by $PL[d] = FPL[d] + 3 + 0.15 \times d$, where $FPL[d]$ is the free-space path-loss, and d is the distance covered by the link, in kilo-metres.

When a_{2k} is transmitting to a_{2k-1}, the received signal level at a_{2k-1} is as given in the LHS of Eq. 1. The interference level is given by the sum of the interference from all other transmitting antennae.

Now, if 2P were operating with perfect synchronization across the nodes in the network, when a_{2k} is transmitting, only the nodes in antennae in partition A_2, i.e. a_{2j}, $j = 1, 2, ... N_A/2$ will be transmitting. However, with loose synchronization, even some antennae in the set A_1 could be transmitting. We are only guaranteed that the ifa-nbrs of a_{2k-1} are not transmitting. The signals from all other $a_j, j \neq 2k, j \neq 2k-1, j \neq ifa\_nbr(2k-1)$ are seen as interference at a_{2k-1}. The total interference can be written as

$$Interf(2k-1) = \sum_{j=1, j \epsilon I(2k)}^{N_A} \frac{P_j \times g(j, 2k-1)}{PL[d(j, 2k-1)]} \quad (2)$$

where the set $I(2k)$ is the set of interfering antennae, $I(2k) = \{j = 1 ... N_A, j \neq 2k, j \neq 2k-1, j \neq ifa\_nbr(2k-1)\}$. Now, for proper reception at $2k-1$, we require the signal level to be above the interference level by SIR_{reqd}. Hence,

$$\frac{P_{2k} \times g(2k, 2k-1)}{PL[d(2k, 2k-1)]} \geq SIR_{reqd} \times Interf(2k-1) \quad (3)$$

We can write a similar equation for transmission from a_{2k-1} to a_{2k}. We call the set of equations Eq. 1 and Eq. 3 (and similar ones for transmission from a_{2k-1} to a_{2k}) as "power-equations". These are a set of equations on the variables P_i, $i = 1, 2 ... N_A$. Equations represented by Eq. 1 are simply bounds on the variables, while Eq. 3 is a linear equation involving the variables. These equations are specific to a topology – the values of $d(i, j)$ and $g(i, j)$ depend heavily on the specific topology.

There are two important parameters in the above equations. The first is the SIR_{reqd}. The theoretical estimate of SIR_{reqd} for a negligible level of BER (Bit Error Rate) of 10^{-6} is about $10dB$ [11]. While solving the above equations, it may be advisable to build in some head-room, by choosing the desired SIR_{reqd} to be a little higher, say about $14dB$ to $16dB$.

The other important parameter is the antenna radiation pattern which decides the $g(i, j)$. For the parabolic grid antennae used in our testbed in Fig. 3, the spatial radiation pattern in the horizontal plane is as shown in Fig. 10 [5].

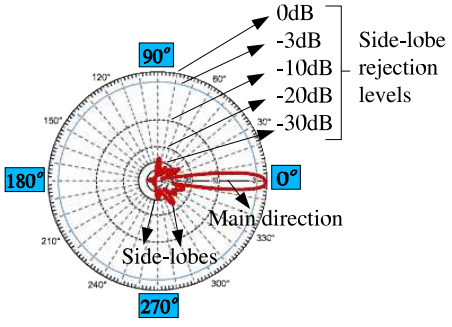

Figure 10: Spatial radn. pattern: parabolic grid antenna

In the above equations, it is easy to incorporate such practical considerations as different kinds of antennae, cable losses, etc. Whether or not the set of power equations

have a feasible solution will determine whether or not 2P is feasible in a given topology. This is the *power constraint* for the topology. We now discuss how, given a collection of village locations, one can plan a network topology to satisfy the bipartition as well as power constraints.

4.2 Topology Formation

With the above constraints, there is now the open question of whether at all it is possible to have topologies in which 2P is feasible. We now present a heuristic-based algorithm for topology creation.

We address the bipartition constraint as follows. In our context, all (or at least most) traffic passes through the land-line node. If we assume that we do not do multi-path routing, a tree topology, rooted at the land-line, is sufficient. And a tree is trivially bipartite.

However, a tree topology is bad in terms of fault-tolerance. But we note that we can *provision* extra links for fault-tolerance, but *turn-on* only a tree at any given time. We term this notion as *morphing* the topology; we discuss this further in Sec. 6.

Our heuristic algorithm is for the creation of a tree topology, in which the power constraints are satisfied. That is, given the locations of $|V|$ nodes, we create a spanning tree with $|V|-1$ links. There are three heuristics we use: (H1) reducing the length of links used, (H2) avoiding "short" angles between links, and (H3) reducing the hop-count. The first heuristic H1 is based on the observation that long-links in a topology are likely to interfere with more number of other links in the region in-between the two ends. This is even more so since a long-link would mean a higher power of transmission. The second heuristic H2 is based on the observation that the side-lobe leakage level of a directional antenna usually decreases with greater angular separation from the main direction (see Fig. 10). That is, the greater the angular separation between the links at a node, the lesser the mutual interference during SynRx or SynTx. The third heuristic H3 avoids cases of very "deep" trees, which can be bad in terms of latency.

