Distributed Interference Pricing for OFDM Wireless Networks with Non-Separable Utilities

Changxin Shi, Randall A. Berry, and Michael L. Honig
Department of Electrical Engineering and Computer Science
Northwestern University, Evanston, Illinois 60208
Email: sze@northwestern.edu, {rberry, mh}@ece.northwestern.edu

***Abstract*—We present a distributed algorithm for allocating power among multiple interfering transmitters in a wireless network using Orthogonal Frequency Division Multiplexing (OFDM). The algorithm attempts to maximize the sum over user utilities, where each user's utility is a function of his total transmission rate. Users exchange *interference prices* reflecting the marginal cost of interference on each sub-channel, and then update their power allocations given the interference prices and their own channel conditions. A similar algorithm was studied earlier assuming that each user's utility function is a separable function of the user's rate per sub-channel. Here, we do not assume this separability. We give a different algorithm for updating each user's power allocation and show that this algorithm converges monotonically. Numerical results comparing this algorithm to several others are also presented.**

I. Introduction

Mitigating interference is critical to enable co-existence of multiple wireless transmitters in a common spectrum band. In an OFDM system interference can be controlled by adjusting each user's power allocation across the available sub-channels. Ideally, this should be done to optimize an overall network objective. In the absence of centralized control (e.g., in an ad hoc network) this optimization must be done in a distributed manner with limited information exchange. In this paper we consider such a distributed algorithm. The objective is to maximize total network performance, namely, total utility summed over all users, where each user's utility is a function of his total rate summed over all sub-channels. Each user's transmission rate per sub-channel depends on the received signal-to-interference plus noise ratio (SINR) on that sub-channel.

The algorithm we study here is an iterative scheme in which each user exchanges *interference prices* with neighboring users on each sub-channel. Each price reflects the marginal cost of increasing interference to that user on the particular sub-channel. Given a set of interference prices, a user then updates his power allocation. For example, one possibility is a "best response" update, in which the user maximizes his own utility minus the "cost" associated with the interference prices. These prices and the users' power allocations are then iteratively updated over time. This *Asynchronous Distributed Pricing (ADP)* algorithm (with best response updates), was introduced in [1] for a single-channel wireless network and the OFDM model studied here. [1]

It was shown in [1] that the ADP algorithm for OFDM converges with a specific class of *channel-separable utilities*, i.e., each user's utility is the sum over utilities associated with each sub-channel. If a user's utility is a function of his total rate, then this limits the results in [1] to utility functions, which are linear in rate. In this paper, we do not make this restriction, i.e., we allow each user's utility function to be a non-separable function of the utilities (or equivalently, powers) assigned across sub-channels. With non-separable utilities, we give a modified version of the ADP algorithm and prove that it converges under suitable conditions.

Other related work on distributed power allocation with frequency-selective channels includes iterative waterfilling introduced in [3]. In that algorithm nodes do not need to exchange any information about interference levels, but in general it may not converge or may converge to a sub-optimal allocation (e.g., see [4]–[6]). Iterative waterfilling can be viewed as a best response update in a non-cooperative game. In [7] a repeated game for power allocation with non-cooperative users is formulated, which enforces a Pareto efficient equilibrium. That generally requires more information exchange than required by the algorithms considered here. In [8] an approach to distributed power allocation in a single channel network is given based on a distributed gradient projection algorithm. Our numerical results compare the performance of a similar gradient algorithm for OFDM with the other distributed algorithms presented.

In the next section, we state the model and the optimization problem. We then decompose the problem into distributed sub-problems for each user to solve. In Section III, we propose a distributed algorithm and compare with some other existing algorithms. Simulation results are presented in Section IV, and conclusions are in Section V.

II. System Model

We consider an OFDM wireless network with M distinct pairs of transmitters and receivers. We will refer to each transmitter-receiver pair as a "user". All transmitters use the

This research was supported in part by ARO under grant W911NF-06-1-0339 and by NSF under grant CNS-0626558.

[1]A similar algorithm for a single channel network was also studied in [2]

same bandwidth of B Hz, which is is divided into K equal-sized sub-channels (i.e., either tones or groups of adjacent tones). Each sub-channel is modeled as a flat fading Gaussian interference channel. The quality of service for each user i is determined by a utility function $u_i(c_i)$, which is a monotonically increasing and concave function of user i's rate c_i. User i's rate is given by the Shannon capacity, assuming interference is treated a noise, i.e.

$$c_i = \sum_{k=1}^{K} \frac{B}{K} \log \left(1 + \frac{p_i^k H_{ii}^k}{n_0 + \sum_{j \neq i} p_j^k H_{ji}^k} \right), \tag{1}$$

where H_{ij}^k denotes the channel gain from transmitter i to receiver j on sub-channel k, p_i^k denotes the power transmitter i allocates to sub-channel k, and n_0 is the total noise power for each sub-channel.

