Distributed resource allocation for single-hop networks under the SINR model

Vinod Sudheesh, Krishna Jagannathan, and Srikrishna Bhashyam
Department of Electrical Engineering
IIT Madras, Chennai 600036, India

***Abstract*—We consider the problem of distributed resource allocation in a single-hop wireless network under the SINR model. The network consists of fixed transmitters and their distinct receivers, and the physical layer model we consider is the SINR threshold model, where the transmitter will be able to send data successfully to its receiver at a fixed rate if the SINR at the receiver is greater than a given threshold. We address the problem of joint power control and link activation in a distributed setting, in order to stably support any traffic arrival process whose rate vector lies inside the capacity region. One of the technical contributions of this paper is in identifying a linear program to determine the optimal durations for employing each activation, whose solution decouples into a distributed algorithm, upon introducing a logarithmic barrier function.**

I. Introduction

In the context of communication networks, link scheduling for maximum throughput is a well studied problem. Maximum weight scheduling, first proposed in [12], [13], forms the basis for much of the literature on this topic. Despite being capable of supporting the largest possible set of arrival rates in a constrained queueing network, maximum weight scheduling suffers from some drawbacks when it comes to practical implementation. Besides being NP-hard, the maximum weight schedule requires global knowledge of the queue length information, which necessitates a centralised network controller. Distributed resource allocation algorithms are preferable in ad-hoc networks, from the points of view of operational simplicity and robustness.

Distributed and greedy scheduling approaches were proposed in [2], [5] and [14]. However, these greedy algorithms are not throughput optimal in general. In a series of recent papers [7], [10], [11], adaptive CSMA (carrier sense multiple access) based distributed algorithms have been proposed, and shown to achieve maximum throughput for a *conflict graph* based interference model. Their framework was a reversible Markov chain based approach that samples independent sets of the network randomly according to a product form distribution. The algorithm adjusts the parameters of the distribution to ensure that the time average service rate of all users exceeds their average arrival rate.

A conflict graph based interference model, in which two users either completely interfere with each other or don't interfere at all, irrespective of the transmission status of other users in the network is too simplistic to capture the complex nature of wireless interference. More specifically, conflict graph based interference models ignore (a) the problem of power allocation for a user, (b) the fact that whether or not two links interfere with each other depends on the transmission power of other users and their spatial location in the network, and (c) the fact that the interference between two users can be asymmetric.

In this paper, we consider the problem of distributed resource allocation under an SINR threshold based interference model. In this setting, the receiver will be able to decode data sent by the transmitter, if the SINR exceeds a predefined threshold β. Hence, the transmission rate is either 1 or 0, depending on whether the SINR threshold is exceeded or not.

Resource allocation algorithms under the SINR model can be categorized under three frameworks, depending on whether the algorithm employs only *power control*, only *scheduling*, or *joint power control and scheduling*. In a setting where only *power control* is used, the users share the channel by limiting the transmission power so that they do not cause more than necessary interference to other users, while maintaining their threshold SINR requirements. In a setting where only *scheduling* is used, the users share the channel in time between some subsets of users with users transmitting at the same fixed power in the active state, and meeting the threshold SINR requirements whenever they are active. In the most general case of *joint power control and scheduling*, the users share the channel in time between some subsets of users, while simultaneously tuning their transmission powers depending on the particular set of active users in order to meet the SINR requirements. We mention some related papers that have addressed resource allocation under the SINR model. In [15], a distributed algorithm was proposed for the SINR threshold model under scheduling. In [3], the authors propose a distributed throughput optimal algorithm under scheduling, for an SINR adaptation model, where the rate on each link is a concave function of the SINR. In [8], the authors propose a distributed algorithm which was shown to be throughput optimal for the SINR adaptation model, under joint power control and scheduling. However this algorithm requires the users to exchange control information, and is therefore not strictly distributed.

In this paper, we propose a throughput optimal distributed algorithm for the SINR threshold model under joint power control and scheduling. Our algorithm proceeds in four phases.

This work was supported in part by the Department of Science and Technology, Government of India.

978-1-4799-8439-8/15/$31.00 ©2015 IEEE

The first phase uniquely indexes the links, while the second phase identifies all the feasible link activation vectors, and determines the power allocation to each link for each feasible activation. Here, we adopt a distributed power allocation algorithm from [6], and show that either the powers converge if the particular link activation is feasible, or become unbounded if the activation is infeasible. In the third phase, the feasible activations are uniquely indexed. In the fourth phase, the links determine the optimal duration of time each feasible activation is employed, by solving a distributed optimization problem. Indeed, the optimization problem that determines the optimal durations for employing each activation is a simple linear program (LP). However, this linear program is not amenable to a distributed solution.

One of the main technical contributions of this paper is in proposing a distributed gradient descent algorithm, which approximates the optimal solution to the above LP to arbitrary precision. Indeed, from the constrained LP, we formulate an unconstrained convex program by introducing a logarithmic barrier function parameterized by a small but positive penalty factor Θ. The optimal solution to the unconstrained convex program can be shown to converge to the optimal solution to the original LP as $\Theta \downarrow 0$. In the unconstrained problem, we show that the gradients can be computed locally by each link, and hence each link can update the activation durations in a distributed fashion. By slowly reducing the value of Θ in the distributed gradient descent, we obtain convergence to the optimal solution to the original LP. Thus, the links obtain the optimal durations for employing each feasible link activation. We believe the above approach could be of independent interest in solving a certain LPs in a distributed fashion.