The algorithm below mimics a natural deployment pattern: we first seek to connect villages (nodes) which are closest to the land-line (one-hop distance). Then we extend it to the next set of closest villages (two-hop), and so on.

Denote the land-line node to be level 0, and let hop h_i links connect between levels $i-1$ and i. The algorithm works by creating h_i links, starting with $i=1$, and proceeding until all nodes are connected, or a failure is declared. This can be summarized as below:

1. Call the set of unconnected nodes at this stage to be U. To create links at h_i, find the set S of all possible links between nodes at level $i-1$ and U.

2. Order the links in set S in increasing order of distance, say $l_1, l_2, l_3 ... l_k$.

3. For each $l_1, l_2, l_3 ... l_k$ (in that order): (a) If l_j forms an angle less than *ang_thr* with any of its neighbouring links, ignore it, continue with l_{j+1} (b) Else add l_j, and see if the resultant tree's set of linear equations is feasible, if feasible, go to step (4), else, remove l_j and continue trying with l_{j+1}.

4. If all nodes have been connected, stop

5. If we were successful in adding a link in step (3), continue with step (1) (no change in i – continue adding links at the same hop)

6. If we were *not* successful in adding any link in step(3), and *some* link has been formed in h_i, do $i = i+1$ and continue with step (1) (i.e. try to add links at the next hop now)

7. If we were *not* successful in adding any link in step (3), and *no* links have so far been formed for h_i, declare failure and stop.

The main steps are (2) and (3), where links are considered for addition according to heuristics H1 and H2. Heuristic H3 is implicit in steps 1-5, where we try to add as many links as possible at h_i, before moving on to the next hop. Step-6 considers the case where at least one link was added at h_i, where we try to form links at the next hop h_{i+1}. Step-7 indicates failure of the algorithm.

There is parameter *ang_thr* used above, to implement heuristic H2. This parameter depends on the side-lobe pattern of the antennae being used. Its use is to ensure that "bad" links are not formed early on in the algorithm, even though at that point of time, the power constraint may be satisfied. Links forming short angles in general cause problems later on in the algorithm, since the antennae have lesser higher side-lobe levels at smaller angles. This heuristic in effect reduces the chance that we declare failure in the algorithm (in step (7)).

For the parabolic grid antennae in use, which have a half-beam width of about 15^o, a value of 30^o to 45^o for the *ang_thr* works well. More details on the evaluation of the heuristic algorithm above are given in the next section.

5. EVALUATION

In this section, we present a detailed evaluation of the 2P MAC. Our evaluation is in terms of both simulations as well as experiments based on our implementation of 2P. We first begin in Sec. 5.1 with an evaluation of the topology creation algorithm, since this is essential for the presentation of further performance studies. Sec. 5.2 presents the details of various ns-2 based simulations. Finally, in Sec. 5.3, we evaluate 2P using our prototype implementation.

5.1 Evaluation of Topology Creation

With respect to the topology creation algorithm, we are interested in knowing the following measures. First, how does the algorithm scale – for how large a network are we able to form a topology compatible with 2P. Second, how is the feasibility affected by SIR_{reqd}. We mentioned in Sec. 4.1 that although the theoretical value of SIR_{reqd} for 11Mbps transmission is about $10dB$, we would like to have extra head-room. We are interested in knowing how much head-room we have for the topologies generated by our algorithm.

Our evaluation is in two parts. We first use collections of villages from the map of Durg district, Chattisgarh, India. Next, we use randomly generated topologies. For the first part, we take four collections of nearby villages, all from the same district. The first collection has 31 villages, and the other three have 32 villages each. We call these as Q_1, Q_2, Q_3, and Q_4. In each case, we choose one of the more prominent villages to be the land-line node. For each Q_i, we run the topology creation algorithm, varying the SIR_{reqd}

from $14dB$ to $20dB$. We fix the ang_thr at 30^o. We use the radiation pattern of the parabolic grid antenna, as shown in Fig. 10. To solve the power equations as part of the algorithm, we use the LP solver in the QSopt package [8]. In each case, we note the number of links which are formed successfully, before the algorithm terminates.

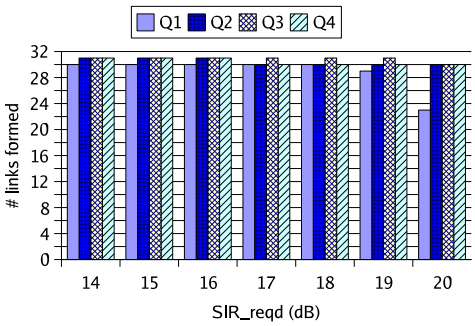

Figure 11: Topology formation on Q_1, Q_2, Q_3, and Q_4

The results are shown in Fig. 11. We see that in all the cases, the entire tree ($|V| - 1$ links) is formed successfully, for up to $SIR_{reqd} = 16dB$. And even for higher values of SIR_{reqd}, we continue to have most of the links formed. That is, a large part of the topology is 2P-compatible, even with a high value for the head-room in SIR_{reqd}. This clearly illustrates that our heuristics are good in achieving their objective. An example tree created by our algorithm for Q_1 is shown in Fig. 12. This is the tree topology we use in our simulations in the next section.

