Cooperative Power Scheduling for a Network of MIMO Links

Xiang Dong, Yue Rong, *Member, IEEE,* and Yingbo Hua, *Fellow, IEEE*

***Abstract*—A cooperative power scheduling algorithm developed by Wang, Krunz and Cui is extended for an ad hoc network of MIMO links. This algorithm, referred to as price-based iterative water filling (PIWF) algorithm, is a distributed algorithm by which each link computes its power scheduling through an iterative and cooperative process. The cooperation among all links is achieved by adaptive price factors applied by each link. Compared to a centralized power scheduling algorithm, the PIWF algorithm is much more efficient in computation although not as efficient in network throughput. Compared to a non-cooperative counter-part by Demirkol and Ingram where all price factors are zero, the PIWF algorithm requires additional in-network computation but is more efficient in network throughput.**

***Index Terms*—Ad hoc network of MIMO links, iterative water filling, power scheduling, network throughput, cooperative game.**

I. Introduction

For scattering-rich environment such as the case where the transmitters and/or receivers are close to ground, a MIMO (multi-input multi-output) wireless link is known to have a higher capacity than a traditional SISO (single-input single-output) wireless link. In recent years, there has been a strong interest in developing basic theories for networking of MIMO links. One of the important issues for a network of MIMO links is power scheduling, e.g., see [1], [2], [3], [4] and the references therein. In [1], a distributed iterative water filling algorithm was proposed where each link tries to maximize its own capacity in a non-cooperative way. This algorithm is also similar to the closed loop multiple-transmitter and multiple-receiver algorithm shown in [2]. In [3], a centralized algorithm was developed to maximize the network throughput where the temporal freedom in time or frequency was however not considered. In [4], both the spatial and temporal freedoms were exploited in a centralized scheduling algorithm.

On the other hand, there have been many research activities on power scheduling for networks of SISO links. Recent examples include [5], [6] and [7], where several different utility functions are considered. The work [7] by Wang, Krunz and Cui is particularly relevant to the studies in [1], [2], [3], [4]. In [7], the authors developed a distributed algorithm, which we refer to as the PIWF algorithm, to maximize a priced-based utility function for an ad hoc network of SISO links. The price factors are chosen in such a way that they are simple, adaptive and useful for an improved network throughput. In this letter, we present an extension of the PIWF algorithm for a network of MIMO links and compare its performance with [1] and [4]. We will also refer to the MIMO version of the PIWF algorithm simply as the PIWF algorithm.

Our study shows that the PIWF algorithm is indeed more efficient in network throughput than the algorithm in [1], referred to as DI algorithm, which is a special case of the PIWF algorithm with the price-factors set to zero. Compared to the centralized algorithm in [4], the PIWF algorithm is not as efficient in network throughput but is much more efficient in computation. We will also evaluate the network throughput of the PIWF algorithm using bits-meter per second per Hertz per node (bits-meter/s/Hz/node), which is a fundamental throughput unit introduced in [8] and also used recently in [9] and [10] for large ad hoc networks. We will show that for a medium sized network (64 nodes), the throughput of the PIWF algorithm can exceed substantially those in [9] and [10]. However, the complexity of the PIWF algorithm increases as the network size increases. For this reason, the PIWF algorithm is not suitable for large ad hoc networks.

II. Network Model and Problem Formulation

We assume that there are N active MIMO links that need to be scheduled to share K frequency bands within a time window of interest. Each MIMO link has N_t transmitting antennas for the transmitter and N_r receiving antennas for the receiver. The channel matrix between the transmitter of link j and the receiver of link i in band k is denoted by $H_{j,i}(k) \in C^{N_r \times N_t}$. The waveform transmitted from the source of link i in band k is $x_i(k) \in C^{N_t \times 1}$. The waveform received by the destination of link i in band k is $y_i(k) \in C^{N_r \times 1}$. The relationship between these waveforms is given by

$$y_i(k) = H_{i,i}(k)x_i(k) + \sum_{j=1, j\neq i}^{N} H_{j,i}(k)x_j(k) + w_i(k) \quad (1)$$

where $w_i(k)$ is the noise vector which is assumed to be white Gaussian with zero mean and the covariance matrix $N_0 I$. Here, N_0 is a scalar and I is an identity matrix.