II. Network Model

We consider a network consisting of N distinct transmitter-receiver node pairs or N distinct links. The locations of the nodes are fixed. The term *network configuration* will denote the location of the different transmitters and receivers. Let P_i denote the transmit power of the i^{th} transmitter. For all $i \in \{1, 2, ..., N\}$, P_i satisfies the condition $P_i \in \mathcal{P}$, where $\mathcal{P}$ is the power constraint set. The power vector is $P = [P_1, P_2, ..., P_N]^T$. Let $g_{ij},\ i, j \in \{1, 2, ..., N\}$ denote the power gain in the channel between the j^{th} transmitter and i^{th} receiver. The interference power I_i at the receiver of link i is given by $I_i = \sum\limits_{j=1,\, j\neq i}^{N} P_j\ g_{ij}$. Let N_0 be the additive Gaussian noise power at each receiver. Then, SINR$_i[P]$, the SINR at the receiver of link i when power vector P is used, is given by

$$\text{SINR}_i[P] = \frac{P_i\ g_{ii}}{N_0 + \sum\limits_{j=1,\, j\neq i}^{N} P_j\ g_{ij}}. \tag{1}$$

Define an $N \times N$ link gain matrix G such that $G(i, j)$, the $(i, j)^{th}$ entry in G, is $g_{ij}, \forall i, j \in \{1, 2, ..., N\}$. We will assume that the matrix G is a full rank matrix. Define another $N \times N$ matrix Z, called the normalized link gain matrix, such that $Z(i, j)$, the $(i, j)^{th}$ entry in Z is $z_{ij} = \dfrac{g_{ij}}{g_{ii}}$ $, \forall i, j \in \{1, 2, ..., N\}$. Note that both Z and G are positive matrices.

Time is slotted, and we assume an ergodic arrival process with arrival rate vector $\lambda = [\lambda_1, \lambda_2, ..., \lambda_N]^T$ units, where λ_i is the arrival rate to the transmitter of link i. We assume that every transmitter i knows its own arrival rate λ_i. We also assume that the arrivals during each slot is upper bounded by a finite constant, for all links. The time average service rate of link i is denoted as S_i. The transmission rate of a link depends on all the entries of the power vector through the SINR. The exact relation between the two depends on the specific model under consideration. Let Rate$_i[P]$ denote the transmission rate of link i when power vector P is used. For the SINR threshold model, Rate$_i[P]$ is given by

$$\text{Rate}_i[P] = \begin{cases} 1 & \text{SINR}_i[P] \geq \beta \\ 0 & \text{SINR}_i[P] < \beta \end{cases}, \tag{2}$$

where β is called the threshold SINR.

III. SINR Threshold Model

In this section, we first analyze the feasibility of satisfying the SINR constraints of all links in the network simultaneously. When this is not feasible, it may be possible to satisfy the SINR constraints for a subset of links called an activation set. When the SINR constraints can be satisfied for an activation set, we will call the activation set *feasible*. An arrival process could be served by optimally time-sharing such feasible activation sets. We define the set of all possible activation sets and discuss the feasibility of satisfying the SINR constraints for each such activation set. We observe that the Foschini-Miljanic power control algorithm (FM-PCA) [6] can be used to find the power vector for feasible activation sets in a *distributed* manner. Finally, we show how the FM-PCA algorithm can also be used to identify if an activation set is not feasible.

A. Feasibility

A threshold SINR of β is feasible for the network if there exists a power vector that can simultaneously achieve the SINR constraint for all the links. This feasibility problem has been studied in [17] assuming that $N_0 = 0$, i.e., there is no receiver noise. For this network of interfering links, there exists a maximum balanced signal to interference ratio (SIR) denoted as β_0. This β_0, the maximum common value of SIR for all links that can be attained by using any power vector P, is given by $\beta_0 = \dfrac{1}{\kappa_Z^1 - 1}$, where κ_Z^1 is the dominant eigenvalue of the normalized link gain matrix Z, i.e., $|\kappa_Z^1| > |\kappa_Z^i|$ $, \forall i \in \{2, 3, ..., N\}$, and $\kappa_Z^1, \kappa_Z^2, ..., \kappa_Z^N$ are the N eigenvalues of the matrix Z [17]. Since Z is a matrix with positive entries, it is known that the dominant eigenvalue κ_Z^1 is real, and that $\kappa_Z^1 > 1$ [17]. Let $J_Z^1, J_Z^2, ..., J_Z^N$ be the N eigenvectors of Z corresponding to eigenvalues $\kappa_Z^1, \kappa_Z^2, ..., \kappa_Z^N$ respectively. The power vector that achieves this maximum balanced SIR (β_0) is the eigenvector J_Z^1 corresponding to the eigenvalue κ_Z^1.

The SINR is upper bounded by the SIR and approaches the SIR when the additive receiver noise becomes insignificant compared to the interference. Therefore, the maximum feasible SINR threshold is also β_0, i.e., $\beta \geq \beta_0$ is infeasible. In the rest of the paper, we will assume that active transmitters can use any positive power or in other words that the power constraint set $\mathcal{P} = (0, \infty)$. We remark that this assumption would only mean that the additive receiver noise (N_0) is sufficiently low when compared with the maximum transmission power limit for the links.

Next, we derive another SINR feasibility condition. In order to do this, we will rewrite the SINR requirements for all users as a single linear matrix inequality. In order to maintain successful transmission for all the links, we need to find a power vector P that satisfies

$$\text{SINR}_i[P] \geq \beta, \ \forall i \in \{1, 2, ..., N\}, \tag{3}$$

or, equivalently,

$$P_i g_{ii} - \sum_{j=1,\, j\neq i}^{N} P_j \beta g_{ij} \geq \beta N_0, \ \forall i \in \{1, 2, ..., N\}. \tag{4}$$

The above N constraints can be expressed in matrix form as

$$\begin{bmatrix} g_{11} & -\beta g_{12} & \dots & -\beta g_{1N} \\ -\beta g_{21} & g_{22} & \dots & -\beta g_{2N} \\ \vdots & \vdots & \dots & \vdots \\ -\beta g_{N1} & -\beta g_{N2} & \dots & g_{NN} \end{bmatrix} \begin{bmatrix} P_1 \\ P_2 \\ \vdots \\ P_N \end{bmatrix} \geq \begin{bmatrix} \beta N_o \\ \beta N_o \\ \vdots \\ \beta N_o \end{bmatrix}.$$

Define an $N \times N$ matrix $\tilde{G}$ with entries $\tilde{G}(i,i) = g_{ii}, \forall i \in \{1, 2, ..., N\}$ and $\tilde{G}(i,j) = -\beta\, g_{ij}, \forall i,j \in \{1, 2, ..., N\}, i \neq j$, and an $N \times 1$ matrix $B = [\beta N_0, \beta N_0, \ldots, \beta N_0]^T$. Then, the above SINR constraints on the power vector can be written succinctly as $\tilde{G}P \geq B$.

