Analysis of Spatial Multiplexing for Cross-Layer Design of MIMO Ad Hoc Networks

Marco Levorato*, Stefano Tomasin*, Paolo Casari*, Michele Zorzi†

*Department of Information Engineering — University of Padova — Via G. Gradenigo, 6/B, I–35131 Padova (PD), Italy
†Department of ECE — University of California at San Diego — 9500 Gilman Drive, La Jolla, CA 92093-0407

E-mail: {firstname.lastname}@dei.unipd.it

Abstract—We consider the application of spatial multiplexing to ad hoc networks where nodes have multiple antennas. At the physical level, we suppose that layered space-time multiuser detection (LAST-MUD) is applied to separate multiple streams arriving at the receiver simultaneously.

Our contributions here consist first in the reproduction of the multiuser detection process performance by a simple analysis, where we also specify when the analytical results are expected to be accurate and why. Second, we use this approximation to perform a cross-layer design of a MIMO ad hoc network where physical layer and Medium Access Control strategies are integrated to maximize the network throughput.

We finally corroborate our conclusions, by comparing analysis with simulation results both at both the link and the network level.

I. INTRODUCTION

THE USE of multiple antennas in wireless communication systems has proven to be effective in increasing throughput both of single links and of an entire network. Considering typical network environments, signal propagation on multiple radio paths allows to model the transmission–reception phase as a Multiple–Input–Multiple–Output (MIMO) system, which can be exploited to increase the transmission rates with respect to systems with one antenna [1]. Recently, interest has arisen on the application of MIMO techniques to ad hoc networks, which are the focus of this paper. In ad hoc networks, nodes are not connected to any underlying infrastructure and are not supervised by any entity. Instead, communication is directly performed between nodes using a flexible architecture, which can be easily adapted to changes in position and number of nodes. Examples of ad hoc networks can be found in military battlefield scenarios, networks for disaster recovery and flexible data sharing among computers.

In particular, in this paper we investigate the cross-layer design of physical and Medium Access Control (MAC) layers of an ad hoc network with multiple antennas. The MIMO technology we consider is the scheme based on spatial multiplexing presented in [2], where each transmit antenna sends a different data stream. Signals from the various receive antennas are then combined and successfully decoded in succession, and the interference among streams is removed after each detection. Since the detection of streams is performed in cooperation by means of an Interference Cancellation (IC) technique, we denote the resulting scheme as LAyered Space–Time (LAST) MultiUser Detection (MUD), i.e. LAST-MUD. The scheme in [2] has been proven to be equivalent to the well known V-BLAST [3], which was one of the pioneering systems exploiting spatial multiplexing for increasing the available transmit data rate, through proper exploitation of a rich fading environment.

Simulations of both the physical and MAC layers require an excessive computational effort even for medium-sized networks. Hence we introduce an analytical model of the bit error rate in a LAST-MUD scenario, based on the exploration of error events in the IC process. The performance of network protocols is then assessed, and impacts on the choice of different MAC strategies on the network throughput are evaluated.

II. SYSTEM DESCRIPTION

We take into account a packet radio network where each node has A antennas. Users choose how many antennas to use for transmission, depending on traffic needs, and transmit a different BPSK stream on each selected antenna. The number of bits per antenna is constrained to a maximum of 1000.

The network is set up in a rich scattering environment, modeled by frequency–flat Rayleigh fading propagation channels. Thus, the channels are represented by a matrix $H = (h_{at})$ having zero–mean, unit–variance, circularly Gaussian complex entries. The entry $h_{at}$ is the complex channel gain between the $a$-th receive antenna and the $t$-th of the $U$ transmitting antennas (which includes all active users).