Next, we perform a similar evaluation using randomly generated topologies. We generate 50 nodes at random locations in a square area of side 'S' km. We chose 'S' such that the node density per unit area was about the same of the real village maps we used (we used S=44km). Among the nodes, we chose one roughly in the center to be the land-line; we picked a random node within 7km of the center of the square area. We generated twenty such 50-node scenarios. We ran the topology creation algorithm on these, much as earlier. Here too, we used $ang_{thr} = 30^o$. The results are as shown in Fig. 13.

We see that for $SIR_{reqd} = 14dB$, all the twenty cases form almost (48-49 links) the entire topology. For $SIR_{reqd} = 16dB$, there are only two cases among the twenty which form less than 30 links. And for $SIR_{reqd} = 18dB$, all but two cases form at least 20 links.

From this, we can conclude that in most cases it is easy to form 25-30 node tree topologies which are 2P-compatible, with a good amount of head-room. We now move on to the evaluation of the 2P protocol itself, using simulations.

5.2 Simulation Studies

Our goal in the analysis of 2P on a simulation platform is three fold. The first is to measure the impact step by step link establishment has on the already loosely synchronized network. Since 2P employs loose synchronization across nodes, our second goal is to study its effect on a large topology with saturation throughput as the metric. We are also

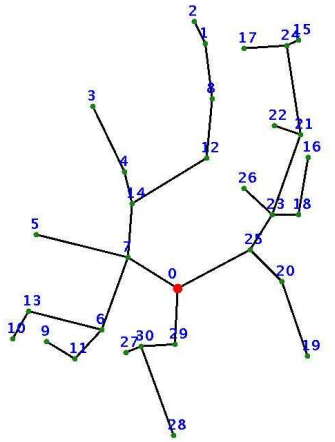

Figure 12: An example tree created on Q_1

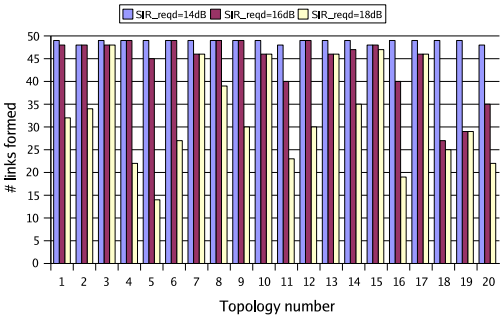

Figure 13: Topology formation on random 50-node cases

interested in quantifying the performance improvement 2P offers over traditional CSMA/CA protocol. Our third goal is to study the performance of TCP over 2P operated networks and the effect channel losses have on both the synchronization in the network and on the higher layer TCP protocol.

We have evaluated the performance of 2P on a simulation setup based on ns-2 [7] (version 2.1b9a). The default network simulator does not support many features needed in our setup. We first begin with the explanation of changes made to ns-2 followed by a detailed evaluation of 2P on an actual district topology. We compare the performance of 2P with the CSMA/CA protocol where appropriate.

5.2.1 *Extensions to ns-2*

We have addressed some of the deficiencies in the modeling of 802.11 MAC layer in the Extended Network Simulator package (ENS). ENS includes features such as adjacent channel interference, support for grey regions, random temporal variations in signal strength, adaptive data-rate transition. Apart from a more realistic implementation of the 802.11 MAC, it also incorporates additional features like multiple interface support for wireless nodes, a static routing

protocol for wireless scenarios, and also features inclusion of simple directional antennas.

The features of ENS relevant to our performance study are:

- *Multiple Interface Support:* In the 2.1b9a version of ns-2, all outbound packets that arrive at a node finally end up at a routing agent (AODV, DSR etc) which unfortunately does not support multiple interfaces. We have extended the "add-interface" functionality in "tcl/lib/ns-mobilenode.tcl" so that a node can now be configured with multiple interfaces (each interface has its own LL, IFq, MAC, NetIF). We have also introduced a new multiple interface aware Wireless Static Routing protocol (WLStatic), where routes can be configured manually. The usage of Address Resolution Protocol (ARP) in these settings turned out to be very tricky since the addressing mechanism used in NS is node based and not interface based. Tackling this in a proper manner requires many changes at various levels in NS. Instead, we took a simplified approach of manually populating the arp table with the necessary node to interface translations.