The channel capacity (in bits/s/Hz) of link i in band k is known [12] to be

$$r_i(k) = \log_2 |I + P_i(k)G_i(k)| \quad (2)$$

where $P_i(k) = E\left\{x_i(k)x_i^H(k)\right\}$, E is the statistical expecta-

X. Dong and Y. Hua (corresponding author) are with the Department of Electrical Engineering, University of California, Riverside, CA, 92521. Emails: xdong001@ucr.edu and yhua@ee.ucr.edu. And Y. Rong is with the Department of Electrical and Computer Engineering, Curtin University of Technology, Bentley, WA 6102, Australia. E-mail: y.rong@curtin.edu.au. This work was supported in part by the U. S. Army Research Laboratory under the Collaborative Technology Alliance Program.

tion, ^{H} is the Hermitian transpose, and

$$G_i(k) = H_{i,i}^H(k)M_i^{-1}(k)H_{i,i}(k) \quad (3)$$

$$M_i(k) = N_0 I + \sum_{j=1,j\neq i}^{N} H_{j,i}(k)P_j(k)H_{j,i}^H(k) \quad (4)$$

Furthermore, if $G_i(k)$ is given and has the eigenvalue decomposition $G_i(k) = Q_i(k)\Lambda_i(k)Q_i^H(k)$ where $Q_i(k)$ is the matrix of its eigenvectors and $\Lambda_i(k) = diag\left[\lambda_i^{(1)}(k)\ \lambda_i^{(2)}(k)\cdots\lambda_i^{(N_t)}(k)\right]$ is the matrix of its eigenvalues, then the optimal structure of $P_i(k)$ to maximize the rate of link i is known [12] to be $P_i(k) = Q_i(k)\Pi_i(k)Q_i^H(k)$ where $\Pi_i(k) = diag\left[\pi_i^{(1)}(k)\ \ \pi_i^{(2)}(k)\ \ \cdots\ \ \pi_i^{(N_t)}(k)\right]$. Given the optimal $P_i(k)$, we can write $r_i(k) = \sum_{l=1}^{N_t}\log_2\left(1+\lambda_i^{(l)}(k)\pi_i^{(l)}(k)\right)$.

The main interest here is to develop a distributed algorithm for computing $\pi_i^{(l)}(k)$ for all i,k,l so that the network throughput is maximized (at least locally) subject to a power constraint. We will assume that there is a fixed value for the power of each link, i.e., we have $\overline{p}_i$ such that

$$\frac{1}{K}\sum_{k=1}^{K} tr\{P_i(k)\} = \frac{1}{K}\sum_{k=1}^{K}\sum_{l=1}^{N_t}\pi_i^{(l)}(k) \le \overline{p}_i \quad (5)$$

Although omitted in this letter, peak power constraints on individual $\pi_i^{(l)}(k)$ can also be introduced as in [7].

III. The PIWF Algorithm

We now follow the same principle in [7] to formulate a cooperative iterative water filling scheduling algorithm for computing $\pi_i^{(l)}(k)$.

First we define a cooperative utility (or objective) function for link i as follows:

$$\hat{u}_i = \frac{1}{K}\sum_{k=1}^{K}\omega_i r_i(k) - \frac{1}{K}\sum_{k=1}^{K}\sum_{l=1}^{N_t}\delta_i^{(l)}(k)\pi_i^{(l)}(k) \quad (6)$$

where ω_i can be any useful weight for link i (which can be a product of the distance of link i, a penalty factor between zero and one for practical coding and modulation, and other parameters), and $\delta_i^{(l)}(k)$ is a (non-negative) price factor for channel k, link i and sub-stream l. The first term in $\hat{u}_i$ is the capacity of link i, and the second term in $\hat{u}_i$ is a penalty term for link i. It is the second term that provides a cooperation between link i and other links. Clearly, the rate of link i is directly affected by the power distribution $\pi_i^{(l)}(k)$ for all l and k. At the same time, $\pi_i^{(l)}(k)$ for all l and k also cause interferences to other links. The second term reflects a price that inhibits link i from increasing its own rate selfishly. Note that each penalty component to the utility function $\hat{u}_i$ is purposely chosen to be simple, which is the product of a price factor and the corresponding power component. If $\delta_i^{(l)}(k)$ for all (k,l) are given, and link i wishes to find $\pi_i^{(l)}(k)$ to maximize $\hat{u}_i$, then the optimal $\pi_i^{(l)}(k)$ for all (k,l) can be shown (see the appendix) to be

$$\pi_i^{(l)}(k) = \left[\frac{\omega_i\log_2 e}{\beta_i + \delta_i^{(l)}(k)} - \frac{1}{\lambda_i^{(l)}(k)}\right]^+ \quad (7)$$

where $[x]^+ = \max(x,0)$, and $\beta_i \ge 0$ is such that $\sum_{k=1}^{K}\sum_{l=1}^{N_t}\pi_i^{(l)}(k) \le K\overline{p}_i$. Furthermore, if there is no positive β_i such that $\sum_{k=1}^{K}\sum_{l=1}^{N_t}\pi_i^{(l)}(k) = K\overline{p}_i$, then choose $\beta_i = 0$. The above is what is called priced-based water filling [7]. If there is a peak power constraint on $\pi_i^{(l)}(k)$, the optimal $\pi_i^{(l)}(k)$ is also given by (7) except for a clipping at the peak value [7].