Among the possible power vectors that satisfy the SINR constraint, we define the *optimal power vector* as the one with the smallest l_1 norm, or sum power. Therefore, the *optimal power vector* P^* is an optimal point of the following linear program (LP).

$$\min \quad L(P) = \sum_{i=1}^{N} P_i$$

$$\text{subject to: } \tilde{G}P \geq B \ , \quad P \geq 0.$$

Theorem 1: 1) The optimal power vector P^* satisfies $\tilde{G}P^* = B$.

2) The threshold SINR β is feasible if and only if the P^* satisfying $\tilde{G}P^* = B$ also satisfies $P^* > 0$.

Proof: Following terminology in [9], we first convert the LP into *standard form* by adding N *surplus variables* $\beta' = [\beta'_1, \ \beta'_2, \ ...\beta'_N]^T$. Now, the LP can be written as

$$\min \quad L(P) = \sum_{i=1}^{N} P_i$$

$$\text{subject to: } [\tilde{G} \ \ -I] \begin{bmatrix} P \\ \beta' \end{bmatrix} = B \ , \quad [P \ \ \beta'] \geq 0.$$

Note that since G was assumed to be a full rank matrix, the matrix $[\tilde{G} \ \ -I]$ is of rank N. Therefore, to find the *optimal power vector*, it is enough to search over *basic feasible solutions*. The absence of any basic feasible solution will indicate that the LP is *infeasible* [9, Section 2.4]. From the $2N$ variables, a basic solution is obtained by setting N variables to be zero and solving the equality constraints for the remaining N variables. Now observe that any basic solution having $P_i = 0$, for some $i \in \{1, 2, ..., N\}$ cannot be feasible as the SINR constraint for the i^{th} link will not be satisfied in that case. Therefore, the only basic solution that can be feasible corresponds to $\beta'_i = 0, \forall i = 1, 2, ..., N$, i.e., the optimal power vector P^* satisfies $\tilde{G}P^* = B$. This optimal power vector is feasible if and only if $P^* > 0$. Hence the result. ■

From the above theorem, we conclude that with an *optimal power vector* of P^*, the SINR for all the links is exactly equal to β. This is feasible under these two equivalent conditions ($\beta \leq \beta_0$) or ($P^* > 0$).

B. Activating a subset of links

When the SINR threshold of β is not feasible for the network of N links, the SINR threshold could be feasible for a sub-network obtained after deactivating some of the links by setting their transmit power to zero. The subset of links activated is called an *activation set*. For a network of N links, we have $2^N - 1$ non empty activation sets. Define the set $E = \{0, 1\}^N \backslash (0, 0, ..., 0)$ that will be used to denote the set of all non empty activation sets. An element $e \in E$, called an *activation vector*, is a binary vector of length N such that the i^{th} component of e, e_i is 1 if link i is in this activation set and 0 otherwise. We will also refer to e as an *activation set* with elements $i \in e$ if and only if $e_i = 1$.

Let $P^e = [P_1^e, \ P_2^e, \ ..., P_N^e]^T$ denote the power vector corresponding to the *activation vector* e. The power vector P^e, satisfies the following constraint: $P_i^e \in \mathcal{P} \ \forall i \in e$ and $P_i^e = 0 \ \forall i \notin e$. A *feasible activation vector* is an *activation vector* $e \in E$ such that there exists a power vector P^e that satisfies $\text{SINR}_i[P^e] \geq \beta \ \forall i \in e$, i.e., the SINR threshold of β is attainable for all the links in that activation vector. Let P^{e*} denote the *optimal power vector* (i.e., the one with the smallest sum power) for a *feasible activation vector*. Let B^e be an $N \times 1$ matrix such that $B^e(i,1) = B(i,1)$ if $i \in e$ and $B^e(i,1) = 0$ if $i \notin e$. Then, P^{e*} solves $\tilde{G}P^{e*} = B^e$. This system of equations corresponds to $n_e = \sum_{i=1}^{N} I_{\{e_i=1\}}$ constraints. From Theorem 1, an *activation vector* $e \in E$ is *feasible* if and only if P^{e*} satisfies $P_i^{e*} > 0 \ \forall i \in e$. Also, let β_0^e denote the maximum common SINR that can be attained for the links in *activation vector* e.

C. Distributed power control in the feasible case ($\beta \leq \beta_0$)

When the SINR threshold of β is feasible, the Foschini Miljanic-Power Control Algorithm (FM-PCA) [6] can be used to find the optimal power vector of P^* in a distributed manner. In FM-PCA, the power P_i of a link i is updated in every time slot as:

$$P_i \leftarrow (1-\epsilon)P_i + \epsilon\, P_i \,\frac{\beta}{\text{SINR}_i[P]} \ , \epsilon \in (0,1]. \tag{5}$$

Starting from any power vector, the above iteration when executed for all the links, was proved to make the power vector converge to the *optimal power vector* P^* if the SINR threshold of β is feasible ($\beta < \beta_0$). Note that this algorithm can be used for any activation set.

D. Using FM-PCA to determine infeasibility ($\beta > \beta_0$)

In this section, we analyze the FM-PCA in the infeasible scenario ($\beta > \beta_0$). We show that the transmit powers diverge and the SINR converges to β_0 in this scenario. Identifying this divergence allows us to determine infeasibility using the FM-PCA algorithm.