- *Directional Antenna Support:* The use of long distance links in our setup lead us to incorporate directional antenna support in ns-2. The radiation pattern can be specified in the form of a configuration file. For our experiments, we used the radiation pattern of a commercially available 24dBi parabolic grid antenna shown in Fig. 10

- *MAC Modifications:* The 802.11 MAC protocol shipped with NS does not factor in the air propagation delay added by the channel models when calculating ACK timeout values. For long distance links, this could result in close to zero throughput due to repeated timeouts. We have modified both the ACK timeout and contention window to factor in air propagation time of the channel. The 2P MAC was coded in NS according to the description provided in the previous sections.

- *LLC Modifications:* Since the MAC layer in 2P provides no acknowledgment mechanism, we have implemented a sliding window based LLC acknowledgment for 2P. The acknowledgment information is carried in the unused 6 byte "Address 4" field. The acknowledgment field consists of 2 parts: a 2 byte sequence number *ackseq* with wrap around and a 4 byte acknowledgment window *ackwin*. The value of ackseq means that all sequence numbers strictly "less than" ackseq need no longer be retransmitted. This need not necessarily mean that all these sequence numbers have been received. The LLC has a MAX_RETRY count for each packet. The receiving end waits for MAX_RETRY retransmissions, beyond which it considers the packet to be lost. The i'th bit (i=0..31) in the ackwin is an acknowledgment bit for $ackseq+i$. In each round/phase, the smallest sequence number(s) that has not yet been acked is sent. The MAX_RETRY limit is set to 4 in our experiments.

5.2.2 Simulation Methodology

Unless otherwise specified, we use the following parameters in our experiments. We consider a 31 node village topology Q_1, shown in Fig. 12. We experimented with other village as well as random topologies, the nature of the results remains the same. We consider both UDP (for saturation throughput measurement) and TCP flows. The UDP flow generates constant bit rate traffic, one packet every 2ms, enough to saturate a 11Mbps half-duplex link. The TCP flow is generated by a file transfer (ftp) for the duration of the simulation. The variant of TCP considered is NewReno with default parameter settings. The packet size in both cases is 1400 bytes. The direction of traffic flow is from the landline node down towards the village nodes. In other words, for each village node, we establish one UDP/TCP connection with the landline node as source and the village node as sink. The results will be similar if the direction of the traffic is reversed. We do not consider bi-directional traffic or asymmetric traffic since it really makes no difference in a half-duplex system for reasons mentioned in Sec. 3.4. The duration of each experiment is 10sec, which is enough time since each round in 2P is about 2.6ms.

Since we are dealing with a wireless environment, we consider complete signal losses (not just CRC losses) and evaluate the effect these losses have on the synchronization of 2P. We consider two types of losses: uniform and bursty. Bursty losses are modeled using a two state Markov chain with an average burst length of 4 packets. In both types of losses, we consider 1% and 5% loss rates.

Wherever appropriate, we compare the performance of 2P with the CSMA/CA protocol. Specific to 2P, we bring all the links up at time 0. We set the number of packets in a phase to one which amounts to a round duration of 2.6ms. We turn on the RTS/CTS option in case of CSMA/CA given that there are many cases of hidden nodes when using directional antennae.

5.2.3 Link Establishment

2P involves synchronizing the different links in the network so that they operate in tandem, utilizing all available capacity. When establishing a link (at startup or recovering after a temporarily shutdown), it is important to ensure that the procedure is fast without disrupting the synchronization of the remaining network.

We perform an experiment where links are added one after another to an existing already synchronized network. The establishment of the very first link in this case took about 12.9ms. After that all other links got established under 4.9ms. The reason for the long duration in establishing the first link is mainly because the first transmission of both ends of the link coincide (which we expect to be rare in practice) and they have to use *bumping* to establish the link. We also did not find any noticeable difference in the throughputs of the already synchronized links when adding the new links.

5.2.4 Saturation Throughput

A plot of the saturation throughput of 2P measured using UDP traffic along with a comparison of CSMA/CA is shown in Fig. 14. Note that the y axis is log scale. The different nodes are ordered in increasing order of throughput. As can be seen, nodes in the 2P scenario are able to achieve at least 3-4 times more bandwidth than nodes in the CSMA/CA case.

The maximum application throughput that can be obtained on a single 11Mbps link in a given direction in ns-2 is

