Distributed Contention Resolution in Wireless Networks*

Thomas Kesselheim Berthold Vöcking

July 13, 2010

Abstract

We present and analyze simple distributed contention resolution protocols for wireless networks. In our setting, one is given n pairs of senders and receivers located in a metric space. Each sender wants to transmit a signal to its receiver at a prespecified power level, e. g., all senders use the same, uniform power level as it is typically implemented in practice. Our analysis is based on the physical model in which the success of a transmission depends on the Signal-to-Interference-plus-Noise-Ratio (SINR). The objective is to minimize the number of time slots until all signals are successfully transmitted.

Our main technical contribution is the introduction of a measure called maximum average affectance enabling us to analyze random contention-resolution algorithms in which each packet is transmitted in each step with a fixed probability depending on the maximum average affectance. We prove that the schedule generated this way is only an $\mathcal{O}(\log^2 n)$ factor longer than the optimal one, provided that the prespecified power levels satisfy natural monontonicity properties. By modifying the algorithm, senders need not to know the maximum average affectance in advance but only static information about the network. In addition, we extend our approach to multi-hop communication achieving the same appoximation factor.

*Department of Computer Science, RWTH Aachen University, Germany. {thomask,voecking}@cs.rwth-aachen.de.
This work has been supported by the UMIC Research Centre, RWTH Aachen University.

1 Introduction

In a wireless network, communication carried out at the same time is not physically separated. Therefore transmissions may collide due to too much interference. The Media Access Control (MAC) layer's task is to coordinate the communication such that simultaneous transmissions do not interfere but that the medium is sufficiently used to allow for optimal throughput. In this paper, we present and analyze distributed contention-resolutions protocols for this scheduling task giving worst-case guarantees.

The interference constraints are modelled by the *physical interference model* [12]. Between any two nodes of the network u and v a distance $d(u,v)$ is defined. The received signal strength decreases when this distance is increasing. More formally, if node u transmits a signal at power level p then it is received by v with strength $p/d(u,v)^\alpha$, where the constant $\alpha > 0$ is the so-called *path-loss exponent*[1]. The node v can successfully decode this signal if the signal strength received from the intended sender is at least β times as large as the signals strengths by interfering transmissions made at the same time plus ambient noise. This is, the *Signal-to-Interference-plus-Noise Ratio (SINR)* is above some threshold $\beta \geq 0$, the so-called *gain*.

In the *interference scheduling problem*, we are given a set of n requests $\mathcal{R} \subseteq V \times V$, corresponding to pairs of nodes from a metric space. For each request $\ell \in \mathcal{R}$, we have to select a power level $p(\ell) > 0$ and a time slot $c(\ell) \in [k] := \{1, \ldots, k\}$ such that for each $\ell = (u,v) \in \mathcal{R}$ the SINR constraint

$$\frac{p(\ell)}{d(u,v)^\alpha} \geq \beta \left(\sum_{\substack{\ell'=(u',v') \in \mathcal{R} \\ c(\ell)=c(\ell')}} \frac{p(\ell')}{d(u',v)^\alpha} + N \right)$$

is fulfilled. The constant $N \geq 0$ expresses ambient noise that all transmissions have to cope with. The objective is to minimize the number of time slots k.

So, in fact, two choices are made: For each request ℓ, we have to select a power level $p(\ell)$ and a time slot in which the transmission should take place (to make the distinction clear the latter problem is also referred to as *coloring*). In this work, we focus on the coloring problem and assume the power assignment to be given such that all senders know a priori at which power to transmit. For example, powers might be given by hardware or depend on the distance between the sender and the receiver as described below.

Our objective is to calculate a schedule whose length is close to the length of the optimal schedule in the same instance. This measure was introduced by Moscibroda et al. as *scheduling complexity* $T(\mathcal{R})$ [18]. As we concentrate on the coloring problem, we compare lengths of the schedules we compute to the optimal schedule length for $\mathcal{R}$ that uses some fixed power assignment p, denoted by $T(\mathcal{R}, p)$.

In order to have an algorithm that is applicable in a realistic environment it has to work in a distributed fashion with as little information as possible. In fact, our algorithms only require static information on the network that can be spread at the time of deployment. Particularly, the number of network nodes, the clock synchronization and the power assignment can be seen as such static information. In contrast, no information about the current state of the network will be necessary. For example, communication requests arise after the deployment and an algorithm has to work without knowledge on which requests have to be served by the network and which of them were already successfully served.

In related work several schemes for assigning the powers have been used. The simplest way is to make all transmissions at the same power level, e. g., the maximum power supported by the hardware. These power assignments are called *uniform* [14, 3].

A more complex solution that still facilitates a distributed implementation at least in theory is to make the power only dependent of the distance between the respective sender and receiver (and therefore

[1]Typically it is assumed that $2 < \alpha < 5$. However, our analysis works for any $\alpha > 0$.

independent of the other nodes). In *linear power assignments* [10] the power for a transmission between a sender and a receiver whose distance is d is chosen proportional to d^α and thus proportional to the minimum transmission power needed to deal with ambient noise. *Square-root* (or mean) power assignments [9, 13] choose the transmission power for distance d proportional to $\sqrt{d^\alpha}$. So the transmission power still grows for increasing distances but not as fast as in a linear power assignment.

For each of the power schemes mentioned above, there are specialized algorithms and several bounds on how fixing to this scheme influences the optimal schedule length. Most of these algorithms are centralized; so far, distributed algorithms with a provable performance guarantee are only known for linear power assignments [10] (cf. Section 1.2). Furthermore, most existing transceivers support only a relatively small, fixed number of possible power levels so that a practical implementation of both linear and square-root power assignments remain a challenge. As a consequence it is necessary to have more general algorithms which do not only work for a certain power scheme.

Our algorithms solve these issues as they work in a distributed fashion and take the power assignment as input. They do not require a certain power scheme but work for every power assignment satisfying the following natural conditions. First, it has to be *non-decreasing* and *sublinear* or *linear*. That means if $d(\ell) \leq d(\ell')$ for two requests $\ell, \ell' \in \mathcal{R}$ then

$$p(\ell) \leq p(\ell') \quad \text{and} \quad \frac{p(\ell)}{d(\ell)^\alpha} \geq \frac{p(\ell')}{d(\ell')^\alpha} \ . \tag{1}$$

So the transmission power of ℓ' has to be at least as large as the one for ℓ. At the same time, the received power at the receiver of ℓ' must not be larger than the one at the receiver of ℓ. This monotonicity condition is very natural and is fulfilled by all previously studied power assignments, particularly the ones mentioned above.

The second condition is that powers are chosen sufficiently large so that ambient noise plays a minor part compared to interference. In particular, we assume the power received at any receiver is at least some constant factor higher than the minimum power that is needed to deal with noise (βN). To simplify notation, we assume this constant factor to be 2. So for all requests $\ell \in \mathcal{R}$

$$\frac{p(\ell)}{d(\ell)^\alpha} \geq 2\beta N \ . \tag{2}$$

This ensures that we actually deal with conflicts due to too large interference and not due to too weak transmission power. Previous approaches also used this assumption but stated it rather implicitly. Indeed it may be violated by uniform and square-root power assignments when considering too large distances. But in this case the problem consists of rather dealing with noise than with interference.