We use the SINR expression (1) in (5) to rewrite the evolution of the transmit power as

$$P_i \leftarrow (1-\epsilon)\left(P_i + \frac{\epsilon\beta}{1-\epsilon}\sum_{j=1,\, j\neq i}^{N} z_{ij}P_j\right) + \frac{\epsilon\beta N_0}{g_{ii}}. \tag{6}$$

In matrix form, the evolution of power vector P can be expressed as

$$P \leftarrow (1-\epsilon)\ A\ P + C, \tag{7}$$

where A is an $N \times N$ matrix such that $A(i,i) = 1, \forall i$, $A(i,j) = c\ Z(i,j)\ ,\ \forall i, j,\, i \neq j$, and $c = \dfrac{\epsilon\beta}{1-\epsilon}$. C is an $N \times 1$ matrix with entries $C(i,1) = \dfrac{\epsilon\beta N_0}{g_{ii}}\ ,\ i = 1, 2, ..., N$.

Next, we state (without proof) two lemmas relating the eigenvectors and eigenvalues of the matrices Z and A before stating the SINR convergence result.

Lemma 1: The matrices Z and A have the same set of eigenvectors.

Lemma 2: Suppose κ_Z^1 is the dominant eigenvalue of Z, i.e., $|\kappa_Z^1| > |\kappa_Z^i|\ ,\forall i\ \in \{2,3,...,N\}$. Then, $\kappa_A^1 = 1 + c\ (\kappa_Z^1 - 1)$ is the dominant eigenvalue for the matrix A, i.e., $|\kappa_A^1| > |\kappa_A^i|\ ,\forall i\ \in \{2,3,...,N\}$.

Theorem 2: Suppose $\beta > \beta_0$ and $\epsilon \in (0,1)$. The power vector updated according to (7) diverges and the SINR converges to β_0.

Proof: The iteration in (7) is different from the iteration in [16] in two ways: (1) the presence of matrix C, and (2) matrix A instead of matrix Z. The two lemmas above show that the eigenvectors of Z and A are identical and that the eigenvector corresponding to the dominant eigenvalue is the same for both matrices Z and A. Using these results, a SINR convergence result can be obtained for the iteration in (7) as in [16].

Consider the following iteration obtained from (7) by ignoring the matrix C

$$P \leftarrow (1-\epsilon)\ A\ P. \tag{8}$$

Let $P(0)$ be the initial power vector. Let $P(1), P(2), \ldots$ denote the updated power vectors at times $t = 1, 2, \ldots$ using the iteration in (7). Starting from the same initial power vector $P(0)$ let $\widetilde{P}(1), \widetilde{P}(2), \ldots$ be the updated power vectors using the iteration in (8). As C contains entries that are strictly positive, it is clear that for all $t = 1, 2, \ldots$ we have $\widetilde{P}(t) < P(t)$. Therefore, if $\widetilde{P}(t)$ diverges, then $P(t)$ also diverges.

Express the initial power vector $P(0)$ as $P(0) = \sum\limits_{i=1}^{N} p_0^i\ J_A^i$ in terms of the eigenvectors of the matrix A. Assume $P(0) > 0$. Then, $\widetilde{P}(t)$ is dominated by the term corresponding to the dominant eigenvector $J_A^1(= J_Z^1$ from lemmas 1 and 2) for large t [16]. We have $\widetilde{P}(t) \to p_0^1(1-\epsilon)^t(\kappa_A^1)^t J_Z^1$, where $p_0^1 > 0$. Now, we have

$$(1-\epsilon)\kappa_A^1 = (1-\epsilon)\left(1 + \frac{\epsilon\beta}{1-\epsilon}(\kappa_Z^1 - 1)\right) = 1 - \epsilon + \frac{\epsilon\beta}{\beta_0} > 1$$

for $\beta > \beta_0$. Therefore, as $t \to \infty$, $\widetilde{P}_i(t) \to \infty, \forall i \in \{1, 2, ..., N\}$, which also implies $P_i(t) \to \infty, \forall i \in \{1, 2, ..., N\}$. Hence, in the infeasble scenario, the interference power at the receiver of all links grows without bounds. This also means that C which depends on the finite receiver noise term of N_0 becomes negligible for large t. Thus, for large values of t, the power vector $P(t)$ that evolves according to (7) behaves similar to the evolution according to (8).

We also note that the power vector is aligned with the dominant eigenvector of Z. This is the power vector corresponding to the balanced SIR of β_0 from [16]. Thus, we have SINR $\to \beta_0$ as $t \to \infty$. ■

From the above theorem, we observe that when the SINR threshold of β is not feasible for the network, the SINR for all the links converges to β_0 if the FM-PCA algorithm is used to update the power vector.

IV. Rate Stability & Capacity Region

The notion of stability we consider is rate stability [4][Section 7]. According to this notion, if the time average service rate for a link i, S_i is strictly greater than its time average arrival rate λ_i, ($S_i > \lambda_i$) then the queue of transmitter i, Q_i is rate stable.

The capacity region consists of all arrival rate vectors that can be stabilized by any scheduling algorithm (possibly centralized). In order to support an arrival rate vector λ for a power constraint set $\mathcal{P}$, the following conditions on the *network configuration* need to be satisfied.

$\exists\ E(f) = \{e^1, e^2, ..., e^K\} \subseteq E$ such that $\forall e^k \in E(f)$, $\exists\ P^{e^k}$ such that

$$\text{SINR}_i[P^{e^k}] \geq \beta, \forall i \in e^k, \tag{9}$$

$$P_i^{e^k} \in \mathcal{P}\ \ \forall e^k \in E(f) \text{ and } \forall i \in e^k. \tag{10}$$

$\exists\ \mu = [\mu_1, \mu_2, ..., \mu_K]^T$ such that

$$\lambda_i = \sum_{k=1}^{K} \mu_k\ I_{\{e_i^k = 1\}}, \forall i \in \{1, 2, ..., N\}, \tag{11}$$

$$0 \leq \mu_k < 1,\ \forall k \in \{1, 2, \ldots, K\}, \text{and } \sum_{k=1}^{K} \mu_k < 1. \tag{12}$$

$E(f)$ is the set of all feasible activation vectors and μ_k is the fraction of time each of the activation vector e^k is used. The conditions for an activation set e^k to be feasible are given by (9) and (10). Conditions (11) through (12) state that the arrival rate must lie in the interior of convex combination of all feasible activation vectors [12].