Assuming that perfect symbol synchronization and channel estimation can be achieved, each receiver processes antenna outputs on a per–symbol basis, according to the LAST–MUD scheme [2]. In particular, let $z_a = [h_{a1} \ldots h_{at}]$ be the row vector of channel gains from all of the $U$ transmit antennas to the $a$-th receive antenna and write the received signal as $r_a = z_a b + n_a$, where $b$ is the column vector of all transmitted symbols and $n_a$ is the noise sample at the $a$-th antenna (a zero–mean complex Gaussian random variable with variance $\sigma^2$). The receiver computes

$$m = \sum_{a=1}^{A} z_a^H (r_a + z_{at}^N b_{int}) = R b + n + i,$$ (1)

Note that this assumption is given here and in the following for the sake of clarity, but could be easily removed to account for the transmission of packets having different, or not multiple of 1000, lengths.
where $R = \sum_{n=1}^{A} z_n^H z_n$ is the $U \times U$ space cross-correlation matrix, $n = \sum_{a=1}^{A} z_n^H p_a$ is the filtered Gaussian noise vector, and $^H$ denotes the complex transpose operator. Note that an unestimated space–filtered interference term is present, i.e., $i = \sum_{n=1}^{A} z_n^H d_{int}$, that is used to model the contribution coming from received streams whose channel cannot be estimated for some reason. The presence of these uncanceled streams is due to limited channel estimation capabilities of wireless terminals.

The receiver performs the detection of the streams in subsequent steps. At each step, the user with the highest Signal–to–Interference–and–Noise Ratio (SINR) is chosen and the corresponding weighing vector $w$ for the statistics in (1) is obtained from the correlation matrix $R$. Namely, if $i$ is the decoding step, and $k_i$ is correspondingly the index of the user selected for decoding, $R$’s Moore–Penrose pseudoinverse [4] is computed, which we call $R^+$, and finally the $k_i$-th column of $R^+$ is extracted for weighing [2]. The inner product between vectors $m$ and $w$ is then calculated, and the obtained scalar is used to feed a hard detector. Finally, the contribution of the estimated symbol is canceled from the sufficient statistics $m$ and the following user (on a SINR basis) is decoded, restarting the whole process from the beginning. The algorithm goes on until all spatially superimposed symbols have been estimated.

### III. Approximate Performance Analysis

In order to be realistic and to produce meaningful results, any simulation aimed at assessing the performance of a network that takes advantage of the described physical layer features has to reproduce properly the behavior of the underlying signal processing structure. To this extent, the most accurate choice that could be preferred would be to simulate the whole algorithm on a per–symbol and per–step basis. While this is indeed a good approach for Bit Error Rate (BER) evaluation, it would not be convenient for network simulation, where many single–link transmissions are performed, therupon enlarging computation times.

In the present work, we devise a method to predict the BER performance of a LAST–MUD receiver as a function of the number of incoming streams, their transmission powers and undergone channel effects. This method relies on the prediction of the probability that an error occurs in the decoding process, conditioned on the probability that previous steps of the algorithm ended up in no errors or in some mistaken symbol detection, thereby including in our analysis the effects of error propagation. An alternative approach, which uses a gaussian model for the effects of error propagation is presented in [5].

First, we observe that the first detected stream is not affected by the error propagation and its error probability depends only on the channel conditions, noise and interference; on the other hand, suppose that $U$ total symbols are to be decoded in $U$ subsequent steps and focus the attention on step $i$ (i.e., the symbol belonging to the $k_i$-th user is being decoded). In general, call $\epsilon_{k_i} = \hat{s}_{k_i} - s_{k_i}$ the error contribution corresponding to the detection of the $k_i$-th user, where $s_{k_i}$ is the symbol originally sent and $\hat{s}_{k_i}$ is the algorithm’s estimate. With BPSK, we have $\epsilon_{k_i} \in \{-2, 0, +2\}$, since erroneous cancellation has the detrimental effect of doubling interference affecting subsequent detections.

Define then the vector $e(i)$, representing the sequence of error contribution up to step $i - 1$, to be

$$e(i) = [\epsilon_{k_1}, \epsilon_{k_2}, \ldots, \epsilon_{k_{i-1}}]$$

where each component is defined as before. Then, the probability that an error is performed at LAST–MUD step $k_i$ is given by

$$P[\epsilon_{k_i} \neq 0] = \sum_{E \in \{-2,0,+2\}^{i-1}} P[\epsilon_{k_i} \neq 0 | e(i) = E] P[e(i) = E]$$

where the summation is taken over all possible error configurations of the $i - 1$ previously decoded streams. We remark that the probability in (3) represents an exhaustive evaluation of error events, which considers any combination of mistakes and correct estimations of superimposed received symbols.