1.1 Our Contribution

We introduce a new measure called *maximum average affectance* $\bar{A}(\mathcal{R}, p)$ that depends on the request set $\mathcal{R}$ and the power assignment p. This measure extends a so-called *measure of interference* for linear power assignments [10] in a non-trivial way towards general power assignments satisfying Conditions 1 and 2. It is the key for analyzing the performance of simple contention resolution protocols in wireless networks with prespecified power assignments and comparing it to the optimum that could be achieved.

For two requests $\ell = (u, v)$ and $\ell' = (u', v')$, and a power assignment p, we define the *affectance* of ℓ on ℓ' by

$$a_p(\ell, \ell') = \min\left\{1, \beta \frac{p(\ell)}{d(u, v')^\alpha} \bigg/ \left(\frac{p(\ell')}{d(u', v')^\alpha} - \beta N\right)\right\} \ .$$

The notion of affectance was introduced by Halldórsson and Wattenhofer [14], which we extended to arbitrary power assignments and bounded by 1. When taking the noise out of consideration, it indicates which amount of interference ℓ induces at ℓ', normalized by the signal strength from the intended sender of ℓ. As a consequence the sum of affectance is at most 1 for a request set that may be assigned to same time slot.

To get the *maximum average affectance* $\bar{A}(\mathcal{R}, p)$, we take the maximum over all subsets of requests and consider the average affectance a link is exposed to from all other requests in this subset.

Definition 1. *The* maximum average affectance *of a request set* $\mathcal{R}$ *and a power assignment* p *is given by*

$$\bar{A}(\mathcal{R}, p) = \max_{M \subseteq \mathcal{R}} \operatorname*{avg}_{\ell' \in M} \sum_{\ell \in M} a_p(\ell, \ell') = \max_{M \subseteq \mathcal{R}} \frac{1}{|M|} \sum_{\ell' \in M} \sum_{\ell \in M} a_p(\ell, \ell') \ .$$

If $\mathcal{R}$ and p are clear from the context, we simply write $\bar{A}$. When replacing the average by the maximum in this definition, we would get a measure that is closely related to the measure of interference in [10].

The *maximum average affectance* enables us to derive lower bounds on the scheduling complexity for all power assignments satisfying Conditions 1 and 2. In particular, we prove $\bar{A}(\mathcal{R}, p)$ is at most a factor $\mathcal{O}(\log n)$ larger than the optimal schedule length $T(\mathcal{R}, p)$. This way it enables us to compare schedules we compute to the optimal schedule length $T(\mathcal{R}, p)$.

We use this measure to analyze random contention-resolution based algorithms. In this kind of algorithms each sender transmits with a certain probability q in each step until one of the transmissions has successfully been received. We first prove a stability result. If $q \leq 1/4\bar{A}$, all transmissions are successful within $\mathcal{O}(\log n/q)$ time slots whp[2]. Thus choosing $q = 1/4\bar{A}$, we generate a schedule of length $\mathcal{O}(\bar{A} \cdot \log n)$ whp, which is at most $\mathcal{O}(T(\mathcal{R}, p) \cdot \log^2 n)$.

To make the algorithm applicable to a distributed setting, we present two modifications. These do not affect the schedule length vitally and we still get schedules of length $\mathcal{O}(\bar{A} \cdot \log n)$ whp. On the one hand, we extend it such that the network nodes do not have to know $\bar{A}$ anymore but adapt the transmission probability q on their own. On the other hand, we find a way to inform each sender if a transmission has successfully been received by transmitting acknowledgement packets. This is not a trivial task because these acknowledgement packets may also interfere.

Altogether, this is the first distributed algorithm to the interference scheduling problem with a guaranteed approximation ratio. The algorithm is distributed in the following sense. It can be run on all senders and receivers of a network such that during the execution no central entity is needed that spreads information about the current state of the network, e. g. which requests have to be scheduled. The nodes only need static information, namely the power assignment, a rough estimation on the total number of nodes and a synchronized clock.

As a further result, we adapt the ideas to a distributed multi-hop algorithm that allows packets to use intermediate relay nodes. For a fixed choice of paths and powers we get an $\mathcal{O}(\log^2 n)$ whp approximation for this problem as well. For most of the other approaches to scheduling such an adaptation is not possible. In particular, for uniform and square-root power assignments, no multi-hop scheduling algorithm with a provable performance guarantee was known up to now.

1.2 Related Work

Scheduling in wireless networks has been subject to reseach for more than four decades up to now. Right from the beginning random access protocols such as ALOHA [1] have been dealt with. Collision avoidance in models featuring binary conflict constraints has been deeply studied over the years (see, e. g.

[2] with high probability: with probability $1 - n^c$ for each constant c

[4, 6, 7]). However, in the physical interference model the interference constraints are not binary but take all other network nodes into consideration in an additive way. Therefore, it brings about new problems and challenges. Depending on the choice of powers interesting phenomena such as nested pairs [9] can be observed. Furthermore, characterizing the optimal schedule length is much more involved.

The analysis of scheduling algorithms in the physical model that deals with arbitrarily (and not randomly) distributed network nodes has been initiated by Moscibroda and Wattenhofer [17]. However, they do not solve the interference scheduling problem but a similar one. Instead, they find and schedule a set of pairs of senders and receivers such that all network nodes are strongly connected. They prove scheduling these pairs is possible in $\mathcal{O}(\log^4 n)$ time steps (independent of the topology) with a certain (sublinear) power assignment. This approach was extended to arbitrary requests sets [18]. However, the lengths of the generated schedules were not compared to the optimal schedule length. In fact, they can be a factor of $\Omega(n)$ away from the optimal one.

There has also been much work on algorithms with an approximation guarantee compared to the optimal schedule length. Most of these algorithms are centralized.

Using uniform power assignments, Halldórsson and Wattenhofer [14] present an $\mathcal{O}(1)$ approximation in the plane compared to uniform power assignment. This problem was proved to be NP-hard by Goussevskaia et al. [11]. Andrews and Dinitz [2] in contrast compare to the optimal power assignment and present an $\mathcal{O}(\log \Delta \cdot \log n)$ approximation in the plane. Here, Δ is the ratio between the longest and the shortest distance between a sender and its receiver. They also prove the joint problem of power control and coloring to be NP-hard. Avin et al. [3] additionally show that the optimal schedule can at most be a factor $\mathcal{O}(\log P_{\max})$ shorter if the ratio between the maximum and the minimum power used is $P_{\max}$.

With special regard to linear power assignments, Fanghänel et al. [10] introduced a measure of interference I as a bound on the optimal schedule length. It holds $I = \mathcal{O}(\log \Delta \cdot \log n \cdot T(\mathcal{R}))$ in general and $I = \mathcal{O}(T(\mathcal{R}, p))$ if p is a linear power assignment. This bound is complemented by an algorithm using $\mathcal{O}(I + \log^2 n)$ time slots which results in an $\mathcal{O}(1)$ approximation in linear power assignments for sufficiently dense instances.

Chafekar et al. [5] also use linear power assignments to get an $\mathcal{O}(\log^2 \Delta \log^2 \Gamma \log n)$ approximation for the joint multi-hop scheduling and routing problem, where Γ is the maximum and the minimum transmission power used by the optimum. This result was improved by Fanghänel et al. [10] to $\mathcal{O}(\log \Delta \log^2 n)$. So far, these were the only algorithms for the multi-hop problem with a performance guarantee.