Let $\mathbf{p_i} = \{p_i^1, p_i^2, \cdots, p_i^K\}$ denote the power profile of user i and let $\mathbf{P} = \{\mathbf{p_1}, \mathbf{p_2}, \cdots, \mathbf{p_M}\}$ be the vector of power profiles across all users. The optimization problem we would like to solve is then given by

$$\begin{aligned} \max_{\mathbf{P}} \quad & \sum_{i=1}^{M} u_i(c_i(\mathbf{P})) && (\mathrm{P}_0) \\ \text{s.t.} \quad & \sum_{k=1}^{K} p_i^k \leq P_i^{max} \qquad \text{for all } i \\ & p_i^k \geq 0 \qquad \text{for all } i \text{ and } k \end{aligned}$$

where P_i^{max} denotes the power constraint of user i. Note that in general even with a concave utility function, the objective of Problem P_0 is not concave in the power allocation and so may have multiple local optima. In a centralized manner, a local optimum can be found using standard optimization algorithms. However, our focus is on a distributed solution.

Any local optimum $\mathbf{P}^* = \{\mathbf{p_1^*}, \mathbf{p_2^*}, \cdots, \mathbf{p_M^*}\}$ of Problem P_0 must satisfy the Karush-Kuhn-Tucker (KKT) conditions [9], i.e. there exist unique Lagrange multipliers $\lambda_i (i = 1, \cdots, M)$ such that for all $i = 1, \cdots, M$:

$$\frac{\partial u_i(c_i(\mathbf{P}^*))}{\partial p_i^{k^*}} + \sum_{j \neq i} \frac{\partial u_j(c_j(\mathbf{P}^*))}{\partial p_i^{k^*}} - \mu_i + \lambda_i^k = 0, \ \forall k, \tag{2}$$

$$\mu_i \left(\sum_{k=1}^{K} p_i^{k^*} - P_i^{max} \right) = 0, \tag{3}$$

$$\lambda_i^k p_i^k = 0, \ \forall k, \tag{4}$$

$$\mu_i \geq 0, \text{ and } \lambda_i^k \geq 0 \qquad \forall k. \tag{5}$$

Using the KKT conditions, we now give a decomposition of the original problem into M subproblems. Following [1] we define the interference price π_i^k for user i on sub channel k as

$$\pi_i^k = -\frac{\partial u_i(c_i(\mathbf{P}))}{\partial I_i^k(\mathbf{P_{-i}})} \tag{6}$$

where $I_i^k(\mathbf{P_{-i}}) = \sum_{j \neq i} p_j^k H_{ji}^k$ is the interference power for user i on sub-channel k, and $\mathbf{P_{-i}}$ denotes the power profile for all users other than user i. This interference price can be viewed as marginal cost to user i per unit of interference power on the k-th sub-channel. Let $\pi_\mathbf{i} = \{\pi_i^1, \cdots, \pi_i^K\}$ denote the vector of interference prices for a given user i.

Using (6), condition (2) can be written as

$$\frac{\partial u_i(c_i(\mathbf{P}^*))}{\partial p_i^{k^*}} - \sum_{j \neq i} \pi_j^k H_{ij}^k = \mu_i - \lambda_i^k \qquad \forall \quad i, k. \tag{7}$$

Given fixed interference prices, and fixing the power profile of each user except user i, it can be seen that (7) and conditions (3)-(5) are the KKT conditions of the following optimization problem.

$$\begin{aligned} \max_{\mathbf{p_i}} \quad & u_i(c_i(\mathbf{p_i})) - \sum_{k=1}^{K} p_i^k \left(\sum_{j \neq i} \pi_j^k H_{ij}^k \right) && (\mathrm{P}_i) \\ \text{s.t.} \quad & \sum_{k=1}^{K} p_i^k \leq P_i^{max}, \\ & p_i^k \geq 0, \text{ for all } k. \end{aligned}$$

Moreover, the objective in this problem is strictly concave and can be shown to have a unique optimal solution (satisfying the KKT conditions).

We can view the preceding procedure as a decomposition of the original problem P_0 into M sub-problems, each with the form of Problem P_i. In each sub-problem P_i, only user i adjusts his power over all sub-channels. Of course, this changes the utility objective for other users because of the changing interference. The interference prices indicate this effect to each user.

