How Optimal are Wireless Scheduling Protocols?

Thomas Moscibroda
Microsoft Research
One Microsoft Way
Redmond, WA 98052
moscitho@microsoft.com

Yvonne Anne Oswald
Computer Engineering
and Networks Laboratory
ETH Zurich, Switzerland
yoswald@tik.ee.ethz.ch

Roger Wattenhofer
Computer Engineering
and Networks Laboratory
ETH Zurich, Switzerland
wattenhofer@tik.ee.ethz.ch

Abstract— **In wireless networks mutual interference impairs the quality of received signals and might even prevent the correct reception of messages. It is therefore of paramount importance to dispose of power control and scheduling algorithms, coordinating the transmission of communication requests. We propose a new measure *disturbance* in order to comprise the intrinsic difficulty of finding a short schedule for a problem instance. Previously known approaches suffer from extremely bad performance in certain network scenarios even if disturbance is low. To overcome this problem, we present a novel scheduling algorithm for which we give analytical worst-case guarantees on its performance. Compared to previously known solutions, the algorithm achieves a speed up, which can be exponential in the size of the network.**

I. Introduction

In the past, the networking community has been remarkably successful in designing protocol standards, such as TCP or IP, many of which have even survived the unexpected and tremendous growth of the Internet. Most of these protocols have in common that they were developed using heuristic (rule of thumb) reasoning, and evaluated through complex simulations. While so far, this approach has worked well, we feel that *wireless networks* are challenging the heuristics/simulation paradigm.

Considering the specific characteristics of wireless networks, designing efficient and reliable wireless protocols is challenging. In order to be credible, one has to consider various difficulties which arise in practice, such as complex wireless channel models, genuine traffic patterns, or strenuous environmental influences. These intricate modeling issues have often prevented researchers from thoroughly analyzing their algorithms and protocols, resorting instead to simulation as the seemingly only feasible method.

On the other hand, it is clear that simulation is problematic, as one can never cover all possible scenarios. What if a heuristic works well in most (simulated) scenarios but is inferior in some other classes of scenarios? What if these devastating classes are important or even critical in practice? In view of these potential problems, it is not surprising that several researchers in the wireless networking community have recently started questioning the relevance of simulations. In contrast, analytic worst-case analysis has the advantage to include all possible cases, and offers strict performance guarantees.

In this paper we study a fundamental problem in wireless networking, which is prototypical for the heuristics/simulation dilemma. Specifically, we consider the following spatial reuse instance, captured by the following well-known *scheduling problem*: Given a set of transmission requests, how do we schedule these requests such that concurrent transmissions do not cause a level of interference preventing the correct reception of messages and that the total time needed to successfully schedule all requests is minimized.

If we consider omnidirectional antennae, every sender causes interference to every receiver, depending on the propagation attenuation of the sender signal. In order to decode a signal the *signal-to-interference-plus-noise-ratio* $(SINR)$ has to be above a threshold that depends on the given hardware. However, even an optimal power control algorithm cannot guarantee acceptable $SINR$ levels for all links concurrently because, in general, only a subset of all links can be scheduled in parallel. It is therefore unavoidable to postpone the transmission of some communication requests to subsequent time slots. As short schedules maximize network throughput, the ultimate aim of any scheduling protocol is hence to find a schedule of minimum (or at least close to minimal) length.

Scheduling in wireless networks being of utmost theoretical and practical importance, it is not surprising that numerous heuristics are known for this problem. In this paper, we prove that for certain problem instances all these heuristics perform poorly (Section V). In addition we present a new scheduling algorithm called LDS that exhibits explicit worst-case guarantees (Section IV). We prove that on a class of scenarios our algorithm performs *exponentially* better than all the previous heuristics we are aware of. In particular, LDS schedules n transmission requests in $O(\log^2 n)$ time whereas previous heuristics require $\Omega(n)$ time.

II. Related Work and Existing Protocols

A wide range of models and various classes of protocols have been suggested in order to solve the problem of scheduling and power control. One line of research uses a graph representation of a wireless network, modeling interference by some (often binary) graph property. Assuming equal transmission powers for all sending nodes, for instance, the set of "interference-edges" contains pairs of nodes within a certain distance to each other, thus modeling interference as a local measure. Graph-based scheduling algorithms usually employ an implicit or explicit coloring strategy, which neglects the aggregated interference of nodes located further away.

More importantly, as shown in [14], graph-based scheduling algorithms are too conservative as they do not tap the full potential of spatial reuse. Overlapping links, for instance, are not scheduled simultaneously in a graph-based scheduling algorithm although this is feasible in practice [14]. The performance of graph-based algorithms is examined in [10] and [2] along with a demonstration that such simplistic graph-based approaches are inferior to algorithms in more realistic $SINR$ models.

As we argue in detail in Section V, algorithms explicitly defined for the $SINR$ model can broadly be classified into three categories. One approach is to assign the same power to all transmitting nodes. In [13] it is shown that protocols with such uniform power assignment can result in long schedules. In the same paper it is proved that the second intuitive procedure, adjusting the power proportionally to the so-called "energy-metric", can lead to long schedules as well.

More sophisticated methods are based on results from [1], where Aein shows how to determine the *maximum achievable* $SINR^*$ in polynomial time for satellite communications system. These results being directly applicable to wireless networks, it is possible to find an optimal power assignment efficiently. However, the problem is that $SINR^*$ may be too low to guarantee correct reception at all receivers. That implies that our problem of partitioning the set of communication requests into time slots meeting the required $SINR$ criteria remains unresolved. A brute force approach for finding the optimal schedule attempts to find for each time slot the largest set of remaining links which can be scheduled simultaneously by checking for each subset of links whether it allows a sufficiently high $SINR$. As there are 2^n subsets, however, the required time complexity grows exponentially with the number of links.

Consequently, many computationally efficient methods for postponing the transmission of links according to some ("link removal") heuristics have been devised. The first among them is presented by Zander [21]. He proposes an algorithm called SRA, which removes nodes from the current time slot by a stepwise approximation criterion. In [20] Zander devises an improved algorithm called LISRA that requires less knowledge of the network and the $SINR$ for each time slot converges to $SINR^*$ in a distributed fashion. Subsequently, several improvements on this convergence procedure have been proposed, e.g. [9], [8]. The idea of Lee et al. [12] is to postpone links which either have a high level of interference at the receiver or links of which the sender causes much interference to other receivers. The distributed algorithm proposed by Wang et al. [19] eliminates links which cause most interference in order to allow the remaining links to reach an acceptable $SINR$ level. Most recently, Brar et al. [4] present a scheduling method that is based on a greedy assignment of weighted colors.

All the above polynomial-time algorithms have one crucial drawback: The authors provide no worst-case analysis on their performance and all assumptions on their algorithm's quality are based on simulations and—in the case of [4]—analysis of randomly deployed networks. In Section V, we show that these link removal heuristics have a bad worst-case performance, creating schedules which are exponentially longer than necessary for certain networks.

The same limitation holds for the influential algorithm for next neighbor transmissions and power control by ElBatt and Ephremides [6]. They combine two heuristics to produce a short schedule and the corresponding power assignment. First a set of *valid* links is selected by greedily choosing nodes such that no node is receiving and transmitting in the same slot (to avoid self-interference) and no sender is situated within a certain range of an already selected receiving node. In a second phase, Zander's LISRA algorithm is applied to these links. As it is possible to construct scenarios, where all links together form a valid set, the worst case behavior of LISRA carries over to the algorithm of [6] as well (see Section V).