Square-root power assignments were introduced by Fanghänel et al. [9]. They proved that in the bidirectional model (featuring undirected requests) it yields schedules that are $\mathcal{O}(\log^{3.5+\alpha} n)$ times as long as the ones using the optimal power assignment. Halldórsson [13] improved this result to $\mathcal{O}(\log n)$ in fading metrics. He also showed that in the unidirectional model (as presented above) the resulting schedule is at most $\mathcal{O}(\log \log \Delta \cdot \log^2 n)$ away from the one using the optimal power assignment.

All of the mentioned approximation guarantees depend on Δ and indeed there are instances [13] with large Δ where the algorithm only computes a (trivial) $\Omega(n)$ approximation. Only very recently [15] an approach has been presented that achieves approximation guarantees poly-logarithmic in n. Nevertheless, optimizing transmission powers in a de-centralized way still remains a challenge. So distance-based power assignments seem to be a reasonable way for distributed environments at least in theory.

2 Distributed Single-Hop Scheduling Algorithms

Random contention-resolution algorithms are probably the most intuitive way to share limited resources among several agents in a distributed fashion. The idea is that each agent accesses the resource in any time slot with a certain probability q until its first success. In case of a collision, none of the involved agents is successful in this round.

This idea is easily applicable in wireless networks by letting each sender transmit its packet in each time slot with probability q until the first success. Due to its simplicity, this and similar approaches are also relevant for practical applications. However, if the transmission probability q is chosen too small, time slots are not sufficiently used. In contrast, if it is chosen too large, conflicts are likely.

In this section, we present an analysis of a random contention-resolution algorithm for the interference scheduling problem. This analysis is based on the *maximum average affectance* $\bar{A}$. We start by analyzing a single time slot in which some senders transmit with probability q while the others remain silent. We prove that if q is chosen small enough, a $q/4$ fraction of the senders taking part succeed.

Lemma 1. *Given a request set $\mathcal{R}' \subseteq \mathcal{R}$. Consider a time slot in which each sender of the requests in $\mathcal{R}'$ transmits with probability $q \leq 1/4\bar{A}$, the others remain silent. Then at least $q/4 \cdot |\mathcal{R}'|$ transmissions are successful in expectation.*

Proof. For $\ell \in \mathcal{R}'$, let X_ℓ be the 0/1 random variable indicating if ℓ transmits, and X'_ℓ be the 0/1 random variable indicating if the transmission is successful.

Note that to have $X'_\ell = 1$, i. e. to make transmission ℓ successful, it suffices to have

$$X_\ell = 1 \qquad \text{and} \qquad \sum_{\ell' \in \mathcal{R}'} a_p(\ell', \ell) X_{\ell'} < 1 \ .$$

By Markov inequality, we have

$$\mathbf{Pr}\left[X'_\ell = 0 \mid X_\ell = 1\right] \leq \mathbf{Pr}\left[\sum_{\ell' \in \mathcal{R}'} a_p(\ell', \ell) X_{\ell'} \geq 1\right] \leq \mathbf{E}\left[\sum_{\ell' \in \mathcal{R}'} a_p(\ell', \ell) X_{\ell'}\right] = \sum_{\ell' \in \mathcal{R}'} a_p(\ell', \ell) q \ .$$

Let

$$M := \left\{ \ell \in \mathcal{R}' \mid \sum_{\ell' \in \mathcal{R}'} a_p(\ell', \ell) \leq 2\bar{A} \right\} \ .$$

By definition of $\bar{A}$, we know that $|M| \geq \frac{1}{2} \cdot |\mathcal{R}'|$.

For a request $\ell \in M$, we have

$$\sum_{\ell' \in \mathcal{R}'} a_p(\ell', \ell) q \leq 2\bar{A}q \leq \frac{1}{2}$$

which implies that $\mathbf{E}\left[X'_\ell\right] = \mathbf{Pr}\left[X'_\ell = 1\right] = \mathbf{Pr}\left[X_\ell = 1\right] \cdot \mathbf{Pr}\left[X'_\ell = 1 \mid X_\ell = 1\right] \geq q/2$. In case $\ell \notin M$, we simply use $\mathbf{E}\left[X'_\ell\right] \geq 0$.

This yields the expected total number of successful transmissions is

$$\mathbf{E}\left[\sum_{\ell \in \mathcal{R}} X'_\ell\right] = \sum_{\ell \in \mathcal{R}} \mathbf{E}\left[X'_\ell\right] \geq \frac{q}{2} \cdot |M| \geq \frac{q}{4} \cdot |\mathcal{R}'| \ .$$

□

We use this lemma to analyze Algorithm 1, which takes the transmission probability q as a parameter. Each sender transmits its packet with probability q until the first success.

The measure $\bar{A}$ allows us to derive a relation between the transmission probability q and the time until the last transmission has successfully taken place. This can be seen as a stability result for a fixed value q. We find out a value such that if $\bar{A}$ is below it, collisions do not take place too often and all packets are successfully delivered fast.

Theorem 2. *If $q \leq 1/4\bar{A}$, Algorithm 1 needs $\mathcal{O}(\log n/q)$ time slots whp.*

Algorithm 1: Scheduling using random contention resolution with transmission probability q.

while *success* $\neq$ *true* **do**
  transmit with probability q;

Proof. Let be n_t the random variable indicating the number of requests that have not been successfully scheduled in the time slots 1, ..., t.

By Lemma 1, we have $\mathbf{E}[n_{t+1} \mid n_t = k] \geq k - \frac{q}{4}k$ and so

$$\mathbf{E}[n_{t+1}] \leq \sum_{k=0}^{\infty} \mathbf{Pr}[n_t = k] \cdot \left(1 - \frac{q}{4}\right) k = \left(1 - \frac{q}{4}\right) \sum_{k=0}^{\infty} k \cdot \mathbf{Pr}[n_t = k] = \left(1 - \frac{q}{4}\right) \mathbf{E}[n_t] \quad .$$

Using $n_0 = n$, this yields

$$\mathbf{E}[n_t] \leq \left(1 - \frac{q}{4}\right)^t n \quad .$$

In particular, after $4c \ln n/q$ time slots for each constant c, the expected number of remaining requests is

$$\mathbf{E}\left[n_{4c \ln n/q}\right] \leq \left(1 - \frac{q}{4}\right)^{4c \ln n/q} n \leq \left(\frac{1}{e}\right)^{c \ln n} n = n^{1-c} \quad .$$

The Markov inequality yields

$$\mathbf{Pr}\left[n_{4c \ln n/q} \neq 0\right] = \mathbf{Pr}\left[n_{4c \ln n/q} \geq 1\right] \leq \mathbf{E}\left[n_{4c \ln n/q}\right] \leq n^{1-c}$$

So we need $\mathcal{O}(\log n/q)$ time slots whp. □

We achieve the best result when choosing $q = 1/4\bar{A}$, which yields a schedule of length $\mathcal{O}(\bar{A} \cdot \log n)$ whp. In Section 4 we will see this yields an $\mathcal{O}(\log^2 n)$ approximation of the optimal schedule. However, two major issues prevent this algorithm from being applied in distributed scenarios. On the one hand, a suitable transmission probability has to be chosen, which requires knowing $\bar{A}$. On the other hand, senders have to know when to stop transmitting. This cannot be determined from the position of the sender node but only from the receiver. Therefore, in a distributed setting, senders have to be informed someway. In the next sections, we will present solutions to cope with these two problems.