In order to determine a right set of the price factors $\delta_i^{(l)}(k)$ for all channels, links and sub-streams, we set

$$\delta_i^{(l)}(k) = -\frac{\partial}{\partial\pi_i^{(l)}(k)}\sum_{j=1,j\neq i}^{N}\omega_j r_j(k) \quad (8)$$

Namely, each price factor for link i is set to be equal to the negative of the derivative of the sum of the weighted rates for all other links with respect to the corresponding power component. This rule for selecting the price factors is heuristically sound. It was also obtained in [7] by comparing the KKT conditions of $\max_{\pi_i^{(l)}(k)}\sum_{j=1}^{N}\hat{u}_j$ and the KKT conditions of $\max_{\pi_i^{(l)}(k)}\hat{u}_i$ for each i.

It should be noted that the domain for the summation in (8) can be reduced to a set only comprising the "neighboring" links of link i. A link j is considered to be a neighbor of link i if $\frac{\partial}{\partial\pi_i^{(l)}(k)}\omega_j r_j(k)$ is not negligible, or equivalently if the channel gain between the transmitter of link i and the receiver of link j is not negligible.

We next provide more explicit expressions for computing the price factors. It follows from (8) that

$$\begin{aligned}
&\delta_i^{(l)}(k)\\
&= -\frac{\partial}{\partial\pi_i^{(l)}(k)}\sum_{j\neq i}\omega_j\log_2|I + P_j(k)G_j(k)|\\
&= -\log_2 e\sum_{j\neq i}\omega_j tr\left[(I+P_j(k)G_j(k))^{-1}P_j(k)\right.\\
&\quad\left.\cdot\frac{\partial}{\partial\pi_i^{(l)}(k)}G_j(k)\right] \quad (9)
\end{aligned}$$

where we have used $\partial\ln|X| = tr(X^{-1}\partial X)$. Furthermore, using (3), (4) and $\partial(X^{-1}) = -X^{-1}(\partial X)X^{-1}$, we can write

$$\begin{aligned}
&\frac{\partial}{\partial\pi_i^{(l)}(k)}G_j(k)\\
&= H_{j,j}^H(k)\left(\frac{\partial}{\partial\pi_i^{(l)}(k)}M_j^{-1}(k)\right)H_{j,j}(k)\\
&= -H_{j,j}^H(k)M_j^{-1}(k)\left(\frac{\partial}{\partial\pi_i^{(l)}(k)}M_j(k)\right)M_j^{-1}(k)H_{j,j}(k)\\
&= -H_{j,j}^H(k)M_j^{-1}(k)H_{i,j}(k)\left(\frac{\partial}{\partial\pi_i^{(l)}(k)}P_i(k)\right)\\
&\quad\cdot H_{i,j}^H(k)M_j^{-1}(k)H_{j,j}(k)\\
&= -T_{i,j}^H(k)\left(\frac{\partial}{\partial\pi_i^{(l)}(k)}diag[\pi_i^{(1)}(k)\cdots\pi_i^{(N_t)}(k)]\right)T_{i,j}(k)\\
&= -T_{i,j}^H(k)diag\left[\overbrace{\underbrace{0\ \cdots\ 0}_{l-1}1\ 0\ \cdots\ 0}^{N_t}\right]T_{i,j}(k) \quad (10)
\end{aligned}$$

where $T_{i,j}(k) = Q_i^H(k)H_{i,j}^H(k)M_j^{-1}(k)H_{j,j}(k)$. Since $T_{i,j}(k)$ is proportional to $H_{i,j}$, the summation in (9) needs to be over the set of all j where $j \neq i$ and $H_{i,j} \neq 0$.

The computation of the price factors $\delta_i^{(l)}(k)$ and the power components $\pi_i^{(l)}(k)$ can be implemented in a sequential or concurrent fashion [7]. In our simulation, we will consider the sequential implementation as follows.

Step 1: The network conducts a full range of channel estimation. By the end of this process, link i knows $H_{j,j}$, $H_{j,i}$ and $H_{i,j}$ for all j. Also, $P_i(k) = 0$, $M_i^{-1}(k) = \frac{1}{N_0}I$ and $G_i(k) = \frac{1}{N_0}H_{i,i}^H H_{i,i}$ are set for all (i,k), and a counter is initialized at $t = 0$.