Recently, polynomial-time algorithms with provable guarantees in physical model environments for specific network topologies were proposed and analyzed in [13], [15]. In this paper, we improve on these algorithms and give strict worst-case guarantees even in scenarios in which no efficient bounds have been derived.

The problem of scheduling broadcast requests has been studied by Ephremides and Truong [7]. They show that in a generalized, *non-geometric* model, finding an optimal schedule is NP-complete, if no interference is tolerated. Other aspects of scheduling and power control are studied for instance in [3], [5], [16], [17], [18].

III. Model

In this paper, we are interested in devising scheduling protocols that exhibit a provably good performance even in non-uniformly distributed networks. We therefore consider the network to consist of a set of n nodes $X = \{x_1, \ldots, x_n\}$ that are arbitrarily (possibly even worst-case) located in the Euclidean plane. The Euclidean distance between two nodes $x_i, x_j \in X$, is denoted by $d(x_i, x_j)$. For simplicity and without loss of generality, we assume that the minimal distance between any two nodes is 1.

A communication request λ_i from a sender $s_i \in X$ to a receiver $r_i \in X$ is represented as a directed link (s_i, r_i) with length $d_i = d(s_i, r_i)$.

A. The Physical SINR Model

A crucial aspect when studying scheduling in wireless networks is to use an appropriate model. In the past, researchers have studied a wide range of communication models, ranging from complex channel models to simplistic graph-based protocol models. A standard model that is realistic, but also concise enough to allow for stringent reasoning and proofs is the Physical *Signal-to-Interference-plus-Noise-Ratio* ($SINR$) model [11]. In this model, the successful reception of a transmission depends on the received signal strength, the interference caused by nodes transmitting simultaneously, and the ambient noise level.

The received power $P_r(s_i)$ of a signal transmitted by sender s_i at an intended receiver r_i is

$$P_r(s_i) \;=\; P(s_i) \cdot g(s_i, r_i),$$

where $P(s_i)$ is the transmission power of s_i and $g(s_i, r_i)$ is the propagation attenuation (link gain) modeled as $g(s_i, r_i) = d(s_i, r_i)^{-\alpha}$. The *path-loss exponent* α is a constant between 2 and 6, whose exact value depends on external conditions of the medium (humidity, obstacles, ...), as well as the exact sender-receiver distance. As common, we assume that $\alpha > 2$ [11].

Given a request $\lambda_i = (s_i, r_i)$, we use the notation $I_r(s_j) = P_r(s_j)$ for any other sender s_j concurrent to s_i, in order to emphasize that the signal power transmitted by s_j is perceived at r_i as interference. The *total interference* I_r experienced by a receiver r_i is the sum of the interference power values created by all nodes in the network transmitting simultaneously (except the intending sender s_i), that is, $I_r := \sum_{s_j \in X \setminus \{s_i\}} I_r(s_j)$. Finally, let N denote the ambient noise power level. Then, r_i receives s_i's transmission if and only if

$$\begin{aligned} SINR(i) &= \frac{P_r(s_i)}{N + \sum_{j \neq i} I_r(s_j)} \qquad (1) \\ &= \frac{P(s_i) g(s_i, r_i)}{N + \sum_{j \neq i} P(s_j) g(s_j, r_i)} \\ &= \frac{\frac{P(s_i)}{d(s_i, r_i)^{\alpha}}}{N + \sum_{j \neq i} \frac{P(s_j)}{d(s_j, r_i)^{\alpha}}} \;\geq\; \beta, \end{aligned}$$

where β is the minimum $SINR$ required for a successful message reception.

In our analysis we often use the gain matrix $G = [g(s_i, r_j)]$ and its normalized correspondent $Z = [\frac{g(s_i, r_j)}{g(s_i, r_i)}]$.

B. Problem Formulation

The aim of a scheduling and power control algorithm is to generate a sequence of power assignment vectors, such that the $SINR$ level is above a threshold β at every intended receiver and all links are scheduled successfully at least once.

More formally, let Λ be a set of *communication requests* λ_i. P_t denotes the *power assignment vector*, where $P_t(s_i)$ determines the transmission power of sender s_i in time slot t. A *schedule* is represented by $\mathcal{S} = (P_1, \ldots, P_T)$. As in [11], it is assumed without loss of generality that transmissions are slotted into synchronized slots of equal length.

Let L_t be the set of all successfully scheduled links in time slot t. The goal is that after as few time slots as possible the union of all sets L_t equals the set of requests Λ. The *scheduling complexity* defined in [13] is a measure that captures the amount of time required by a scheduling protocol to schedule requests in the Physical $SINR$ model.

Definition 3.1: The scheduling problem for Λ is to find a schedule $\mathcal{S}$ of minimal length T such that the union of all successfully transmitted links $\bigcup_{t=1}^{T(\mathcal{S})} L_t$ equals Λ. An algorithm's *scheduling complexity* is the length of the schedule generated.

The scheduling complexity of a protocol reflects the protocol's quality. Ideally, a wireless scheduling protocol should achieve a good (close to optimal) scheduling complexity in all networks and for arbitrary communication requests. Understanding the scheduling complexity of different protocols in arbitrary networks is therefore of fundamental practical and theoretical interest in wireless networking.

C. The Disturbance

Since we study arbitrary, possibly worst-case network and request settings, we introduce a formal measure that captures the intrinsic difficulty of scheduling a given set of communication requests.

For a given set of communication requests Λ and some constant $\rho \geq 1$, we define the *ρ-disturbance* as the maximal number of senders (receivers) that are in close physical proximity (depending on the parameter ρ) of any sender (receiver). Consider disks S_i and R_i of radius d_i/ρ around sender s_i and receiver r_i, respectively. Formally, the *ρ-disturbance of a link* λ_i is the larger of either the number of senders in S_i or the number of receivers in R_i. The *ρ-disturbance of* Λ is then the maximum ρ-disturbance of any link $\lambda_i \in \Lambda$.

Definition 3.2: Given a set of requests Λ. The ρ-disturbance, denoted as χ_ρ of Λ is defined as

$$\chi_\rho := \max_{\lambda_i \in \Lambda} \chi_\rho(\lambda_i),$$

where the disturbance $\chi_\rho(\lambda_i)$ for request λ_i is the maximum of $|\{r_j \mid d(r_j, r_i) \leq d_i/\rho\}|$ and $|\{s_j \mid d(s_j, s_i) \leq d_i/\rho\}|$.

As it turns out, the *disturbance* of a set of requests indeed captures the fundamental difficulty of scheduling these requests. Solving problem instances with low disturbance efficiently is very important in practice since in realistic networks one always tries to prevent situations with many receivers clustered in the same area. Section IV presents LDS, a scheduling protocol that achieves a provably fast performance for all networks and requests that have low disturbance. On the other hand, we prove in Section V that currently known scheduling protocols may perform highly sub-optimally even in instances with low disturbance. In fact, the number of time slots required by any such protocol may be exponentially higher than the optimum.

IV. Efficient Scheduling Protocol

In this section, we propose a novel scheduling protocol, called the *Low-Disturbance Scheduling Protocol (LDS)*, which achieves provable performance guarantees even in worst-case networks. In particular, given a network and a set of communication requests, LDS computes a schedule whose length is within a polylogarithmic factor of the network's disturbance.

A. LDS Protocol

The protocol consists of three parts: a pre-processing step, the main scheduling-loop, and a test-subroutine that determines whether a link is to be scheduled in a given time slot.