2.1 Determining the Optimal Transmission Probability

One major drawback of Algorithm 1 is that it needs to get the transmission probability as a parameter, which has to be chosen suitably to guarantee short schedules. If senders do not know the network or the request this is not possible. We solve this problem by applying the idea of an *exponential backoff* as follows. Algorithm 2 works the same way as Algorithm 1 but does not have the parameter q anymore. Instead, it starts with a high transmission probability and reduces it if the transmission has not been successful during a longer period. That causes eventually the transmission probability to be small enough that no collisions occur.

Although the algorithm is much more complex, we get a guarantee that is not essentially worse than the one for Algorithm 1.

Theorem 3. *When all nodes apply Algorithm 2, scheduling takes $\mathcal{O}(\bar{A} \cdot \log n)$ time slots whp.*

Algorithm 2: A Distributed Single-Hop Scheduling Algorithm

$k := 0$;
while *success* $\neq$ *true* **do**
  run Algorithm 1 for $8 \ln n/q$ time slots with parameter $q = \frac{1}{4 \cdot 2^k}$;
  $k := k + 1$;

Proof. We proved above that in the kth run of Algorithm 1 the probability that any of the (remaining) requests is not successfully scheduled is at most $1/n$ if $k \geq \lceil \log \bar{A} \rceil$. Ignoring all success made during the first $\lceil \log \bar{A} \rceil - 1$ executions, we can conclude that the probability that there is some unsuccessful request left after the first $c + \lceil \log \bar{A} \rceil - 1$ executions is at most $1/n^c$.

These executions include

$$\sum_{k=0}^{c+\lceil \log \bar{A} \rceil - 1} 2^{k+5} \cdot \lceil \ln n \rceil = \mathcal{O}(2^c \cdot \bar{A} \cdot \log n)$$

time slots. So, we can conclude that scheduling is completed after $\mathcal{O}(\bar{A} \cdot \log n)$ time slots whp. □

3 Sending Acknowledgements

Taking the model again into consideration, it only states feasibility of one-way communication. This yields senders do not know if a transmission has successfully been received. Our solution to this problem is to use acknowledgement packets to inform a sender that its transmission was successfully received by the intended receiver. These acknowledgement packets also need one time slot to be transmitted. In the final algorithm, they will be transmitted in even time slots, whereas the actual data packets are transmitted in odd time slots.

We need to assign powers to the acknowledgement transmissions as well. For these transmissions, the original senders act as receivers and vice versa. Using the same power as for the other transmission does not work in general, because there are instances and power assignments in which the optimal schedule length increases by $\Omega(n)$ when exchanging senders and receivers.

3.1 Dual Instances

For a request $\ell = (u, v)$, we define the *dual request* ℓ^* by (v, u). Analogously, for a request set $\mathcal{R}$ the *dual request set* $\mathcal{R}^*$ is defined by $\mathcal{R}^* = \{\ell^* \mid \ell \in \mathcal{R}\}$. For a request set $\mathcal{R}$ and a power assignment $p \colon \mathcal{R} \to \mathbb{R}_{>0}$, we define the *dual power assignment* $p^* \colon \mathcal{R}^* \to \mathbb{R}_{>0}$ by

$$p^*(\ell^*) = \frac{p(\ell')^2}{d(\ell')^\alpha} \cdot \frac{d(\ell)^\alpha}{p(\ell)} \qquad \text{where } \ell' = \arg\max_{\ell \in \mathcal{R}} d(\ell) \ .$$

Note that if p fulfills Conditions 1, then p^* also does and $(p^*)^* = p$. Note that the ℓ' factor is only necessary to ensure Condition 2 holds. It could also be chosen much larger. So when switching to a subset of $\mathcal{R}$, we do not have to use a different dual power assignment.

We can observe that in the dual power assignment p^* the affectance of a dual request ℓ^* on another dual request ℓ'^* is bounded by the affectance of ℓ' on ℓ in the power assignment p.

Observation 4. *Given two requests $\ell, \ell' \in \mathcal{R}$ and some power assignments p, we have $a_{p^*}(\ell^*, \ell'^*) \leq 2a_p(\ell', \ell)$.*

Proof. Let be $\ell_1^* = (v_1, u_1)$, $\ell_2^* = (v_2, u_2)$. We have

$$a_{p^*}(\ell_1^*, \ell_2^*) = \min \left\{ 1, \beta \frac{p^*(\ell_1^*)}{d(v_1,u_2)^\alpha} \middle/ \left(\frac{p^*(\ell_2^*)}{d(v_2,u_2)^\alpha} - N \right) \right\} \ .$$

Since $p(\ell)/d(\ell)^\alpha \geq 2\beta N$ for all $\ell \in \mathcal{R}$, this is at most

$$2 \min \left\{ 1, \beta \frac{p^*(\ell_1^*)}{d(v_1,u_2)^\alpha} \middle/ \frac{p^*(\ell_2^*)}{d(v_2,u_2)^\alpha} \right\} \ .$$

By definition of ℓ_1^*, ℓ_2^*, and p^* this is equal to

$$2 \min \left\{ 1, \beta \frac{d(u_1,v_1)^\alpha}{p(\ell_1) d(v_1,u_2)^\alpha} \middle/ \frac{1}{p(\ell_2)} \right\} = 2 \min \left\{ 1, \beta \frac{p(\ell_2)}{d(u_2,v_1)^\alpha} \middle/ \frac{p(\ell_1)}{d(u_1,v_1)^\alpha} \right\} \ .$$

This is at most

$$2 \min \left\{ 1, \beta \frac{p(\ell_2)}{d(u_2,v_1)^\alpha} \middle/ \left(\frac{p(\ell_1)}{d(u_1,v_1)^\alpha} - N \right) \right\} = 2 a_p(\ell_2, \ell_1) \ .$$

□

This observation directly yields the maximum average affectance for a dual request set under the dual power assignment differs by at most a factor of 2 from the original one.

Lemma 5. *For all request sets $\mathcal{R}$ and power assignments p, we have $\bar{A}(\mathcal{R}^*, p^*) \leq 2\bar{A}(\mathcal{R}, p)$.*

Proof. Let be $\mathcal{R}' \subseteq \mathcal{R}$. The above yields

$$\frac{1}{|\mathcal{R}'^*|} \sum_{\ell_1 \in \mathcal{R}'^*} \sum_{\ell_2 \in \mathcal{R}'^*} a_{p^*}(\ell_1^*, \ell_2^*) \leq 2 \frac{1}{|\mathcal{R}'|} \sum_{\ell_1 \in \mathcal{R}'} \sum_{\ell_2 \in \mathcal{R}'} a_p(\ell_1, \ell_2) \leq 2\bar{A}(\mathcal{R}, p) \ .$$

This holds for all $\mathcal{R}'^*$, so we also have $\bar{A}(\mathcal{R}^*, p^*) \leq 2\bar{A}(\mathcal{R}, p)$. □

So, we found a power assignment for the dual request set whose maximum average affectance is not much higher than the one for the original request set. Therefore it is suitable for the acknowledgement transmissions. For the algorithm, we again assume that all transmission powers are a priori known to the senders. This is, the receivers know the dual power assignment.

3.2 Scheduling Algorithm

In order to implement acknowledgement transmissions, we let each receiver transmit a packet back to its sender immediately in the time slot after having received a packet. However, as these transmissions may still interfere each other, each one is only transmitted with probability $1/8$. Otherwise, no acknowledgement is transmitted yielding a retransmission. Each sender only stops transmissions after having successfully received an acknowledgement. Algorithm 3 extends Algorithm 1 by these ideas. We can still adapt the approach of Algorithm 2 that assigns different values for the parameter q.