Step 2: For each new value of t, set $i = 0$.

Step 3: For each new value of i, link i makes sure that it has collected the most recent $P_j(k)$, $M_j^{-1}(k)$ and $G_j(k)$ for all $j \neq i$ and all k from other links. Then, link i computes (or updates) $M_i^{-1}(k)$ and $G_i(k)$ according to (4) and (3). From $G_i(k)$, link i obtains its eigenvalues $\lambda_i^{(l)}(k)$ for all (l,k) and its eigenvector matrix $Q_i(k)$ for all k. Then, link i computes the price factors $\delta_i^{(l)}(k)$ for all (l,k) according to (9)-(10). Finally, link i updates its power components $\pi_i^{(l)}(k)$ for all (l,k) according to (7) and consequently updates $P_i(k)$ for all k.

Step 4: Set $i = i + 1$ and go to Step 3 until $i = N$.

Step 5: Set $t = t + 1$ and go to Step 2 until convergence.

It should be noted that in practice there are various factors that can reduce the actual network throughput, which include non-ideal coding and modulation, channel estimation errors, quantization errors, etc. However, these errors can be controlled by the network designers for certain applications. The channel estimation issue studied in [13] is among many useful considerations in practice. There are also many different ways for channel estimation. The PIWF algorithm clearly consumes an additional amount of the network resources. This overhead should be relatively small compared to the resources used for data transmissions (apart from the information exchanges for in-network computations governed by the PIWF algorithm), or otherwise the PIWF algorithm should not be used. How to minimize the overhead of the PIWF algorithm without reducing its performance is a useful topic for future research. How to quantify the impact of channel estimation errors and other sources of errors on the performance of the PIWF algorithm is also an important future topic.

IV. Performance Evaluation

A. Comparison with the OPS and DI Algorithms

We first demonstrate a comparison of the PIWF algorithm with the OPS algorithm [4] and the DI algorithm [1]. The DI algorithm is simply the PIWF algorithm with all price factors set to zero. The OPS algorithm is a centralized algorithm which maximizes directly the network throughput using a gradient-based search. We will choose the same parameters as used in [4], i.e., the signal model for link i ($i = 1, 2, ..., N$) and channel k ($k = 1, 2, ..., K$) is

$$y_i(k) = H_{i,i}x_i(k) + \gamma \sum_{j=1, j\neq i}^{N} H_{j,i}x_j(k) + w_i(k) \qquad (11)$$

where the channel matrices $H_{j,i} \in C^{N_r \times N_t}$ are assumed to be invariant to the channel index k and consist of i.i.d. complex Gaussian random entries with zero mean and unit variance, and $w_i(k)$ is complex Gaussian white noise vector with zero mean and the covariance matrix $N_0 I$, and γ is a parameter used to model the total interference for each link. We further assume that $N = 6$, $K = 6$, $N_t = N_r = 2$, and $10\log_{10}\frac{\bar{p}_i}{N_0} = 20$ where $\bar{p}_i = \frac{1}{K}\sum_{k=1}^{K} E\{\|x_i(k)\|^2\} = 1$. A nominal interference to noise ratio (INR) in dB is defined as $10\log_{10}\frac{\gamma^2}{N_0}$. Fig. 1 shows the sum capacity of all links

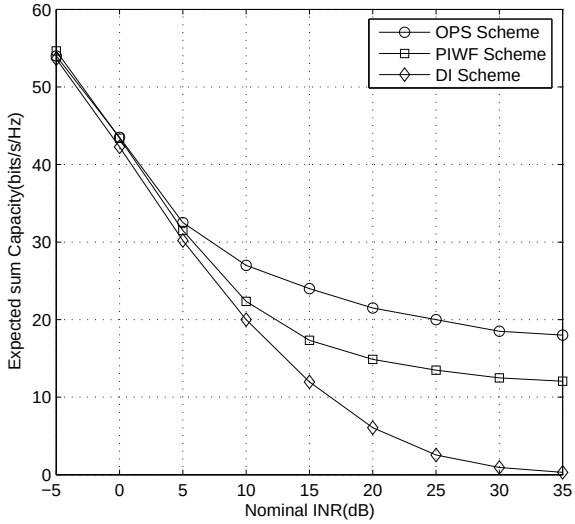

Fig. 1. Sum capacity in bits/s/Hz versus nominal INR based on the same data as in [4].