III. Distributed Pricing Algorithms

We are now ready to discuss distributed algorithms for solving Problem P_0. We begin by reviewing the ADP algorithm from [1]. The main idea behind this algorithm is for each user i to iteratively solve the sub-problems P_i given the current interference prices and power profile of the other users and then to recalculate the corresponding interference price and repeat. Formally we summarize this as follows:

1) Each user i chooses an initial power profile $\mathbf{p_i}$ satisfying the power constraint.
2) Using (6), each user i calculates the interference price vector $\pi_\mathbf{i}$ given the current power profiles and announces this to every other user.
3) At each time n, one user i is randomly selected to maximize its payoff function $s_i(\mathbf{p_i})$ and update its power profile, given the other user's power profiles $\mathbf{P_{-i}}$ and price vectors π, i.e.,

$$\mathbf{p_i}(n+1) = \arg \max_{\mathbf{p_i} \in \Pi_i} s_i(\mathbf{p_i}; \mathbf{P_{-i}}(n)), \tag{8}$$

where

$$s_i(\mathbf{p_i}; \mathbf{P_{-i}}) = u_i(c_i(\mathbf{P})) - \sum_{k=1}^{K} p_i^k \left(\sum_{j \neq i} \pi_j^k H_{ij}^k \right) \tag{9}$$

and Π_i denotes the set of feasible power allocations for user i.

4) Go to step 2 and repeat.

In [1], a version of this algorithm was studied for both a single channel (i.e. K=1) wireless network and a multi-channel (i.e. OFDM) network. A key idea is to view the steps in this algorithm as best response updates in a non-cooperative game in which each player maximizes a pay-off function given by (9). For a single channel network convergence of the algorithm was proved by showing that under suitable assumption this is a supermodular game. This analysis was then extended to a multi-channel model assuming that each users utility was separable across carriers.[2] Here we do not require separable utilities, which makes the corresponding game no longer supermodular and so the analysis in [1] does not apply.[3]

A. Modified ADP Algorithm

We now give a modified version of the ADP algorithm, which we will show converges for the non-separable case. There are two main modifications in this algorithm, which we make to simplify the analysis.

The first modification is to linearize each users utility around the current rate before calculating its best response as in (8). In other words, when a user updates his power profile in step 3 at time n, he does this assuming that his utility is given by

$$\tilde{u}_i(c_i(\mathbf{p_i})) = u_i(c_i(\mathbf{p_i}(n)) + u_i' \times c_i(\mathbf{p_i}) \qquad (10)$$

where $u_i' := u_i'(c_i(\mathbf{p_i}(n))$. This linearization enables us to write a user's best response as

$$p_i^{k^*} = \frac{u_i'}{\sum_{j\neq i} \pi_j^k H_{ij}^k + \mu_i - \lambda_i^k} - \frac{p_i^k}{\gamma_i^k} \qquad (11)$$

where μ_i and λ_i^k are the Lagrange multipliers as in (3)-(5) and γ_i^k is the received SINR for user i on the k-th sub-channel. By complementary slackness, if $\sum_{k=1}^{K} p_i^{k^*} < P_i^{max}$, then $\mu_i = 0$, and if $\sum_{k=1}^{K} p_i^{k^*} = P_i^{max}$, then $\mu_i > 0$. If $p_i^k > 0$, then $\lambda_i^k = 0$, and if $p_i^k = 0$, then $\lambda_i^k > 0$. Note that p_i^k/γ_i^k is in fact a constant independent of $\mathbf{p_i}$.

In the ADP algorithm each user allocates his power to maximize the pay-off $s_i(\mathbf{p_i})$ in (9) given the current interference prices, which linearize the effects of that user's interference on other users. The difficulty in analyzing this is that when a user makes a large change in the power allocated to a sub-channel, this interference price may no longer accurately model its effect on the other users' utilities. Our second modification is to replace this with a more conservative strategy in which each user choses his new power profile to be a convex combination of his best response and the current power profile, i.e.,

$$\mathbf{p_i}(n+1) = (1-\alpha_i)\mathbf{p_i}(n) + \alpha_i \arg\max_{\mathbf{p_i}\in\Pi_i} s_i(\mathbf{p_i}; \mathbf{P_{-i}}(n)), \qquad (12)$$

[2]For the multi-channel case, a relaxed version of this algorithm was studied, in which the total power constraint for each node was relaxed by introducing the "power price" which was adjusted over a slower time-scale.

[3]In the special case of two users and two channels this game can still be shown to be supermodular by a suitable transformation of the strategy space.

where s_i is the users pay-off assuming the linearized utility as in (10). Here $\alpha_i \in (0, 1)$ is a fixed parameter that can be viewed as a normalized step-size along the direction of the (linearized) best response. Note that if $\alpha_i = 1$, then the modified ADP becomes the original ADP for the linearized utility.