The purpose of the pre-processing phase is to assign two values $\tau(i)$ and $\gamma(i)$ to every request λ_i. The value $\gamma(i)$ is an integer values between 1 and $\lceil \log(3n\beta) + \rho \log \alpha \rceil$. The idea is that only requests with the same $\gamma(i)$ values are considered for scheduling in the same iteration of the main scheduling-loop (Lines 2 and 3 of the main scheduling-loop). The second assigned value, $\tau(i)$, further partitions the requests. In particular, it holds that the length of all requests that have the same $\gamma(i)$ and $\tau(i)$ differ by at most a factor two. On the other hand, we show in Lemma 4.4 that if two requests λ_i and λ_j satisfy $\tau(i) < \tau(j)$, then the length of λ_i, d_i, is at least by a factor $\frac{1}{2}(3n\beta\rho^\alpha)^{\tau(j)-\tau(i)}$ longer than d_j. Generally speaking, the assignment of $\tau(i)$ ensures that the smaller the value $\tau(i)$ assigned to a requests λ_i, the longer the corresponding communication link, and vice versa.

In summary, the pre-processing phase partitions the set of requests in such a way that two requests λ_i and λ_j that are assigned the same $\gamma(i)$ have either almost equal length (if, $\tau(i) = \tau(j)$) or very different length. This partition will turn out to be crucial in the actual scheduling process, which takes part in the subsequent main scheduling-loop.

Each for-loop iteration of the main scheduling-loop schedules the set of requests having the same $\gamma(i)$ values, denoted by $\mathcal{F}_k$. As long as not all requests of $\mathcal{F}_k$ have been successfully scheduled, the algorithm considers the remaining requests in $\mathcal{F}_k$ in decreasing order of their length d_i. Specifically, the algorithm checks for each request whether it can safely be scheduled alongside the longer links that have already been selected. If a request is chosen to be scheduled in time slot t, it is added to L_t, otherwise it remains in $\mathcal{F}_k$.

The decision whether a request λ_i is selected for scheduling or not takes place in the $\mathbf{allowed}(\lambda_\mathbf{i}, \mathbf{L_t})$ subroutine. For each (longer) request $\lambda_j \in L_t$ that has already been chosen to be scheduled in time slot t, the subroutine checks three conditions. Only if none of them is violated, λ_i is added to L_t. Notice, however, that the selection-criteria are significantly more complex than the simple "reuse-distance" argument that has been used in previous work (e.g. [6]). In particular, the second criterion states that λ_i is scheduled only if for all longer requests $\lambda_j \in L_t$, it holds that $d_i \cdot (3n\beta\rho^\alpha)^{\frac{\tau(i)-\tau(j)+1}{\alpha}} > d(s_i, r_j)$ if $\tau(i) > \tau(j)$. That is, the distance that must be maintained between the sender s_i of λ_i and the receiver of r_j of some $\lambda_j \in L_t$ depends on the relative values of $\tau(i)$ and $\tau(j)$ assigned in the pre-processing phase.

The definition of the three selection-criteria guarantees that all simultaneously transmitted requests in a single time slot are received successfully by the intended receivers. Additionally, the subsequent analysis section shows that all requests can be scheduled efficiently even in worst-case networks.

B. Analysis

In this section, we prove that the LDS protocol is both correct (i.e., all requests scheduled during the protocol's execution are received successfully at the intended receivers) and fast. Specifically, we prove that every set of requests can be scheduled efficiently even in worst-case networks provided that the ρ-disturbance of the requests is small. As we show in Section V, this distinguishes the LDS protocol from all existing protocols, that may perform badly even if the disturbance is small.

Algorithm 1 The LDS Protocol for requests Λ

Pre-processing phase:
```
 1: $\tau\text{cur} := 1;\quad \gamma\text{cur} := 1;\quad \text{last} := d_1;$
 2: Consider all requests $\lambda_i \in \Lambda$ in decreasing order of $d_i$:
 3: for each $\lambda_i \in \Lambda$ do
 4:    if $\text{last}/d_i \geq 2$ then
 5:       if $\gamma\text{cur} < \lceil \log(3n\beta) + \rho \log \alpha \rceil$ then
 6:          $\gamma\text{cur} := \gamma\text{cur} + 1;$
 7:       else
 8:          $\gamma\text{cur} := 1;\ \tau\text{cur} := \tau\text{cur} + 1;$
 9:       end
10:       $\text{last} := d_i;$
11:    end
12:    $\gamma(i) := \gamma\text{cur};\ \ \tau(i) := \tau\text{cur};$
13: end
```
Main scheduling-loop:
```
 1: Define constant $\nu$ such that $\nu := 4N;$
 2: $t := 1;$
 3: for $k = 1$ to $\lceil \log(3n\beta) + \rho \log \alpha \rceil$ do
 4:    Let $\mathcal{F}_k$ be the set of all requests $\lambda_i$ with $\gamma(i) = k$.
 5:    while not all requests in $\mathcal{F}_k$ have been scheduled do
 6:       $L_t := \emptyset;$
 7:       Consider all $\lambda_i \in \mathcal{F}_k$ in decreasing order of $d_i$:
 8:          if $allowed(\lambda_i, L_t)$ then
 9:             $L_t := L_t \cup \{\lambda_i\};\quad \mathcal{F}_k := \mathcal{F}_k \setminus \{\lambda_i\}$
10:          end if
11:       Schedule all $\lambda_i \in E_t$ in time slot $t$, assigning $s_i$
          a transmission power of $P_i = \nu \cdot d_i^\alpha \cdot (3n\beta\rho^\alpha)^{\tau(i)};$
12:       $t := t + 1;$
13:    end while
14: end for
```
$\mathbf{allowed}(\lambda_\mathbf{i}, \mathbf{L_t})$
```
1: Define constant $\mu$ such that $\mu := 4\sqrt[\alpha]{\frac{120\beta(\alpha-1)}{\alpha-2}};$
2: for each $\lambda_j \in L_t$ do
3:    $\delta_{ij} := \tau(i) - \tau(j);$
4:    if $\tau(i) = \tau(j)$ and $\mu \cdot d_i > d(s_i, s_j)$
5:    or $\tau(i) > \tau(j)$ and $d_i \cdot (3n\beta\rho^\alpha)^{\frac{\delta_{ij}+1}{\alpha}} > d(s_i, r_j)$
6:    or $\tau(i) > \tau(j)$ and $d_j/\rho > d(s_j, r_i)$
7:    then return false
8: end for
9: return true
```

We begin with two simple lemmas that bound the amount of interference created by simultaneously scheduled senders s_j at an intended received r_i.

Lemma 4.1: Let λ_i and λ_j be two requests with $\tau(i) \neq \tau(j)$ the protocol selects for the same time slot. The interference at r_i created by s_j is at most $I_r(s_j) \leq \nu \cdot \rho^{\alpha\tau(i)} \cdot (3n\beta)^{\tau(i)-1}$, where $\nu = 4N$.