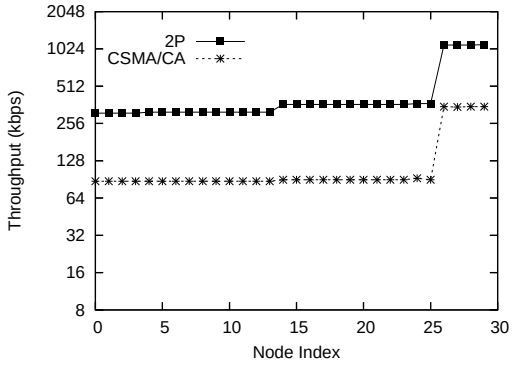

Figure 14: Saturation Throughput

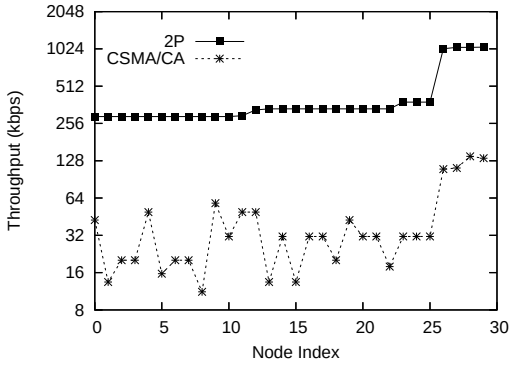

Figure 15: TCP Performance: No Losses

about 4.4Mbps. This is half of the 11Mbps raw rate, minus the packet header overheads. A quick calculation reveals that 2P is able to achieve the same rate on all its saturated (first-hop) links. If we were to add the application throughputs of all nodes on a given branch from the landline node, they sum to approximately 4.4Mbps. For example, the four nodes with the highest throughput correspond to nodes 27, 28, 29, and 30 in Fig. 12 and their application throughputs sum to 4.4Mbps.

The throughputs obtained by the nodes when operating in CSMA/CA mode can also be easily explained. Due to potential interference, only one of the links at the landline node can operate at any given time. Given there are 3 links at the landline node, this brings down the throughput of all nodes by a factor of 3. There is also the further overhead of RTS/CTS and unnecessary backoffs.

5.2.5 TCP Performance

We have also evaluated TCP performance for three scenarios: loss free, uniform and bursty losses. Fig. 15 shows the TCP throughput at the different nodes for the loss free case (Again the y-axis is log scale). There is about a 8-20 fold improvement in throughput in case of 2P compared to CSMA/CA. In fact, when compared to Fig. 14, one will notice that there is not much difference between the TCP and the saturation UDP throughput in case of 2P. TCP performance over CSMA/CA in a multi-hop setting is a very well studied area and is known to be very poor [12]. This is due to a combination of the exposed node (interface) problem, carrier sense based backoffs and poor interaction between TCP data and acknowledgments in either direction.

The performance of CSMA/CA in presence of losses only gets worse. So we exclude results for CSMA/CA in this setting. In case of 2P, we like to make two observations here: 1) When losses are isolated, the interface perceiving a loss will timeout and in the very next round regains synchronization. In the simulations, the timeout value is set to 1.25 times the phase duration. Hence the excess delay induced by a timeout, compared with a no-loss case is about $320\mu s$. Since the LLC implements a retransmission strategy, the losses are hidden from higher layer TCP connection. 2) When losses happen in bursts, time to resynchronize an interface will involve multiple timeouts, depending on the burst length. Since the retry limit employed by the LLC is 4, bursts above 5 will result in TCP having to cope with packet losses.

A plot of TCP performance for uniform and bursty loss rates along with no loss case is as shown in Fig. 16. In lieu of the above arguments, it is easy to see that the throughputs achieved are similar to the no loss case except when LLC is unable to recover the packet loss. For the 5% bursty loss scenario, in a few cases the throughput is lower. A study of the trace reveals that in these cases, the onus of loss recovery falls on TCP lowering its congestion window and hence the throughput. In some cases, the throughput experienced by a node with 5% bursty loss rate is more than the same node experiencing uniform 1% loss rate. This is because this node gets more bandwidth at the expense of some other node which experienced more losses.

It should be noted that in 2P, any event on a link that causes a timeout propagates the additional delay so added (320us) to other links in the network due to the loose synchronization mechanism in place. However since this overhead is small, we observed that its effect on application throughput was negligible in the simulations.

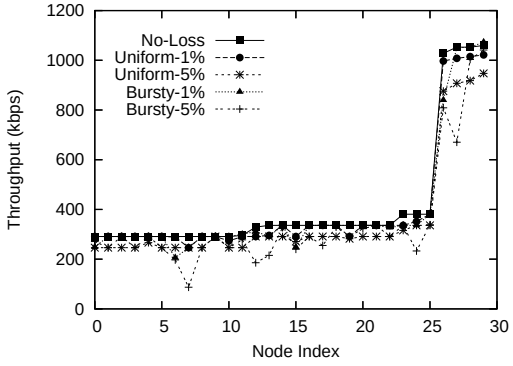

Figure 16: TCP Performance: Uniform and Bursty Losses

In summary, 2P is able to operate all the links simultaneously without mutual interference with minimal overhead and good robustness against losses. Not only does it show

up to 20 times better performance than CSMA/CA, but also it has close to *optimal* performance by operating all links at all times. That is, it is bound to perform as good as, or better than *any* MAC. In fact, its operation on a wireless mesh network resembles wired networks so closely to the extent that the results would be similar had we replaced the wireless links altogether with wired half-duplex links with 11Mbps capacity.

5.3 Implementation-based Evaluation

We now present an evaluation of 2P based on our prototype implementation. We used HostAP v0.2.4 on Linux v2.4.20-8. We present the evaluation in three stages, as below.