To summarize, the modified ADP (MADP) algorithm can be stated in the same way as the original one except that step 3 is replaced with the following step:

3') At each time n, one user i is randomly selected to update its power profile according to

$$\mathbf{p_i}(n+1) = \mathbf{p_i}(n) + \alpha_i(\mathbf{p_i^*} - \mathbf{p_i}(n)), \qquad (13)$$

where p_i^* is given in (11).

B. Convergence Analysis of MADP Algorithm

Our main result, stated in Proposition 1 below is that for small enough values of α_i for each user i, the MADP algorithm converges monotonically to a fixed point. Let $U(n)$ denote the total utility summed over all users before the n-th iteration of the MADP algorithm.

Proposition 1: There exists an $\alpha_i > 0$ for each user i so that $\{U(n)\}$ is a monotonically increasing and convergent, i.e., $U(n+1) \geq U(n)$ for all n and $U(n) \to U^*$ as $n \to \infty$.

Since each user's total power is bounded, the sequence $\{U(n)\}$ will also be bounded. Hence, showing that $\{U(n)\}$ is monotonically increasing implies that it is convergent. Thus to prove this proposition we only need to show that $\{U(n)\}$ is monotonically increasing. To show this it suffices to consider a given iteration n in which user i is selected to update its power profile, and show that $U(\mathbf{p_i}(n+1)) \geq U(\mathbf{p_i}(n))$, where the total utility U is now regarded as a function of $\mathbf{p_i}$ because only the power profile of user i is updated. The proof of this can be found in the Appendix. In this proof, a bound to ensure that the step size α_i satisfies this proposition is given. however numerical examples indicate that this bound is quite conservative. Moreover, it requires global knowledge to calculate.

Proposition 1 guarantees that the MADP algorithm will converge. The next result characterizes the limit points of this algorithm.

Proposition 2: If the MADP algorithm converges, it converges to a solution satisfying the KKT condition of Problem P_0.

Proof: Assume that the algorithm converges to a fixed point U^* and let $U(n) = U^*$ for some time n. Then since this is a fixed point, it follows that $\mathbf{p_i}(n) = \mathbf{p_i^*}$ for every user i. It can then be seen that for all i $\mathbf{p_i}(n)$ must be an optimal solution to Problem P_i given the other users' current power profiles and interference price vectors (note that at a fixed point an optimal solution of Problem P_i with the linearized utility will also be optimal without the linearization). Hence, $\mathbf{P}(n) = \{\mathbf{p_1}(n), \cdots, \mathbf{p_M}(n)\}$ will satisfy the KKT condition of Problem P_0, because Problem P_0 and Problem P_i share the same KKT conditions for all i. ■

Note that we are not claiming that the final solution is an optimal solution to Problem P_0, or even that it is a local optimal.

C. Comparison of the MADP Algorithm with Alternative Algorithms

In this section we briefly compare the MADP algorithm with two other possibilities with similar overhead: the original ADP algorithm and a distributed gradient projection algorithm.

The MADP algorithm is derived from the ADP algorithm. From numerical simulation (some of which are reported in the next section) the ADP algorithm still works well in most cases, although we have not been able to prove convergence in general. On the other hand for the MADP algorithm Proposition 1 enables us to guarantee convergence. Another advantage of the MADP algorithm compared to the ADP is in terms of computing a user's new power allocation. For the ADP algorithm, this requires solving the non-linear optimization problem in Problem P_i. In general this requires using a numerical algorithm. For the MADP algorithm the linearization of the utility simplifies this updating. Specifically, one simply need to determine the dual variable μ_i and λ_i^k in (11), which can be done easily. For example, the parameter μ_i can be determined by the following iteration:

1) Initialize μ_i with any arbitrary $\mu_i^0 > 0$.
2) Calculate the temporary $\mathbf{p_i}^n$ by equation (11).
3) Reestimate μ_i^n by

$$\mu_i^{n+1} = \left[\mu_i^n + \alpha_{\mu_i}\Big(\sum_{k=1}^{K} {p_i^k}^n - P_i^{max}\Big)\right]^+$$

where α_μ is a step-size, which is required to be small enough, and $[\bullet]^+ = \max\{\bullet, 0\}$.

It can be shown that μ_i^n converges to the desired limit very fast.