Theorem 6. *If $q \leq 1/4\bar{A}$, Algorithm 3 needs $\mathcal{O}(\log n/q)$ time slots whp.*

Algorithm 3: An extended algorithm implementing acknowledgements.

```
while success ≠ true do
    transmit with probability q (otherwise wait one time slot);
    wait for acknowledgment (one time slot);
    if acknowledgement has been received then
        success := true;
```

Proof. Let us first consider a single iteration of the while loop. Let $\mathcal{R}_{\text{remain}}$ be the set of requests that have not been successfully scheduled up to now. Let $\mathcal{R}_{\text{receive}}$ be the set of requests for which the transmission is successfully decoded at the receiver. Let $\mathcal{R}_{\text{success}}$ in turn be the set of requests whose sender receives an acknowledgement.

In Lemma 1 we proved that

$$\mathbf{E}\left[|\mathcal{R}_{\text{receive}}|\right] \geq \frac{q}{4} \cdot |\mathcal{R}_{\text{remain}}| \enspace .$$

Furthermore, we know that $\bar{A}(\mathcal{R}_{\text{receive}}, p) \leq 1$ and so by Lemma 5 we have $\bar{A}(\mathcal{R}^*_{\text{receive}}, p^*) \leq 2$. All receivers transmit their acknowledgment with probability $1/8$, so we can apply Lemma 1 once more to get

$$\mathbf{E}\left[|\mathcal{R}_{\text{success}}|\right] \geq \frac{1}{32} \cdot |\mathcal{R}_{\text{receive}}| \enspace .$$

Let n_t be the random variable indicating the number of requests that have not been successfully scheduled until the tth iteration. We can conclude that

$$\mathbf{E}\left[n_{t+1}\right] \leq \mathbf{E}\left[n_t\right] - \frac{q}{4} \cdot \frac{1}{32} \cdot \mathbf{E}\left[n_t\right] = \left(1 - \frac{q}{128}\right) \mathbf{E}\left[n_t\right]$$

and thus

$$\mathbf{E}\left[n_t\right] \leq \left(1 - \frac{1}{128}\right)^t n \enspace .$$

The probability that not all requests are successful within $128 \cdot c \cdot \lceil \ln n \rceil / q$ iterations is

$$\mathbf{Pr}\left[n_{128c\ln n/q} \geq 1\right] \leq \mathbf{E}\left[n_{128c\ln n/q}\right] \leq \left(1 - \frac{q}{128}\right)^{128c\ln n/q} n \leq \frac{1}{e^{2c\ln n}} n = n^{1-c} \enspace .$$

Each iteration consists of 2 time slots. Altogether, we can conclude that we need $\mathcal{O}(\log n/q)$ time slots whp. ☐

As we see, we only lose a constant factor in the schedule length when using the acknowledgement packets as above. In order to adapt the approach of Algorithm 2, the only modification needed is that for each possible value of q the algorithm has now to be run for $256\ln n/q$ steps. However, this also changes the resulting schedule length by only a constant factor, still ensuring $\mathcal{O}(\bar{A} \cdot \log n)$ whp.

In total, we get an algorithm that is fully distributed in the sense that nodes do not need any additional information about the current state of the network. The only assumption we need is all nodes have to know a rough estimation of the total number of requests n and which powers to use and they have a synchronized clock.

4 Comparison to the Optimal Schedule

To this point, we have presented several algorithms, each with a performance bound of $\mathcal{O}(\bar{A} \cdot \log n)$ whp. However, it still remains to show that $\bar{A}$ is not far from the optimal schedule length in order to compare the performance of this algorithm to the optimal schedule. In contrast, for all similar measures this is not guaranteed – they can differ by a factor of $\Omega(n)$ from the optimal schedule length. As a matter of fact, this would also happen if we used the *maximum* instead of the *average*. In particular, we will prove that $\bar{A} = \mathcal{O}(T(\mathcal{R}, p) \cdot \log n)$. This will prove that the schedules calculated by our algorithms are at most a factor of $\mathcal{O}(\log^2 n)$ whp away from the optimal schedule.

As a first step towards this result, we consider a set of requests $\mathcal{R}$ that can be scheduled in a single time slot. Informally spoken, we add a request ℓ that is shorter than all requests in $\mathcal{R}$ but does not have to be scheduled in the same time slot as $\mathcal{R}$. We derive a bound on how much affectance this request ℓ is exposed to. This lemma can be seen as a generalization of Theorem 1 in [10].

Lemma 7. *Given a set $\mathcal{R}$ of requests that may be scheduled in a single time slot using some power assignment p fulfilling Conditions 1 and 2, and another request ℓ with $d(\ell) \leq d(\ell')$ for all $\ell' \in \mathcal{R}$, then we have*

$$\sum_{\ell' \in \mathcal{R}} a_p(\ell', \ell) = \mathcal{O}(1) \ .$$

By decomposing the set $\mathcal{R}$ corresponding to the schedule this directly yields the following generalization where $\mathcal{R}$ may be scheduled in T time slots.

Lemma 8. *Given a set $\mathcal{R}$ of links that may be scheduled in T slots using some power assignment p fulfilling Conditions 1 and 2, and another request ℓ with $d(\ell) \leq d(\ell')$ for all $\ell' \in \mathcal{R}$, then we have*

$$\sum_{\ell' \in \mathcal{R}} a_p(\ell', \ell) = \mathcal{O}(T) \ .$$

Proof of Lemma 7. Let be $\ell = (u, v)$ and $\mathcal{R} = \{\ell_1, \ldots, \ell_{\bar{n}}\}$ with $\ell_i = (u_i, v_i)$.

To bound the sum, we distinguish between two kinds of summands. Let j be the index of the receivers $v_1, \ldots, v_{\bar{n}}$ that is closest to v, that is $j \in \arg\min_{i \in [\bar{n}]} d(v_i, v)$. We define a set U of indices of requests whose senders u_i lie within a distance of at most $\frac{1}{2} d(v_j, v)$ from v, i.e. $U = \{i \in [\bar{n}] \mid d(u_i, v) \leq \frac{1}{2} d(v_j, w)\}$. Using the triangle inequality we can conclude for all $i \in U$:

$$d(u_i, v_j) \leq d(u_i, v) + d(v, v_j) \leq \frac{3}{2} d(v_j, w) \ . \tag{3}$$

In addition, we have

$$\begin{aligned} d(v_j, v) &\leq d(v_i, v) && \text{since } v_j \text{ is the closest receiver} \\ &\leq d(v_i, u_i) + d(u_i, v) && \text{by triangle inequality} \\ &\leq d(v_i, u_i) + \frac{1}{2} d(v_j, w) && \text{by definition of } U \ . \end{aligned}$$

This implies

$$d(v_j, w) \leq 2 d(u_i, v_i) \ . \tag{4}$$

Combining Equation 3 and Equation 4 we get $d(u_i, v_j) \leq 3 d(u_i, v_i)$. We now consider the sender k that uses the smallest transmission power among all senders in U. For $k \in \arg\min_{i \in U} p_i$ we get

$$\sum_{\substack{i \in U \\ i \neq k}} \frac{p_i}{d(u_i, v_k)^\alpha} \geq \sum_{\substack{i \in U \\ i \neq k}} \frac{p_i}{3^\alpha d(u_k, v_k)^\alpha} = \frac{1}{3^\alpha d(u_k, v_k)^\alpha} \sum_{\substack{i \in U \\ i \neq k}} p_i \geq \frac{p_k}{3^\alpha d(u_k, v_k)^\alpha} |U \setminus \{k\}| \ .$$

The transmission from u_k to v_k is also successful, which means the SINR constraint is fulfilled

$$\sum_{\substack{i \in U \\ i \neq k}} \frac{p_i}{d(u_i, v_k)^\alpha} \leq \frac{1}{\beta} \frac{p_k}{d(u_k, v_k)^\alpha} \ .$$

In combination this yields $|U| \leq \frac{3^\alpha}{\beta} + 1$.