Confirmation of SynOp with Prism2 cards

As a first step, we first confirmed the operation of SynOp on the testbed shown in Fig. 3 (outlined in Sec. 2). We used the mechanisms explained in Sec. 3.1 to achieve SynOp on off-the-shelf PCMCIA cards. The power settings we had to use for the four interfaces were: $0dBm$ at A, $5dBm$ at B, and $0dBm$ at both the interfaces at N. We had saturating UDP flow on each link, with 1400 byte packets being sent every $1.5ms$ (enough to saturate the link). With this, SynRx as well as SynTx showed an average throughput of about $6.5Mbps$ on each link at the same time. This is close to the maximum throughput which can be achieved in practice with $11Mbps$ raw transmission, after accounting for the PHY/MAC headers and the carrier-sense based backoff. We observed the same maximum throughput even with just a single link active. We also confirmed with the same testbed setting that Mix-Rx-Tx was not feasible, just as we expected.

2P performance on a single link

We now look at the performance of 2P on a single link. We have performed this experiment indoors, as well as using the $A-N$ link of the outdoor testbed for this purpose, with similar results. We used Prism2 PCMCIA cards at both ends, attached to laptops running Linux 2.4.20-8. The traffic used was similar to that in the SynOp experiment. We now had bidirectional traffic, with each end sending 1400 byte UDP packets to the other side, every $3ms$. We measured the received throughput averaged over every 5 seconds.

We observed an average throughput of about $3.05Mbps$ at either end. This is lower than the $4.4Mbps$ observed in simulations, since several overheads come into play in implementation. Also, the overall throughput on the link is $3.05 \times 2 = 6.1Mbps$. This is slightly lesser than the $6.5Mbps$ we observed in the case of unidirectional traffic on a single link. The overheads which cause this can be accounted for as follows.

The expected overhead in the 2P prototype are two-fold: the marker packet, and the changing of the $antsel\_rx$ in the Prism2 cards. The marker packet being transmitted at $1Mbps$ raw transmission rate, is expected to have an air-time of about $192+50+240+320 \simeq 800\mu s$. The four components above are the PHY header transmission time, DIFS, MAC header/payload transmission time, and the random backoff. With a slot-time of $20\mu s$ and a minimum contention window CW_{min} of 32, we have an average of $320\mu s$ backoff per packet.

Note that this random backoff is different from the carrier-sense based backoff. This random backoff happens when a station has a continuous stream of packets, even when the medium is sensed to be idle. 2P synchrony is definitely affected by this randomness, as we shall see in the two-link case below. Ideally we would like to set $CW_{min} = 1$ to sidestep this randomness. This is in fact allowed by Atheros 802.11 chipsets, but not by the Prism2 chipsets we used.

The changing of $antsel\_rx$ takes $140\mu s$, according to the Prism2 specifications, and also as observed by us in practice. Hence the overhead per phase is about $940\mu s$. The transmission time of a single 1400 byte packet is about $192 + 50 + 1120 + 320 \simeq 1680\mu s$. With $B = 10KB$, 7 packets are transmitted per phase. Hence a phase duration is $940 + 1680 * 7 = 12700\mu s$. Hence the expected combined throughput is $1400 * 7 * 8bits/12.7ms = 6.17Mbps$. This is about the throughput we observe experimentally as well.

With a value of $B = 4.2KB$, we expect a total throughput of $5.6Mbps$, which is also what we observed in practice.

2P performance on a pair of links

The 2P code development and testing has been done using an indoor setup, where we try to mimic the outdoor setup. As of this writing, we have evaluated 2P in this setting[1]. The setting consists of four interfaces, like in the SynOp testbed (Fig. 3). We term these as A, N_1, N_2, and B, where N_1 and N_2 are the two interfaces at the intermediate node.

We took care in placing the nodes such that: (a) A was able to hear N_1, but not N_2 or B, (b) B was able to hear N_2, but not N_1 or A, and importantly (c) N_1 and N_2 were able to hear each other if and only if they had their respective antsel_rx set to the connected antenna.

This roughly mimics the outdoor setup: just as in the outdoor setup, we were able to achieve SynRx and SynTx, but not Mix-Rx-Tx. The fact that we were not able to achieve Mix-Rx-Tx in the indoor setting implies that the two links are not really isolated from one another.

In this setting, we generated UDP traffic much as earlier, and measured the received throughput averaged over 5 second intervals. The communication between ifa-nbrs N_1 and N_2, as required by 2P, was via ethernet. We had bidirectional UDP traffic on both the links $A \leftrightarrow N_1$ and $N_2 \leftrightarrow B$. The measured throughput is shown in Tab. 1. We compare it with the throughput as achieved by CSMA/CA in the same setting. We have 24 readings, each of 5 second durations, for a total of 120 seconds. The table shows the average and standard deviation of these 24 readings.