Another alternative algorithm is a distributed gradient projection algorithm (see e.g. [10]). Given that the users exchange interference prices, the algorithm can be directly applied here (one update is done according to the current gradient of the total utility which each user can determine using the interference price vectors from other users and the gradient of one's own utility function). Applying standard results from [10] it can be shown that such an algorithm will converge monotonically provided that a small enough step size is used. As with the MADP algorithm, the limit will again satisfy the KKT conditions for Problem P_0 (and again it will not necessarily be an optimal solution to the problem). Although both the MADP algorithm and the gradient projection algorithm can guarantee that the total utility is monotonically increasing, the direction of update in the two algorithms is quite different. The direction for the MADP algorithm is pointing to the optimum of the (linearized) payoff function, which is an approximation of the total utility. On the other hand, the gradient projection algorithm always moves along the ascent direction of the gradient. In Fig. 1 we illustrate this difference for a simple case of 2 dimensions. With proper scaling, the direction of B can be preferable to that of A.

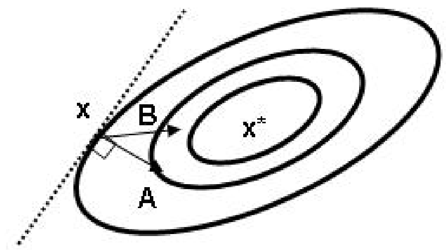

Fig. 1. Illustration of the direction taken by the gradient projection algorithm and the MADP algorithm. The arrow A indicates the direction for the gradient projection algorithm, while the arrow B is the direction taken by the MADP algorithm (which attempts to move toward the optimum point x*).

IV. Simulation Results

In this section we provide some simulation results, which compare the convergence of the MADP algorithm with the ADP and gradient projection algorithms. In the simulation model 10 transmitter-receiver pairs (users) share 10 subchannels. The users are randomly placed within an area, which is 1 km $\times$ 1 km, and the sequence of users who update powers at each iteration is also randomly generated.

The sub-channel gains are assumed to be *iid* Rayleigh random variables, where the mean is determined by distance attenuation. Namely, the average channel gain is $h(d) = h_0(\frac{d}{100})^{-4}$, where d is the separation in meters and h_0 is the reference channel gain at a distance of 100 m, which is the minimum separation between a paired transmitter and receiver. The maximum power, noise, and h_0 are selected so that the average received SNR at 100 m is 300 (about 25 dB). The utility function is $u(c) = 1 - \exp(-0.1c)$, which is assumed to be the same for all users.

Fig. 2 shows total utility versus number of iterations for a particular model realization, starting from all-zero power profiles. For this example the ADP converges rapidly and monotonically to the stationary point. The MADP converges more slowly than the ADP in this example. Here the update coefficient $\alpha = 0.2$, which is much larger than the value of α needed to guarantee monotonic convergence in the proof of Proposition 1. Even so, the results show that the MADP curve is relatively smooth and monotonic. Faster convergence can be obtained by increasing α; however, further simulations have shown that the MADP can exhibit oscillations, and may not converge. In contrast, we have observed in additional simulations that the ADP always converges monotonically.

The gradient projection algorithm converges much more slowly than the MADP algorithm. This is partially due to the small step-size selected, which is needed for monotonic convergence. Increasing the step-size can speed up convergence somewhat, although it is still significantly slower than that shown for the other algorithms. These results also indicate

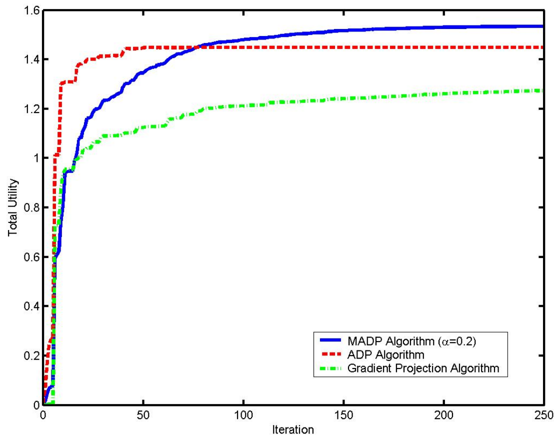

Fig. 2. Illustration of convergence for the ADP, MADP, and gradient projection algorithms.

TABLE I
NUMBER OF ITERATIONS REQUIRED FOR CONVERGENCE AVERAGED OVER 100 CHANNEL REALIZATIONS. STEP-SIZES ARE $\alpha = 0.2$ FOR THE MADP AND 5×10^{-7} FOR THE GRADIENT ALGORITHM.

Threshold	MADP	ADP	Gradient
1%	72.92	39.71	> 98.99
0.5%	102.71	46.32	> 147.42
0.1%	180.02	61.6	> 230.36

TABLE II
NUMBER OF ITERATIONS REQUIRED FOR CONVERGENCE AVERAGED OVER 100 CHANNEL REALIZATIONS. STEP-SIZES ARE $\alpha = 0.4$ FOR THE MADP AND 1×10^{-6} FOR THE GRADIENT ALGORITHM.