Let us now proceed to the summands that do not belong to U. For all $i \in [\bar{n}] \setminus U$ it holds that

$$\begin{aligned} d(u_i, v_j) &\leq d(u_i, w) + d(w, v_j) && \text{by triangle inequality} \\ &\leq d(u_i, w) + 2d(u_i, w) && \text{by definition of } U \\ &= 3d(u_i, w) \ . \end{aligned}$$

Now, we take the sum of $a_p(\ell, \ell_i)$ for all $i \in [\bar{n}] \setminus U$, $i \neq j$. Note that $p(\ell)/d(u,v)^\alpha \geq p(\ell_i)/d(u_i,v_i)^\alpha$ due to Condition 1. Using the above calculations and the fact the SINR constraint is fulfilled, we get

$$\sum_{\substack{i \in [\bar{n}] \setminus U \\ i \neq j}} a_p(\ell_i, \ell) \leq \sum_{\substack{i \in [\bar{n}] \setminus U \\ i \neq j}} \beta \frac{p(\ell_i)}{d(u_i, v)^\alpha} \Bigg/ \left(\frac{p(\ell)}{d(u, v)^\alpha} - N \right) \leq \sum_{\substack{i \in [\bar{n}] \setminus U \\ i \neq j}} \beta \frac{p(\ell_i)}{\frac{1}{3^\alpha} d(u_i, v_j)^\alpha} \Bigg/ \frac{p(\ell_j)}{d(u_j, v_j)^\alpha}$$

$$= 3^\alpha \beta \sum_{\substack{i \in [\bar{n}] \setminus U \\ i \neq j}} \frac{p(\ell_i)}{d(u_i, v_j)^\alpha} \Bigg/ \frac{p(\ell_j)}{d(u_j, v_j)^\alpha} \leq 3^\alpha \ .$$

Summing up all $i \in [\bar{n}]$ gives

$$\sum_{i \in [\bar{n}]} a_p(\ell_i, \ell) \leq |U| + \sum_{\substack{i \in [\bar{n}] \setminus U \\ i \neq j}} a_p(\ell_i, \ell) + a_p(\ell_j, \ell) \leq \frac{3^\alpha}{\beta} + 1 + 3^\alpha + 1 = \mathcal{O}(1) \ .$$

□

For the same situation as above, we now bound the sum of affectance that ℓ causes at all requests in $\mathcal{R}$.

Lemma 9. *Given a set $\mathcal{R}$ of links that may be scheduled in T time slots using power assignment p and another link ℓ with $d(\ell) \leq d(\ell')$ for all $\ell' \in \mathcal{R}$, then we have*

$$\sum_{\ell' \in \mathcal{R}} a_p(\ell, \ell') = \mathcal{O}(T \cdot \log n) \ .$$

Proof. To prove this lemma, we make use of the results on dual instances we presented in Section 3.1. For the the dual instance $\mathcal{R}^*$ and the dual power assignment p^* we showed

$$\sum_{\ell' \in \mathcal{R}} a_p(\ell, \ell') \leq 2 \sum_{\ell'^* \in \mathcal{R}^*} a_{p^*}(\ell'^*, \ell^*) \ .$$

Furthermore p^* fulfills Conditions 1 and 2. This allows us to apply Lemma 8 and get

$$\sum_{\ell'^* \in \mathcal{R}^*} a_{p^*}(\ell'^*, \ell^*) = \mathcal{O}(T(\mathcal{R}^*, p^*)) \ .$$

So it only remains to show that $T(\mathcal{R}^*, p^*) = \mathcal{O}(T(\mathcal{R}, p) \cdot \log n)$. Let $\mathcal{R}_1, \ldots, \mathcal{R}_T$ be the decomposition of $\mathcal{R}$ made by the schedule. We know that each of the sets $\mathcal{R}_t$ fulfills the SINR constraint using the power assignment p. This implies that $\bar{A}(\mathcal{R}_t, p) \leq 1$ for all $t \in [T]$.

By Lemma 5, we have $\bar{A}(\mathcal{R}_t^*, p^*) \leq 2$. Thus, using Algorithm 1 there is a schedule of length $\mathcal{O}(\log n)$ for each request set $\mathcal{R}_t^*$ using power assignment p^*. A concatenation of these schedules gives us a schedule of length $\mathcal{O}(T \cdot \log n)$ for the entire set $\mathcal{R}^*$. □

The combination of both above lemmas allows us to compare $\bar{A}(\mathcal{R}, p)$ to the optimal schedule length $T(\mathcal{R}, p)$.

Theorem 10. *Given a set $\mathcal{R}$ of links that may be scheduled in T time slots using some power assignment p fulfilling Conditions 1 and 2. Then $T = \Omega(\bar{A}(\mathcal{R},p)/\log n)$.*

Proof. We show that $\bar{A}(\mathcal{R}, p) = \mathcal{O}(T \cdot \log n)$. Let be $\mathcal{R}' \subseteq \mathcal{R}$ and $\mathcal{R}' = \{\ell_1, \ldots, \ell_{\bar{n}}\}$ with $d(\ell_1) \leq d(\ell_2) \leq \ldots \leq d(\ell_{\bar{n}})$. Observe that in this notation Lemmas 8 and 9 yield

$$\sum_{\substack{j \in [\bar{n}] \\ j > i}} a_p(\ell_i, \ell_j) = \mathcal{O}(T) \qquad \text{and} \qquad \sum_{\substack{j \in [\bar{n}] \\ j > i}} a_p(\ell_j, \ell_i) = \mathcal{O}(T \cdot \log n) \enspace .$$

So we get

$$\frac{1}{|\mathcal{R}'|} \sum_{\ell \in \mathcal{R}'} \sum_{\ell' \in \mathcal{R}'} a_p(\ell, \ell') = \frac{1}{|\mathcal{R}'|} \sum_{i \in [\bar{n}]} \sum_{j \in [\bar{n}]} a_p(\ell_i, \ell_j) = \frac{1}{|\mathcal{R}'|} \sum_{i \in [\bar{n}]} \sum_{\substack{j \in [\bar{n}] \\ j > i}} a_p(\ell_i, \ell_j) + a_p(\ell_j, \ell_i) \enspace .$$

This yields the claim. □

So, in total, this guarantees that the schedules calculated by our algorithms are at most a factor of $\mathcal{O}(\log^2 n)$ longer than the optimal one that uses the same power assignment. Interestingly, this optimal schedule is not required to use acknowledgement packets and therefore not be computable in a distributed way.

5 Multi-Hop Scheduling

In the multi-hop variant of the interference scheduling problem, packets are routed via intermediate networks nodes until reaching their final destination. As we do for the powers, we assume also the paths to be fixed. This is, each packet has a predefined path, which is for example given by routing tables in the network nodes.

The total number of packets is denoted by m, the maximum number of hops that a packet has to cross is denoted by D (dilation). For the node that is the jth hop destination of the ith packet we write $P_{i,j}$.