averaged over $R = 100$ channel realizations versus the INR in dB for each of the three algorithms: OPS, PIWF and DI, all with the unit weight $\omega_i = 1$. The averaged sum capacity is computed by $\frac{1}{KR}\sum_{r=1}^{R}\sum_{k=1}^{K}\sum_{i=1}^{N} r_i^{(r)}(k)$ where $r_i^{(r)}(k)$ is the value of $r_i(k)$ after the rth run. We see from this figure that when the INR is small, all three algorithms yield the same capacity, which is expected. But when the INR is large, the capacity differences of the three algorithms become large. The OPS has the largest capacity, the DI has the smallest capacity, and the PIWF is somewhere in between. This figure also shows that the capacity gap between the OPS and the PIWF is still quite large when the INR is large. However, the advantage of the PIWF over the OPS is that the computational cost of the PIWF is much smaller. Indeed, for a medium-sized network as shown next, the complexity of the OPS algorithm became so overwhelming that its simulation was not feasible. Compared to the DI, however, the PIWF is more costly due to the computation of the price factors.

B. Throughput of the PIWF algorithm for a larger network

We now consider a larger network of MIMO links as illustrated in Fig. 2, where there are 64 nodes distributed randomly (with a uniform statistical distribution) in a square area, and 32 links ($N = 32$) are formed randomly among these nodes. The node density (i.e., the number of nodes in

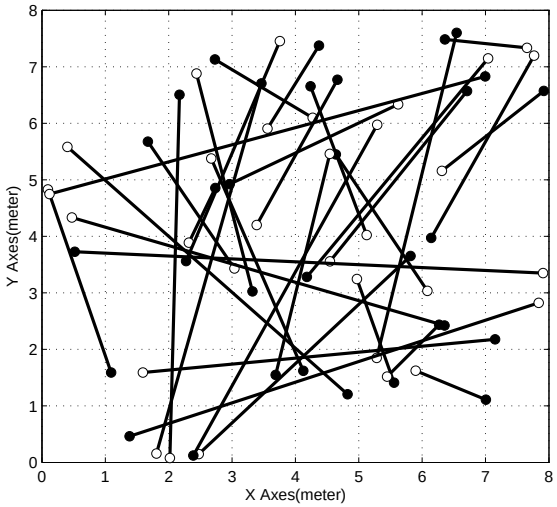

Fig. 2. A network of 64 randomly distributed nodes and 32 links for random source-destination pairs. Node density is one.

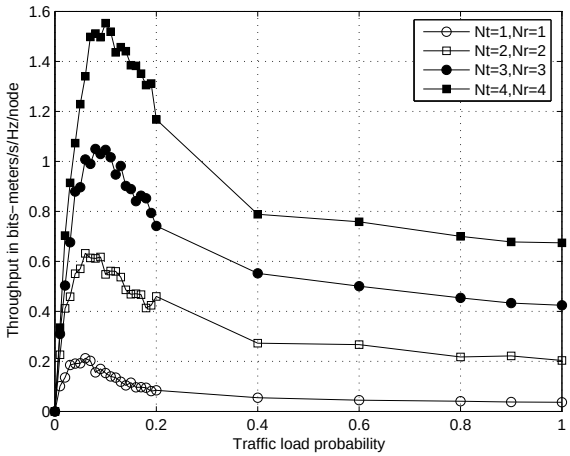

Fig. 3. The network throughput in bits-meter/s/Hz/node versus the traffic load probability ζ, where $K = 1$. The network topology is shown in Fig. 2.

a unit area) is one. In our simulation, we had 100 network-wise channel realizations, and for each channel realization we formed a new random set of 64 nodes and a new random set of 32 links. The signal model for this network is given by (1), where we assume that $H_{i,j}(k) = \frac{1}{d_{i,j}^{\alpha/2}} \tilde{H}_{i,j}(k)$, $d_{i,j}$ is the distance between the transmitter of link i and the receiver of link j, $\alpha = 3$ is the path loss factor, and $\tilde{H}_{i,j}(k)$ for all (i, j, k) consist of i.i.d. complex Gaussian random entries of zero mean and unit variance. The noise variance N_0 is such that $10 \log_{10} \frac{\bar{p}_i}{N_0} = 40$ where $\bar{p}_i = 1$.