	Avg (SD) thrpt at A (Mbps)	Avg (SD) thrpt at N_1 (Mbps)	Avg (SD) thrpt at N_2 (Mbps)	Avg (SD) thrpt at B (Mbps)
2P	2.70 (0.31)	2.06 (0.24)	2.81 (0.15)	2.81 (0.10)
CSMA	2.07 (0.13)	1.13 (0.22)	1.90 (0.15)	3.11 (0.14)

Table 1: 2P on two links, versus CSMA

First, we observe that 2P performance is uniformly higher than that of CSMA, except in the case of B, where 2P performance is $0.3Mbps$ lower. Summed over all four nodes, the throughput of 2P is $2.17Mbps$ higher. This is primar-

[1]We expect to repeat the evaluation in the outdoor setup in the coming month.

ily because of the (loose) synchronous operation of the two links which mimimizes mutual interference.

Next, we observe that the per interface throughput in the case of 2P is lower than the $3.05Mbps$ we observed with just a single 2P link. The value is especially low at N_1. An examination of pieces of the trace collected by the driver revealed that there were many cases where the synchronization of 2P was being affected by delays. We believe that the main reason for this is the random backoff even in the absence of carrier-sense, as explained earlier. Ideally, we would like to set $CW_{min} = 1$ to avoid any randomness altogether.

We believe there are also other shortcomings of the driver-level approach to 2P implementation. 2P needs tight control over when packets are sent and received over the wireless interface. In addition, it also needs similar tight control over the ethernet messages sent and received between N_1 and N_2. There are several sources of CPU scheduling delays.

In the version of Linux we used, the scheduling granularity is $10ms$. Moreover, we do not have sub-millisecond kernel timers (the default Linux kernel does not come with this functionality, and we have not yet tried the patch for this functionality). We use the periodic timer interrupts received from the Prism2 hardware for this purpose. We used a timer duration of $2ms$, below which the overhead became quite high. Apart from the timer interrupt, the driver receives interrupts for (a) each packet transmission, (b) each packet reception, and (c) every switch to $antsel\_rx$. Further, we also have to process the kernel callbacks for every ethernet message received (we implemented the ifa-nbr communication using netfilter hooks). While we expect to have tight control over packet transmission/reception, this may not necessarily be the case. For instance, a packet may get sent much after the command was given to the hardware. Or, we may receive a netfilter callback on a received ethernet message much after the packet was actually received.

Furthermore, there were other stresses on the CPU scheduling as well. The UDP traffic generation itself was making a system call for packet transmission or reception every $1.5ms$. And the wireless PCMCIA cards we use do not have support for DMA (Direct Memory Access) and the CPU has to be involved in the copying of tx/rx bytes to/from the hardware. Given all this, the sub-optimal performance of the 2P implementation is understandable.

Our prototype implementation of 2P at the driver level is meant for proof of concept. The above performance issues can be overcome by alternate implementation strategies. An ideal possibility is the firmware level implementation mentioned earlier. If firmware level access were not available, we could push the time-critical tasks of the 2P driver on to a separate processor hardware, which can in turn interface with the off-the-shelf 802.11 hardware (still preserving the low-cost benefits).

6. DISCUSSION AND CONCLUSIONS

In this section, we present some points of discussion beyond the detailed presentation so far, and subsequently conclude the paper.

Prior Work

In style, 2P is nothing but Spatial reuse Time Division Multiple Access (STDMA) scheduling. Researchers have long considered STDMA techniques (see [16] and references thereof) in multi-hop networks. However, work in this domain has mainly revolved around scenarios where a node can receive from only one neighbor at a time. With respect to transmission, a node may be able to broadcast to all neighbors, but not transmit independent information to neighbors simultaneously. What separates our work from past work is the use of a) multiple radios per node, b) directional antennae, and c) knowledge of exact location of nodes so that power levels on links can be engineered to reject interfering transmissions. This lends itself to simultaneous synchronous operation and the simple 2P protocol that achieves loose synchronization. The loose synchronization aspect is significant given that tight synchronization is often quite difficult to design and implement.

Wider Applicability of 2P

While we have focused mostly on rural networking in this paper, 2P can be used in other settings such as community/campus networks to provide a wireless back-haul. These settings do not normally have spectrum licensing issues, thereby permitting usage of multiple channels. A growing concern here is the problem of "RF pollution". A policy we advocate in these settings is to reserve one channel exclusive for any back haul network use. A university or a municipality can make exclusive use of that channel in its jurisdiction to build an interference free wireless backbone based on 2P, to serve its members. The other channels are free for use by its members for any purpose whatsoever including providing local access. The practicality and performance of 2P in such settings are worthy of further exploration.

The concept behind 2P is spatial reuse and this concept has applicability in any domain where spectrum is scarce. There are two dependences for 2P: (a) the static nature of the network, and (b) multiple radios at a node. Its dependence on the PHY is minimal, and is related to two things: the side-lobe patterns as determined by the antenna design, and SIR_{reqd}, as determined by the modulation. Both these parameters are captured in the equations in Sec. 4.1. The equations thus apply to any PHY, and not just to 802.11b.