V. Throughput Optimal Resource Allocation

In this section, we derive a distributed throughput optimal resource allocation algorithm. As suggested by the characterization of the capacity region in the previous section, the resource allocation algorithm consists of a *power allocation* problem, and a *link scheduling* problem. We will derive distributed algorithms to optimally perform power allocation as well as link scheduling.

A. Power Allocation Problem

The power allocation problem consists of finding the set of all feasible activation vectors $E(f) = \{e^1, e^2, ..., e^K\}$, and the corresponding power vectors P^{e^k} that satisfy the SINR constraints for all the users simultaneously. However, from Theorem 1, the feasibility or infeasibility of an activation vector is decided by the *optimal power vector* $P^{e^{k*}}$. Hence, by executing FM-PCA for a link i in each activation vector $e \in E$, we can determine if the activation vector is feasible or not as follows.

- If power P_i converges to a constant P_i^{e*}, and SINR converges to β, the activation vector e is feasible.
- If power P_i diverges and SINR converges to $\beta_0^e < \beta$, the activation vector e is infeasible.

B. Link Scheduling Problem

The scheduling problem entails finding the parameters μ that will make the time average service rate for each link greater than its time average arrival rate. We first formulate an LP whose optimal point will be shown to satisfy this service rate requirement, for any arrival rate vector in the capacity region.

Consider the following Linear Program (LP) in variables $\nu = [\nu_1,\ \nu_2,\ ...,\ \nu_k]^T$.

Problem SCHEDULING LP:

$$\min\ L(\nu) = \sum_{k=1}^{K} \nu_k$$

$$\text{subject to:} \quad s_i \geq \lambda_i\ ,\ i = 1, 2, ..., N,$$

$$\nu_k \geq 0\ ,\ k = 1, 2, ..., K,$$

where $s_i = \sum_{k=1}^{K} \nu_k\, I_{\{e_i^k=1\}}\ ,\ i = 1, 2, ..., N.$

Let ν^* be the optimal point for the above LP.

In the above formulation, ν_k is the duration of time that the feasible activation vector $e^k \in E(f)$ is used. If the channel is never left idle, then we have the following relation between the variables μ_k and ν_k, $\mu_k = \nu_k / \sum_{l=1}^{K} \nu_l$. The term s_i will be called *virtual service rate* of link i, and it is related to the time average service rate S_i as $S_i = s_i / \sum_{l=1}^{K} \nu_l$.

Theorem 3: If the arrival rate vector λ lies in the interior of the capacity region, then at the optimal solution ν^*, we have $S_i > \lambda_i\ , \forall i \in \{1, 2, ..., N\}$.

Proof: Since the arrival rate vector λ lies in the interior of the capacity region, there must exist a $\mu' = [\mu'_1\ , \mu'_2\ , ..., \mu'_K]$ that satisfies $\lambda_i = \sum_{k=1}^{K} \mu'_k\ I_{\{e_i^k=1\}}$, $\forall i \in \{1, 2, ..., N\}$, and hence $s_i = \lambda_i\ , \forall i \in \{1, 2, ..., N\}$. The point μ' lies in the feasible set of the Linear Program. Now, since ν^* is the optimal point of the LP, we have $\sum_{k=1}^{K} \nu_k^* \leq \sum_{k=1}^{K} \mu'_k < 1$. Also, we have $\sum_{k=1}^{K} \nu_k^* > 0$, or else $s_i \geq \lambda_i$ condition cannot be met for any link i with a strictly positive arrival rate λ_i.

Now, we have at ν^*, $s_i \geq \lambda_i$, $\forall i \in \{1, 2, ..., N\}$ or $S_i = s_i / \sum_{l=1}^{K} \nu_l > s_i \geq \lambda_i$. ■

Theorem 4: Any arrival rate λ in the capacity region can be supported by using at most N activation vectors.

Proof: As in Theorem 1, we can convert the LP into standard form. Then, the result follows from [9, Section 2.4]. ■

C. Distributed solution to SCHEDULING LP

Unfortunately, the problem SCHEDULING LP cannot be solved directly in a distributed manner. In order to find the optimal point ν^* in a distributed manner, we formulate an unconstrained convex optimization problem, parametrized by a scalar $\Theta > 0$. This convex program is an approximation to the linear program SCHEDULING LP. The scalar $\Theta > 0$ is called the penalty factor and the approximation gets better as $\Theta \downarrow 0$.

Problem BARRIER:

$$\min L(\nu, \Theta) = \sum_{k=1}^{K} \nu_k - \Theta \sum_{i=1}^{N} \log(s_i - \lambda_i) - \Theta \sum_{k=1}^{K} \log(\nu_k),$$

where $s_i = \sum_{k=1}^{K} \nu_k\, I_{\{e_i^k=1\}}$, $i = 1, 2, ..., N$ and $\Theta > 0$.

Let $\nu^*(\Theta)$ be the solution to the problem BARRIER. From [1, Section 11.2.1], we have the relation,

$$\nu^* = \lim_{\Theta \downarrow 0} \nu^*(\Theta). \tag{13}$$

As we show shortly, a key property of the problem BARRIER is that its optimal point $\nu^*(\Theta)$ can be obtained in a distributed manner. In particular, the gradient of the objective function $L(\nu, \Theta)$ can be computed by individual links, thus enabling us to solve the scheduling problem in a distributed manner by choosing an appropriately small value for the penalty Θ.