Fig. 3 shows the network throughput in bits-meter/s/Hz/node versus the traffic load probability ζ. Here, we define ζ as the probability that each link (or link i) has a packet to transmit. For a realization, if a link has no packet to transmit, it will not take part in the power scheduling. The throughput in bits-meter/s/Hz/node is

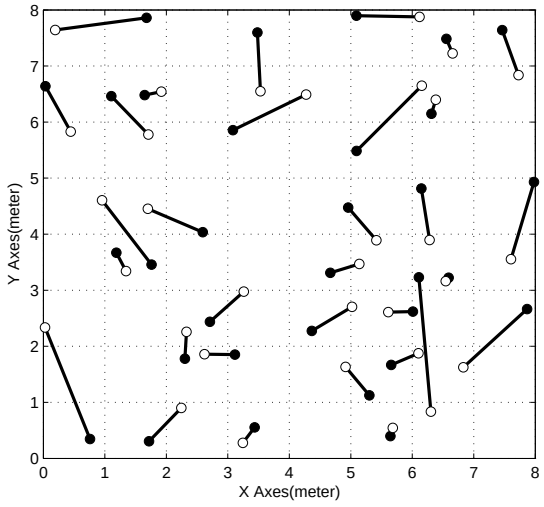

Fig. 4. A network of 64 randomly distributed nodes and 32 links between (approximately) nearby nodes.

computed by

$$C_{mean} = \frac{1}{2NKR} \sum_{r=1}^{R} \sum_{i=1}^{N} \sum_{k=1}^{K} \omega_i r_i^{(r)}(k) \qquad (12)$$

where $\omega_i = d_{i,i}$. For this figure, only a single channel is used, i.e., $K = 1$. We see that the capacity increases as the number of antennas increases, which is expected. Interestingly, we also see that for $1 \leq N_t = N_r \leq 4$, the capacity increases almost linearly with the number of antennas. Another useful observation is that when ζ is relatively large, the capacity of the PIWF for such a network topology as in Fig. 2 decreases as ζ increases. This observation is complementary to an observation shown in [10] that the shortest distance transmissions are the most efficient in network throughput for large networks unless the traffic load is low. Indeed, our simulations also confirmed that if we form all links only between nearby neighbors, e.g., see Fig. 4, the network throughput of the PIWF algorithm becomes an increasing function of ζ for the entire range $0 \leq \zeta \leq 1$. However, a disadvantage of the nearby neighbor transmissions is the requirement of routing and flow control protocols between sources and destinations, which adds more complexity.

Alternatively, if we increase the number of bands, as shown in Fig. 5 where $N_t = N_r = 1$, the capacity at higher traffic load can be improved.

In Table I, we show a range of values of the network throughput under the PIWF algorithm for $1 \leq N_t = N_r \leq 6$ and $K = 1, 2, 4, 8, 16, 32$. We see that the throughput in bits-meter/s/Hz/node is an increasing function of $N_t = N_r$ and K.

It is useful to note that the values of the network throughput in bits-meter/s/Hz/node (under unit node density) shown in Fig 3, Fig 5 and Table I (especially for large $N_t = N_r$ and large K) exceed those shown in [9] and [10]. However, the network considered in [9] and [10] is much larger than the one in Fig. 2, and the PIWF algorithm is much more complex than those

TABLE I
THE NETWORK THROUGHPUT IN BITS-METER/S/HZ/NODE OF THE PIWF ALGORITHM VERSUS THE NUMBER OF ANTENNAS AND THE NUMBER OF CHANNELS. $\bar{N} = N_t = N_r$. $\zeta = 0.5$. THE NETWORK TOPOLOGY IS SHOWN IN FIG. 2.

C_{mean}	$K=1$	$K=2$	$K=4$	$K=8$	$K=16$	$K=32$
$\bar{N}=1$	0.0513	0.2096	0.4005	0.5553	0.5897	0.6114
$\bar{N}=2$	0.2641	0.5544	0.9545	1.1616	1.1713	1.1793
$\bar{N}=3$	0.5149	1.0609	1.5562	1.7353	1.7607	1.7682
$\bar{N}=4$	0.7988	1.5954	2.0954	2.2455	2.2996	2.3509
$\bar{N}=5$	1.0952	2.1062	2.6106	2.7446	2.8000	2.8100
$\bar{N}=6$	1.4140	2.4604	3.1140	3.2915	3.4740	3.6033

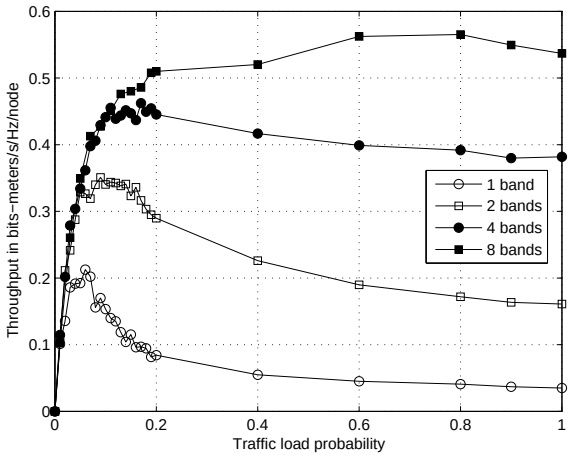

Fig. 5. The network throughput in bits-meter/s/Hz/node versus the traffic load probability ζ, where $N_t = N_r = 1$. The network topology is shown in Fig. 2.