Proof: We distinguish two cases, depending on the relative values of $\tau(i)$ and $\tau(j)$.
a) $\tau(i) < \tau(j)$: In this case, we know that $d_i > d_j$ by the definition of Line 6 in the main scheduling-loop. Hence, by

the time λ_j is added to L_t by the $\mathbf{allowed}(\ell_\mathbf{i}, \mathbf{L_t})$ subroutine, λ_i is already in L_t. Because $\mathbf{allowed}(\ell_\mathbf{i}, \mathbf{L_t})$ evaluated to $\mathbf{true}$, the distance $d(s_j, r_i)$ is at least $d_i \cdot (3n\beta\rho^\alpha)^{\frac{\delta_{ij}+1}{\alpha}}$, where $\delta_{ij} := \tau(i) - \tau(j)$. Hence the interference of s_j at r_i is at most

$$\begin{aligned} I_r(s_j) &= \frac{P_j}{d(s_j, r_i)^\alpha} \leq \frac{\nu \cdot d_j^\alpha \cdot (3n\beta\rho^\alpha)^{\tau(j)}}{d_j^\alpha \cdot (3n\beta\rho^\alpha)^{\delta_{ij}+1}} \\ &= \nu \cdot (3n\beta\rho^\alpha)^{\tau(i)-1}, \end{aligned}$$

which is smaller than the upper-bound claimed in the lemma. b) $\tau(i) > \tau(j)$: In this case, it holds that $d_i < d_j$. Because both links have been selected by the protocol, it follows that $d(s_j, r_i) \geq d_j/\rho$. Furthermore, it holds that $\tau(i) \geq \tau(j) + 1$, thus the maximum amount of interference that can be caused by s_j at r_i is

$$\begin{aligned} I_r(s_j) &= \frac{P_j}{d(s_j, r_i)^\alpha} \leq \frac{\nu \cdot d_j^\alpha \cdot (3n\beta\rho^\alpha)^{\tau(j)}}{(d_j/\rho)^\alpha} \\ &= \nu \cdot \rho^{\alpha(\tau(j)+1)} \cdot (3n\beta)^{\tau(j)} \\ &\leq \nu \cdot \rho^{\alpha\tau(i)} \cdot (3n\beta)^{\tau(i)-1}. \end{aligned}$$

□

The next lemma bounds the total interference created by all nodes transmitting simultaneously for which $\tau(i) = \tau(j)$.

Lemma 4.2: Given a request λ_i, the total interference I_r^0 at r_i created by all senders s_j transmitting simultaneously for which $\tau(i) = \tau(j)$ is at most $I_r^0 \leq \frac{\nu}{4}\beta^{\tau(i)-1}(3n\rho^\alpha)^{\tau(i)}$.

Proof: By the pre-processing phase, it holds that if both $\tau(i) = \tau(j)$ and $\gamma(i) = \gamma(j)$, then $\frac{d_j}{2} \leq d_i \leq 2d_j$ is satisfied. Thus, all requests have roughly the same lengths and we can bound the total interference using a standard area argument. Specifically, by Line 3 of the $\mathbf{allowed}(\lambda_\mathbf{i}, \mathbf{L_t})$ subroutine, λ_i and λ_j being scheduled in the same time slot implies that $\mu \cdot d_i > d(s_i, s_j)$, where $\mu := 4\sqrt[\alpha]{120\beta(\alpha-1)/\alpha-2}$. Now, consider all concurrently transmitting nodes s_j for which $\tau(i) = \tau(j)$ and consider disks D_j of radius $\frac{\mu d_i}{4}$ centered at each such sender. Because of the required spatial reuse distance and the fact that the length of two requests differs by at most a factor two, it holds that $d(s_j, s_{j'}) > \frac{\mu d_i}{2}$ and hence, disks D_j do not overlap. The area of each such disk is $A(D_i) \geq (\frac{\mu d_i}{4})^2\pi$.

Consider rings R_k of width μd_i around r_i, consisting of all senders s_j transmitting simultaneously for which $\tau(i) = \tau(j)$ and $\frac{k\mu}{2}d_i \leq d(s_j, r_i) \leq \frac{(k+1)\mu}{2}d_i$. Notice that by the first condition of the subroutine, R_1 must be empty. Consider a ring R_k and the transmitters contained in it. All corresponding disks D_i must be entirely located in an "extended" ring R_k^* of area

$$\begin{aligned} A(R_k^*) &= \left[\left(\frac{(k+1)\mu d_i}{2} + \frac{\mu d_i}{2}\right)^2 - \left(\frac{k\mu d_i}{2} - \frac{\mu d_i}{2}\right)^2\right]\pi \\ &= \frac{3(2k+1)}{4}\mu^2 d_i^2\pi. \end{aligned}$$

The distance of a sender s_j in R_k from r_i has a lower bound of $\frac{k\mu}{2}d_i$. Furthermore, each such sender transmits at a power at most $\nu \cdot (2d_i)^\alpha \cdot (3n\beta\rho^\alpha)^{\tau(i)}$. Using the fact that the disks D_i do not overlap, we can bound the interference at r_i from nodes in ring R_k by

$$\begin{aligned} I_r^0(R_k) &\leq \frac{A(R_k^*)}{A(D_i)} \cdot \frac{\nu(3\beta n\rho^\alpha)^{\tau(i)} \cdot (2d_i)^\alpha}{(\frac{k\mu}{2}d_i)^\alpha} \\ &< \frac{12(2k+1)\nu(3\beta n\rho^\alpha)^{\tau(i)} \cdot 2^{2\alpha}}{(k\mu)^\alpha} \\ &\leq \frac{30\nu(3\beta n\rho^\alpha)^{\tau(i)} \cdot 2^{2\alpha}}{k^{\alpha-1}\mu^\alpha}, \end{aligned}$$

where the last inequality follows because only rings where $k \geq 2$ need to be considered. Summing up the interference generated by all rings results in a total interference of

$$\begin{aligned} I_r^0 &< \sum_{k=1}^{\infty} I_r^0(R_k) \leq \frac{30\nu(3\beta n\rho^\alpha)^{\tau(i)} \cdot 2^{2\alpha}}{\mu^\alpha} \sum_{k=1}^{\infty} \frac{1}{k^{\alpha-1}} \\ &< \frac{30\nu(3\beta n\rho^\alpha)^{\tau(i)} \cdot 2^{2\alpha}}{\mu^\alpha} \cdot \frac{\alpha-1}{\alpha-2} \\ &< \frac{\nu}{4}\beta^{\tau(i)-1}(3n\rho^\alpha)^{\tau(i)}, \end{aligned}$$

where the second-to-last inequality follows from a bound on Riemann's zeta-function and the last one from plugging in the definition of μ. This concludes the proof. □

Using the previous two lemmas, it can now be shown that every message scheduled for transmission by the algorithm can be decoded successfully by the intended receiver.

Theorem 4.3: The schedule computed by the protocol allows all requests to be successfully received by the intended receiver.

Proof: Using Lemmas 4.1 and 4.2, we bound the total interference I_r created by concurrent senders as

$$\begin{aligned} I_r &\leq \frac{\nu}{4}\beta^{\tau(i)-1}(3n\rho^\alpha)^{\tau(i)} + \sum_{s_j:\tau(i)\neq\tau(j)} \nu\rho^{\alpha\tau(i)}(3n\beta)^{\tau(i)-1} \\ &\leq (\nu/4 + \nu/3)\,(3n\rho^\alpha)^{\tau(i)}\beta^{\tau(i)-1}. \end{aligned}$$

The theorem follows from verifying that the resulting $SINR$ is sufficiently high and by noting that every request is scheduled for transmission exactly once by the algorithm.

$$SINR(r_i) \geq \frac{\nu \cdot (3n\beta\rho^\alpha)^{\tau(i)}}{N + \left(\frac{\nu}{3} + \frac{\nu}{4}\right)(3n\rho^\alpha)^{\tau(i)}\beta^{\tau(i)-1}} > \beta.$$

□

So far, we have proven that the produced schedule is correct in the sense that all messages are actually received successfully. It now remains to show that the schedule is short and includes all requests. For this reason, we bound the number of time slots required to schedule all requests that have the same $\gamma(i)$ value. That is, we bound the amount of time used for one iteration of the for-loop in the main scheduling-loop. We begin with two simple lemmas.