For a more efficient computation of (3) we can consider a subset $S \subset \{-2,0,+2\}^{i-1}$ that contains only those error configurations where at most $\ell$ errors have occurred. The rationale behind this choice is twofold. First it allows for much shorter computation times, since it considers a smaller set of events. Second, for increasing $\ell$, all events describing more than $\ell$ errors represent progressively minor contributions on the overall $P[\epsilon_{k_i} \neq 0]$. For this reason, they may be set aside from calculation, since there is a low probability that interference originated from wrong cancellation at a certain step (say, $i$) allows for correct detections of symbols at steps $i + 1, i + 2, \ldots, U$. Hence, we restrict the summation in (3) to subset $S$ and, in this paper, we also set $\ell = 1$.

In order to be more general, we indeed set aside configurations outside set $S$ as explained, but do not neglect them completely. We consider instead that when more than $\ell$ decoding errors occur, the resulting SINR is so low that we can approximate the probability of error with 0.5 at each subsequent step.

In Fig. 1, we report the BER for the chosen case $\ell = 1$. There, the notation $(T; R)$ accounts for the presence of $T$ transmit and $R$ receive antennas. Because we will use 8 antennas per node in our following network design, a comparison with simulated results is provided for a total of 5 different configurations, all having a fixed number $R = 8$ of receive antennas. The approximation accuracy is encouraging, as BER curves begin to separate significantly at high load levels only. Anyway, note that for BER calculation purposes, all $T$ transmitters are placed at the same distance from the receiver, and each of them uses a single antenna at the full available power. This is indeed a very uncommon situation in an ad hoc network, where random radio access is expected to

---

2We have also tested system performance for other values of $\ell$, without obtaining appreciable differences, either in physical or network layer performance. Hence, we choose $\ell = 1$ because it offers the lowest computational effort at an almost equal level of precision.
be ruled by some protocol in order to maximize aggregate throughput, typically by using limiting techniques for the interference level at each receiver. In Sec. IV-B, we will give more detailed insights on how Packet Error Rate (PER) is affected by superimposed transmissions in a properly designed MIMO network.

IV. NETWORK PERFORMANCE EVALUATION

In this section, we describe the network settings that we used to test the accuracy of the approximate approach to BER calculation, when applied to the performance simulation of an ad hoc network with multiple antennas and MIMO communications.

A. Parameters and MAC Protocol

Our protocol is based on an exchange of control messages preceding data transmission. We call these messages Request–to–Send (RTS) and Clear–to–Send (CTS), according to the common 802.11 DCF terminology [6]. Each correct data reception is followed by an ACK message. We suppose that transmissions are organized in frames, so that all nodes wanting to transmit send first an RTS, then all pinged receivers may respond with CTS if they want to grant the transmission: data and ACKs also follow the same synchronization. Even if it may seem a major assumption, note that this kind of frame–level synchronization will be applicable to our network in an easy way, as we will study protocol performance on a completely connected network. Moreover, recall that slot synchronization is currently achievable in the 802.11 family of protocols. A study on the impact of unsynchronized transmissions is beyond the scope of this work.

For our study, we consider a total of 25 nodes on a $5 \times 5$ square grid with a 25 m spacing between neighbors. Nodes have 8 antennas each and use BPSK for transmission. The number of bits per packet is constrained to 200 for RTSs, CTSs and ACKs, and to 1000 bits per transmit antenna for data. For example, a 2000-bit packet could be either sent in a single frame by feeding two different 1000-bit streams to two different antennas, or split between two frames, using a single transmit antenna in each of them.

Packets are generated according to a Poisson process whose intensity is $\lambda$ packets per second (per node) and randomly assigned a length in the set $1, 2, 3, 4 \times 1000$. Unsent packets are backlogged.