Our algorithm applies the technique of random delays that has successfully been applied to scheduling in wired [16] and also in wireless [5, 10] networks. That is, time is divided into time frames such that each packet attempts to cross one hop in each time frame after having waiting the initial delay at its origin node.

However, in a distributed environment, determining the maximum delay is more involved as we assume that all nodes only know static information on the network. They neither know which packets have to be scheduled in general nor which is the future path some packet will take. Algorithm 4 deals with this problem as follows. It works in phases. In phase k each packet is assigned a delay independently uniformly at random that is at most 2^k. The phase consists of $\mathcal{O}(2^k \log^2 n)$ time slots that are grouped to 2^{k+1} time frames each of length $2^{12} \cdot 18 \cdot \lceil \ln^2 n \rceil$.

During each of these phases Algorithm 3 is executed, where in each step each node works as a receiver if it does not decide to transmit in this step. In each time frame, each packet attempts to cross one hop. If a packet fails to cross a hop in the respective time slot, it is not considered anymore during this phase but deferred to the next one starting at the node where the failure occurred.

Theorem 11. *If T is the optimal schedule length, Algorithm 4 results in a schedule length of $\mathcal{O}(T \cdot \log^2 n)$ whp.*

Proof. Let $\mathcal{R} = \{(u, v) \in V \times V \mid s = P_{i,j}, r = P_{i+1,j}$ for some $i, j\}$ be the set of all hops and $C = \max_{\ell \in \mathcal{R}} \sum_{\ell' \in \mathcal{R},\, d(\ell') \geq d(\ell)} a_p(\ell, \ell') + a_p(\ell', \ell)$. We bound the probability that all packets are delivered to their final destination in phase k where $2^k \geq C/9 \log n + D$. Note that this choice also ensures the phase consists of enough time frames that even a packet of path length D and the maximum delay could reach its final destination within this phase.

Let $X_{\ell,\ell'}$ be a 0/1 random variable indicating if ℓ and ℓ' are allocated to the same time frame. By definition $\mathbf{E}[X_{\ell,\ell'}] \leq 9 \ln n / C$ for all $\ell, \ell' \in \mathcal{R}$. Furthermore define for all $\ell \in \mathcal{R}$ a random variable by $C_\ell = \sum_{\ell' \in \mathcal{R},\, d(\ell') \geq d(\ell)} (a_p(\ell, \ell') + a_p(\ell', \ell)) X_{\ell,\ell'}$. We have $\mathbf{E}[C_\ell] \leq 9 \ln n$.

The random variables $X_{\ell,\ell'}$ are *negatively associated* as defined by Dubhashi and Ranjan [8]. This allows us to use a Chernoff bound to get $\mathbf{Pr}[C_\ell \geq 18 \ln n] \leq \exp(-3 \ln n) \leq 1/n^3$ for all $\ell \in \mathcal{R}$. So, we get $\mathbf{Pr}[\exists \ell \in \mathcal{R} :\ C_\ell \geq 18 \ln n] \leq 1/n^2$. Furthermore, if all $C_\ell < 18 \ln n$ then also $\bar{A}(\mathcal{R}_t, p) \leq 18 \ln n$ for all t. This is, with probability at least $1 - 1/n^2$, the condition $q \leq 1/4\bar{A}(\mathcal{R}_t,p)$ is satisfied for all t.

For Algorithm 3, we proved that the probability that some of the requests is not successfully scheduled during $2^9 \cdot \lceil \ln n \rceil / q$ time slots is at most $1/n^3$. This is the failure probability for each time frame. Since there are at most n time frames used, we get a total failure probability of at most $1/n^2$ if $q \leq 1/4\bar{A}(\mathcal{R}_t,p)$ for all t. We proved above the probability of the latter event not to happen is also $1/n^2$. So all packets reach their final destination during phase k with probability at least $1 - 1/n$.

Observe that for this analysis any progress made in past phases only increases the probability of success. So for all constants c the probability that not all packet reach their final destinations within the first $c + \lceil \log (C/9 \log n + D) \rceil - 1$ phases is at most $1/n^c$. These phases consist of $\mathcal{O}(2^c(C \log n + D \log^2 n))$ time slots. In other words, we are finished after $\mathcal{O}(C \log n + D \log^2 n)$ time slots whp.

It now remains to compare this schedule length to T. In Lemmas 8 and 9, we proved that $C = \mathcal{O}(T \cdot \log n)$. Furthermore, we obviously have $D \leq T$. Thus $C \log n + D \log^2 n = \mathcal{O}(T \cdot \log^2 n)$. □

Algorithm 4: A Distributed Multi-Hop Scheduling Algorithm

```
for k = 1 to ∞ do
    assign each waiting packet a delay δ_i ∈ [2^k];
    for t = 1 to 2^{k+1} do
        set R_t = {(P_{i,j}, P_{i,j+1}) | j = t + δ_i};
        run Algorithm 3 with q = 1/72 · ⌈ln n⌉ on R_t for 2^10 · ⌈ln n⌉/q time slots;
        foreach transmission that fails during the execution do
            δ_i := ∞;
```

6 Adaptation to Different Scenarios

6.1 Devices without Power Control

As uniform power assignments obviously fulfill Condition 1, and also Condition 2 if the power is large enough, the basic scheduling results are immediately applicable even if the senders do not support transmitting at smaller power levels than the maximum. However, to transmit the acknowledgement packets we used the dual power assignment p^*, which is not uniform in general. So, in order to apply this algorithm, devices have to be able to control their transmission power.

Nevertheless, if devices are not able of power control we can still use the key techniques. Suppose all transmissions have to be made at the same power level, say $\hat{p}$. If $\beta > 1$, we can prove the following result, which is similar to Observation 4.

Observation 12. *If ℓ and ℓ' can be transmitted in the same time slot using power $\hat{p}$ and $\beta > 1$, then*

$$a_{\hat{p}}(\ell^*, \ell'^*) \leq \left(\frac{\beta+1}{\beta-1}\right)^\alpha \cdot a_{\hat{p}}(\ell, \ell') \ .$$

Proof. Let $\ell = (u, v)$ and $\ell' = (u', v')$. The SINR constraint gives us

$$d(u', v')^\alpha \leq \frac{1}{\beta} d(u, v')^\alpha \qquad \text{and} \qquad d(u, v)^\alpha \leq \frac{1}{\beta} d(u', v)^\alpha \ .$$

The triangle inequality furthermore yields

$$d(u, v') \leq d(u, v) + d(v, u') + d(u', v') \leq \frac{1}{\beta} d(u', v) + d(u', v) + \frac{1}{\beta} d(u, v') \ .$$

We can conclude that $d(u', v) \geq ({\beta-1}/{\beta+1}) \cdot d(u, v')$.

For $a_{\hat{p}}(\ell^*, \ell'^*)$, we get

$$\begin{aligned} a_{\hat{p}}(\ell^*, \ell'^*) &= \min\left\{1, \beta \frac{\hat{p}}{d(v, u')^\alpha} \Big/ \left(\frac{\hat{p}}{d(v', u')^\alpha} + N\right)\right\} \\ &\leq \min\left\{1, \beta \left(\frac{\beta+1}{\beta-1}\right)^\alpha \frac{\hat{p}}{d(u, v')^\alpha} \Big/ \left(\frac{\hat{p}}{d(u', v')^\alpha} + N\right)\right\} \\ &\leq \left(\frac{\beta+1}{\beta-1}\right)^\alpha \min\left\{1, \beta \frac{\hat{p}}{d(u, v')^\alpha} \Big/ \left(\frac{\hat{p}}{d(u', v')^\alpha} + N\right)\right\} \\ &= \left(\frac{\beta+1}{\beta-1}\right)^\alpha \cdot a_{\hat{p}}(\ell, \ell') \ . \end{aligned}$$

□

This observation yields $\bar{A}(\mathcal{R}^*, \hat{p}) = \mathcal{O}(1)$ if $\mathcal{R}$ can be scheduled in a single time slot. So, we can send the acknowledgements in a similar manner as in Algorithm 3 using again the transmission power $\hat{p}$ rather than the dual power assignment. These changes do not affect the core of the analysis but only the constant factors involved. However, this result only holds for the case that all transmissions use the same transmission power $\hat{p}$.