Solving the barrier program using gradient descent

As the objective function, $L(\nu, \Theta)$, in the barrier program is a convex function, its optimal point can be reached using gradient descent. The gradient for the objective function $L(\nu, \Theta)$

is denoted $\nabla_\nu L(\nu,\Theta)$ and its k^{th} element is given by

$$\frac{\partial L(\nu,\Theta)}{\partial \nu_k} = 1-\Theta\frac{1}{\nu_k}-\Theta\sum_{i=1}^{N}\frac{I_{\{e_i^k=1\}}}{s_i-\lambda_i},\ \ k=1,2,...,K. \quad (14)$$

The gradient descent algorithm for finding the parameters ν works as follows. Every discrete time slot indexed by t, update the parameter ν as

$$\nu(t+1) \leftarrow \nu(t) - \alpha(t)\ \nabla_{\nu(t)}\ L(\nu(t),\Theta), \qquad (15)$$

where $\alpha(t),\ t=1,2,...$ is a sequence such that $\alpha(t)>0$, $\sum_{t=1}^{\infty}\alpha(t)=\infty,\ \sum_{k=1}^{\infty}\alpha(t)^2<\infty$.

VI. Distributed Algorithm

In this section, we present our distributed algorithm for the SINR threshold model under *joint power control and scheduling*. We assume that there is a feedback channel from the receiver to the transmitter that reports the SINR at the receiver and that this feedback is instantaneous. We also assume that the receiver will be able to detect any infinitesimal change in interference in the network. Recall that every transmitter i is assumed to know its own arrival rate λ_i.

The distributed algorithm has the following four phases: Initialization Phase (IP), Activation Vector and Transmit Power Identification Phase (AVTPIP), Activation Vector Indexing Phase (AVIP), Data Frame Phase (DFP).

A. Initialization Phase

In this phase, the objective is to establish a unique index for all the links in the network and also to identify the total number of links in the network. We assume that time is divided into slots of duration σ. During odd slots, links that have not been indexed will transmit with some probability $p>0$. If there is no collision, they get indexed. The other links in the network will listen to this transmission and increment the number of indexed links. In case of a collision during the odd slot, the links involved in the collision will transmit in the next even slot to inform other links in the network about the collision. This helps the other links to keep track of the number of indexed links. There is also an upper limit on the number of consecutive idle odd slots W, and all the unindexed links will transmit message in the odd slot with probability 1 if there have been $W-1$ consecutive idle odd slots already. Therefore, W consecutive idle odd slots would indicate to the links that all links have been indexed and the IP is over.

Each link i executes the following algorithm. After the execution of this algorithm, link i will have a unique index given by $Link\_index_i$ and all links will know the total number of links N.

```
Initialization:
  • Number of links that have been indexed $index = 0$.
  • Link $i$ indexed status $indexed_i = No$.
  • No. of consecutive odd idle slots, $idle\_slots = 0$.
  • Indicator of collision for link $i$, $collide_i = 0$.
every time slot t, do
1) if $t$ is an odd slot
  • if $indexed_i = No$, transmit in this slot with probability $p = \frac{idle\_slots+1}{W}$.
    – If transmission has been attempted by link $i$
      ∗ Update $idle\_slots = 0$ ,   $index = index + 1$
      ∗ if collision occurs, i.e., two or more links tried to use the channel:
          · Update $collide_i = 1$,   $index = index - 1$
      ∗ if collision did not occur. Indexing is done in the current time slot.
          · Update $Link\_index_i = index$, $indexed_i = Yes$.
    – If transmission has not been attempted by link $i$
      ∗ if channel is Busy. At least one link in the network tried to get an index in the current time slot.
          · Update $index = index + 1$ ,   $idle\_slots = 0$
      ∗ if channel is Idle.
          · Update $idle\_slots = idle\_slots + 1$.
  • if $indexed_i = Yes$
    – if channel is Busy. At least one link in the network tried to get an index in the current time slot.
      ∗ Update $idle\_slots = 0$ ,   $index = index + 1$
    – if channel is Idle.
      ∗ Update $idle\_slots = idle\_slots + 1$
      ∗ if $idle\_slots = W$
          · Set $N = index$. Exit IP.
2) if $t$ is an even slot
  • if $collide_i = 1$ (link $i$ encountered a collision in the previous odd slot)
    – Transmit, Update $collide_i = 0$
  • if $collide_i = 0$ (link $i$ did not encounter a collision in the previous odd slot)
    – if channel is Busy
      ∗ Update $index = index - 1$.
```

B. Activation Vector and Transmit Power Identification Phase

Let $E_i(f)$ be the set of all feasible *activation vectors* that link i is contained in. In this phase, every link i in the network determines $E_i(f)$ and also corresponding the transmission power $P_i^e,\ \forall e\in E_i(f)$. For $m\in\{0,1,...,2^N-1\}$, $\{m\}_2$ will denote the N length binary equivalent of decimal number m. Each link i executes the following algorithm.

```
Initialization:
  • $m = 2^N - 1$, activation vector $e = \{m\}_2$.
  • Transmit power of link $i$, $p_i = 0$.
  • SINR at the receiver of link $i$ $\text{SINR}_i[0] = 0$.
every time slot $t$, do
1) if $t$ is an odd slot
  • if link $i$ is contained in activation vector $e$
    – Update power $p_i$ as
          $p_i \leftarrow (1-\epsilon)p_i + \epsilon\, p_i\, \frac{\beta}{\text{SINR}_i[t-1]}$.
```

```
        – Transmit with power $p_i$
        – Update $I_i[t]$ ,SINR$_i[t]$ from channel
        – if $\{I_i[t]\}$ has converged: Activation vector $e$ is feasible
            ∗ Update $exit_i = 1$, $P_i^e = p_i$, $E_i(f) = E_i(f) \cup e$.
            ∗ Update $p_i = 0$.
        – if $\{I_i[t]\}$ has not converged, but $\{\text{SINR}_i[t]\}$ has converged: Activation vector $e$ is infeasible
            ∗ Update $exit_i = 1$, $p_i = 0$.
    • if link $i$ is not contained in activation vector $e$
        – Update $I_i[t]$ from channel
2) if $t$ is an even slot
    • if link $i$ is contained in activation vector $e$
        – Transmit with power $p_i$
        – if $exit_i = 0$
            ∗ Update $I_i[t]$ from channel
            ∗ if $I_i[t]$ is less than $I_i[t-1]$
                · if SINR$_i[t-1] \geq \beta$. Activation vector $e$ is feasible
                  Update $P_i^e = p_i$, $E_i(f) = E_i(f) \cup e$.
                  Update $p_i = 0$.
                · if SINR$_i[t-1] < \beta$. Activation vector $e$ is infeasible
                  Update $p_i = 0$.
            ∗ Update $m = m - 1$, $e = \{m\}_2$, SINR$_i[t] = 0$.
        – if $exit_i = 1$
            ∗ Update $exit_i = 0$
            ∗ Update $m = m - 1$, $e = \{m\}_2$, SINR$_i[t] = 0$.
    • if link $i$ is not contained in activation vector $e$
        – Update $I_i[t]$ from the channel
        – if $I_i[t]$ is less than $I_i[t-1]$
            ∗ Update $m = m - 1$, $e = \{m\}_2$, SINR$_i[t] = 0$.
    • if $m = 0$
        – Exit AVTPIP.
```