In order to allow receivers to estimate channel effects undergone by incoming streams, each transmitted packet contains a proper training sequence. Nodes can anyway estimate up to a maximum of $N_s$ training sequences, thus limiting the number of sumperimposed receivable flows. Each unestimated signal produces an un cancellable interference contribution as in (1). We set $N_s = 32$.

RTS messages are sent to the destinations of the first packets in the queue and, if possible, including multiple requests toward multiple nodes. Each request carries the corresponding number of transmit antennas to be used. When composing CTSs to allow one or more nodes to transmit, receivers apply a Neighboring Traffic Adaptation (NTA) policy that aims at protecting data from undesirable interference, while at the same time preserving at least some wanted incoming streams. More in detail, NTA saves $N_s/2$ sequences for interference cancellation purposes only. The remaining $N_s/2$ are allocated by giving the priority to at least one 1000-bit stream meant for the receiver, but then choosing what channels to estimate in order of decreasing received stream power. If the stream considered is directed to the node, then it grants it in the CTS, otherwise it saves a training sequence for interference estimation and cancellation.

This structure is flexible and offers senders the chance to address multiple receivers in a single frame. If such behavior should turn into unmanageable traffic requests, receivers protect themselves from excess traffic by granting and specifically decoding (in the extreme case) one single 1000-bit stream, while saving all other training sequences to estimate and cancel the interfering flows. We stress that NTA is a good example of cross–layer design, as the MAC layer is informed by the physical layer of per–stream received power, and thereby enabled to decide the preferred subset of received streams to be explicitly decoded, either for throughput or interference cancellation purposes. These decisions are then fed back to the PHY layer which performs spatial demultiplexing according to MAC layer orders.

Data is then stream–wise ACK’ed upon correct reception, an important feature that enables decoded 1000-bit streams not to be wastefully sent twice. When all streams belonging to a packet are ACK’ed, then the packet is removed from the node queue. Just like RTSs and CTSs, ACKs also include all acknowledgments to be sent by a node to its transmitters in that frame.

Two backoff techniques are used to limit excess transmission attempts in congested portions of the network. If the CTS corresponding to an RTS is not received, the node defers transmissions for a number of frames uniformly chosen in the interval $[1, B_{max}]$. $B_{max}$ is increased exponentially in
the number of failures $N_{\text{fail}}$, following the relation $B_{\text{max}} = W \cdot 2^{N_{\text{fail}}-1}$, with $W$ a start value which we set to 16.\(^3\) We distinguish between restraining from starting any other transmission upon any failure (Node–Lock), which is a commonly used backoff technique (e.g., similarly to standard 802.11 [6]) and blocking transmissions to a specific unavailable destination at a time (Dest–Lock), expecting the first to be aggressive, and the second to be a more conservative policy.

### B. Results

We have implemented a MATLAB simulator to test our network scenario described, both with per-bit simulation and using the approximate approach devised in Sec. III, with $\ell = 1$.

As a first result we wish to show that the network setup described in Sec. IV-A reaches the objective to allow for multiple communications to simultaneously co-exist in the same network neighborhood, without necessarily destroying one another. To this extent, in Fig. 2 we depict the average network throughput as a function of offered traffic $\lambda$, defined as before. Throughput is defined as the average number of 1000-bit streams that are heard and decoded correctly by their intended receiver (per frame). Depending on the backoff policy employed, throughput reaches a maximum (simulated) value of about 11.4 for Dest–Lock and about 8.3 for Node–Lock, showing that the first policy is more aggressive in transmission attempts as expected. Notice also that in an 802.11 completely connected network, the maximum number of such streams that could have been transmitted per frame is bounded by 1, as RTSs or CTSs reach all nodes during preliminary handshakes, thus blocking all but one radio link.