Threshold	MADP	ADP	Gradient
1%	56.17	39.24	> 90.07
0.5%	72.98	47.03	> 130.85
0.1%	122.12	62.43	> 230.38

that the different algorithms can converge to different limits.

Tables I and II compare the time it takes for each algorithm to converge when averaged over 100 channel realizations. For a particular realization we define convergence as occuring when the difference in total utility between two consecutive iterations is less than a threshold (e.g., 1%, 0.5%, or 0.1%). Of course, the convergence time for the MADP and gradient projection algorithms depends on the step-size. A smaller step-size gives a longer convergence time, but the utility increases more smoothly to the stationary value.

V. CONCLUSIONS

We have presented a distributed power control algorithm, the MADP algorithm, for a wireless network with frequency-selective channels. The algorithm is a modification of the ADP in [1], in which users announce the marginal cost per unit interference power as a set of interference prices, and each user updates a power profile given a set of interference prices and knowledge of incoming and outgoing channels. In the MADP the best response objective is linearized, and a partial step in that direction is taken. We have shown that the MADP converges monotonically provided that the steps are sufficiently small. Furthermore, because the steps can be much larger than in a gradient projection algorithm, the MADP algorithm is less likely to converge to a suboptimal local maximum due to a bad initialization.

The proof of convergence for the MADP algorithm relies on a rather conservative estimate of the maximum step that can be taken at each iteration. An open issue is how to choose this step to optimize the convergence speed. Extensions to other network models with different performance objectives and constraints are also interesting topics for future work.

APPENDIX A
PROOF OF PROPOSITION 1

Proof: Our goal is prove that $U(\mathbf{p_i}(n+1)) \geq U(\mathbf{p_i}(n))$. To do this we will use the following lemma to bound $U(\mathbf{p_i}(n+1))$:

Lemma 1 (Descent Lemma [10]): If $F : \Re^n \mapsto \Re$ is continuously differentiable and its gradient has the following property

$$\|\nabla F(x) - \nabla F(y)\|_2 \leq K\|x-y\|_2 \qquad \forall x, y \in \Re^n \tag{14}$$

then

$$F(x+y) \leq F(x) + y'\nabla F(x) + \frac{K}{2}\|y\|_2^2 \qquad \forall x, y \in \Re^n$$

Property (14) is a constraint that ∇F be Lipschitz continuous. One sufficient condition for Lipschitz continuity is that the L^2-norm of the Hessian matrix of F is bounded, in which case this bound can be used for the Lipschitz constant K. It can shown that is true for $U(\mathbf{p_i})$. Specifically, there exists a constant B_{Ui} which upper bounds the L^2-norm of the Hessian matrix of $U(\mathbf{p_i})$ independent of others' power profiles.

Applying the Descent Lemma to $-U(\mathbf{p_i})$, we get

$$\begin{aligned} & U(\mathbf{p_i}(n+1)) \\ & \geq U(\mathbf{p_i}(n)) + [\mathbf{p_i}(n+1) - \mathbf{p_i}(n)]'\nabla_{\mathbf{p_i}} U(\mathbf{p_i}(n)) \\ & \quad - \frac{B_{Ui}}{2}\|\mathbf{p_i}(n+1) - \mathbf{p_i}(n)\|_2^2. \end{aligned}$$

Hence to show $U(\mathbf{p_i}(n+1)) \geq U(\mathbf{p_i}(n))$, it suffices to show that

$$\begin{aligned} & [\mathbf{p_i}(n+1) - \mathbf{p_i}(n)]'\nabla_{\mathbf{p_i}} U(\mathbf{p_i}(n)) \\ & \geq \frac{B_{Ui}}{2}\|\mathbf{p_i}(n+1) - \mathbf{p_i}(n)\|_2^2. \end{aligned} \tag{15}$$

Using the power update strategy in (13), the inequality in (15) is equivalent to

$$[\mathbf{p_i^*} - \mathbf{p_i}(n)]'\nabla_{\mathbf{p_i}} U(\mathbf{p_i}(n)) \geq \alpha_i \frac{B_{Ui}}{2}\|\mathbf{p_i^*} - \mathbf{p_i}(n)\|_2^2, , \tag{16}$$

where $\mathbf{p_i}^*$ is given in (11).