Lemma 4.4: Consider two requests λ_i and λ_j with $\gamma(i) = \gamma(j)$. If $\tau(i) \geq \tau(j)$ it holds that

$$d_j \geq 1/2(3n\beta\rho^\alpha)^{\tau(i)-\tau(j)} \cdot d_i.$$

Proof: If two requests λ_i and λ_j have the same γ value but different τ values, $\gamma(i)$ has been increased at least

$(\tau(i)-\tau(j))\lceil\log(3n\beta)+\rho\log\alpha\rceil$ times since processing λ_j. The reason is that $\gamma(i)$ must be increased exactly $\lceil\log(3n\beta)+\rho\log\alpha\rceil$ times (and reset to 0 once) in order to reach $\gamma(i)=\gamma(j)$ for the next higher value of τ. Due to Line 4, each but one such increase implies a halving of the length d_j. Hence,

$$d_j \geq d_i \cdot 2^{(\tau(i)-\tau(j))(\log(3n\beta)+\rho\log\alpha)} \geq d_i \cdot (3n\beta\rho^\alpha)^{\tau(i)-\tau(j)}.$$

□

Lemma 4.5: In any disk D of radius R, there can be at most χ_ρ receivers r_i of requests λ_i with length $d_i \geq 2\rho R$.

Proof: If $d_i \geq 2\rho R$ for all λ_i, the disk of radius d_i/ρ around each receiver fully covers D. The claim now follows from the definition of χ_ρ. □

In order to bound the number of time slots required to schedule all requests in the same iteration of the main loop, we define the notion of *blocking requests*.

Definition 4.1: λ_j is a *blocking request* for λ_i if $\gamma(i)=\gamma(j)$, $d_j \geq d_i$, and $\mathbf{allowed}(\lambda_\mathbf{i}, \mathbf{L_t})$ evaluates to $false$ if $\lambda_j \in L_t$. B_i denotes the set of blocking requests of λ_i.

Consequently, blocking requests $\lambda_j \in B_i$ are those requests that can "block" a request λ_i from being scheduled in a given time slot. Because each such blocking request can prevent λ_i from being scheduled only in a single time slot (when it is scheduled itself), it holds that λ_i is scheduled in time slot $|B_i|+1$ or earlier of the for-loop iteration when requests with $\gamma(i)$ are scheduled. We distinguish three kinds of blocking requests, depending on which of the three conditions in the $\mathbf{allowed}(\lambda_\mathbf{i}, \mathbf{L_t})$ subroutine is responsible for the blocking, and we bound the number of blocking requests in each category independently.

Lemma 4.6: Let B_i^1 be the set of blocking requests $\lambda_j \in B_i$ with $\tau(i)=\tau(j)$ and $\mu d_i > d(s_i,s_j)$. For all λ_i it holds that $|B_i^1| \leq 4\rho^2(\mu+2)^2\chi_\rho$.

Proof: From $\tau(i)=\tau(j)$, it follows by Lemma 4.4 that $d_i \leq d_j \leq 2d_i$ for all $\lambda_j \in B_i^1$. By Lemma 4.5, we know that there can be at most χ_ρ receivers of blocking requests with length at least d_i in any disk of radius $d_i/(2\rho)$. Because $\mu d_i > d(s_i,s_j)$ holds for any blocking request in B_i^1, any receiver corresponding to a blocking request must be located inside a disk of radius $(\mu+2)d_i$ centered at s_i. Thus,

$$|B_i^1| \;\leq\; \chi_\rho \cdot \frac{\pi(\mu+2)^2 d_i^2}{\frac{1}{(2\rho)^2}\pi d_i^2} \;=\; 4\rho^2(\mu+2)^2\chi_\rho.$$

□

The next lemma is key to our worst-case result and bounds the number of blocking requests that prevent a shorter request by the second condition of the $\mathbf{allowed}(\lambda_\mathbf{i}, \mathbf{L_t})$ subroutine.

Lemma 4.7: Let B_i^2 be the set of blocking requests $\lambda_j \in B_i$ with $\tau(i) > \tau(j)$ and $d_i \cdot (3n\beta\rho^\alpha)^{\delta_{ij}+1/\alpha} > d(s_i,r_j)$. For all λ_i it holds that $|B_i^2| \leq 16\log(n+1)\chi_\rho$.

Proof: First we show that for any integer $\varphi \geq -1$, there can be $O(\chi_\rho)$ different blocking requests $\lambda_j \in B_i^2(\varphi)$ where

$$(3n\beta\rho^\alpha)^{\alpha^\varphi} \cdot d_i \;<\; d(s_i,r_j) \;\leq\; (3n\beta\rho^\alpha)^{\alpha^{\varphi+1}} \cdot d_i.$$

By the definition of the second condition in the $\mathbf{allowed}(\lambda_\mathbf{i}, \mathbf{L_t})$ subroutine, each such request $\lambda_j \in B_i^2(\varphi)$ must satisfy $\frac{\delta_{ij}+1}{\alpha} > \alpha^\varphi$, and hence $\delta_{ij} \geq \alpha^{\varphi+1}$. By Lemma 4.4, we know that each such blocking request $\lambda_j \in B_i^2(\varphi)$ with $d(s_i,r_j)$ in the range specified above must be of length at least $d_j \geq \frac{1}{2}(3n\beta\rho^\alpha)^{\alpha^{\varphi+1}} \cdot d_i$.

It remains to show that there can be at most $O(\chi_\rho)$ such requests $\lambda_j \in B_i^2(\varphi)$. For simplicity, define $K := (3n\beta\rho^\alpha)^{\alpha^{\varphi+1}} \cdot d_i$. By Lemma 4.5 and the above lower bound on d_j, at most χ_ρ receivers of requests in $B_i^2(\varphi)$ can be in any disk of radius $\frac{K}{4\rho}$. By definition all these receivers r_j must be within distance K of s_i, thus that there can be at most $16\rho^2\chi_\rho$ blocking requests in $B_i^2(\varphi)$ by the classic area argument.

We know that for any integer $\varphi > -1$, there are at most $16\rho^2\chi_\rho$ blocking requests in $B_i^2(\varphi)$. The value δ_{ij} between two requests λ_i and λ_j cannot exceed n and hence, the furthest distance $d(s_i,r_j)$ of any blocking request λ_j can be $(3n\beta\rho^\alpha)^{\frac{n+1}{\alpha}} d_i$. It follows that $|B_i^2(\varphi)|=0$ for all $\varphi > \frac{n+1}{\alpha}$. Finally, because $\alpha^{(\varphi+1)} > \frac{n+1}{\alpha}$ for some $\varphi \geq \log_\alpha(n+1)$, it follows that there are at most $O(\log n)$ many "rings", each of which can contain at most $16\rho^2\chi_\rho$ blocking receivers. Hence,

$$|B_i^2| \;=\; \sum_{\varphi:=-1}^{\infty} |B_i^2(\varphi)| \;\leq\; \log_\alpha(n+1)\cdot 16\rho^2\chi_\rho.$$

□

Finally, we bound the number of blocking requests that can block a request r_i due to the third constraint in the $\mathbf{allowed}(\lambda_\mathbf{i}, \mathbf{L_t})$ subroutine.