C. Activation Vector Indexing Phase

In this phase, every link learns the set of all feasible activation vectors $E(f)$ and indexes the vectors. Note that each link i already knows $E_i(f)$ from the previous phase. Each link i executes the following algorithm.

```
  Initialization:
    • $m = 2^N - 1$, activation vector $e = \{m\}_2$.
    • Set of all feasible activation vectors $E(f) = \phi$.
every time slot $t$, do
1) if link $i$ is contained in activation vector $e$
    • if $e \in E_i(f)$: Activation vector $e$ is feasible.
        – Transmit using power $P_i^e$
        – Update $k = k + 1$, $e^k = e$, $E(f) = E(f) \cup \{e^k\}$.
2) if link $i$ is not contained in activation vector $e$
    • if channel is Busy: (this indicates that activation vector $e$ is feasible)
        – Update $k = k + 1$, $e^k = e$, $E(f) = E(f) \cup \{e^k\}$.
3) Update $m = m - 1$, $e = \{m\}_2$.
4) if $m = 0$, set $K = k$, exit AVIP.
```

D. Data Frame Phase

After the first three phases described earlier, each link i is aware of the set $E(f)$ and their corresponding optimal power P_i^{e*} corresponding to each of these activation vectors $e \in E(f)$. Now, they need to determine the parameters $\nu_k^*, \forall k \in \{1, 2, ..., K\}$, where ν_k^* is the solution to SCHEDULING LP. Let *data frame* be the period of time during which all the activation vectors $e \in E(f)$ are executed once sequentially. In each *data frame*, one link updates the parameters $\nu_k, \forall k \in \{1, 2, ..., K\}$, and this will be observed from the channel by other links and stored. After N *data frames*, all the links update the parameters based on the updated parameters in the previous N *data frames*.

Each link i executes the following algorithm.

```
  Initialization:
    • $m = 0$, $\mu = [\mu_1, \mu_2, \ldots, \mu_K] = [1, 1, ..., 1]$ $s_i = |E_i(f)|$.
    • $scale\_1 = 1$, $scale\_2 = 1$, $\Theta = \Theta_0$.
for each data frame $m$, do
  1) Update $m = m + 1$.
  2) The index of the link that will update the parameters in the current data frame is $idx$, given by $idx = 1 + mod(m - 1, N)$.
  3) Link $i$ now proceeds to transmit data in the data frame as follows:
    1) if $Link\_index_i = idx$: (link $i$ is responsible for updating the parameter $\nu$ in the current data frame)
       for each $e^1, e^2, ..., e^K$ in $E(f)$
        • Transmit data with power $P_i^{e^k}$ for duration $\nu_k$, where
          $$\nu_k = \mu_k - \frac{\epsilon_1}{scale\_1}\left(1 - \frac{\Theta}{\mu_k} - \frac{\Theta N}{(s_i - \lambda_i)} I_{\{e_i^k=1\}}\right) \tag{16}$$
        • Wait for $\sigma$ amount of time.
        • Store $u_k^{idx} = \nu_k$.
    2) if $Link\_index_i \neq idx$: (link $i$ is not responsible for updating the parameter $\nu$ in the current data frame)
       for each $e^1, e^2, ..., e^K$ in $E(f)$
        • Transmit data with power $P_i^{e^k}$ for duration $\hat{\nu}_k$, where
          $$\hat{\nu_k} = \mu_k - \frac{\epsilon_1}{scale\_1}\left(1 - \frac{\Theta}{\mu_k}\right) \tag{17}$$
        • Observe $\nu_k$ from the channel.
        • Wait for $\sigma$ amount of time.
        • Store $u_k^{idx} = \nu_k$.
4) if $idx = N$: Update $\mu$ as
   $$\mu = (1 - \epsilon_2)\,\mu + \epsilon_2 \frac{1}{N}\sum_{i=1}^{N} u^i, \tag{18}$$
   where $u^i = [u_1^i, u_2^i, \ldots, u_K^i]$.
   $s_i = \sum_{k=1}^{K} \mu_k \; I_{\{e_i^k=1\}}$, $scale\_1 = scale\_1 + 1$.
5) if $mod(m - 1, MN) = MN - 1$:
```

- $scale\_1 = 1$, $scale\_2 = scale\_2 + 1$, $\Theta = \frac{\Theta}{scale\_2}$.

Theorem 5: In the proposed distributed algorithm, ν converges to ν^*.