Note that

$$\frac{\partial U(\mathbf{p_i})}{\partial p_i^k}\Big|_{\mathbf{p_i}=\mathbf{p_i}(n)} = u_i' \cdot \frac{1}{1+\gamma_i^k} \cdot \frac{\gamma_i^k}{p_i^k(n)} - \sum_{j \neq i} \pi_j^k H_{ij}^k, \tag{17}$$

where π_i^k is defined in (6). Substituting this for the components of $\nabla_{\mathbf{p_i}} U(\mathbf{p_i}(n))$, the left hand side (LHS) of (16) can be simplifed as follows

$$\begin{aligned}
\text{LHS} &= \sum_{k=1}^{K} (u_i' \cdot \frac{1}{1+\gamma_i^k} \cdot \frac{\gamma_i^k}{p_i^k(n)} - \sum_{j\neq i} \pi_j^k H_{ij}^k)(p_i^{k^*} - p_i^k(n)) \\
&= \sum_{k-1}^{K} \frac{\sum_{j\neq i}\pi_j^k H_{ij}^k}{p_i^k(n)\frac{1+\gamma_i^k}{\gamma_i^k}} (\frac{u_i'}{\sum_{j\neq i}\pi_j^k H_{ij}^k} - \frac{p_i^k(n)}{\gamma_i^k} - p_i^k(n)) \\
&\quad \cdot (p_i^{k^*} - p_i^k(n)) \\
&= \sum_{k=1}^{K} \frac{\sum_{j\neq i}\pi_j^k H_{ij}^k}{p_i^k(n)\frac{1+\gamma_i^k}{\gamma_i^k}} (p_i^{k^*} - p_i^k(n))^2 + \sum_{k=1}^{K} \frac{1}{p_i^k(n)\frac{1+\gamma_i^k}{\gamma_i^k}} \\
&\quad \cdot \frac{u_i'(\mu_i - \lambda_i^k)}{\sum_{j\neq i}\pi_j^k H_{ij}^k + \mu_i - \lambda_i^k}((p_i^{k^*} - p_i^k(n)) \\
&= \sum_{k=1}^{K} \frac{\sum_{j\neq i}\pi_j^k H_{ij}^k}{p_i^k(n)\frac{1+\gamma_i^k}{\gamma_i^k}} (p_i^{k^*} - p_i^k(n))^2 \\
&\quad + \sum_{k=1}^{K} \frac{\mu_i - \lambda_i^k}{p_i^k(n)\frac{1+\gamma_i^k}{\gamma_i^k}} (p_i^{k^*} - p_i^k(n))^2 \qquad (18) \\
&\quad + (\mu_i - \lambda_i^k) \sum_{k=1}^{K} (p_i^{k^*} - p_i^k(n)) \\
&= \sum_{k=1}^{K} \frac{\sum_{j\neq i}\pi_j^k H_{ij}^k + \mu_i - \lambda_i^k}{p_i^k(n)\frac{1+\gamma_i^k}{\gamma_i^k}} (p_i^{k^*} - p_i^k(n))^2 \\
&\quad + \sum_{k=1}^{K} (\mu_i - \lambda_i^k)(p_i^{k^*} - p_i^k(n)). \qquad (19)
\end{aligned}$$

Next we consider two cases for the values for the values of the Lagrange multiplies in this expression.

Case 1: If $\mu_i = 0$ and $\lambda_i^k = 0$ for all k, (19) can be written as

$$\text{LHS} = \sum_{k=1}^{K} \frac{\sum_{j\neq i}\pi_j^k H_{ij}^k}{p_i^k(n)\frac{1+\gamma_i^k}{\gamma_i^k}} (p_i^{k^*} - p_i^k(n))^2. \qquad (20)$$

Note that

$$\begin{aligned}
\frac{\sum_{j\neq i}\pi_j^k H_{ij}^k}{p_i^k(n)\frac{1+\gamma_i^k}{\gamma_i^k}} &= \frac{u_i'}{p_i^{k^*} + p_i^k(n)/\gamma_i^k} \cdot \frac{1}{p_i^k(n)(1+\gamma_i^k)/\gamma_i^k} \\
&\geq \frac{u_{imin}'}{P_i^{max} + (n_0 + \sum_{j\neq i} P_j^{max} H_{ji}^k)/H_{ii}^k} \\
&\quad \cdot \frac{H_{ii}^k}{(n_0 + \sum_{j\neq i} P_j^{max} H_{ji}^k)(1 + P_i^{max} H_{ii}^k/n_0)} \\
&> 0.
\end{aligned}$$

Therefore, there exists a minimum value $A_i = \min_k(\sum_{j\neq i}\pi_j^k H_{ij}^k)/(p_i^k(n)\frac{1+\gamma_i^k}{\gamma_i^k})$, which is positive and independent of other users' power profiles and interference price vectors. Then the left hand side of (16) can be lower bounded as

$$\text{LHS} \geq A_i \sum_{k=1}^{K} (p_i^{k^*} - p_i^k(n))^2 = A_i \|\mathbf{p_i^*} - \mathbf{p_i}(n)\|_2^2. \qquad (21)$$

Hence in this case choosing $\alpha_i \leq \min\{\frac{2A_i}{B_{Ui}}, 1\}$ ensures that the inequality (16) will be satisfied.