Lemma 4.8: Let B_i^3 be the set of requests $\lambda_j \in B_i$ with $\tau(i) > \tau(j)$ and $\frac{d_j}{\rho} > d(s_j,r_i)$. It holds $|B_i^3| \leq 6\chi_\rho\ \forall\lambda_i$.

Proof: Assume for contradiction that there are more than $6\chi_\rho$ such blocking requests $\lambda_j \in B_i^3$. For each of these $d_j > d_i$. Partition the area around r_i into cones of angle $\pi/3$. At least one of these cones must contain the senders s_j of $\chi_\rho+1$ or more blocking requests. The angle of this cone being $\pi/3$, the distance of the furthest such sender s'_j to each of the other blocking senders s_j in this cone is at most $d(s'_j,s_j) < d(s'_j,r_i)$, and hence, $d(s'_j,s_j) < d_i/\rho < d'_j/\rho$. There are at least $\chi_\rho+1$ senders within distance d'_j/ρ of s'_j, which contradicts χ_ρ's definition. □

As every blocking request can block a request λ_i at most once, we combine the above and prove the following theorem.

Theorem 4.9: The number of time slots required by Algorithm 1 to successfully schedule all requests $\lambda_i \in \Lambda$ is at most $O\left(\chi_\rho\rho^2\log n\cdot(\log n+\rho)\right)$.

Proof: By Lemmas 4.6, 4.7, and 4.8, any request λ_i can be blocked by at most

$$B_i^1+B_i^2+B_i^3 \leq 4\rho^2(\mu+2)^2\chi_\rho + 16\rho^2\log(n+1)\chi_\rho + 6\chi_\rho$$

blocking requests. Thus, after at most $O(\chi_\rho\rho^2\cdot\log n)$ iterations of the while-loop, all requests having the same $\gamma(i)$ value are scheduled successfully. The theorem follows as the number of for-loop iterations is $\lceil\log(3n\beta)+\rho\log\alpha\rceil$. □

The next section shows that our algorithm significantly outperforms other known scheduling protocols in many settings.

Algorithm 2 Generic Link Removal Algorithm

```
1:  time slot $t := 1$;
2:  while there are links to schedule do
3:      compute $SINR^*$ and $\mathbf{P}^*$ from $Z$;
4:      while $SINR^* \leq \beta$ do
5:          remove links $\lambda_k$ for which $CON$ is satisfied;
6:          compute $SINR^*$ and $\mathbf{P}^*$ from new $Z$;
7:      end while
8:      schedule the links of $Z$ in time slot $t$ and assign $\mathbf{P}^*$;
9:      time slot $t := t + 1$;
10:     compute new $Z$ for unscheduled links;
11: end while
```

V. Inefficiency of Existing Protocols

Intuitively, the disturbance of a set of requests in a network characterizes the difficulty of scheduling these requests in a wireless communication environment. Therefore, an efficient scheduling protocol should be capable of generating short schedules in settings with low disturbance. Unfortunately, all previously known scheduling protocols may require a linear number of time slots in order to schedule a set of requests even if their ρ-disturbance is as low as 1.

Existing scheduling algorithms and protocols for the $SINR$ model can be classified into three classes[1]:

- *uniform power assignment*: the transmission power of all nodes is the same.
- *linear power assignment*: the transmission power for a link of length d_i is set to a value proportional to d_i^α. Protocols analyzed using the so-called 'energy-metric" belong to this category.
- *link removal heuristics*

Recently, it has been proven in [13] that every protocol employing a *uniform or linear power assignment* scheme has a poor worst-case efficiency. In particular, any such protocol may require a linear number of time slots even if every node merely wants to transmit to its closest neighbor in the network.

Theorem 5.1 ([13]): Every protocol employing a uniform or linear power assignment scheme has a worst-case scheduling complexity of $\Omega(n)$ even in settings with ρ-disturbance 1.

Theorem 5.1 indicates that a large number of scheduling algorithms proposed in the literature has bad worst-case behavior, including for instance the recent algorithm in [4] for which the authors prove guarantees in randomly deployed networks.

In contrast to these intuitive, but inefficient scheduling schemes, *link removal heuristics* are much more sophisticated. The heuristics known in the literature are all based on a generic link removal algorithm.

The idea of these algorithms is to postpone the transmission of a link λ_k from the set of the links if some condition CON holds, until the minimal $SINR$ level for successful reception is met. Then the optimal power vector is assigned and the procedure is repeated with the remaining links.

We scrutinize the four algorithms $SRA, SMIRA, WCRP$ and $LISRA$, which follow the execution of the generic algorithm and differ only in the condition CON.

SRA (Stepwise Removal Algorithm), devised by Zander in [21], iteratively removes the link with the largest row or column sum of Z, since these sums provide a bound on the maximal eigenvalue, until the required $SINR$ level is met.

$$CON : \max\{\sum_j Z_{kj}, \sum_j Z_{jk}\} \ is\ maximimal\ for\ k.$$

SMIRA (Stepwise Maximum Interference Removal Algorithm), by Lee et al. [12], excludes links which cause or receive the most interference when power is assigned optimally, taking the normalized link gain matrix Z and the corresponding optimal power vector into account.

$$CON : \max\{\sum_{j\neq k} P_j Z_{kj}, P_k \sum_{j\neq k} Z_{jk}\} \ is\ maximimal\ for\ k.$$

Lee et al. suggest versions of this algorithm considering only $\max_k(\sum_{j\neq k} P_j Z_{kj})$ or $\max_k(P_k \sum_{j\neq k} Z_{jk})$ in the condition and demonstrate with simulations, that they perform worse than SMIRA. Our analysis can be adapted easily to these cases with the same complexity result.

WCRP is a (distributed) algorithm presented in [19]. When adapted to our model, it first computes for each row i the value $MIMSR$ (maximum interference to minimum signal ratio), defined by

$$MIMSR(i) = \max\{\frac{\beta G(i,j)}{G(i,i)} | j \neq i \wedge j\ not\ scheduled\}$$

and removes links with MIMSR above a threshold ζ. We present here a simplified and centralized version, which produces schedules of at most the same length as the original algorithm. $CON : MIMSR(k) > \zeta.$

LISRA (Limited Information Stepwise Removal Algorithm), described in [20], postpones the transmission of the links with the lowest $SINR$ when all sender transmit with equal power, to increase the probability for the remaining links to reach the $SINR$ threshold[2]. To generate schedules with LISRA we replace Step 5 of the generic with

5a: set $\mathbf{P} = 1$ and compute $SINR$;
5b: remove links γ_k for which $\min_i SINR(i) = SINR(k)$;

$$CON : SINR(k)\ is\ minimal\ for\ k.$$

These algorithms have all been tested in situations with nodes distributed uniformly at random. No worst case analysis has been done and the authors do not give any guarantees on their behavior. To prove our point we construct an example where the schedules these algorithms produce are extremely long.

[1]Notice that protocols based on graph-models can typically be characterized as either employing a uniform or linear power assignment scheme.

[2]In its original version step 3 contains the execution of an iterative distributed algorithm based on locally available information. The number of rounds is fixed beforehand, hence the quality of the results depend on the convergence speed of the algorithm. As we are most interested in the maximal length of the schedules LISRA produces, we replace the algorithm in step 3 by a (centralized) eigenvalue decomposition.