Fault Tolerance and Morphing

In the topology creation algorithm, we have considered only the formation of trees. But trees are not good for fault tolerance. However, as mentioned earlier, so long as we do not have multi-path routing, we only need a tree active at any point of time. Hence we can *provision* additional links, but turn them *on* only as needed. That is, *morph* the topology in the event of a failure. An algorithm for determining the set of back-up links to provision for a tree topology, is an area for future work.

The notion of morphing can be used in another significant way, as follows. The usage of rural Internet (e.g. www.n-logue.com) has several interesting characteristics: (a) the usage is *not* 24x7, even discounting night-time idle periods, (b) there are a focused set of applications such as VoIP, eGovernment, telemedicine, etc., and (c) there is a shared usage model; a single PC is used in a time-shared model. Given (a), only a relevant part of the network needs to be up at any time. Observations (b) and (c) lead to a situation where users are willing to tolerate delays of several minutes (waiting for their turn, during busy times). Together, these mean that PCs, network interfaces, and any supporting networking equipment can all be turned-off when not needed, and brought up on demand. A minute or two delay for

bringing up the system (booting) is very much tolerable, in the interest of saving power in the long-run.

For such morphing, a node downstream from the landline needs to be able to bring up a node upstream, for availing connectivity as needed. In turn, this requires engineering (low-power) hardware which senses radio activity. Once activity is sensed, the relevant part of the system can be brought up. After this, 2P link formation itself takes little time.

Topology morphing as described above, for fault-tolerance as well as power savings is an interesting area for further exploration.

Conclusions

Cost reduction is of utmost importance in rural networking, and 802.11 is a *cost-effective* technology. In this paper, we have explored how to make it *performance-effective* for our intended use of the technology. The 2P MAC protocol is a replacement for the 802.11 CSMA/CA, and achieves maximal efficiency while using a single channel, and without requiring tight time synchronization. We have shown through simulations that 2P can perform several times better than CSMA/CA in a long-distance 802.11 mesh network. Our prototype implementation demonstrates the viability of 2P in practice, building on off-the-shelf 802.11 hardware.

Acknowledgment

We thank Media Lab Asia for financial support of this project. The testbed experiments described in this paper would not have been possible without the technical support of the Digital Gangetic Plains project team.

7. REFERENCES

[1] Cisco a350 Data Sheet. `www.cisco.com/warp/public/cc/pd/witc/ao350ap/prodlit/a350a_ds.htm`.

[2] Community Wireless / Rooftop Systems. `www.practicallynetworked.com/tools/wireless_articles_community.htm`.

[3] DjurslandS.net: The story of a project to support the weak IT infrastructure in an low populated area of Denmark. `http://djurslands.net/biblioteket/international/djurslands_net_english_presentation.ppt`.

[4] Host AP driver for Intersil Prism2/2.5/3. `http://hostap.epitest.fi/`.

[5] HyperGain HG2424G Reflector Grid Antenna. `http://www.hyperlinktech.com/web/hg2424g.php`.

[6] IEEE P802.11, The Working Group for Wireless LANs. `http://grouper.ieee.org/groups/802/11/`.

[7] Network Simulator. ns2. `http://www.isi.edu/nsnam/ns`.

[8] QSopt Linear Programming Solver. `http://www.isye.gatech.edu/~wcook/qsopt/`.

[9] Technology and Infrastructure for Emerging Regions. `http://tier.cs.berkeley.edu/`.

[10] Pravin Bhagwat, Bhaskaran Raman, and Dheeraj Sanghi. Turning 802.11 Inside-Out. In *HotNets-II*, Nov 2003.

[11] A. Chindapol, A. Stephens, and J Lansford. *IEEE P802.15-02/069r0, Change request for 802.15 Recommended Practice PHY text.*

[12] Z. Fu, P. Zerfos, H. Luo, S. Lu, L. Zhang, and M. G erla. The Impact of Multihop Wireless Channel on TCP Throughput and Loss . In *INFOCOM*, Apr 2003.

[13] Rajesh Gandhi. Empirical Path Loss Models for 802.11b Links. Master's thesis, Indian Institute of Technology, Kanpur, 2003.

[14] R. Karrer, A. Sabharwal, and E. Knightly. Enabling Large-scale Wireless Broadband: The Case for TAPs. In *HotNets-II*, Nov 2003.

[15] Bhaskaran Raman and Kameswari Chebrolu. Revisiting MAC Design for an 802.11-based Mesh Network. In *HotNets-III*, Nov 2004.

[16] A. Sen and M. L. Huson. A New Model for Scheduling Packet Radio Networks. In *INFOCOM*, 1996.

[17] The New York Times. Philadelphia Hopes to Lead the Charge to Wireless Future, 17 Feb 2005.