in [9] and [10]. The PIWF algorithm requires and exploits the knowledge of all interfering sources for each link. For the case of 32 links, 4 bands and 4 antennas on each node, for example, it took about 5 minutes (on a desktop) for the PIWF algorithm to complete the computation for a single channel realization under full load condition. For the network size (over 200 nodes) considered in [9] and [10], the simulation of the PIWF algorithm would be infeasible. Although the neighborhood of each link as used in (8) may be much smaller than the network size, the coupling between many neighborhoods makes it difficult for the PIWF algorithm to converge. In fact, we also observed that if the radius of support (with respect to link i) for the summation in (9) reduces, the network throughput produced by the PIWF algorithm also reduces.

V. CONCLUSION

In this letter, we have presented an extension and evaluation of the PIWF algorithm by Wang, Krunz, and Cui [7] for an ad hoc wireless network of MIMO links. The extension of the PIWF algorithm to the MIMO links requires each link to have a full knowledge of the channel matrices with respect to all other links, and the computations need to be further increased. The study shown in this letter provides a useful perspective of the PIWF algorithm along with other related algorithms, which is not available in [7]. This work also suggests that there is a wide range of complexity-performance tradeoffs in the power scheduling algorithms. Different distributed algorithms may have different complexities and require different levels of cooperations between links. The PIWF algorithm is a distributed algorithm which needs a high level cooperation between links, which may or may not be feasible depending on specific applications or network environment. The mathematical analysis of the convergence behaviors of the PIWF algorithm remains an open problem.

APPENDIX A
PROOF OF (7)

The optimization problem here is that for each i, $\min_{\{\pi_i^{(l)}(k)\}} -K\hat{u}_i$ subject to $-\pi_i^{(l)}(k) \le 0$ and $\sum_{k=1}^{K}\sum_{l=1}^{N_t}\pi_i^{(l)}(k) \le K\bar{p}_i$. The corresponding Lagrangian function is

$$L_i = -K\hat{u}_i - \sum_{k=1}^{K}\sum_{l=1}^{N_t}\alpha_i^{(l)}(k)\pi_i^{(l)}(k) + \beta_i\left(\sum_{k=1}^{K}\sum_{l=1}^{N_t}\pi_i^{(l)}(k) - K\bar{p}_i\right) \quad (13)$$

where $\hat{u}_i$ is given in (6), $\alpha_i^{(l)}(k) \ge 0$ and $\beta_i \ge 0$ are Lagrange multipliers. The K.K.T. conditions [11] for the optimal $\pi_i^{(l)}(k)$ are given as follows:

$$\frac{\partial}{\partial \pi_i^{(l)}(k)} L_i = -\frac{\partial}{\partial \pi_i^{(l)}(k)}\omega_i r_i(k) + \delta_i^{(l)}(k) - \alpha_i^{(l)}(k) + \beta_i = 0 \quad (14)$$

$$\pi_i^{(l)}(k) \ge 0 \quad (15)$$

$$\alpha_i^{(l)}(k)\pi_i^{(l)}(k) = 0 \quad (16)$$

$$\sum_{k=1}^{K}\sum_{l=1}^{N_t}\pi_i^{(l)}(k) - K\bar{p}_i \le 0 \quad (17)$$

$$\beta_i\left(\sum_{k=1}^{K}\sum_{l=1}^{N_t}\pi_i^{(l)}(k) - K\bar{p}_i\right) = 0 \quad (18)$$

From (14) and (16), for those (l,k) where $\pi_i^{(l)}(k) > 0$, we have $\alpha_i^{(l)}(k) = 0$ and

$$\frac{\partial}{\partial \pi_i^{(l)}(k)}\omega_i r_i(k) - \delta_i^{(l)}(k) - \beta_i = 0 \quad (19)$$

We also know

$$\begin{aligned}
&\frac{\partial}{\partial \pi_i^{(l)}(k)} r_i(k) \\
= \;& \frac{\partial}{\partial \pi_i^{(l)}(k)} \log_2 \left| I + P_i(k) G_i(k) \right| \\
= \;& \frac{\partial}{\partial \pi_i^{(l)}(k)} \sum_{l=1}^{N_t} \log_2 \left(1 + \pi_i^{(l)}(k) \lambda_i^{(l)}(k) \right) \\
= \;& (\log_2 e) \frac{\lambda_i^{(l)}(k)}{1 + \pi_i^{(l)}(k) \lambda_i^{(l)}(k)} \qquad (20)
\end{aligned}$$