Consider a scenario S with $k = \frac{n}{2}$ communication requests where all the sender and receiver nodes are situated on a straight line with the following distance to 0: Sender node $s_i = -2^i$, receiver node $r_i = 2^i, \forall 0 < i \leq k$. We set $\alpha = 3$, the noise level $N = 0$ and the minimum $SINR$ necessary for successful transmission to $\beta = 2$. For this situation all the algorithms described above perform poorly, namely they schedule each link individually and require $\Omega(n)$ time slots, even though we prove $O(\log n)$ time slots to be sufficient. Because the 3-disturbance of the above scenario S is $\chi_3 = 1$, our example demonstrates that these algorithms exhibit severe worst-case problems even in networks with low disturbance.

Theorem 5.2: SRA, LISRA, SMIRA and WCRP produce a schedule of length $\Omega(n)$ for the scenario S in which the 3-disturbance χ_3 is 1.

Proof: Starting from SRA, we prove the claim for each algorithm individually.

SRA: As we cannot schedule all links in the same slot, we compute the column and row sums of Z to decide which links we postpone to subsequent time slots. The sum for row i is $R_i = \sum_{j=1}^{n} z(i,j) = \sum_{j=1}^{n} \left(\frac{2^{i+1}}{2^j+2^i}\right)^{\alpha}$, which is maximal when $i = n$. Analogously the sum for column i is $C_i = \sum_{j=1}^{n} z(j,i) = \sum_{j=1}^{n} \left(\frac{2^{j+1}}{2^j+2^i}\right)^{\alpha}$. This sum is largest when $i = 1$, since i only appears in the denominator. Hence we have to determine $\max\{R_n, C_1\}$.

The summands of C_1 grow with j whereas the summands of R_n decrease. As a consequence we can simplify the analysis by comparing $\frac{2^{n+1}}{2^{n-j+1}+2^n}$ to $\frac{2^{j+1}}{2^j+2}$.

$$\frac{2^{n+1}}{2^{n-j+1}+2^n} = \frac{2^j}{1+2^{j-1}} = \frac{2^{j+1}}{2+2^j} \qquad \forall 0 < j \leq n.$$

Hence we know that the largest row sum is equal to the largest column row, which causes either the shortest or the longest link to be removed from the set of links to schedule in the next time slot. Without loss of generality we assume that we postpone the transmission of the shortest link.

Without the first link we have to deal with almost the same situation, the only difference is the start of the sums with $j = 2$ instead of 1. Again we remove the shortest link. This game continues until only one link is left, since two links next to each other cannot be scheduled in the same slot.

Lemma 5.3: Two links λ_i and λ_{i+1} cannot be scheduled in the same slot.

Proof: Let $\lambda_i = (-2^i, 2^i), \lambda_j = (-2^j, 2^j)$. We compute $Z = \begin{pmatrix} 1 & \left(\frac{2^{i+1}}{2^j+2^i}\right)^{\alpha} \\ \left(\frac{2^{j+1}}{2^j+2^i}\right)^{\alpha} & 1 \end{pmatrix}$ and set $j = i+1$. Now the larger eigenvalue is

$$\begin{aligned} \lambda^* &= 1/2\left(z_{1,1} + z_{2,2} + \sqrt{4z_{1,2}z_{2,1} + (z_{1,1} - z_{2,2})^2}\right) \\ &= 1/2\left(1 + 1 + \left(\sqrt{4 \cdot 2^{i+j+2}/(2^i+2^j)^2}\right)^{\alpha}\right) \\ &\overset{j=i+1}{=} 1 + \left(\frac{\sqrt{2^{2i+3}}}{2^i+2^{i+1}}\right)^{\alpha} = 1 + \left(\frac{\sqrt{8}}{3}\right)^{\alpha} > 1.83. \end{aligned}$$

Consequently $SINR^* = \frac{1}{\lambda^*-1} < 1.19$, implying that the links λ_i and λ_{i+1} cannot be transmitted simultaneously. □

We can derive from the above, that SRA schedules all links individually, i.e. the length of the schedule is $\Omega(n)$.

SMIRA: The transmission of link λ_i is postponed if either the interference received and the interference caused by link λ_i is above a certain threshold. As the receiving node of link 1 suffers from the highest level of interference we remove it. This situation occurs again in the next time slot, hence each link is scheduled individually, leading to a complexity of $\Omega(n)$.

WCRP: We compute the MIMSR value for each link i.

$$MIMSR(i) = \max_j \frac{\beta \cdot G(i,j)}{L \cdot G(i,i)} = \beta \cdot \max_i \left(\frac{2^{i+1}}{2^i+2^j}\right)^{\alpha}.$$

As $MIMSR(i)$ cannot exceed $\beta 2^{\alpha}$, we define $\zeta = 10$. Hence all links apart from the three shortest links are removed. Let us assume for simplicity that those can be scheduled in one slot. If we repeat this step, again the three shortest links remain and we can conclude that this method produces a schedule of length $\lceil n/3 \rceil \in \Omega(n)$

LISRA: The same holds for LISRA, although with a slightly different reasoning. LISRA iteratively removes the link which achieves the lowest $SINR$ with equal power distribution until β is reached. In our example, the link to be postponed will always be the longest link. As we have seen above, two neighboring links cannot be scheduled in the same time slot, hence LISRA also needs $\Omega(n)$ slots.

□

All four algorithms produce a schedule of length $\Omega(n)$ for this example. However, it is possible to construct a much shorter schedule. We present a schedule that needs as few as $O(\log n)$ time slots for the $n/2$ links.

Theorem 5.4: There exists a scheduling and power assignment scheme which produces a schedule of length $O(\log n)$ for scenario S for all $n > 16$.

Proof: Consider the schedule where every $\log n^{th}$ link starting with 1 is selected for transmission in slot 1, every $\log n^{th}$ link starting with 2 for slot 2, etc. More formally, we schedule $\{\lambda_t, \lambda_{t+\log n}, \lambda_{t+2\log n}, \ldots\}$ in time slot t. We construct a power assignment $P(s_i)$ such that every link exceeds a signal-to-interference-ratio of 2.

Let us have a closer look at the set Λ_t containing the links scheduled for time slot t. There are at most $\lceil \frac{n}{2\log n} \rceil$ links scheduled in this slot, of which we select link $\lambda_i = (-2^i, 2^i)$, the τ_i^{th} longest link. Consider the assignment $P(s_i) = (2n)^{\tau_i} 2^{\alpha(i+1)}$ to s_i and recall that $SINR(i) = P_r(s_i)/\sum_{\lambda_j \in \Lambda_t \setminus \{\lambda_i\}} I_i(s_j)$

We note that the largest interference is caused by the neighboring links $\lambda_{(i-\log n)}$ and $\lambda_{(i+\log n)}$. Moreover, the interference power is cut in half for each link further away from λ_i.