6.2 Bidirectional Model

In the bidirectional model [9], requests are undirected because both involved nodes act as a sender and a receiver at the same time. In order to estimate the interference between two requests ℓ and ℓ' the smallest of the distances is relevant. For this model, we can replace the affectance definition by

$$a_p(\ell, \ell') = \min\left\{1, \beta \frac{p(\ell)}{\min\{d(u, u')^\alpha, d(u, v')^\alpha, d(v, u')^\alpha, d(v, v')^\alpha\}} \Big/ \left(\frac{p(\ell')}{d(u', v')^\alpha} - \beta N\right)\right\} \ .$$

With this definition, the above results can be transferred to the bidirectional model. Particularly, we get distributed algorithms with an approximation factor of $\mathcal{O}(\log^2 n)$ whp.

7 Discussion and Open Problems

While previous algorithms are mostly centralized, the algorithms and analyses we presented seem to be much closer to realistic scenarios. However, we made use of three assumptions, namely the power assignment is given, each node has a rough estimation of the total number of nodes and the clocks are synchronized. These are suitable assumptions as this knowledge can be spread when deploying the network. On the contrary, the information that arises over time, e. g. which transmissions have to be made is not necessary.

Nevertheless, it is an interesting question which performance can still be achieved without this knowledge. Unfortunately, we cannot get rid of any of these assumptions in a non-trivial way. For example, for the clock synchronization the standard ALOHA trick [19] does not work. However, concerning the number of nodes and the clock synchronization there are various results in other scenarios that could possibly be transferred.

For the power assignment problem the best solution for distributed settings up to now is to take distance-based power schemes such as the square-root power assignment. With our algorithms this yields an $\mathcal{O}(\log\log\Delta \cdot \log^3 n)$ approximation compared to the optimal schedule using the optimal power assignment. Very recently [15] an approach has been presented calculate a power assignment achieving an approximation ratio that is independent of Δ. However this algorithm only works in a centralized way. So there is still much space for future research in distributed protocols for these problems.

8 Acknowledgements

We like to thank Alexander Fanghänel for valuable discussions and comments.

References

[1] Norman Abramson. The aloha system: another alternative for computer communications. In *AFIPS '70 (Fall): Proceedings of the November 17-19, 1970, fall joint computer conference*, pages 281–285, 1970.

[2] Matthew Andrews and Michael Dinitz. Maximizing capacity in arbitrary wireless networks in the sinr model: Complexity and game theory. In *Proceedings of the 28th Conference of the IEEE Communications Society (INFOCOM)*, 2009.

[3] Chen Avin, Zvi Lotker, and Yvonne Anne Pignolet. On the power of uniform power: Capacity of wireless networks with bounded resources. In *Proceedings of the 17th annual European Symposium on Algorithms (ESA)*, pages 373–384, 2009.

[4] Reuven Bar-Yehuda, Oded Goldreich, and Alon Itai. On the time-complexity of broadcast in multi-hop radio networks: An exponential gap between determinism and randomization. *Journal of Computer and System Sciences*, 45(1):104 – 126, 1992.

[5] Deepti Chafekar, V. S. Anil Kumar, Madhav V. Marathe, Srinivasan Parthasarathy, and Aravind Srinivasan. Cross-layer latency minimization in wireless networks with SINR constraints. In *Proceedings of the 8thACM International Symposium Mobile Ad-Hoc Networking and Computing (MOBIHOC)*, pages 110–119, 2007.

[6] Andrea E.F. Clementi, Pilu Crescenzi, Angelo Monti, Paolo Penna, and Riccardo Silvestri. On computing ad-hoc selective families. In *Proceedings of the 5th International Workshop on Randomization and Approximation Techniques in Computer Science (RANDOM-APPROX)*, pages 211–222, 2001.

[7] Erik D. Demaine, Mohammad Taghi Hajiaghayi, Uriel Feige, and Mohammad R. Salavatipour. Combination can be hard: approximability of the unique coverage problem. In *Proceedings of the 17th ACM-SIAM Symposium on Discrete Algorithms (SODA)*, pages 162–171, 2006.

[8] Devdatt P. Dubhashi and Desh Ranjan. Balls and bins: A study in negative dependence. *Random Structures and Algorithms*, 13(2):99–124, 1998.

[9] Alexander Fanghänel, Thomas Kesselheim, Harald Räcke, and Berthold Vöcking. Oblivious interference scheduling. In *Proceedings of the 28th ACM symposium on Principles of distributed computing*, pages 220–229, 2009.

[10] Alexander Fanghänel, Thomas Kesselheim, and Berthold Vöcking. Improved algorithms for latency minimization in wireless networks. In *Proceedings of the 36th International EATCS Colloquium on Automata, Languages and Programming (ICALP)*, pages 447–458, 2009.

[11] Olga Goussevskaia, Yvonne Anne Oswald, and Roger Wattenhofer. Complexity in geometric SINR. In *Proceedings of the 8th ACM International Symposium Mobile Ad-Hoc Networking and Computing (MOBIHOC)*, pages 100–109, New York, NY, USA, 2007.

[12] Piyush Gupta and P. R. Kumar. The capacity of wireless networks. *IEEE Transactions on Information Theory*, 46:388–404, 2000.

[13] Magnús M. Halldórsson. Wireless scheduling with power control. In *Proceedings of the 17th annual European Symposium on Algorithms (ESA)*, pages 361–372, 2009.

[14] Magnús M. Halldórsson and Roger Wattenhofer. Wireless communication is in APX. In *Proceedings of the 36th International EATCS Colloquium on Automata, Languages and Programming (ICALP)*, pages 525–536, Berlin, Heidelberg, 2009. Springer-Verlag.

[15] Thomas Kesselheim. A constant-factor approximation for wireless capacity maximization with power control in the SINR model. *CoRR*, abs/1007.1611, 2010.

[16] F. T. Leighton, Bruce M. Maggs, and Satish B. Rao. Packet routing and job-shop scheduling in O(congestion+dilation) steps. *Combinatorica*, 1994.

[17] Thomas Moscibroda and Roger Wattenhofer. The complexity of connectivity in wireless networks. In *Proceedings of the 25th Conference of the IEEE Communications Society (INFOCOM)*, pages 1–13, 2006.

[18] Thomas Moscibroda, Roger Wattenhofer, and Aaron Zollinger. Topology control meets SINR: The scheduling complexity of arbitrary topologies. In *Proceedings of the 7th ACM International Symposium Mobile Ad-Hoc Networking and Computing (MOBIHOC)*, pages 310–321, 2006.

[19] Lawrence G. Roberts. Aloha packet system with and without slots and capture. *SIGCOMM Comput. Commun. Rev.*, 5(2):28–42, 1975.