Case 2: If $\mu_i > 0$ or $\lambda_i^k > 0$ for some k, i.e., $\sum_{k=1}^{K} p_i^{k^*} = P_i^{max} \geq \sum_{k=1}^{K} p_i^k(n)$ or $p_i^k = 0 \leq p_i^k(n)$, then

$$\begin{aligned}
&\sum_{k=1}^{K} (\mu_i - \lambda_i^k)(p_i^{k^*} - p_i^k(n)) \\
&= \mu_i \sum_{k=1}^{K} (p_i^{k^*} - p_i^k(n)) + \sum_{k=1}^{K} \lambda_i^k (p_i^k(n) - p_i^{k^*}) \\
&\geq 0.
\end{aligned}$$

Therefore, (19) can be lower bounded as

$$\text{LHS} = \sum_{k=1}^{K} \frac{\sum_{j\neq i}\pi_j^k H_{ij}^k + \mu_i - \lambda_i^k}{p_i^k(n)\frac{1+\gamma_i^k}{\gamma_i^k}} (p_i^{k^*} - p_i^k(n))^2. \qquad (22)$$

As in Case 1, we can lower bound the quantities $(\sum_{j\neq i}\pi_j^k H_{ij}^k + \mu_i - \lambda_i^k)/(p_i^k(n)\frac{1+\gamma_i^k}{\gamma_i^k})$ by the positive constant A_i. The lower bound of the left hand side of (16) then becomes

$$\text{LHS} \geq A_i \sum_{k=1}^{K} (p_i^{k^*} - p_i^k(n))^2 = A_i \|\mathbf{p_i^*} - \mathbf{p_i}(n)\|_2^2. \qquad (23)$$

Setting $\alpha_i \leq \min\{\frac{2A_i}{B_{Ui}}, 1\}$ again ensures the desired inequality. ■

REFERENCES

[1] J. Huang, R.A. Berry, and M.L. Hoing, "Distributed interference compensation for wireless networks," *IEEE Journal on Selected Areas in Communications*, vol.24, no.5, pp.1074-1084, May 200

[2] Jun Yuan and Wei Yu, "Distributed Cross-layer Optimization of Wireless Sensor Networks: A Game Theoretic Approach", *IEEE Global Telecommunications Conference (Globecom)*, San Francisco, CA, Nov. 2006.

[3] W. Yu, G. Ginis, and J. Cioffi, "Distributed Multiuser Power Control for Digital Subscriber Line," *IEEE J. Sel. Areas Comm*, vol. 20, no. 5, pp. 1105-1115, 2002.

[4] Z. Luo and J. Pang, "Analysis of iterative waterfilling algorithm for multiuser power control in digital subscriber lines," *EURASIP J. Appl. Signal Process.*, vol. 2006, no. 1, Jan. 2006.

[5] K.W. Shum, K.-K. Leung, and C.W. Sung, "Convergence of iterative waterfilling algorithm for gaussian interference channels," *IEEE Journal on Selected Areas in Communications*, vol.25, no.6, pp.1091-1100, Aug. 2007

[6] G. Scutari, D. Palomar, and S. Barbarossa, "Asynchronous Iterative Waterfilling for Gaussian Frequency-Selective Interference Channels" to appear *IEEE Trans. on Information Theory*. 6

[7] R. Etkin, A. P. Parekh and D.Tse, "Spectrum Sharing in Unlicensed Bands", *IEEE Journal on Selected Areas of Communication*, vol. 25, no. 3, pp. 517-528, April 2007.

[8] M. Chiang, "Balancing Transport and Physical Layers in Wireless Multihop Networks: Jointly Optimal Congestion Control and Power Control," *IEEE J. Sel. Areas Comm*, vol. 23, no. 1, January 2005.

[9] D. Bertsekas, *Nonlinear Programming*, 2nd ed. Belmont, MA: Athena Scientific, 1999

[10] D.P. Bertsekas and J.N. Tsitsiklis *Parallel and Distributed Computation:Numerical Methods*. Englewood Cliffs, NJ: Prentice-Hall, 1989