Claim 5.5: The following two inequalities hold:

$$\begin{aligned} I_i(s_{i-j\log n}) &> 2I_i(s_{i-(j+1)\log n}) \qquad \text{and} \\ I_i(s_{i+j\log n}) &> 2I_i(s_{i+(j+1)\log n}) \qquad \forall 0 < j < n, n > 4. \end{aligned}$$

Proof: The first inequality holds because of

$$\frac{I_i(s_{i-j\log n})}{I_i(s_{i-(j+1)\log n})} = \frac{P(s_{i-j\log n})g(s_{i-j\log n},r_i)}{P(s_{i-(j+1)\log n})g(s_{i-(j+1)\log n},r_i)}$$

$$= \frac{(2n)^{\tau_i+j}2^{\alpha(i-j\log n+1)}(2^i+2^{i-j\log n})^{-\alpha}}{(2n)^{\tau_i+j+1}2^{\alpha(i-(j+1)\log n+1)}(2^i+2^{i-(j+1)\log n})^{-\alpha}}$$

$$= \frac{n^\alpha}{2n\cdot n^\alpha}\frac{(1+n^{j+1})^\alpha}{(1+n^j)^\alpha} \geq 2 \qquad \forall n \geq 3.$$

The other inequality can be proved analogously. □

Applying Claim 5.5 we can bound $SINR(i)$ as follows

$$SINR(i) = \frac{P_r(s_i)}{\sum_{\lambda_j\in\Lambda_t\setminus\{\lambda_i\}} I_i(s_j)} \geq \frac{P_r(s_i)}{2(I_i(s_{i-\log n})+I_i(s_{i+\log n}))}$$

$$= \frac{\frac{(2n)^{\tau_i}2^{\alpha(i+1)}}{2^{\alpha(i+1)}}}{2\left(\frac{(2n)^{\tau_i+1}2^{\alpha(i-\log n+1)}}{(2^i+2^{i-\log n})^\alpha}+\frac{(2n)^{\tau_i-1}2^{\alpha(i+\log n+1)}}{(2^i+2^{i+\log n})^\alpha}\right)}$$

$$= \frac{n(n+1)^\alpha}{2^\alpha(4n^2+n^\alpha)} \geq 2 \qquad \forall n > 16.$$

Since the above holds for all communication requests in all slots, we have proved that this schedule allows the successful transmission of all links in $O(\log n)$ time slots. □

Let us now examine the schedule our LDS-protocol creates for this scenario. The 3-disturbance χ_3 of setting S is 1. Consequently, we obtain a schedule of length $O(\log^2 n)$ by plugging in the value $\rho = 3$ into the bound of Theorem 4.9. Notice that this is *exponentially shorter* than the schedules generated by any uniform or linear power assignment protocol as well as any of the known link removal heuristics.

Corollary 5.6: For $\rho = 3$, the LDS scheduling algorithm produces a schedule of length $O(\log^2 n)$ for scenario S.

The LDS algorithm thus significantly outperforms existing scheduling strategies in worst-case scenarios. Nonetheless, the analysis of the power assignment $P(\cdot)$ of Theorem 5.4 demonstrates that an even better solution with complexity $O(\log n)$ exists. Hence, the aim for future research remains to devise algorithms, with results even closer to the optimum.

VI. Discussion and Conclusions

In this paper, we have shown that all scheduling protocols studied so far may have an extremely suboptimal performance in worst-case networks. In order to ameliorate this situation, we propose the LDS scheduling algorithm. By employing a novel power assignment scheme and reuse distance criterion, our algorithm achieves a provably efficient performance in any network and request setting that features low disturbance. Thereby, we prove our solution to outperform all currently existing scheduling protocols and algorithms by as much as an exponential factor.

In its current state, the LDS protocol is centralized and hence suited to be employed in static networks with known traffic patterns only. Finding a distributed algorithm in a manner similar to the LDS protocol is an exciting open problem. Ideally, such a distributed worst-case efficient scheduling algorithm could lead to improved MAC-layer solutions, as combined power control and scheduling are crucial to a theoretical understanding of media access control problems.

In general, it can be argued that the network topologies and request sequences found in real-world applications may not have an explicit worst-case structure. We hope, however, that our novel power assignment strategy in combination with the theoretical insights gained from our worst-case analysis will ultimately lead to a significant increase in bandwidth and capacity beyond heuristics in real networks. Further investigation in this direction are bound to prove useful in areas such as wireless mesh networks, sensor networks, or even cellular networks.

References

[1] J. Aein. Power balancing in systems employing frequency reuse. *COMSAR Tech. Rev.*, vol. 3, 1973.

[2] A. Behzad and I. Rubin. On the Performance of Graph-based Scheduling Algorithms for Packet Radio Networks. In *Proceedings of the IEEE Global Telecommunications Conference (GLOBECOM)*, 2003.

[3] P. Björklund, P. Värbrand, and D. Yuan. A column generation method for spatial TDMA scheduling in ad hoc networks. *Ad Hoc Networks*, 2(4):405–418, 2004.

[4] G. Brar, D. Blough, and P. Santi. Computationally Efficient Scheduling with the Physical Interference Model for Throughput Improvement in Wireless Mesh Networks. In *Proceedings of the 12th International Conference on Mobile Computing and Networking (MOBICOM)*, 2006.

[5] R. Cruz and A. Santhanam. Optimal routing, link scheduling and power control in multi-hop wireless networks, 2003.

[6] T. A. ElBatt and A. Ephremides. Joint scheduling and power control for wireless ad-hoc networks. In *INFOCOM*, 2002.

[7] A. Ephremides and T. V. Truong. Scheduling broadcasts in multihop radio networks. *IEEE Trans. Communications*, 38(4):456–460, Apr. 1990.

[8] G. J. Foschini and Z. Miljanic. A simple distributed autonomous power control algorithms and its convergence. *IEEE Transactions on Vehicular Technology*, 40(4):641–646, Nov. 1993.

[9] S. A. Grandhi, R. Vijayan, and D. J. Goodman. Distributed power control in cellular radio systems. *IEEE Trans. on Communications*, 42, 1994.

[10] J. Grönkvist and A. Hansson. Comparison Between Graph-Based and Interference-Based STDMA Scheduling. In *Proceedings of the 2nd ACM International Symposium on Mobile Ad Hoc Networking & Computing (MOBIHOC)*, pages 255–258, 2001.

[11] P. Gupta and P. R. Kumar. The Capacity of Wireless Networks. *IEEE Trans. Information Theory*, 46(2):388–404, 2000.

[12] T.-H. Lee, J.-C. Lin, and Y. T. Su. Downlink power control algorithms for cellular radio systems. *IEEE Trans. Veh. Technol.*, 44, 1995.

[13] T. Moscibroda and R. Wattenhofer. The Complexity of Connectivity in Wireless Networks. In *Proc. of the 25th IEEE INFOCOM*, 2006.

[14] T. Moscibroda, R. Wattenhofer, and Y. Weber. Protocol Design Beyond Graph-based Models. In *Proceedings of the 5th ACM SIGCOMM Workshop on Hot Topics in Networks (HotNets)*, 2006.

[15] T. Moscibroda, R. Wattenhofer, and A. Zollinger. Topology Control meets SINR: The Scheduling Complexity of Arbitrary Topologies. In *Proc. of the 7th ACM Symposium on Mobile Ad Hoc Networking and Computing (MOBIHOC)*, 2006.

[16] S. Papavassiliou and L. Tassiulas. Joint optimal channel base station and power assignment for wireless access. *IEEE/ACM Trans. Netw*, 4(6):857–872, 1996.

[17] B. Radunovic and J.-Y. Le Boudec. Optimal Power Control, Scheduling, and Routing in UWB Networks. *Journal on Selected Areas in Communications*, 2(7), 2004.

[18] B. Radunovic and J.-Y. Le Boudec. Rate Performance Objectives of Multi-hop Wireless Networks. In *Proc. 23th IEEE INFOCOM*, 2004.

[19] K. Wang, C. Chiasserini, R. Rao, and J. Proakis. A joint solution to scheduling and power control for multicasting in wireless ad hoc networks. *EURASIP Journal on Applied Signal Processing*, 2005.

[20] J. Zander. Performance of optimum transmitter power control in cellular radio systems. *IEEE Trans. Veh. Technol.*, 41, 1992.

[21] J. Zander. Distributed cochannel interference control in cellular radio systems. *IEEE Trans. Veh. Technol.*, vol. 41, Aug. 1992.