Proof: Substituting (16) in (18) we have, μ_k, the k^{th} component of μ, updated after N *data frames* as

$$\mu_k \leftarrow (1-\epsilon_2)\,\mu_k + \frac{\epsilon_2}{N}\left(\sum_{i=1}^{N}\mu_k - \frac{\epsilon_1}{scale\_1}\left(1 - \frac{\Theta}{\mu_k} - \frac{\Theta\, N}{(s_i-\lambda_i)}\, I_{\{e_i^k=1\}}\right)\right)$$

Therefore, we have

$$\mu_k \leftarrow \mu_k - \frac{\epsilon_2\ \epsilon_1}{scale\_1}\left(1 - \frac{\Theta}{\mu_k} - \sum_{i=1}^{N}\frac{\Theta}{(s_i-\lambda_i)}\, I_{\{e_i^k=1\}}\right), \qquad (19)$$

which is nothing but the gradient descent algorithm that minimizes the Loss function $L(\nu, \Theta)$. Therefore, the distributed algorithm drives the parameter μ to $\nu^*(\Theta)$ for a fixed Θ. If Θ is decremented sufficiently slowly, it can be shown that $\nu^*(\Theta) \to \nu^*$ as $\Theta \downarrow 0$ as desired. ■

VII. Simulations

The proposed distributed algorithm was simulated for a network of 5 links. The standard path loss function $g_{ij} = \alpha\, r_{ij}{}^{-\eta}\ \forall\ i, j \in \{1, 2, ..., N\}$ was used. The table below lists the different parameters and results of the simulation. Figure 1 shows the Queue lengths as a function of time for each of the links. Note that $t = 0$ in the plot corresponds to the starting point for the DFP. A loading factor of 0.99 was used for the simulation. It is clear that the queue lengths stay bounded, even for this high loading.

Param.	**Value**	**Param.**	**Value**	**Param.**	**Value**
N	5	W	12	M	10
α	1	η	3	N_0	10^{-9}
Θ_0	1	ϵ_1	0.005	ϵ_2	1
σ	10^{-4}	β	10	ϵ	0.50

Parameter	**Value**
$E(f)$	$\{3\}\quad\{1\}\quad\{4\}\quad\{3,4\}$ $\{2\}\quad\{2,3\}\quad\{5\}\quad\{4,5\}$

VIII. Conclusion

In this paper, we proposed a distributed algorithm that is throughput optimal for an SINR threshold model under joint power control and scheduling. Our algorithm runs in four phases, and determines the feasible activations as well as the transmit powers of each link in each feasible activation. We posed the problem of determining the optimal durations of employing the feasible activations as an LP. A key technical component of the work involved solving the LP by introducing a logarithmic barrier, upon which the problem decoupled into a distributed gradient descent at each link.

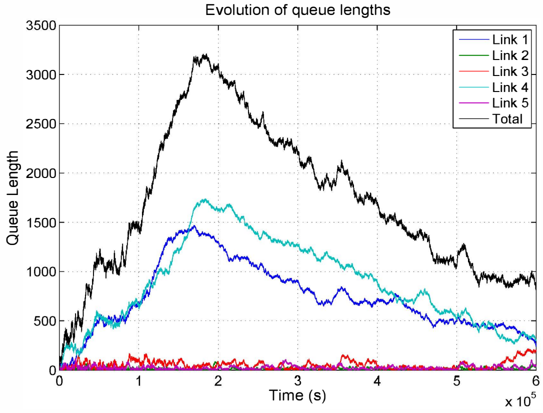

Fig. 1. Queue Lengths

References

[1] S. Boyd and L. Vandenberghe. *Convex Optimization*. Cambridge University Press, 2004.

[2] P. Chaporkar, K. Kar, X. Luo, and S. Sarkar. Throughput and fairness guarantees through maximal scheduling in wireless networks. *IEEE Transactions on Information Theory*, 54(2):572, 2008.

[3] P. Chaporkar and A. Proutiere. Optimal distributed scheduling in wireless networks under sinr interference model. *in Proc. of 51st Allerton Conference on Communication, Control and Computing*, 2013.

[4] P. Chaporkar and A. Proutiere. Optimal distributed scheduling in wireless networks under the SINR interference model. *http://arxiv.org/abs/1305.0384*, 2013.

[5] A. Dimakis and J. Walrand. Sufficient conditions for stability of longest-queue-first scheduling: Second-order properties using fluid limits. *Advances in Applied Probability*, pages 505–521, 2006.

[6] G. Foschini and Z. Miljanic. A simple distributed autonomous power control algorithm and its convergence. *IEEE Transactions on Vehicular Technology*, 42(4):641–646, 1993.

[7] L. Jiang and J. Walrand. A distributed csma algorithm for throughput and utility maximization in wireless networks. *IEEE/ACM Transactions on Networking (TON)*, 18(3):960–972, 2010.

[8] H. W. Lee, E. Modiano, and L. B. Le. Distributed throughput maximization in wireless networks via random power allocation. *IEEE Transactions on Mobile Computing*, 11(4):577–590, 2012.

[9] D. G. Luenberger and Y. Ye. *Linear and Nonlinear Programming*. Springer, 2008.

[10] J. Ni, B. Tan, and R. Srikant. Q-CSMA: Queue-length based CSMA/CA algorithms for achieving maximum throughput and low delay in wireless networks. In *INFOCOM, 2010 Proceedings IEEE*. IEEE, 2010.

[11] S. Rajagopalan, D. Shah, and J. Shin. Network adiabatic theorem: an efficient randomized protocol for contention resolution. In *Proceedings of the Eleventh International Joint Conference on Measurement and Modeling of Computer Systems*, pages 133–144. ACM, 2009.

[12] L. Tassiulas and A. Ephremides. Stability properties of constrained queueing systems and scheduling policies for maximum throughput in multihop radio networks. *IEEE Transactions on Automatic Control*, 37(12):1936–1948, 1992.

[13] L. Tassiulas and A. Ephremides. Dynamic server allocation to parallel queues with randomly varying connectivity. *IEEE Transactions on Information Theory*, 39(2):466–478, 1993.

[14] X. Wu, R. Srikant, and J.R. Perkins. Queue-length stability of maximal greedy schedules in wireless networks. In *Proceedings of Information Theory and Applications Inaugural Workshop*, pages 6–10, 2006.

[15] Y. Yi and G. D. Veciana. On optimal MAC scheduling with physical interference. *in Proc. of 26th IEEE INFOCOM*, 2007.

[16] J. Zander. Distributed co-channel interference control in cellular radio systems. *IEEE Transactions on Vehicular Technology*, 41:305–311, 1992.

[17] J. Zander. Performance of optimum transmitter power control in cellular radio systems. *IEEE Transactions on Vehicular Technology*, 41:57–62, 1992.