Then, using (20) in (19) yields

$$\pi_i^{(l)}(k) = \left[\frac{\omega_i \log_2 e}{\beta_i + \delta_i^{(l)}(k)} - \frac{1}{\lambda_i^{(l)}(k)} \right]^+ \qquad (21)$$

where $[x]^+ = \max(x, 0)$ is chosen to accommodate those (l, k) where $\pi_i^{(l)}(k) = 0$. Namely, for those (l, k) where $\pi_i^{(l)}(k) = 0$, there must be $\alpha_i^{(l)}(k) \geq 0$ such that (14) holds, which is equivalent to the operation $[x]^+$ used in (21). From (17) and (21), the parameter $\beta_i \geq 0$ should be such that

$$\sum_{k=1}^{K} \sum_{l=1}^{N_t} \left[\frac{\omega_i \log_2 e}{\beta_i + \delta_i^{(l)}(k)} - \frac{1}{\lambda_i^{(l)}(k)} \right]^+ \leq K \overline{p}_i \qquad (22)$$

Note that because of the price factors $\delta_i^{(l)}(k)$, it is possible that the equality in (22) may not need to be achieved for some cases. Indeed, since $\pi_i^{(l)}(k)$ as in (21) is a decreasing function of β_i, if $\sum_{k=1}^{K} \sum_{l=1}^{N_t} \left[\frac{\omega_i \log_2 e}{\delta_i^{(l)}(k)} - \frac{1}{\lambda_i^{(l)}(k)} \right]^+ < K \overline{p}_i$, then the optimal β_i is zero according to (18).

REFERENCES

[1] M. F. Demirkol and M. A. Ingram, "Power-controlled capacity for interfering MIMO links," in *Proc. IEEE VTC*, Atlantic City, NJ, Oct. 2001, vol. 1, pp. 187-191.

[2] F. R. Farrokhi, A. Lozano, G. J. Foschini, and R. A. Valenzuela, "Spectral efficiency of FDMA/TDMA wireless systems with transmit and receive antenna arrays," *IEEE Transactions on Wireless Communications*, Vol. 1, No. 4, pp. 591-599, Oct 2002.

[3] S. Ye and R. S. Blum, "Optimized signaling for MIMO interference systems with feedback," *IEEE Trans. Signal Processing*, vol. 51, pp. 2839-2848, Nov. 2003.

[4] Y. Rong and Y. Hua, "Optimal power schedule for distributed MIMO links," *IEEE Transactions on Wireless Communications*, Vol. 7. No. 8, pp. 2896-2900, August 2008.

[5] C. U. Saraydar, N. B. Mandayam, and D. J. Goodman, "Efficient power control via pricing in wireless data networks," *IEEE Transactions on Communications*, vol. 50, no. 2, pp. 291-303, February 2002

[6] M. Xiao, N. B. Shroff, E. K. P. Chong, "A utility-based power-control scheme in wireless cellular systems," *IEEE/ACM Transactions on Networking*, Volume 11, Issue 2, April 2003

[7] F. Wang, M. Krunz, and S. Cui, "Price-based spectrum management in cognitive radio networks," *IEEE Journal of Selected Topics in Signal Processing*, Vol. 2, No. 1, pp. 74-87, February 2008.

[8] P. Gupta and P. R. Kumar, "The capacity of wireless networks," *IEEE Transactions on Information Theory*, vol. 46, no. 2, pp. 388-404, 2000.

[9] B. Zhao and Y. Hua, "A distributed medium access control scheme for a large network of wireless routers," *IEEE Transactions on Wireless Communications*, vol. 7, no. 5, pp. 1614-1622, 2008.

[10] Y. Yu, Y. Huang, B. Zhao, and Y. Hua, "Further development of synchronous array method for ad hoc wireless networks," *EURASIP Journal on Advances in Signal Processing - Special Issue on Cross-Layer Design for the Physical, MAC, and Link Layer in Wireless Systems*, Vol. 2009, Article ID 873202, 14 pages, September 2008.

[11] D. Bertsekas, *Nonlinear Programming*, Second Edition, Athena Scientific, 1995.

[12] D. Tse and P. Viswanath, *Fundamentals of Wireless Communications*, Cambridge Press, 2005.

[13] D. Chiarotto, P. Casari, and M. Zorzi, "On the statistics and MAC implications of channel estimation errors in MIMO ad hoc networks," *Proc of IEEE Globecom 2008*, New Orlean, LA.