In Fig. 2, the performance prediction obtained using the approximation method of Sec. III is also shown. We see that both Node–Lock and Dest–Lock behavior is predicted with high precision in the uncongested region, where the network is sufficiently unloaded, so that throughput increases linearly with offered traffic. As the saturation region is approached, Node–Lock is very well approximated by the analytical approach, whereas for Dest–Lock the predicted maximum throughput is lower than the simulated one. As a matter of fact, the approximation accuracy is strongly dependent on the amount of unestimated and uncanceled interference affecting the received signal. Note that this kind of impairment does not arise due to the low number of lockable training sequences $N$, but rather due to the very protocol used. When performing RTS/CTS exchange in the first part of a frame, all transmitters send their RTSs, so that all receivers hear them simultaneously and build an as precise as possible estimate of the data traffic that will follow. RTSs also contain the number of antennas to be used, thus receivers may forecast how many sequences to reserve for interference cancellation. If, due to the CTS construction policy, a node is allowed transmission with fewer antennas than asked for, then some reserved sequences could be wasted for transmissions that do not actually take place. This phenomenon occurs much more frequently at higher traffic values, and even more for Dest–Lock, which allows more transmission attempts. Thus the approximation performs worse at high traffic, for aggressive MAC policies.

In Fig. 3, further evidence of this behavior is given, by showing the average ratio of transmitted 1000-bit streams per frame to correctly decoded ones. This quantity is meant to give a measure of packet success ratio in a more precisely characterized network scenario, where effects due to, e.g., access protocols and different received stream powers are included, thus depicting a globally more realistic performance than could be predicted by BER evaluation as in Fig. 1. As expected, there is a small difference between simulated and approximate success ratios for Node–Lock (around 2%), which grows for Dest–Lock up to 7.5%, explaining the behaviors in the throughput graph.

Other relevant performance metrics are predicted with fairly good precision by the analytical approximation. In Figs. 4

---

\(^3\)We have found that this is a good value to balance defence from network congestion (thus backoff performance) and average packet transmission delay, which increases with $W$, on the average. We will not focus on this trade–off here due to lack of space.
and 5, we depict the average time elapsed (in frames) from when a packet is generated to the successful transmission of all its 1000-bit streams, and the average number of activated transmitter–receiver links per frame, respectively.

The first metric is accumulated for those packet that incur complete successful transmission only (i.e., those that are not discarded due to timeouts), and is consistent with the trends observed in the throughput and packet success ratio results. Average delay is in fact very well predicted for NTA along with the Node–Lock policy, but a slight increase for Dest–Lock is experienced. The same reasons that explain why success ratio and throughput are lower in this case also apply here, because the higher number of transmission failures, predicted by approximation at high traffic, tends to lower throughput and also augment delay for complete packet transmission. Node–Lock instead is able to reduce the amount of unestimated interference affecting the data transmissions in each frame, but indeed leads to greater delays, as it expectedly forces any failing node to stop any transmission for the prescribed time.

The number of links activated in the network per frame (each possibly implying transmission with more than one antenna) is shown in Fig. 5. There a further clue about the high degree of parallelism reachable with our devised MAC protocol is given, along with insights on the differences between Node–Lock and Dest–Lock. Since the approximation gives satisfying results in both cases, we wish to focus here on network behavior. In saturation, Node–Lock can support a lower number of activated links than Dest–Lock, as expected, but allows for a stronger network activity for low traffic. This is due to the aggressiveness of Dest–Lock, which upon failure in receiving a CTS, lets nodes try a transmission to different neighbors in the following frames. Hence, more requests are injected in the network and receivers are forced to use CTSs for reducing the amount of spatial multiplexing allowed to senders, explaining the lowest slope of the Dest–Lock curve before reaching higher performance in the saturation zone.

V. CONCLUSIONS

When simulating network level performance in ad hoc networks, a feasible and sufficiently accurate analytical approximation of physical behavior is of paramount importance in order to get statistically meaningful results in reasonable time.

In our work, we considered a MIMO ad hoc network where spatial multiplexing capability offered by multiple antennas and properly designed receivers is exploited. In order to speed up simulation results, we deployed an analytical approximation for the algorithm driving signal separation at the receiving stage, showing that it offers satisfactory results and characterizing its performance under different network protocols. A cross–layer designed MAC policy including backoff has been evaluated through simulation and approximation, specifying when and why analytical predictions are more or less accurate.

ACKNOWLEDGEMENT

This work is supported in part by the U. S. Army Research Office under the Multi-University Research Initiative (MURI) grant no. W911NF-04-1-0224.

REFERENCES